Unit 7
Clustering

Prepared By

Arjun Singh Saud, Asst. Prof. CDCSIT

Introduction

- The process of grouping a set of physical or abstract objects into classes of *similar* objects is called clustering. It is an unsupervised learning technique.
- A cluster is a collection of data objects that are *similar* to one another within the same cluster and are *dissimilar* to the objects in other clusters.
- Clustering can also be used for outlier detection, where outliers may be more interesting than common cases.

Introduction

- Applications of outlier detection include the detection of credit card fraud and the monitoring of criminal activities in electronic commerce.
- For example, exceptional cases in credit card transactions, such as very expensive and frequent purchases, may be of interest as possible fraudulent activity.
- The data used in cluster analysis can be interval, ordinal or categorical. However, having a mixture of different types of variable will make the analysis more complicated.

Similarity and Dissimilarity

- Distance measures are used in order to find similarity or dissimilarity between data objects.
- The most popular distance measure is Euclidean distance, which is defined as:

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

Where, $x = (x_1, y_1)$ *and* $y = (x_2, y_2)$

- Another well-known metric is Manhattan (or city block) distance, defined as

$$d(x,y) = |x_2 - x_1| + |y_2 - y_1|$$

Similarity and Dissimilarity

- *Minkowski distance* is a generalization of both Euclidean distance and Manhattan distance. It is defined as

$$d(x, y) = \left(|x_2 - x_1|^p + |y_2 - y_1|^p \right)^{1/p}$$

- Where p is a positive integer, such a distance is also called L_p norm, in some literature. It represents the Manhattan distance when p = 1 (i.e., L_1 norm) and Euclidean distance when p = 2 (i.e., L_2 norm).

Categories of Clustering Algorithms

- Many clustering algorithms exist in the literature. In general, the major clustering methods can be classified into the following categories.

1. **Partitioning methods**: Given a database of n objects or data tuples, a partitioning method constructs k partitions of the data, where each partition represents a cluster and $k < n$. Given k, the number of partitions to construct, a partitioning method creates an initial partitioning. It then uses an iterative relocation technique that attempts to improve the partitioning by moving objects from one group to another.

Categories of Clustering Algorithms

2. **Hierarchical methods:** A hierarchical method creates a hierarchical decomposition of the given set of data objects. A hierarchical method can be classified as being either *agglomerative* or *divisive*. The *agglomerative approach* follows the *bottom-up* approach. It starts with each object forming a separate group. It successively merges the objects or groups that are close to one another, until a termination condition holds. The *divisive approach* follows the *top-down* approach. It starts with all of the objects in the same cluster. In each successive iteration, a cluster is split up into smaller clusters, until a termination condition holds.

Categories of Clustering Algorithms

- **Density-based methods:** Most partitioning methods cluster objects based on the distance between objects. Such methods can find only spherical-shaped clusters and encounter difficulty at discovering clusters of arbitrary shapes. Other clustering methods have been developed based on the notion of *density*. Their general idea is to continue growing the given cluster as long as the density (number of objects or data points) in the neighborhood exceeds some threshold.
- **Model-based methods:** Model-based methods hypothesize a model for each of the clusters and find the best fit of the data to the given model. EM is an algorithm that performs expectation-maximization analysis based on statistical modeling.

K-Means Algorithm

- k-means is one of the simplest partitioning based clustering algorithm. The procedure follows a simple and easy way to classify a given data set into a certain number of clusters (assume k clusters) fixed apriori.
- The main idea is to define k centers, one for each cluster. These centers should be selected cleverly because of different location causes different result. So, the better choice is to place them as much as possible far away from each other.

