

UNSUPERVISED LEARNING
K-MEANS, DBSCAN AND ENUMERATION OF CLUSTERS
A SET OF SLIDES ON A CS460 LECTURE

Mihir Chandra & Ratul Das

School of Computer Sciences
National Institute of Science Education and Research, Bhubaneswar
Homi Bhabha National Institute

March 2, 2023

PART I: INTRODUCTION

- 1 Unsupervised Learning 5**

- 2 Clustering 6**
 - 2.1 What makes clustering unsupervised? 7
 - 2.2 Clustering Algorithms 8

PART II: CLUSTERING ALGORITHMS

1	k-means	10
1.1	Introduction	10
1.2	Algorithm	11
1.3	Inductive bias and hyperparameter	13
1.4	Choosing a good starting point	14
1.5	Advantages and Disadvantages	20
2	Hierarchical Clustering	21
2.1	Introduction	21
2.2	Agglomerative and Divisive clustering	23
2.3	Advantages and disadvantages	26
3	DBSCAN	27
3.1	Introduction	27
3.2	Algorithm	29
3.3	Advantages and Disadvantages	31
4	H-DBSCAN	32
4.1	Introduction	32
4.2	Algorithm	33
4.3	Advantages and Disadvantages	34

PART III: IMPLEMENTING CLUSTERING

1	k-means	36
1.1	Pseudocode	36
2	Hierarchical Clustering	37
2.1	Pseudocode	37
3	DBSCAN	38
3.1	Pseudocode	38
4	H-DBSCAN	40
4.1	Pseudocode	40

Part I

INTRODUCTION

UNSUPERVISED LEARNING

- ▶ In unsupervised learning, algorithms are designed with the intent to **analyse** and **cluster** data that is **neither classified into groups nor labelled**.
- ▶ The goal here is to discover hidden patterns or data that falls in similar classes, without the need for interference on the part of a human.
- ▶ Unsupervised learning algorithms can be broadly grouped into 3 categories:
 - Clustering
 - Anomaly Detection
 - Density estimation
- ▶ In the following slides, we will focus our attention specifically on Clustering.

CLUSTERING

- ▶ Clustering, as a concept, is based around utilising the modularity of a given dataset.
- ▶ We try to increase the intracuster distance while trying to increase the intercluster distance with the choice of metric that we make.
- ▶ If the ground truth for the data is known, then the problem is one of classification. This is what we have seen in supervised learning.
- ▶ But when ground truth is unavailable due to whatever reason, we only have modularity as a tool to identify clusters.

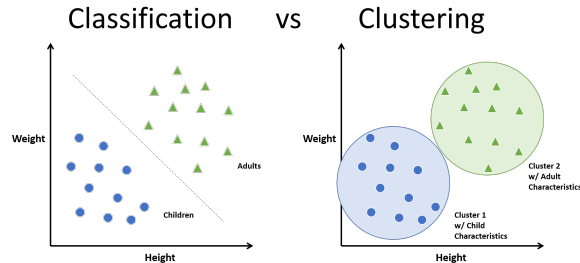


Figure. Difference between classification and clustering [Keerthana 2022]

CLUSTERING

WHAT MAKES CLUSTERING UNSUPERVISED?

Let us take a concrete example to decide where we want to make use of clustering. Say, we have a manufacturing production line where thousands of items are produced every day. If we wanted to come up with a system to tell us if an item is defective, then we would have to train it on a dataset where every single item is labelled as “defective” or “normal”. This would constitute a regular binary classifier. Such an effort will generally require a costly effort to manually label humongous datasets ¹. To top it off, say if the product requires some small change, then the whole labelling and training process would have to be redone. Hence, an algorithm which could just exploit the unlabeled data without needing humans to label every data point turns out to be very useful for such a case. This is exactly where we make use of unsupervised learning.

¹Just a gentle reminder - simply labelling datasets is a whole industry in itself!

