Hierarchical Clustering

09 October 2025 18:25

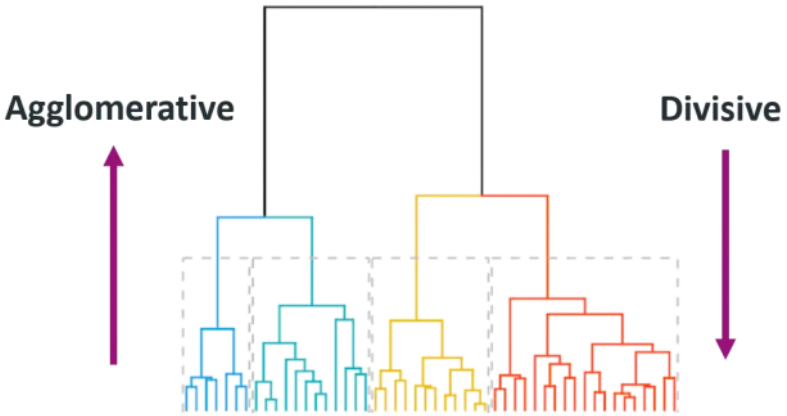
1. Introduction

- **Hierarchical clustering** is an **unsupervised machine learning** technique used to group similar data points into clusters.
- Unlike **partition-based algorithms** (e.g., K-Means, which assumes a predefined number of clusters k), **hierarchical clustering** builds a **multi-level hierarchy** (tree structure) of clusters, often visualized using a **dendrogram**.
- It aims to reveal the nested structure in data, showing how individual samples are grouped at different similarity thresholds.

2. Key Concept: Hierarchy of Clusters

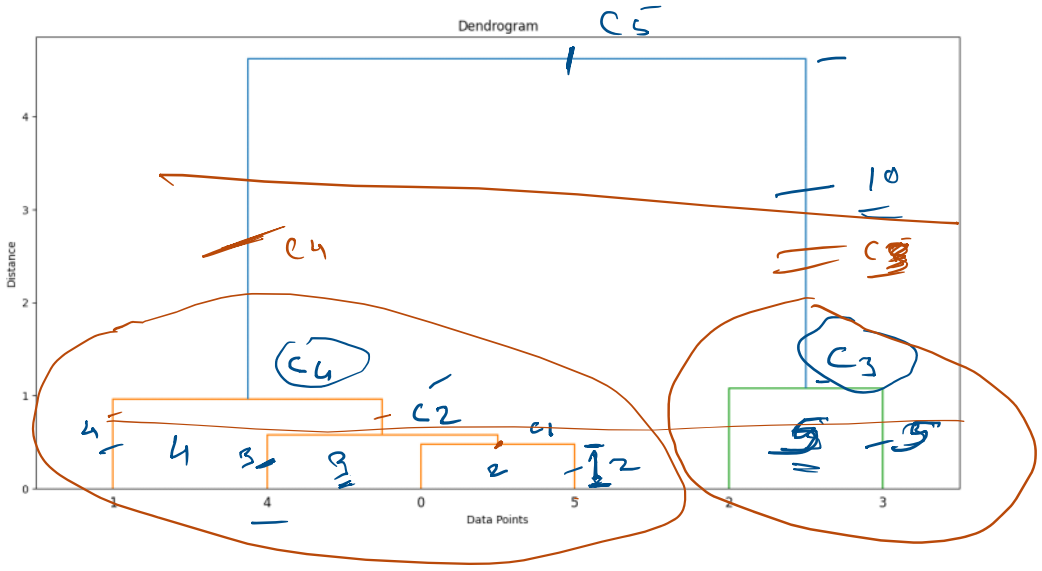
The algorithm successively merges (or splits) clusters based on a **distance (similarity) measure**, leading to a **hierarchical tree**.

Two major types:



Type	Approach	Description
Agglomerative (Bottom-Up)	Start with each data point as its own cluster and iteratively merge the closest clusters.	Most common approach.
Divisive (Top-Down)	Start with all data points in one cluster and recursively split into smaller clusters.	Computationally expensive.

Dendrograms:



- A dendrogram is a tree-like structure that visualizes the process of hierarchical clustering.
- Each level of the tree represents a merge or split operation, and the height of the branches represents the distance (or dissimilarity) at which clusters were joined.

Here's why it's important:

- Allows visual inspection of clusters at different levels.
- By "cutting" the dendrogram at a certain height, you can choose the number of clusters that best fits the data.

3. Mathematical Foundation

Let $X = \{x_1, x_2, \ldots, x_n\}$ be the set of n data points, each $x_i \in \mathbb{R}^d$.