K-Means Algorithm

- Let $X = \{x_1, x_2, x_3, \ldots\ldots, x_n\}$ be the set of data points and $C = \{c_1, c_2, \ldots\ldots, c_k\}$ be the set of cluster centers.
  1. Randomly select *k* cluster centers.
  2. Calculate the distance between each data point and cluster centers.
  3. Assign the data point to the cluster center whose distance from the cluster center is minimum of all the cluster centers.
  4. If
     - No data is reassigned then terminate
  5. Else
     - Recalculate the new cluster center using centroid.
     - Recalculate the distance between each data point and new cluster centers.
     - Go to step 3

K-Means Algorithm

Example: Divide the data points {(2,10), ((2,5), (8,4), (5,8), (7,5), (6,4)} into two clusters.

Solution

Let **p1=(2,10)** **p2=(2,5)** p3=(8,4) **p4=(5,8)** p5=(7,5) p6=(6,4)

Initial step

Choose Cluster centers randomly

Let c1=(2,5) and c2=(6,4) are two initial cluster centers.

K-Means Algorithm

Iteration 1: Calculate distance between clusters centers and each data points

d(c1,p1)=5	d(c2,p1)=7.21
d(c1,p2)=0	d(c2,p2)=4.12
d(c1,p3)=6.08	d(c2,p3)=2
d(c1,p4)=4.24	d(c2,p4)=4.12
d(c1,p5)=5	d(c2,p5)=1.41
d(c1,p6)=4.12	d(c2,p6)=0

Thus, Cluster1={p1,p2} cluster2={p3,p4,p5,p6}

K-Means Algorithm

Iteration 2: *New Cluster centers: c1=(2,7.5) c2=(6.5,5.25)*

Again, Calculate distance between clusters centers and each data points

d(c1,p1)=2.5	d(c2,p1)=6.54
d(c1,p2)=2.5	d(c2,p2)=4.51
d(c1,p3)=6.95	d(c2,p3)=1.95
d(c1,p4)=3.04	d(c2,p4)=3.13
d(c1,p5)=4.59	d(c2,p5)=0.56
d(c1,p6)=5.32	d(c2,p6)=1.35
Thus, Cluster1={p1,p2,p4}	cluster2={p3,p5,p6}

K-Means Algorithm

Iteration 3: *New Cluster centers: c1=(3,7.67) c2=(7,4.33)*

Again, Calculate distance between clusters centers and each data points

d(c1,p1)=2.54	d(c2,p1=7.56
d(c1,p2)=2.85	d(c2,p2)=5.04
d(c1,p3)=6.2	d(c2,p3)=1.05
d(c1,p4)=2.03	d(c2,p4)=4.18
d(c1,p5)=4.81	d(c2,p5)=0.67
d(c1,p6)=4.74	d(c2,p6)=1.05
Thus, Cluster1={p1,p2,p4}	cluster2={p3,p5,p6}

K-Means Algorithm

Since, No data points are re-assigned

Final clusters are: Cluster1={p1,p2,p4} cluster2={p3,p5,p6}

K-Means++ Algorithm

- Randomization of picking k cluster centers in K-means algorithm results in the problem of initialization sensitivity.
- This problem tends to affect the final formed clusters. The final formed clusters depend on how initial cluster centers were picked.
- To overcome the above-mentioned drawback we use K-means++. This algorithm ensures a smarter initialization of the cluster centers and improves the quality of the clustering.
- Apart from initialization, the rest of the algorithm is the same as the standard K-means algorithm.

K-Means++ Algorithm

Initialization Algorithm

1. Randomly select the first cluster center from the data points.
2. For each data point compute its distance from the nearest, previously chosen cluster center.
3. Select the next cluster from the data points such that the probability of choosing a point as cluster center is directly proportional to its distance from the nearest, previously chosen cluster center.
4. Repeat steps 2 and 3 until K cluster centers have been sampled

K-Means++ Algorithm

Example: Consider the data points {(2,10), ((2,5), (8,4), (5,8), (7,5), (6,4),(3,2),(4,6)}. Select three cluster centers using K-means++ algorithm. Select data point with highest probability as next cluster center.