CLUSTERING

CLUSTERING ALGORITHMS

Going further we will cover each of the following algorithms:

- ▶ k-means
- ▶ Hierarchical clustering
- ▶ DBSCAN
- ▶ H-DBSCAN

Part II

CLUSTERING ALGORITHMS

K-MEANS

INTRODUCTION

- ▶ **k-means clustering** is an Unsupervised Learning algorithm, which groups unlabeled datasets into different clusters.
- ▶ It iteratively partitions the dataset into k clusters.
- ▶ Each data point belongs to that cluster whose centroid is at a minimum distance (Euclidean distance) to that point.
- ▶ The value of k is predetermined in this algorithm

K-MEANS

ALGORITHM

The k-means algorithm consists of the following steps:

- ▶ Step 1: Pick k random cluster heads. (centroids)
- ▶ Step 2: Calculate the euclidean distance from each point to the cluster head.
- ▶ Step 3: Assign the closest centroid to each data point.
- ▶ Step 4: Calculate the arithmetic mean of all data points in the cluster and assign that as new centroid.
- ▶ Step 5: Repeat steps 2-4 until the cluster doesn't change or shifts by a very small distance

K-MEANS ALGORITHM

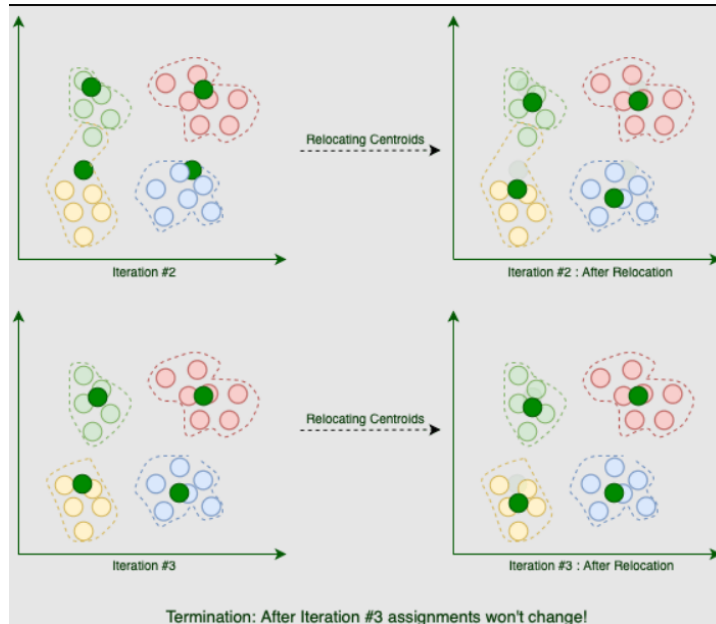


Figure. k-means implementation step-by-step. The centroid is recalculated at every iteration. [Dehghani 2022]

K-MEANS

INDUCTIVE BIAS AND HYPERPARAMETER

Following are the **inductive biases** in this algorithm:

- ▶ Nearby instances should have the same label
- ▶ All features are equally important
- ▶ Complexity is tuned by the k parameter

In this algorithm, **k is the hyperparameter**

- ▶ If $k = 1$ every training example has its own neighborhood and if $k = N$ the entire feature space is one neighborhood
- ▶ Higher k yields smoother decision boundaries

K-MEANS

CHOOSING A GOOD STARTING POINT

The initial clustering depends on the starting point. The same algorithm can lead to different clustering based on the starting point. There are two methods to choose the starting point.

- ▶ Silhouette Score
- ▶ Elbow method

K-MEANS

CHOOSING A GOOD STARTING POINT

Silhouette score is a measure of how similar an object is to its own cluster compared to other clusters. It ranges from -1 to $+1$

- ▶ $+1$ indicates sample is far away from its neighboring cluster.
- ▶ 0 indicates that the sample is on or very close to the decision boundary separating two neighboring clusters.
- ▶ -1 indicates that the samples have been assigned to the wrong clusters.