We define a **distance metric** $D(x_i, x_j)$, e.g., Euclidean, Manhattan, or Cosine distance.

For **agglomerative clustering**, we define a **linkage criterion** that determines the distance between clusters.

---

4. Distance (Similarity) Measures

Common pairwise distances between individual points:

1. **Euclidean Distance**

$$D(x_i, x_j) = \sqrt{\sum_{k=1}^{d} (x_{ik} - x_{jk})^2}$$

2. **Manhattan Distance**

$$D(x_i, x_j) = \sum_{k=1}^{d} |x_{ik} - x_{jk}|$$

3. **Cosine Distance**

$$D(x_i, x_j) = 1 - \frac{x_i \cdot x_j}{\|x_i\| \|x_j\|}$$

4. **Mahalanobis Distance**
   Accounts for correlation between features:

$$D(x_i, x_j) = \sqrt{(x_i - x_j)^T S^{-1} (x_i - x_j)}$$

   where S is the covariance matrix.

---

5. Linkage Criteria (Cluster-to-Cluster Distance)

When clusters contain multiple points, we need a rule to define **distance between clusters** A and B.

Let:

- $|A|, |B|$: number of points in each cluster
- $D(x_i, x_j)$: distance between points $x_i \in A$ and $x_j \in B$

Different **linkage methods** define this as:

Linkage	Formula	Intuition	Characteristics
Single Linkage	$D(A,B) = \min_{x_i \in A, x_j \in B} D(x_i, x_j)$	Closest points	Tends to form "chains" (non-spherical clusters)
Complete Linkage	$D(A,B) = \max_{x_i \in A, x_j \in B} D(x_i, x_j)$	Farthest points	Produces compact clusters
Average Linkage (UPGMA)	(D(A,B) = \frac{1}{	A	
Centroid Linkage	$D(A,B) = \|\mu_A - \mu_B\|$	Distance between cluster centroids	May cause "inversions" (non-monotonic merges)
Ward's Linkage	$D(A,B) =$ increase in total within-cluster variance after merging A and B	Variance-based	Prefers spherical clusters, similar to k-means

Ward's method is mathematically defined as:

$$D(A,B) = \frac{|A||B|}{|A|+|B|} \|\mu_A - \mu_B\|^2$$

where μ_A and μ_B are centroids of clusters.

6. Agglomerative Hierarchical Clustering Algorithm

Step-by-Step Procedure

1. **Initialization:**
   Each data point starts as its own cluster (n clusters).
2. **Compute Distance Matrix:**
   Compute pairwise distances between all clusters (initially, all data points).
3. **Find Closest Clusters:**
   Identify the pair of clusters (A, B) with the smallest inter-cluster distance using the chosen linkage criterion.
4. **Merge:**
   Merge clusters A and B into a new cluster C.
5. **Update Distance Matrix:**
   Recompute distances between C and all other clusters.
6. **Repeat:**
   Continue steps 3–5 until all data points are in one cluster.

7. Dendrogram — Visual Representation

A **dendrogram** is a tree diagram that records the sequence of merges.

- The **x-axis**: individual observations or clusters.
- The **y-axis**: distance (or dissimilarity) at which clusters were merged.

Interpreting a Dendrogram:

- The **height of a merge** represents the distance at which clusters combined.
- To **choose number of clusters (k)**: draw a horizontal line that cuts the dendrogram at a given height — the number of intersections equals k.

8. Stopping Criteria

You can stop merging:

- When a desired number of clusters k is reached.
- When the distance between clusters exceeds a threshold dmaxd_{max}dmax.
- Based on metrics such as **inconsistency coefficient** or **cophenetic correlation coefficient**.

9. Divisive (Top-Down) Hierarchical Clustering

Less common, but works as follows:

1. Start with all data in one cluster.

2. Recursively split clusters using a partitioning criterion (e.g., variance or distance).
3. Continue splitting until each data point is in its own cluster.

This is conceptually similar to **decision tree splitting**, but computationally more expensive: O(2^n).

10. Advantages and Limitations

Advantages	Limitations
No need to pre-specify number of clusters k.	Computationally expensive O(n3).
Dendrogram provides interpretability.	Sensitive to noise and outliers.
Works with different distance metrics and linkage types.	Difficult to handle large datasets.
Can find nested structures in data.	Choice of linkage and distance affects results.

