Lecture Note: Hierarchical Clustering

Luu Minh Sao Khue

1 Introduction

Many clustering methods, such as K-Means, require specifying the number of clusters K *before* running the algorithm. In practice, this requirement can be problematic. We may not know how many groups exist in advance, the data may exhibit meaningful structure at multiple resolutions (for example, large groups that further split into smaller subgroups), or we may want an interpretable view of how clusters are formed.

Hierarchical clustering addresses these issues by constructing a **hierarchy** of clusters rather than a single flat partition. The result is a **tree structure**, called a **dendrogram**, which represents how clusters are merged or split across different scales. By cutting the dendrogram at different heights, we can obtain different clusterings without rerunning the algorithm.

A key idea of hierarchical clustering is that it produces a **sequence of nested clusterings** rather than a single final solution:

$$\text{each point} \;\rightarrow\; \text{small clusters} \;\rightarrow\; \text{larger clusters} \;\rightarrow\; \text{one cluster containing all points.}$$

This property makes hierarchical clustering particularly useful for exploratory data analysis, as it reveals how the dataset is organized at different levels of granularity.

Hierarchical clustering methods are commonly divided into two categories:

- **Agglomerative (bottom-up) clustering**, which starts with each data point as its own cluster and repeatedly merges the closest clusters until a single cluster remains.
- **Divisive (top-down) clustering**, which starts with all points in one cluster and repeatedly splits clusters into smaller ones.

In practice, agglomerative clustering is more widely used because it is easier to implement and is supported by most standard libraries, whereas divisive methods are often more computationally expensive.

2 Setup

Assume we are given a dataset consisting of n data points, each with d features:

- n: number of data points (samples),
- d: number of features,
- $x_i \in \mathbb{R}^d$: the i-th data point, for $i = 1, \ldots, n$.

Hierarchical clustering relies on pairwise comparisons between data points or clusters. To perform these comparisons, two components are required:

- a **distance** function $d(x_i, x_j)$ between data points,
- a rule for defining the **distance between clusters**, known as the **linkage** criterion.

In its classical form, hierarchical clustering makes use of the **distance matrix** $D \in \mathbb{R}^{n\times n}$, defined as:

$$D_{ij} = d(x_i, x_j).$$

The distance matrix has the following properties:

- $D_{ij} \geq 0$ for all i, j,
- $D_{ij} = D_{ji}$ for common distance measures,
- $D_{ii} = 0$ for all i.

Practical implication. Storing the distance matrix requires $O(n^2)$ memory, which can become a major limitation for large datasets. This quadratic memory requirement is one of the main reasons hierarchical clustering does not scale as well as methods such as K-Means.

3 Cluster Distance: Linkage Criteria

When we have clusters (sets of points), we need a definition of distance between clusters. Let A and B be two clusters, where:

$$A = \{x_i : i \in I_A\}, \quad B = \{x_j : j \in I_B\}.$$

Here I_A and I_B are the index sets of points in the clusters.

3.1 Single linkage (minimum distance)

$$d(A, B) = \min_{x\in A,\, y\in B} d(x, y).$$

Meaning: define cluster distance as the smallest distance between any pair of points across the two clusters.

Single linkage can connect clusters through chains of points. This often creates long, thin clusters even if the true groups are separate (**chaining effect**). It can work well when clusters are connected by dense paths, but it is sensitive to noise bridges.

3.2 Complete linkage (maximum distance)

$$d(A, B) = \max_{x\in A,\, y\in B} d(x, y).$$

Meaning: define cluster distance as the largest distance between any pair of points across clusters.

Complete linkage tends to produce compact clusters because it refuses to merge clusters if *any* pair of points would be very far apart. It is more robust to chaining than single linkage but can split large clusters if they have wide spread.

3.3 Average linkage (mean distance)

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

Here $|A|$ is the number of points in cluster A and $|B|$ is the number of points in cluster B.

Average linkage is a compromise between single and complete linkage. It reduces sensitivity to extreme pairs (a single very close or very far pair), and often gives balanced cluster shapes.