Silhouette Score is calculated using:

$$\text{Silhouette score} = \frac{(b - a)}{\max(a, b)}$$

where,

a = average intracluster distance i.e., the average distance between each point within a cluster.

b = average intercluster distance i.e., the average distance between all clusters.

K-MEANS

CHOOSING A GOOD STARTING POINT

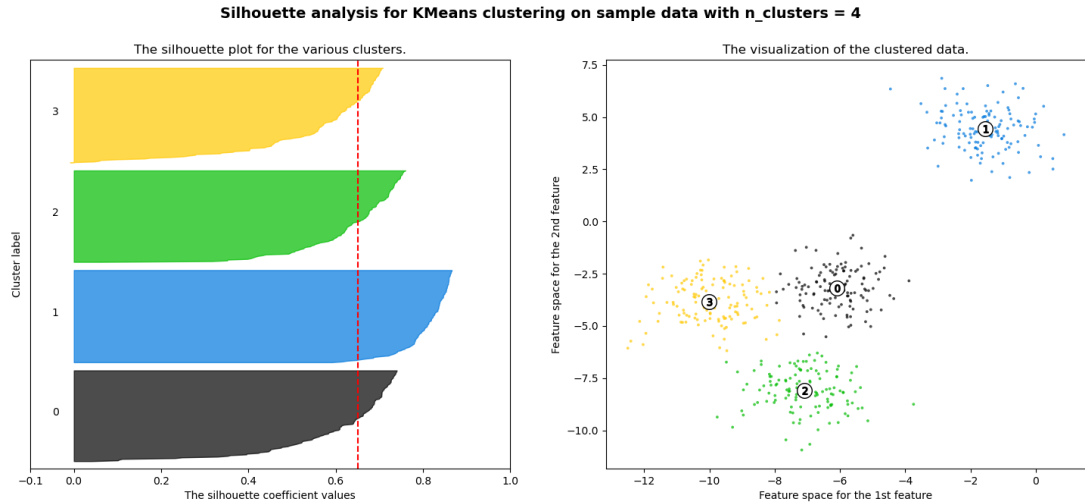


Figure. Visualizing silhouette score of 4 clusters with each iteration [developers 2023]

K-MEANS

CHOOSING A GOOD STARTING POINT

In **elbow method** we vary the number of clusters (k) from 1-10 and for each k , we calculate WCSS (Within-Cluster Sum of Square). As the full-form suggests, WCSS is the sum of squared distance between each point and the centroid in a cluster.

The plot of k vs WCSS represents an elbow. As the number of clusters increases, the WCSS value will start to decrease. WCSS value is largest when $K = 1$. The graph rapidly changes at a point and thus creating an elbow shape. From this point, the graph starts to move almost parallel to the X-axis. The K value corresponding to this point is the optimal K value or an optimal number of clusters. For Elbow method we define two quantities:

- ▶ Distortion: Average of the euclidean squared distance from the centroid of the respective clusters.
- ▶ Inertia: Sum of squared distances of samples to their closest cluster centre.

