Lecture 06

Density Based Clustering

It's a clustering method where **clusters are formed as areas of high data density**, separated by areas of low density.

<u>Formal Definition</u>: Given a fixed radius ϵ around a point, if there are at least $\boldsymbol{min\_pts}$ number of points in that $\boldsymbol{area}$, then this area is dense.

- **Core point**: if its ϵ - neighbourhood contains at least min_pts.
- **Border point**: if it is in the ϵ - neighbourhood of a core point.
- **Noise point:** if it is neither a core nor border point.

DBScan Algorithm

<u>Parameters</u>: ϵ and $\boldsymbol{min\_pts}$

<u>Steps:</u>

1. Find the ϵ - neighbourhood of each point.
2. Label the point as **core** if it contains at least $\boldsymbol{min\_pts}$**.**
3. For each **core** point, assign to the same cluster all **core** points in its neighbourhood (crux of the algorithm).
4. Label points in its neighbourhood that are not **core** as **border**.
5. Label points as **noise** if they are neither **core** nor **border**.
6. Assign border points to nearby clusters.

<u>Advantages:</u>

1. Finds arbitrary-shaped clusters (e.g., spiral, curved).
2. Automatically detects outliers.
3. No need to specify number of clusters in advance.
4. Resistant to noise.

<u>Limitations:</u>

1. Sensitive to choice of ϵ and $\boldsymbol{min\_pts}$.
2. Doesn't work well with clusters of varying densities.
3. Notion of density is problematic in high-dimensional spaces

Hierarchical Clustering

It builds a hierarchy of clusters that can be visualized as a tree-like diagram called a dendrogram.

Two Types:

1. Agglomerative (bottom-up):
   a. *Start:* Each data point is its own cluster.
   b. *Iteratively:* Merge the two most similar clusters.
   c. *End:* Continue until all points are in one big cluster.
   d. Most common in practice.
2. Divisive (top-down):
   a. *Start:* All data points are in one cluster.
   b. *Iteratively:* Split clusters into smaller ones.
   c. *End:* Continue until each data point is in its own cluster.
   d. Less common but useful in some cases.

Hierarchical Clustering- Distance Functions

Two types of Distance Functions:

1. Point-to-point distance: measures similarity between individual data points.
2. Cluster-to-cluster distance (linkage): defines how to measure distance between groups of points once clusters start forming.

Point-to-point distance:

These are the basic ways of measuring distance between two data points in feature space.

1. Euclidean distance (L2 norm): (Works well with spherical clusters.)

$$d(x,y) = \sqrt{\sum_i (x_i - y_i)^2}$$

2. Manhattan distance (L1 norm): (Good for grid-like data (e.g., city block distances).)

$$d(x,y) = \sum_i |x_i - y_i|$$

3. Minkowski distance (generalization of Euclidean & Manhattan):

$$d(x,y) = \left(\sqrt{\sum_i (x_i - y_i)^2}\right)^{\frac{1}{p}}$$

4. Cosine distance: (Useful for high-dimensional sparse data (e.g., text, documents).)

$$d(x,y) = 1 - \frac{x \cdot y}{\|x\|\|y\|}$$

5. Correlation distance: (Focuses on relationships rather than magnitudes.)

$$d(x,y) = 1 - corr(x,y)$$

Cluster-to-Cluster Distance

Once clusters start forming, we need a rule for measuring distance between clusters. This is where linkage functions come in.

1. Single linkage (minimum distance)

$$D(A, B) = \min d(x, y) \mid x \in A, y \in B$$

    1. Tends to form chain-like clusters.
    2. Sensitive to noise and outliers.
2. Complete linkage (maximum distance)

$$D(A, B) = \max d(x, y) \mid x \in A, y \in B$$

    1. Produces compact, spherical clusters.
    2. Sensitive to outliers.
3. Average linkage (UPGMA)

$$D(A, B) = \frac{1}{\|A\|\|B\|} \sum_{x \in A, y \in B} d(x, y)$$

    1. Balances chaining and compactness.
    2. Works well in practice.
4. Centroid linkage

$$D(A, B) = d(\mu_A, \mu_B)$$

    1. Can sometimes cause **inversions** (dendrogram not monotonic).
5. Ward's Distance
    1. Not based on direct distances, but on variance minimization:

    $$\Delta(A, B) = \text{ } increase\ in\ within\ cluster\ variance\ when\ merging\ A\ and\ B$$

    2. Tends to create clusters of similar size and compact shape.

🔸 Example Setup

Suppose we have two clusters:

- **Cluster A**: points $(1, 1), (2, 1)$
- **Cluster B**: points $(4, 3), (5, 4)$

We'll compute **Euclidean distance** between them using different **linkage methods**.