Solution

Let p1=(2,10) p2=(2,5) p3=(8,4) p4=(5,8) p5=(7,5) p6=(6,4) p7=(3,2) p8=(4,6)

Select the first cluster center randomly. Let c1=(2,5)

d(c1,p1)=5 d(c1,p2)=0 d(c1,p3)=6.08 d(c1,p4)=4.24

d(c1,p5)=5 d(c1,p6)=4.12 d(c1,p7)=3.6 d(c1,p8)=2.12

K-Means++ Algorithm

Thus, probabilities of data points being selected as next cluster center is directly proportional to:{5,0,6.08,4.24,5,4.12,3.6,2.12}

The data point p3=(8,4) has the highest probability of being selected as cluster center. Thus, c2=p3=(8,4)

Again, for each data point compute its distance from the nearest cluster center:

min(d(c1,p1), d(c2,p1))=5 min(d(c1,p2), d(c2,p2))=0

min(d(c1,p3), d(c2,p3))=0 min(d(c1,p4), d(c2,p4))=4.24

min(d(c1,p5), d(c2,p5))=5 min(d(c1,p6), d(c2,p6))=4.12

min(d(c1,p7), d(c2,p7))=5.39 min(d(c1,p8), d(c2,p8))=4.47

K-Means++ Algorithm

The data point p7=(3,2) has the highest probability of being selected as next cluster center.

Thus three cluster centers are: c1=(2,5) c2=(8,4), c3=(3,2)

Mini-batch K-means Algorithm

- The Mini-batch K-means clustering algorithm is a version of the K-means algorithm which can be used instead of the K-means algorithm when clustering on huge datasets.
- Sometimes it performs better than the standard K-means algorithm while working on huge datasets because it doesn't iterate over the entire dataset.
- It creates random batches of data to be stored in memory, then a random batch of data is collected on each iteration to update the clusters.

Mini-batch K-means Algorithm

- The main advantage of using the Mini-batch K-means algorithm is that it reduces the computational cost of finding a cluster.
- You may prefer to use the standard K-means algorithm, but when working on a huge dataset, you should prefer to use the mini-batch approach.

K-Medoids Algorithm

- K-Medoids is also a portioning based clustering algorithm. It is also called as Partitioning Around Medoid (PAM) algorithm.
- A medoid can be defined as the point in the cluster, whose dissimilarities with all the other points in the cluster is minimum.
- It majorly differs from the K-Means algorithm in terms of the way it selects the cluster centers.
- K-Means algorithm selects the average of a cluster's points as its center whereas the K-Medoid algorithm always picks the actual data points from the clusters as their centers

K-Medoids Algorithm

- The dissimilarity of the medoid(Ci) and object(Pi) is calculated by using D = |Pi - Ci|.
- The cost in K-Medoids algorithm is given as

$$c = \sum_{Ci} \sum_{pi \in ci} |pi - ci|$$

K-Medoids Algorithm

1. Select k random points out of the n data points as the medoids.
2. Associate each data point to the closest medoid.
3. Repeat while the cost decreases:
4. For each medoid m
    - For each non-medoid point o
        - Swap m and o
        - Associate each data point to the closest medoid
        - Re-compute the cost
5. If the total cost is more than that in the previous step
    - undo the swap.

K-Medoids Algorithm

Example: Divide the data points {(2,10), ((2,5), (8,4), (5,8), (7,5), (6,4)} into two clusters.

Solution

Let p1=(2,10) **p2=(2,5)** p3=(8,4) p4=(5,8) p5=(7,5) **p6=(6,4)**

Initial step

Let m1=(2,5) and m2=(6,4) are two initial cluster centers (medoid).