K-MEANS

CHOOSING A GOOD STARTING POINT

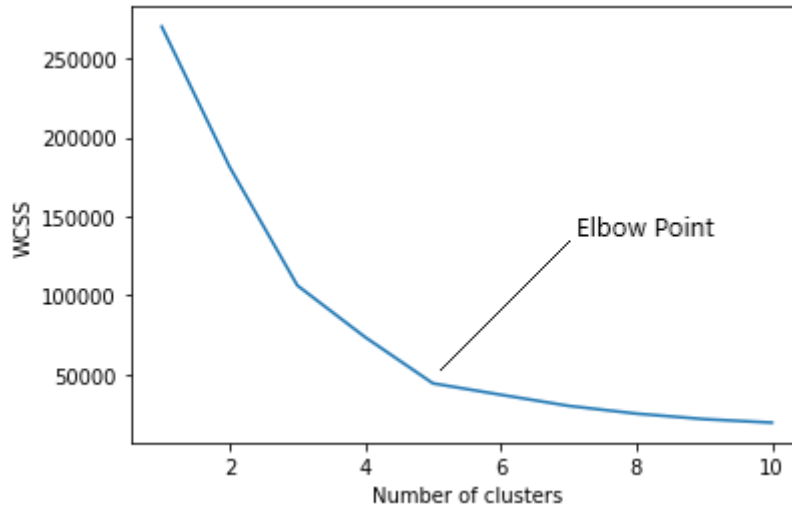


Figure. Plot of WCSS vs Number of clusters, showing the elbow point [Saji 2022]

K-MEANS

CHOOSING A GOOD STARTING POINT

Let's compare silhouette score against the elbow method. We have the following key differences between the two.

- ▶ Elbow only calculates the euclidean distance whereas silhouette takes into account variables such as variance, skewness, high-low differences, etc.
- ▶ Elbow method is computationally easier.
- ▶ Elbow method is generally better for smaller dataset to save time and computational power.
- ▶ Silhouette score is better when we'd want to evaluate our cluster on multiple different criteria to get the optimal number of clusters.

K-MEANS

ADVANTAGES AND DISADVANTAGES

Advantages

- ▶ Easy to implement.
- ▶ Relatively fast on a large dataset.
- ▶ An instance can change cluster (move to another cluster) when the centroids are recomputed.

Disadvantages

- ▶ Difficult to predict the number of clusters. (K-Value)
- ▶ Assumes spherical shapes of clusters (with radius equal to the distance between the centroid and the furthest data point) and doesn't work well when clusters are in different shapes.
- ▶ The initial choice, and the order of data has impact on the clustering.

HIERARCHIAL CLUSTERING

INTRODUCTION

It works via grouping data into a tree of clusters. It treats every data point as a separate cluster. In Hierarchical Clustering, the aim is to produce a hierarchical series of nested clusters. A diagram called Dendrogram (a tree-like diagram that statistics the sequences of merges or splits) graphically represents this hierarchy and is an inverted tree that records the sequences of merges or splits. It is done in two ways:

- ▶ Agglomerative clustering
- ▶ Divisive clustering

HIERARCHIAL CLUSTERING

INTRODUCTION

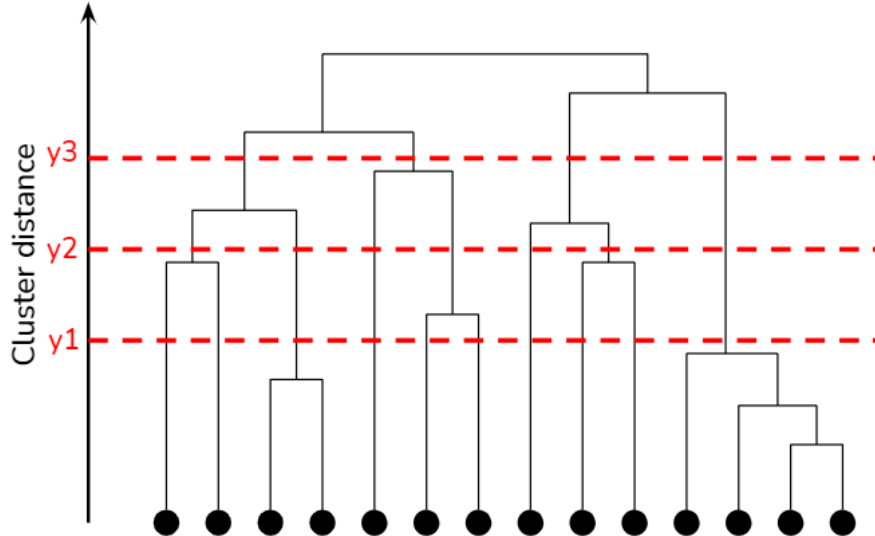


Figure. Dendrogram [Pai 2021]