1. Single Linkage (minimum distance)

Take the **closest pair** of points between A and B.

Distances:

- d((1,1), (4,3)) = √((4−1)² + (3−1)²) = √(9+4) = √13 ≈ 3.61
- d((1,1), (5,4)) = √((5−1)² + (4−1)²) = √(16+9) = √25 = 5
- d((2,1), (4,3)) = √((4−2)² + (3−1)²) = √(4+4) = √8 ≈ 2.83 ✅
- d((2,1), (5,4)) = √((5−2)² + (4−1)²) = √(9+9) = √18 ≈ 4.24

Single linkage = min distance = 2.83

2. Complete Linkage (maximum distance)

Take the **farthest pair** of points between A and B.

From above, max = **5.0**

Complete linkage = 5.0

3. Average Linkage

Take the **average of all pairwise distances** between clusters.

$$\text{Average} = \frac{3.61 + 5 + 2.83 + 4.24}{4} = \frac{15.68}{4} \approx 3.92$$

Average linkage = 3.92

4. Centroid Linkage

Use the centroids (mean points) of each cluster.

- Centroid(A) = ((1+2)/2, (1+1)/2) = (1.5, 1)
- Centroid(B) = ((4+5)/2, (3+4)/2) = (4.5, 3.5)

Distance = $\sqrt{(4.5-1.5)^2 + (3.5-1)^2} = \sqrt{9 + 6.25} = \sqrt{15.25} \approx 3.91$

Centroid linkage = 3.91

5. Ward's Method (Variance Increase)

Ward doesn't use direct distance but looks at **increase in within-cluster variance** when merging.

For simplicity, approximate with **squared distance between centroids × (n_A × n_B)/(n_A+n_B)**.

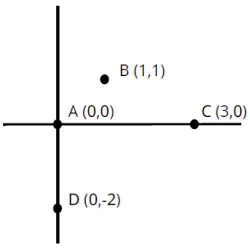
$$\Delta = \frac{n_A \cdot n_B}{n_A + n_B} \cdot d(\mu_A, \mu_B)^2$$

- n_A = 2, n_B = 2
- Centroid distance^2 = 15.25

$$\Delta = \frac{2 \cdot 2}{2+2} \cdot 15.25 = 1 \cdot 15.25 = 15.25$$

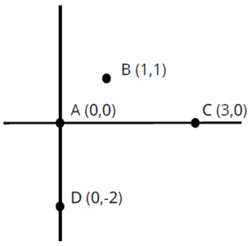
Ward's method = 15.25 (variance increase measure)

Example (Euclidian Distance)



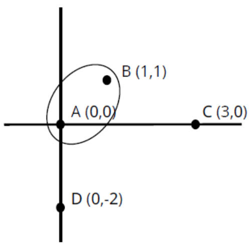
Distance Matrix

	A	B	C	D
A				
B				
C				
D				



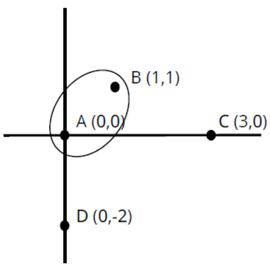
Distance Matrix

	A	B	C	D
A	0	√2	3	2
B	√2	0	√5	√10
C	3	√5	0	√13
D	2	√10	√13	0

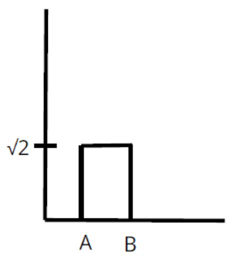


Distance Matrix

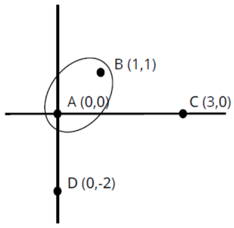
	A	B	C	D
A	0	√2	3	2
B	√2	0	√5	√10
C	3	√5	0	√13
D	2	√10	√13	0



Dendrogram

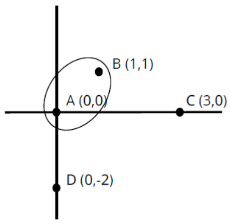


Distance Matrix



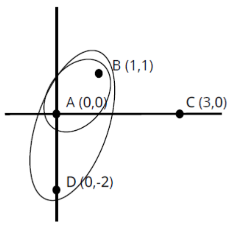
	A & B	C	D
A & B	0		
C		0	√13
D		√13	0

Distance Matrix



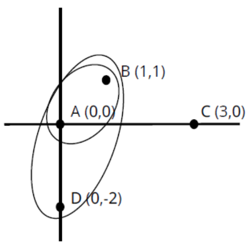
	A & B	C	D
A & B	0	√5	2
C	√5	0	√13
D	2	√13	0