K-Medoids Algorithm

Iteration 1: Calculate distance between medoids and each data points

d(m1,p1)=5	d(m2,p1)=10
d(m1,p2)=0	d(m2,p2)=5
d(m1,p3)=7	d(m2,p3)=2
d(m1,p4)=6	d(m2,p4)=5
d(m1,p5)=5	d(m2,p5)=2
d(cm1,p6)=5	d(m2,p6)=0

Thus, Cluster1={p1,p2} cluster2={p3,p4,p5,p6}

Total Cost=5+0+2+5+2+0=14

K-Medoids Algorithm

Iteration 2: *swap m1 with p1, m1 =(2,10) m2=(6,4)*

Calculate distance between medoids and each data points

d(m1,p1)=0	d(m2,p1)=10
d(m1,p2)=5	d(m2,p2)=5
d(m1,p3)=12	d(m2,p3)=2
d(m1,p4)=5	d(m2,p4)=5
d(m1,p5)=10	d(m2,p5)=2
d(m1,p6)=10	d(m2,p6)=0

Thus, Cluster1={p1,p2,p4} cluster2={p3,p5,p6}

Total Cost=0+5+2+5+2+0=14

K-Medoids Algorithm

Iteration 3: *swap m1 with p3, m1 =(8,4) m2=(6,4)*

Calculate distance between medoids and each data points

d(m1,p1)=12 d(m2,p1)=10

d(m1,p2)=7 d(m2,p2)=5

d(m1,p3)=0 d(m2,p3)=2

d(m1,p4)=7 d(m2,p4)=5

d(m1,p5)=2 d(m2,p5)=2

d(m1,p6)=2 d(m2,p6)=0

Thus, Cluster1={p3,p5} cluster2={p1,p2,p4,p6}

Total Cost=10+5+0+5+2+0=22 => Undo Swapping

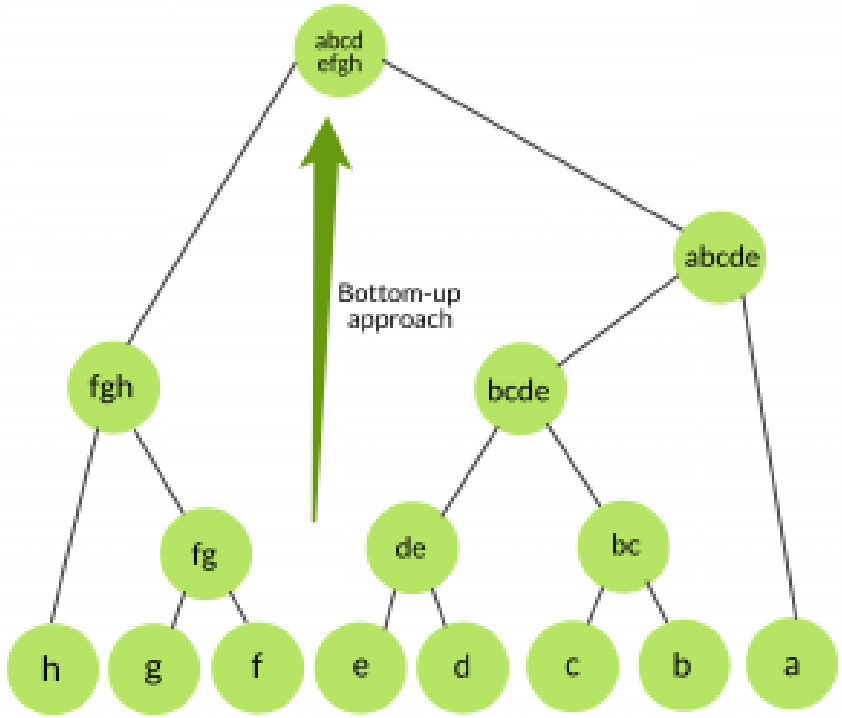
Continue this process…

Hierarchical Clustering

- In data mining and statistics, **hierarchical clustering** (also called **hierarchical cluster analysis** or **HCA**) is a method of cluster analysis which seeks to build a hierarchy of clusters. Strategies for hierarchical clustering generally fall into two types:
  - **Agglomerative**
  - **Divisive**

Agglomerative Clustering Algorithm

- This is a bottom up approach. Initially, each observation is considered in separate cluster and pairs of clusters are merged as one moves up the hierarchy.
- This process continues until the single cluster or required number of clusters are formed
- Distance matrix is used for deciding which clusters to merge.