HIERARCHIAL CLUSTERING

AGGLOMERATIVE AND DIVISIVE CLUSTERING

In the **Agglomerative**, or the "**bottom-up**" **approach**, each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy. **Algorithm:**

1. Create each data point as a single cluster. Consider N data points, so the number of clusters will also be N .
2. Take two closest data points or clusters and merge them to form one cluster. So, there will now be $N-1$ clusters.
3. Again, take the two closest clusters and merge them together to form one cluster. There will be $N-2$ clusters.
4. Repeat Step 3 until only one cluster left.

HIERARCHIAL CLUSTERING

AGGLOMERATIVE AND DIVISIVE CLUSTERING

In the **Divisive**, or the "**top-down**" **approach**, all observations start in one cluster, and splits are performed recursively as one moves down the hierarchy. It is opposite of Agglomerative approach. We assign all of the observations to a single cluster and then partition the cluster to two least similar clusters. Finally, we proceed recursively on each cluster until there is one cluster for each observation.

HIERARCHIAL CLUSTERING

AGGLOMERATIVE AND DIVISIVE CLUSTERING

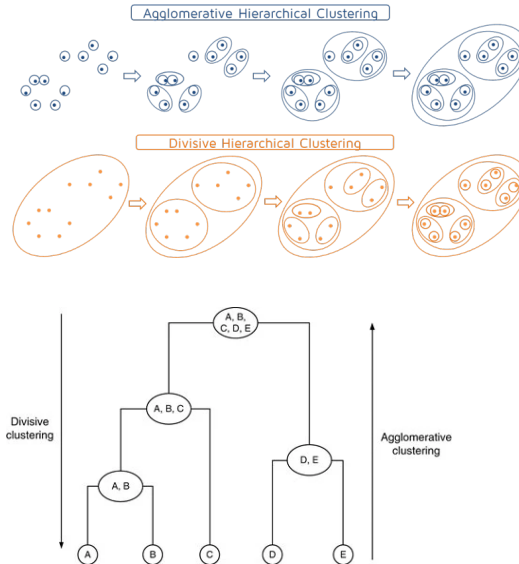


Figure. Comparison of agglomerative and divisive clustering

HIERARCHIAL CLUSTERING

ADVANTAGES AND DISADVANTAGES

Advantages

- ▶ Dendrograms help us in clear visualization, which is practical and easy to understand.
- ▶ We don't have to pre-specify any particular number of clusters and we can obtain any desired number of clusters by cutting the Dendrogram at the proper level

Disadvantages

- ▶ Not optimal on a large dataset due to higher time complexity
- ▶ In hierarchical Clustering, once a decision is made to combine two clusters, it can not be undone.
- ▶ The initial choice, and the order of data has impact on the clustering

DBSCAN

INTRODUCTION

DBSCAN stands for Density Based Spatial Clustering of Applications with Noise. It is based on the idea that a cluster in data space is a continuous region of high point density, separated from other such clusters by continuous regions of low point density.

DBSCAN algorithm uses two parameters:

- ▶ **min points (d):** The minimum number of points (a threshold) clustered together for a region to be considered dense.
- ▶ **eps (ϵ):** specifies how close points should be to each other to be considered a part of a cluster. If the distance between two points is lower or equal to this value, these points are considered neighbors.

DBSCAN works by asking if within ϵ distance whether there are d points.

DBSCAN

INTRODUCTION

We have 3 types of data points in DBSCAN:

1. **Core Point:** If it has at least `minPoints` number of points in its ϵ neighborhood
2. **Border Point:** A point which has fewer than `minPoints` within ϵ but it is in the neighborhood of a core point.
3. **Noise or outlier:** A point which is not a core point or border point. It doesn't have any neighbors.