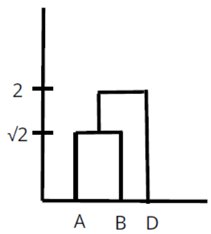
Distance Matrix



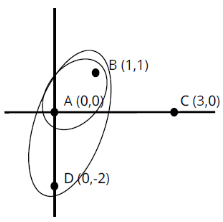
	A & B	C	D
A & B	0	√5	2
C	√5	0	√13
D	2	√13	0

Dendrogram

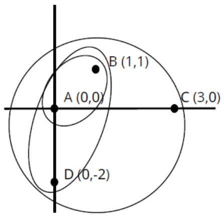




Distance Matrix

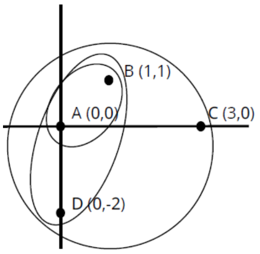


	A & B & D	C
A & B & D	0	
C		0

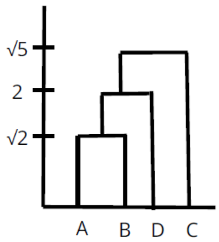


Distance Matrix

	A & B & D	C
A & B & D	0	√5
C	√5	0



Dendrogram



<u>Example 2:</u>

Step 1: Points Coordinates

- A = (0,0)
- B = (0,2)
- C = (2,0)
- D = (5,5)
- E = (6,5)

Step 2: Calculate the Distance Matrix

Euclidean distance between two points (x_1, y_1) and (x_2, y_2) is:

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

Pair	Calculation	Distance
AB	$\sqrt{(0-0)^2 + (2-0)^2} = 2$	2.00
AC	$\sqrt{(2-0)^2 + (0-0)^2} = 2$	2.00
AD	$\sqrt{(5-0)^2 + (5-0)^2} = \sqrt{25+25} = \sqrt{50}$	7.07
AE	$\sqrt{(6-0)^2 + (5-0)^2} = \sqrt{36+25} = \sqrt{61}$	7.81
BC	$\sqrt{(2-0)^2 + (0-2)^2} = \sqrt{4+4} = \sqrt{8}$	2.83
BD	$\sqrt{(5-0)^2 + (5-2)^2} = \sqrt{25+9} = \sqrt{34}$	5.83
BE	$\sqrt{(6-0)^2 + (5-2)^2} = \sqrt{36+9} = \sqrt{45}$	6.71
CD	$\sqrt{(5-2)^2 + (5-0)^2} = \sqrt{9+25} = \sqrt{34}$	5.83
CE	$\sqrt{(6-2)^2 + (5-0)^2} = \sqrt{16+25} = \sqrt{41}$	6.40
DE	$\sqrt{(6-5)^2 + (5-5)^2} = \sqrt{1+0} = 1$	1.00

Distance Matrix (symmetric):

	A	B	C	D	E
A	0	2.0	2.0	7.07	7.81
B	2.0	0	2.83	5.83	6.71
C	2.0	2.83	0	5.83	6.40
D	7.07	5.83	5.83	0	1.00
E	7.81	6.71	6.40	1.00	0

Step 3: Hierarchical Clustering Using Complete Linkage

Complete linkage: Distance between two clusters is the maximum distance between any point in cluster 1 and any point in cluster 2.

Initial Clusters: {A}, {B}, {C}, {D}, {E}

Iteration 1: Find two clusters with minimum distance

- Minimum pairwise distance: 1.00 (between D and E)
- Merge clusters: {D, E}

Clusters now: {A}, {B}, {C}, {D, E}

Iteration 2: Update distance matrix involving cluster {D, E}
Calculate max distance between {D,E} and other clusters:

- Distance({D,E}, A) = max(d(D,A), d(E,A)) = max(7.07, 7.81) = 7.81
- Distance({D,E}, B) = max(d(D,B), d(E,B)) = max(5.83, 6.71) = 6.71
- Distance({D,E}, C) = max(d(D,C), d(E,C)) = max(5.83, 6.40) = 6.40

Current distances:

Pair	Distance
A-B	2.0
A-C	2.0
B-C	2.83
{D,E}-A	7.81
{D,E}-B	6.71
{D,E}-C	6.40

Minimum = 2.0 (A-B or A-C)

Choose {A, B} (or {A, C}); conventionally choose alphabetically first among same distance pairs.