4 Agglomerative Clustering Algorithm

Algorithm 1 Agglomerative Hierarchical Clustering

```
Require: Data points $\{x_i\}_{i=1}^n$, distance function $d(\cdot,\cdot)$, linkage rule, optional target number of clusters $K$
Ensure: Dendrogram (hierarchy) and cluster labels after cutting the tree
 1: Initialize clusters:
        $\mathcal{C}^{(0)} = \{\{x_1\},\{x_2\},\dots,\{x_n\}\}$
 2: Set iteration counter $t \leftarrow 0$
 3: while stopping criterion not met do
 4:     Compute distances $d(A,B)$ for all pairs $A \neq B \in \mathcal{C}^{(t)}$ using the linkage rule
 5:     Find the closest pair of clusters:
            $(A^*,B^*) = \arg\min_{A\neq B\in\mathcal{C}^{(t)}} d(A,B)$
 6:     Merge clusters:
            $C_{\text{new}} = A^* \cup B^*$
 7:     Update the cluster set:
            $\mathcal{C}^{(t+1)} = \big(\mathcal{C}^{(t)} \setminus \{A^*,B^*\}\big) \cup \{C_{\text{new}}\}$
 8:     Update distances between $C_{\text{new}}$ and all remaining clusters
 9:     Increment iteration counter $t \leftarrow t+1$
10: end while
11: Construct the dendrogram from the sequence of merges
12: Obtain cluster labels by cutting the dendrogram
```

5 Divisive Clustering Algorithm

Algorithm 2 Divisive Hierarchical Clustering

```
Require: Data points $\{x_i\}_{i=1}^n$, distance function $d(\cdot,\cdot)$, splitting criterion, optional target
    number of clusters $K$
Ensure: Dendrogram (hierarchy) and cluster labels after cutting the tree
 1: Initialize the cluster set with all points:
        $$\mathcal{C}^{(0)} = \big\{\{x_1, x_2, \ldots, x_n\}\big\}$$
 2: Set iteration counter $t \leftarrow 0$
 3: while stopping criterion not met do
 4:     Select a cluster $A \in \mathcal{C}^{(t)}$ to split
 5:     Partition $A$ into two subclusters:
            $$A \longrightarrow A_1 \cup A_2, \quad A_1 \cap A_2 = \varnothing$$
 6:     Update the cluster set:
            $$\mathcal{C}^{(t+1)} = \big(\mathcal{C}^{(t)} \setminus \{A\}\big) \cup \{A_1, A_2\}$$
 7:     Increment iteration counter $t \leftarrow t + 1$
 8: end while
 9: Construct the dendrogram from the sequence of splits
10: Obtain cluster labels by cutting the dendrogram
```

6 Dendrogram

A dendrogram is a tree where:

- leaves are individual data points,
- internal nodes represent merges of clusters,
- the height of a merge corresponds to the distance (or merge cost) at which two clusters were combined.

If we draw a horizontal line across the dendrogram at height h, each connected component below that line corresponds to one cluster. In other words:

- A low cut (small h) produces many small clusters.
- A high cut (large h) produces fewer large clusters.

This provides a flexible way to obtain clusterings at different levels.

7 Complexity, Strengths, and Limitations

Hierarchical clustering relies on pairwise distance computations. This leads to relatively high computational cost. In particular:

- storing the distance matrix requires $O(n^2)$ memory,
- repeated merging and distance updates typically require $O(n^2)$ time or more.

Therefore, hierarchical clustering is most suitable for small or medium-sized datasets, or when interpretability is more important than scalability.

Hierarchical clustering has several practical advantages:

- the number of clusters does not need to be specified in advance,
- the dendrogram provides an interpretable view of cluster structure,
- classical agglomerative clustering is usually deterministic.

It also has important limitations:

- limited scalability due to quadratic memory and time complexity,
- sensitivity to noise and outliers,
- greedy merges that cannot be undone.

8 Summary

When applying hierarchical clustering in practice:

- clean the data and handle missing values or strong outliers,
- scale features so that distances are comparable,
- choose an appropriate distance metric and linkage rule,
- inspect the dendrogram and select a suitable cut or number of clusters.

In summary, hierarchical clustering builds a hierarchy of nested clusters and allows exploration of data structure at multiple resolutions. It is more interpretable than K-Means and does not require choosing K in advance, but it is less scalable due to pairwise distance computations.