DBSCAN

ALGORITHM

This is how the DBSCAN algorithm works:

1. Find all the neighbor points within ϵ and identify the core points or visited with more than minPoints 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 exist a point c which has a sufficient number of points in its neighbors and both the points a and b are within the ϵ distance.
4. Iterate through the remaining unvisited points in the dataset. Those points that do not belong to any cluster are noise.

DBSCAN

ALGORITHM

Below are a few examples of a DBSCAN algorithm running for various distributions of data points taken from Naftali Harris' blog

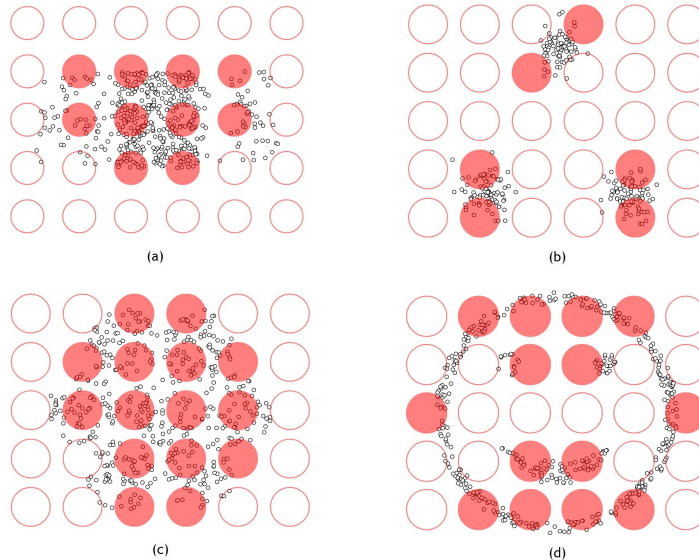


Figure. DBSCAN clustering points in Gaussian mixture, Packed circles, Density bars, and Smiley distributions

DBSCAN

ADVANTAGES AND DISADVANTAGES

Advantages

1. No need to specify the number of clusters.
2. It can find arbitrarily-shaped clusters. It can even find a cluster completely surrounded by (but not connected to) a different cluster.
3. More robust to outliers by classifying them as noise

Disadvantages

1. DBSCAN cannot cluster data sets well with large differences in densities, since the $\min(P) \text{--} \epsilon$ combination cannot then be chosen appropriately for all clusters.
2. It can be confused when there is a border point that belongs to two clusters, depending on the order the data are processed.
3. The algorithm is dependent on the distance metric

H-DBSCAN

INTRODUCTION

H-DBSCAN simply stands for Hierarchical-DBSCAN. It takes DBSCAN and extends it into a hierarchical clustering algorithm. It is considered to be better than DBSCAN at extracting flat clustering based on density of clusters in a dataset.

- ▶ Unlike vanilla DBSCAN, H-DBSCAN uses only one parameter - a minimum cluster size.
- ▶ It extends DBSCAN by converting it into a hierarchical clustering algorithm, and then using a technique to extract a flat clustering based in the stability of clusters.
- ▶ In H-DBSCAN, we only distinguish between core points and noise points.

H-DBSCAN

ALGORITHM

1. Step 1: Transform the space according to density/sparsity
2. Step 2: For the distance-weighted graph, build a minimum spanning tree
3. Step 3: Build the cluster hierarchy for connected components
4. Step 4: Condense the cluster tree on the basis of minimum cluster size
5. Step 5: Extract the clusters that are stable from the condensed tree

H-DBSCAN

ADVANTAGES AND DISADVANTAGES

Advantages

1. No need to specify the number of clusters.
2. It can find arbitrarily-shaped clusters. It can even find a cluster completely surrounded by (but not connected to) a different cluster.
3. More robust to outliers by classifying them as noise

Disadvantages

1. As seen for DBSCAN, H-DBSCAN can still be confused when there is a border point that belongs to two clusters.
2. This algorithm is dependent on the distance metric too.