Merge: {A, B}

Clusters now: {A, B}, {C}, {D, E}

Iteration 3: Calculate new distances:

- Distance({A,B}, C) = max(d(A,C), d(B,C)) = max(2.0, 2.83) = 2.83
- Distance({A,B}, {D,E}) = max(d(A,D), d(A,E), d(B,D), d(B,E)) = max(7.07, 7.81, 5.83, 6.71) = 7.81

Distances:

Pair	Distance
{A,B} - C	2.83
{A,B} - {D,E}	7.81
C - {D,E}	6.40

Minimum = 2.83 → Merge {A,B} and C? But 2.83 is minimum, merge {A,B} and C:

Clusters now: {A,B,C}, {D,E}

Iteration 4: Calculate distance:
Distance({A,B,C}, {D,E}) = max of all pairwise distances between points in {A,B,C} and points in {D,E}.

Pairs:

- A-D = 7.07
- A-E = 7.81
- B-D = 5.83
- B-E = 6.71
- C-D = 5.83
- C-E = 6.40

Max = 7.81

Iteration 5:
Only two clusters left: {A,B,C} and {D,E}. Distance = 7.81. Merge into one cluster.

Complete Linkage Dendrogram:

1. Merge D and E (distance=1.00)
2. Merge A and B (distance=2.00)
3. Merge cluster {A,B} with C (distance=2.83)
4. Merge cluster {A,B,C} with cluster {D,E} (distance=7.81)

Step 4: Hierarchical Clustering Using Average Linkage

Average linkage: Distance between two clusters is the average distance between all pairs of points (one from each cluster).

Iteration 1: Again, minimum distance 1.00 (D-E). Merge.

Clusters: {A}, {B}, {C}, {D,E}

Iteration 2: Calculate distances between {D,E} and others:

- Distance({D,E}, A) = average(d(D,A), d(E,A)) = (7.07 + 7.81)/2 = 7.44
- Distance({D,E}, B) = (5.83 + 6.71)/2 = 6.27
- Distance({D,E}, C) = (5.83 + 6.40)/2 = 6.12

Distances left:

Pair	Distance
A-B	2.0
A-C	2.0
B-C	2.83
{D,E}-A	7.44
{D,E}-B	6.27
{D,E}-C	6.12

Minimum = 2.0 (A-B or A-C) → Merge {A,B}

Clusters: {A,B}, {C}, {D,E}

Iteration 3: Calculate distances:

- Distance({A,B}, C) = average of d(A,C)=2.0 and d(B,C)=2.83 → (2.0 + 2.83)/2 = 2.415
- Distance({A,B}, {D,E}) = average of distances between points in {A,B} and {D,E}:

Pairs:

- A-D = 7.07
- A-E = 7.81
- B-D = 5.83
- B-E = 6.71

Average = (7.07 + 7.81 + 5.83 + 6.71) / 4 = (27.42) / 4 = 6.855

- Distance(C, {D,E}) = previously calculated = 6.12

Distances:

Pair	Distance
{A,B} - C	2.415
{A,B} - {D,E}	6.855
C - {D,E}	6.12

Minimum = 2.415 → merge {A,B} with C

Clusters: {A,B,C}, {D,E}

Iteration 4: Calculate new distance:

Distance({A,B,C}, {D,E}) = average of distances between all points in {A,B,C} and points in {D,E}

Pairs:

- A-D =7.07
- A-E =7.81
- B-D =5.83
- B-E =6.71
- C-D =5.83
- C-E =6.40

Average:

(7.07 + 7.81 + 5.83 + 6.71 + 5.83 + 6.40) / 6 = (39.65) / 6 ≈ 6.61

Iteration 5: Merge {A,B,C} and {D,E} (distance=6.61)

Average Linkage Dendrogram:

1. Merge D and E (distance=1.00)
2. Merge A and B (distance=2.00)
3. Merge cluster {A,B} with C (distance=2.415)
4. Merge cluster {A,B,C} with cluster {D,E} (distance=6.61)

Checking Distance matrix

1. Non-Negativity: All values in the matrix are either 0, positive integers, or square roots of positive integers, which satisfies non-negativity.
2. Identity of Indiscernible: $d(x,y) = 0 \ if \ and \ only \ if \ x = y$. The diagonal values (A to A, B to B, etc.) are all 0, indicating that each element has zero distance from itself.
3. Symmetry: The matrix is symmetric, as the value in row A against column B is equal to the value in row B against column A and similarly for all other pairs.
4. Triangle Inequality: $d(x,z) \leq d(x,y) + d(y,z)$
   To confirm if the matrix satisfies the triangle inequality, take three points (example: B, D, E):
   $$d(B,E) = 5$$
   $$d(B,D) = \sqrt{17} \approx 4.12$$
   $$d(D,E) = \sqrt{20} \approx 4.47$$
   For the triangle inequality to hold,
   $$d(B,E) \leq d(B,D) + d(D,E)$$
   $$5 \leq 4.12 + 4.475$$
   This works for this example, but checking all possible combinations across the matrix might lead to violations.