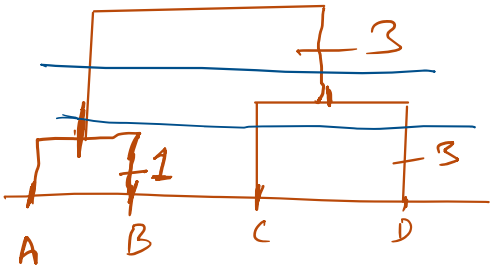
11. Example Use Cases

- **Genomics:** Building phylogenetic trees.
- **Marketing:** Customer segmentation.
- **Document clustering:** Hierarchical topic discovery.
- **Medical imaging:** Grouping similar disease patterns.
- **Social network analysis:** Detecting hierarchical communities.

Dataset (very small & integer)
Points (1-D) labeled A, B, C, D:

- A = 1
- B = 2
- C = 5
- D = 8

Index them: A(1), B(2), C(5), D(8).



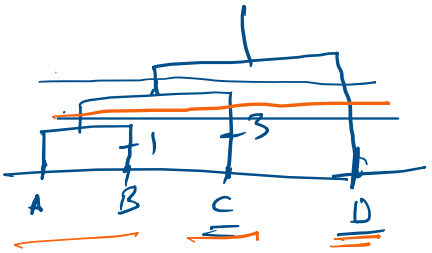
1) Compute the pairwise Euclidean distances
Distance d(X,Y) = |X - Y| because 1-D.

- d(A,B) = |1 − 2| = 1
- d(A,C) = |1 − 5| = 4
- d(A,D) = |1 − 8| = 7
- d(B,C) = |2 − 5| = 3
- d(B,D) = |2 − 8| = 6
- d(C,D) = |5 − 8| = 3

Distance matrix (symmetric):

	A	B	C	D
A	0	1	4	7
B	1	0	3	6
C	4	3	0	3
D	7	6	3	0

Agglomerative process — common initial step
Start with clusters: {A}, {B}, {C}, {D}.
At each iteration we merge the two clusters with the smallest distance (ties resolved as noted).
Smallest distance from matrix is **1 (A,B)**, so first merge is always **A and B** for all linkage methods (because single/complete/avg/Ward all consider pairwise distances or variance and A,B are closest).
Merge 1: {A} + {B} → {A,B} at height (merge distance) = **1**.
Remaining clusters after step 1: {A,B}, {C}, {D}.



We now need distances between the new cluster {A,B} and the others.
We'll compute these for each linkage method. Also keep d(C,D)=3 unchanged.
Distances from cluster {A,B} to C and D (we need them to decide the next merge)
Pairwise original distances to use:

- d(A,C)=4, d(B,C)=3
- d(A,D)=7, d(B,D)=6

Single linkage (minimum distance between any members)

- d_single({A,B}, C) = min(d(A,C), d(B,C)) = min(4, 3) = **3**
- d_single({A,B}, D) = min(d(A,D), d(B,D)) = min(7, 6) = **6**
- d(C,D) = **3**

So single-link distances: {A,B}-C = 3, C-D = 3, {A,B}-D = 6.
There is a tie for smallest distance (3) between {A,B}-C and C-D. A

- d(C,D) = **3**

So single-link distances: {A,B}-C = 3, C-D = 3, {A,B}-D = 6.
There is a tie for smallest distance (3) between {A,B}-C and C-D. A tie can be broken arbitrarily; common choice is to merge the pair that appears first or by index order. I'll merge **C and D** (but note merging {A,B} with C would give a different dendrogram — both are valid under single linkage when ties occur).

For Single linkage:

- **Merge 2:** {C} + {D} → {C,D} at height = **3** (since d(C,D)=3).
- Clusters now: {A,B}, {C,D}.
- Now compute d_single({A,B}, {C,D}) = min(d(A,C), d(A,D), d(B,C), d(B,D)) = min(4,7,3,6) = **3**.
- **Merge 3:** {A,B} + {C,D} → all at height = **3**.

Dendrogram heights (Single linkage): merges at 1, 3, 3.

Complete linkage (maximum distance between any members)

- d_complete({A,B}, C) = max(d(A,C), d(B,C)) = max(4, 3) = **4**
- d_complete({A,B}, D) = max(d(A,D), d(B,D)) = max(7, 6) = **7**
- d(C,D) = **3**

Complete-link distances: {A,B}-C = 4, C-D = 3, {A,B}-D = 7.
Smallest is C-D = 3.

For Complete linkage:

- **Merge 2:** {C} + {D} → {C,D} at height = **3**.
- Clusters now: {A,B}, {C,D}.
- Now compute d_complete({A,B}, {C,D}) = max(d(A,C), d(A,D), d(B,C), d(B,D)) = max(4,7,3,6) = **7**.
- **Merge 3:** {A,B} + {C,D} → all at height = **7**.

Dendrogram heights (Complete linkage): merges at 1, 3, 7.