Agglomerative Clustering Algorithm

Algorithm

- Compute the distance matrix between the input data points
- Let each data point be a cluster
- **Repeat**
  - Merge the two closest clusters
  - Update the distance matrix
- **Until** only K clusters remains

Agglomerative Clustering Algorithm

Example

Cluster the data points (1,1), (1.5,1.5), (5,5), (3,4), (4,4), (3, 3.5) into two clusters.

Solution

Let A=(1,1), B= (1.5,1.5), C=(5,5), D=(3,4), E=(4,4), F=(3,3.5)

Distance Matrix

Dist	A	B	C	D	E	F
A	0.00	0.71	5.66	3.61	4.24	3.20
B	0.71	0.00	4.95	2.92	3.54	2.50
C	5.66	4.95	0.00	2.24	1.41	2.50
D	3.61	2.92	2.24	0.00	1.00	0.50
E	4.24	3.54	1.41	1.00	0.00	1.12
F	3.20	2.50	2.50	0.50	1.12	0.00

Agglomerative Clustering Algorithm

The closest cluster are cluster F and D with shortest distance of 0.5. Thus, we group cluster D and F into cluster (D, F). Then we update the distance matrix.

Min Distance (Single Linkage)

Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	3.20	4.24
B	0.71	0.00	4.95	2.50	3.54
C	5.66	4.95	0.00	2.24	1.41
D, F	**3.20**	**2.50**	**2.24**	0.00	1.00
E	4.24	3.54	1.41	**1.00**	0.00

We can see that the closest distance between cluster B and cluster A is now 0.71. Thus, we group cluster A and cluster B into a single cluster named (A, B). The updated distance matrix is given below.

Agglomerative Clustering Algorithm

Min Distance (Single Linkage)

Dist	A,B	C	(D, F)	E
A,B	0	4.95	2.50	3.54
C	**4.95**	0	2.24	1.41
(D, F)	**2.50**	2.24	0	1.00
E	**3.54**	1.41	1.00	0

we can see that the closest distance between clusters happens between cluster E and (D, F) at distance 1.00. Thus, we cluster them together into cluster ((D, F), E). The updated distance matrix is given below

Agglomerative Clustering Algorithm

Min Distance (Single Linkage)

Dist	(A,B)	C	(D, F), E
(A,B)	0.00	4.95	2.50
C	4.95	0.00	1.41
(D, F), E	**2.50**	**1.41**	0.00

After that, we merge cluster ((D, F), E) and cluster C into a new cluster name (((D, F), E), C). The updated distance matrix is shown in the figure below.

Agglomerative Clustering Algorithm

Min Distance (Single Linkage)