Part III

IMPLEMENTING CLUSTERING

K-MEANS

PSEUDOCODE

We have the pseudocode for the k-means algorithm [Unzueta 2022] as follows:

Initialise Cluster Centers

while convergence is not reached do

 for each iteration l do

 compute r_{nk} :

(r_{nk} is used to denote if points belongs to cluster)

 for each data point x_n do

 Assign each data point to a cluster:

 for each cluster k do

 if $k == \operatorname{argmin} \|x_n - \mu_k^{l-1}\|$ then

(calculating Euclidean distance)

$r_{nk} = 1$

(point belongs to cluster)

 else

$r_{nk} = 0$

(point doesn't belong to cluster)

 end if

 end for

 end for

 for each cluster k do

(recalculating mean of each cluster)

 update cluster centers as the mean of each cluster:

$$\mu_k^l = \frac{\sum r_{nk} x_n}{\sum r_{nk}}$$

 end for

 end for

HIERARCHICAL CLUSTERING

PSEUDOCODE

We have the pseudocode for the Hierarchical Clustering algorithm [Unknown 2019] as follows:

```
for  $n \leftarrow 1$  to  $N$ 
do for  $i \leftarrow 1$  to  $N$ 
  do  $C[n][i] \leftarrow \text{SIM}(d_n, d_i)$ 
 $I[n] \leftarrow 1$       (keeps track of active clusters)
 $A \leftarrow []$       (assembles clustering as a sequence of merges)
for  $k \leftarrow 1$  to  $N - 1$ 
do  $\langle i, m \rangle \leftarrow \max(C[i][m])$ 
   $A \text{ APPEND } (\langle i, m \rangle)$       ( store merge )
  for  $j \leftarrow 1$  to  $N$ 
  do  $C[i][j] \leftarrow \text{SIM}(i, m, j)$ 
   $C[j][i] \leftarrow \text{SIM}(i, m, j)$ 
   $I[m] \leftarrow 0$       (deactivate cluster)
return  $A$ 
```

DBSCAN

PSEUDOCODE

We have the pseudocode for DBSCAN [Gao 2012] as follows:

DBSCAN(D , eps , MinPts)

$C = 0$

for each unvisited point P in dataset D

mark P as visited

$\text{NeighborPts} = \text{region Query}(P, \text{eps})$

if $\text{sizeof}(\text{NeighborPts}) < \text{MinPts}$

mark P as NOISE

else

$C = \text{next cluster}$

expandCluster(P , NeighborPts , C , eps , MinPts)

expandCluster(P , NeighborPts , C , eps , MinPts)

add P to cluster C

for each point P' in NeighborPts

if P' is not visited

mark P' as visited

DBSCAN

PSEUDOCODE

```
NeighborPts' = regionQuery( P', eps)
if sizeof(NeighborPts') >= MinPts
    NeighborPts = NeighborPts joined with NeighborPts'
if P' is not yet member of any cluster
    add P' to cluster C
regionQuery(P, eps)
    return all points within P' s eps-neighborhood (including P )
```


H-DBSCAN

PSEUDOCODE

We have the pseudocode for H-DBSCAN [McInnes, Healy, and Astels 2016] as follows:

```
function H-DBSCAN(points, eps, minPts, clusterthreshold) :  
  clusters = empty list  
  noise = empty list  
  visited = empty set  
  core_samples = empty set  
  for each point in points:  
    if point in visited:  
      continue  
  visited.add(point)  
  neighbors = get_neighbors(point, points, eps)  
  if len(neighbors) < minPts:  
    noise.append(point)  
  else:  
    c = create_cluster()  
    clusters.append(c)  
    core_samples.add(point)  
    expand_cluster(point, c, neighbors, points, visited, eps, minPts, core_samples)
```

H-DBSCAN

PSEUDOCODE

merge clusters with small sizes

while True:

 sizes = [len(c.points) for c in clusters]

 if min(sizes) >= cluster_threshold:

 break

 smallest_cluster_index = sizes.index(min(sizes))

 smallest_cluster = clusters.pop(smallest_cluster_index)

 merge_with_nearest(smallest_cluster, clusters, eps)

return clusters, noise

function expand_cluster(point, cluster, neighbors, points, visited, eps, minPts, core_samples):

 cluster.add_point(point)

 for neighbor in neighbors:

 if neighbor not in visited:

 visited.add(neighbor)

 new_neighbors = get_neighbors(neighbor, points, eps)

 if len(new_neighbors) >= minPts:

 core_samples.add(neighbor)

 neighbors = merge(neighbors, new_neighbors)

 if neighbor not in core_samples:

 continue

H-DBSCAN

PSEUDOCODE







```
for c in clusters:
    if c.has_point(neighbor):
        continue
    if neighbor.distance_to(c.centroid) < eps:
        c.add_point(neighbor)
        if c not in cluster.neighbors:
            cluster.neighbors.append(c)
        if cluster not in c.neighbors:
            c.neighbors.append(cluster)
function merge_with_nearest(cluster, clusters, eps):
    closest_cluster_distance = float('inf')
    closest_cluster_index = -1
    for i in range(len(clusters)):
        if cluster.distance_to(clusters[i].centroid) < closest_cluster_distance:
            closest_cluster_distance = cluster.distance_to(clusters[i].centroid)
            closest_cluster_index = i
    closest_cluster = clusters[closest_cluster_index]
    closest_cluster.merge(cluster)
    clusters.pop(clusters.index(cluster))
```

H-DBSCAN




PSEUDOCODE

```
function get_neighbors(point, points, eps):  
    neighbors = []  
    for p in points:  
        if p.distance_to(point) < eps:  
            neighbors.append(p)  
    return neighbors
```

REFERENCES I

-  **Dehghani, Ali (Nov. 2022).** *The K-means clustering algorithm in Java.* URL: <https://www.baeldung.com/java-k-means-clustering-algorithm>.
-  **developers, scikit-learn (2023).** *Selecting the number of clusters with silhouette analysis on KMeans clustering - scikit-learn 1.2.1 documentation.* URL: https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html (visited on 02/19/2023).
-  **Gao, Jing (2012).** “Clustering - Lecture 4: Density-based Methods by Jing Gao at SUNY Buffalo”. In: URL: https://cse.buffalo.edu/~jing/cse601/fa13/materials/clustering_density.pdf.
-  **Keerthana, V (2022).** *Spectral Clustering: What, why and how of Spectral Clustering!* URL: <https://www.analyticsvidhya.com/blog/2021/05/what-why-and-how-of-spectral-clustering/>.
-  **McInnes, Leland, John Healy, and Steve Astels (2016).** *How HDBSCAN Works — hdbscan 0.8.1 documentation.* URL: https://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html (visited on 03/01/2023).
-  **Pai, Prasad (May 2021).** *Hierarchical clustering explained.* URL: <https://towardsdatascience.com/hierarchical-clustering-explained-e59b13846da8>.

REFERENCES II

-  **Saji, Basil (Dec. 2022).** *K means clustering: K means clustering algorithm in machine learning.* URL: <https://www.analyticsvidhya.com/blog/2021/01/in-depth-intuition-of-k-means-clustering-algorithm-in-machine-learning/>.
-  **Unknown (2019).** *Hierarchical agglomerative clustering: How to update distance matrix?* URL: <https://stackoverflow.com/questions/58068794/hierarchical-agglomerative-clustering-how-to-update-distance-matrix>.
-  **Unzueta, Diego (Apr. 2022).** *Unsupervised learning: K-means clustering.* URL: <https://towardsdatascience.com/unsupervised-learning-k-means-clustering-6fd72393573c>.