Dist	(A,B)	(D, F), E),C
(A,B)	0.00	2.50
((D, F), E),C	**2.50**	0.00

Thus, Final clusters are: {A, B} and {C, D, E, F}

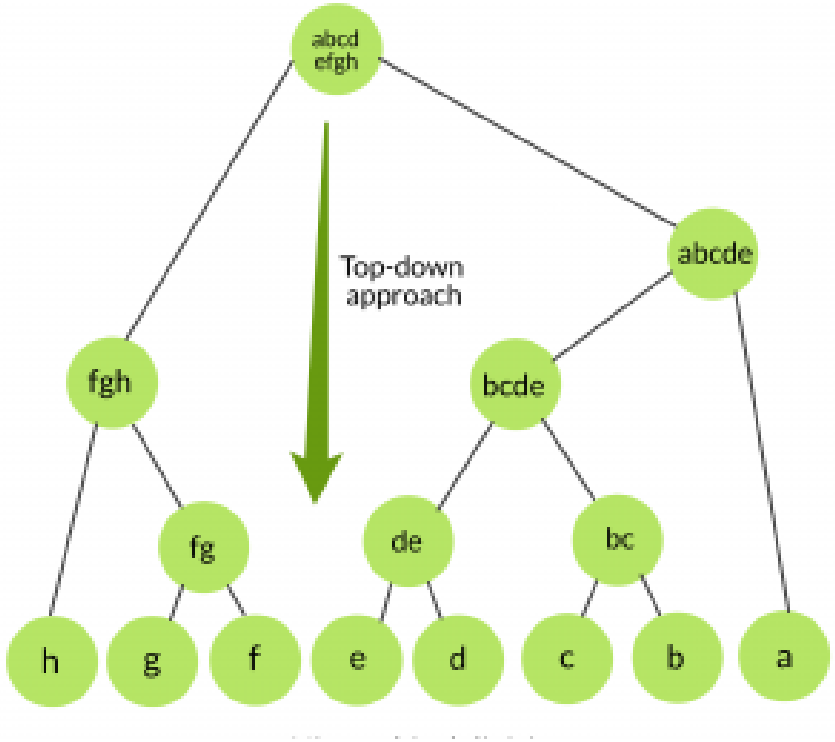
Divisive Clustering Algorithm

- A cluster hierarchy can also be generated top-down. This variant of hierarchical clustering is called *top-down clustering* or *divisive clustering*.
- We start at the top with all data in one cluster. The cluster is split two clusters such that the objects in one subgroup are far from the objects in the other.
- This procedure is applied recursively until required numbers of clusters are formed.
- This method is not considered attractive because there exist $O(2^n)$ ways of splitting each cluster.

Divisive Clustering Algorithm

Algorithm

- Start with all data point in single cluster
- **Repeat**
    - Choice of the cluster to be split
    - Split of this cluster
- **Until** only K clusters are created



DBSCAN Clustering Algorithm

- DBSCAN is density based clustering method. It stands for **Density-based spatial clustering of applications with noise** clustering method.
- DBSCAN algorithm is based in the notion that "clusters are dense regions in the data space, separated by regions of the lower density points".
- The key idea of DBSCAN algorithm is that for each point of a cluster, the neighborhood of a given radius has to contain at least a minimum number of points.

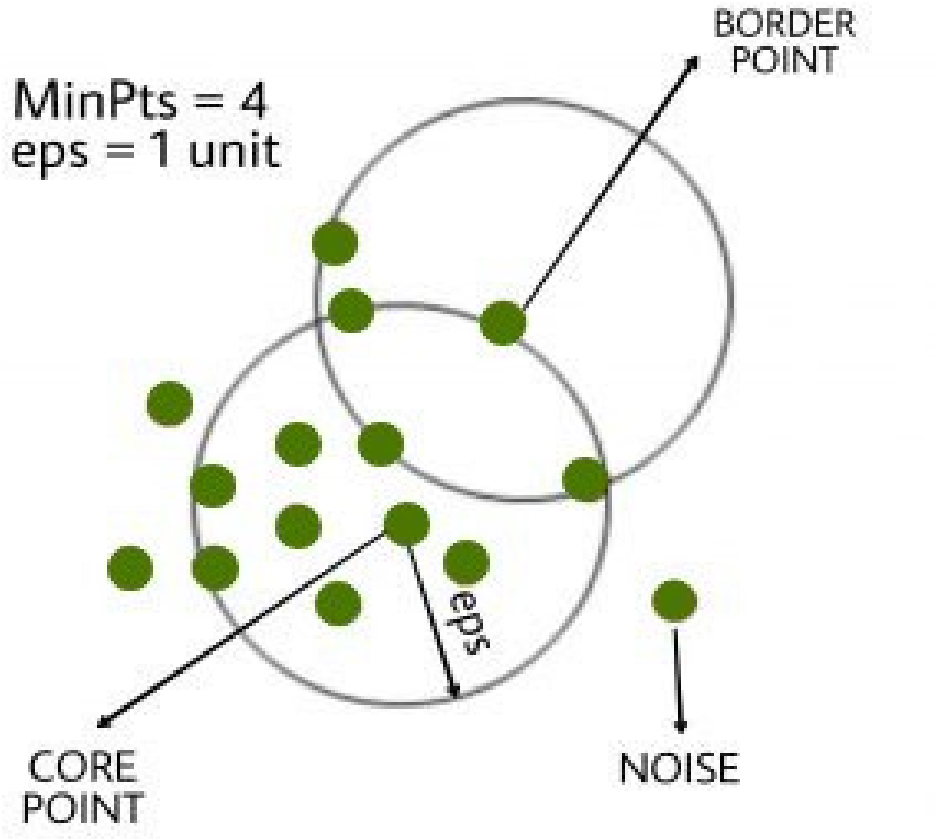
DBSCAN Clustering Algorithm

- Partitioning methods and hierarchical clustering work for finding spherical-shaped clusters or convex clusters. 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. However, real life data may contain irregularities, like:
  - Clusters can be of arbitrary shape.
  - Data may contain noise.

DBSCAN Clustering Algorithm

- The DBSCAN algorithm uses two parameters:
  - **minPts:** The minimum number of points (a threshold) clustered together for a region to be considered dense.
  - **eps (ε):** A distance measure that will be used to locate the points in the neighborhood of any point.
- 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.

DBSCAN Clustering Algorithm



DBSCAN Clustering Algorithm

- The DBSCAN algorithm uses two parameters:
  - **minPts:** The minimum number of points (a threshold) clustered together for a region to be considered dense.
  - **eps (ε):** A distance measure that will be used to locate the points in the neighborhood of any point.
- 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.

DBSCAN Clustering Algorithm

Algorithmic steps for DBSCAN clustering

- The algorithm proceeds by arbitrarily picking up a point in the dataset (until all points have been visited).
- If there are at least minPts points within a radius of ε to the point then we consider all these points to be part of the same cluster.
- The clusters are then expanded by recursively repeating the neighborhood calculation for each neighboring point

DBSCAN Clustering Algorithm

Example: Cluster the following data points {(2,10),(2,5),(8,4),(5,8),(7,5),(6,4),(1,2),(4,9)} using DBSCAN algorithm. Assume eps=2 and MinPts=2

Solution:

Let A=(2,10), B=(2,5), C=(8,4), D=(5,8), E=(7,5), F=(6,4), G=(1,2), H=(4,9)

DBSCAN Clustering Algorithm

Solution: Let A=(2,10), B=(2,5), C=(8,4), D=(5,8), E=(7,5), F=(6,4), G=(1,2), H=(4,9)

Distance Matrix:

	A	B	C	D	E	F	G	H
A	0	5	8.49	3.61	7.07	7.21	8.06	2.24
B	5	0	6.08	4.24	5	4.12	3.16	4.47
C	8.49	6.08	0	5	1.41	2	7.28	6.4
D	3.61	4.24	5	0	3.61	4.12	7.21	1.41
E	7.07	5	1.41	3.61	0	1.41	6.71	5
F	7.21	4.12	2	4.12	1.41	0	5.39	5.39
G	8.06	3.16	7.28	7.21	6.71	5.39	0	7.62
H	2.24	4.47	6.4	1.41	5	5.39	7.62	0

Number of data points within eps=2 from each data points is given below:

A:1(itself) B:1(itself)
C:3(C,E,F) D:2(D,H)
E:3(C,E,F) F:3(C,E,F)
G:1(itself) H:2(D,H)

Thus, Core Points are: C,D,E,F,H

Outliers are: A, B,G

Hence, two clusters are formed:

Cluster1={C,E,F} Cluster2={D,H}

Outlier Analysis

- Outliers are extreme values that deviate from other observations on data.
- In other words, an outlier is an observation that diverges from an overall pattern on a sample.
- Most common causes of outliers are: Data entry errors (human errors), Measurement errors (instrument errors), Experimental errors, Intentional (dummy outliers made to test detection methods), Data processing errors, Sampling errors (extracting or mixing data from wrong or various sources), Natural (not an error, novelties in data).

Outlier Analysis

- Clustering algorithms can be used to detect outliers. Data points that do not belong to any cluster are considered as outliers.
- In some applications such as fraud detection, the rare events can be more interesting than the more regularly occurring one and hence, the outlier analysis becomes important in such case.