Advanced Machine Learning

Lecture 3: Hierarchical Clustering

Nora Ouzir : nora.ouzir@centralesupelec.fr
Lucca Guardiola : lucca.guardiola@centralesupelec.fr

Oct. - Nov. 2020

CentraleSupélec

Content

Today's Lecture

1. Introduction

2. Reminders on Clustering
   1. Types of methods and clusters
   2. Distance and Dissimilarity
   3. Clustering Quality

3. From Partitional to Hierarchical Clustering
   1. K-means
   2. Hierarchical Clustering
   3. DBSCAN
   4. HDBSCAN

Key references

- Tan, P. N., Steinbach, M., Kumar V., *Data mining cluster analysis: basic concepts and algorithms. Introduction to data mining*. 2013.
- Bishop, C. M. *Pattern Recognition and Machine Learning.* Springer, 2006.
- Hastie, T., Tibshirani, R. and Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Second edition. Springer, 2009.
- James, G., Witten, D., Hastie, T. and Tibshirani, R. *An Introduction to Statistical Learning, with Applications in R.* Springer, 2013

Today's Lecture

1. Introduction
2. Reminders on Clustering
   1. Types of methods and clusters
   2. Distance and Dissimilarity
   3. Clustering Quality
3. From Partitional to Hierarchical Clustering
   1. K-means
   2. Hierarchical Clustering
   3. DBSCAN
   4. HDBSCAN

Clustering: An Unsupervised Approach

- Extract homogeneous meaningful or useful categories from the data
- Discover/learn how the data is organized, natural structure
- No ground-truth outputs for training : unsupervised

Objectives

1. Understanding: Biology and medicine, finance, text mining, web, ...
2. Utility: Use cluster characteristics instead of the original data (dimension reduction, regression of high-dimensional data, ...)

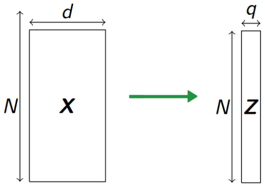
The labels are unknown!

Dimension reduction vs Clustering

Let $\boldsymbol{X} = (x_1, \dots, x_N)$ be a set of N training samples

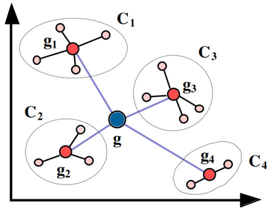
Dimension reduction

- Project $\boldsymbol{X} \in \mathbb{R}^{N,d}$ onto $\boldsymbol{Z} \in \mathbb{R}^{N,q}$ with $q < d$
- Visualize, denoise, reduce computational cost, ...



Clustering

- Groupe similar samples $\boldsymbol{x}_i$ into clusters C_k
- Based on a dissimilarity metric $\mathcal{D}(C_1, C_2)$



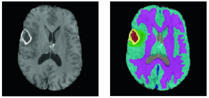
Clustering Applications

Market segmentation

- x: purchase history
- C_k: market segments

Medical image segmentation

- x: image pixels, voxels
- C_k: blood, muscle, tumor, ...



Text mining

- x: text, e-mails, ...
- C_k: folders, themes, ...

Key Questions on Clustering

- Types of clustering ?
- How to characterize a cluster ?
- How to define similarity or dissimilarity between samples ?
- The real/optimal number of clusters ?
- What algorithms can we use and when ?
- How to evaluate a clustering result ? (subjectivity)

Today's Lecture

1. Introduction

2. Reminders on Clustering
    1. Types of methods and clusters
    2. Distance and Dissimilarity
    3. Clustering Quality

3. From Partitional to Hierarchical Clustering
    1. K-means
    2. Hierarchical Clustering
    3. DBSCAN
    4. HDBSCAN

Today's Lecture

1. Introduction

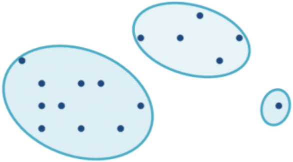
2. Reminders on Clustering
    1. Types of methods and clusters
    2. Distance and Dissimilarity
    3. Clustering Quality

3. From Partitional to Hierarchical Clustering
    1. K-means
    2. Hierarchical Clustering
    3. DBSCAN
    4. HDBSCAN

Types of clustering: Partitional vs Hierarchical

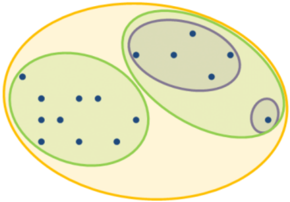
Partitional

- Division into non-overlapping subsets
- Each data point is in exactly one subset



Hierarchical

- Clusters can have sub-clusters
- Set of nested clusters, organized as a tree



Types of Clusters

- Well-separated: Any point in a cluster is closer (or more similar) to every other point in the cluster than to any point not in the cluster.
- Prototype-Based: an object in a cluster is closer (more similar) to the "center" of a cluster, than to the center of any other cluster → Assumptions about shape
  - Center = centroid (average) or medoid (most representative)
- Density-based: dense region of points, which is separated by low-density regions, from other regions of high density. Used when the clusters are irregular or intertwined, and when noise and outliers are present → Is data driven
- Others... graph-based...

Distinctions between sets of clusters

- Exclusive vs non-exclusive (overlapping): separate clusters vs points may belong to more than one cluster
- Fuzzy vs non-fuzzy: each observation $\mathbf{x}_i$ belongs to every cluster $\mathcal{C}_k$ with a given weight $w_k \in [0, 1]$ and $\sum_{k=1}^{K} w_k = 1$ (Similar to probabilistic clustering).
- Partial vs Complete: all data are clustered vs there may be non-clustered data, e.g., outliers, noise, "uninteresting background"...
- Homogeneous vs Heterogeneous: Clusters with $\neq$ size, shape, density...

Today's Lecture

1. Introduction
2. Reminders on Clustering
    1. Types of methods and clusters
    2. Distance and Dissimilarity
    3. Clustering Quality
3. From Partitional to Hierarchical Clustering
    1. K-means
    2. Hierarchical Clustering
    3. DBSCAN
    4. HDBSCAN

Dissimilarity Measures

Dissimilarity is a function of the pair (x, y): $\mathcal{D} : \mathbb{E} \times \mathbb{E} \to \mathbb{R}^+$ s.t

$$\mathcal{D}(x, y) = \mathcal{D}(y, x) \geq 0 \quad \text{and} \quad \mathcal{D}(x, x) = 0 \ \forall x \in \mathbb{E}$$

Distance is a dissimilarity measure that satisfies also

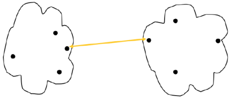
1. $\mathcal{D}(x, y) = 0 \iff x = y$
2. $\mathcal{D}(x, y) \leq \mathcal{D}(x, z) + \mathcal{D}(z, y)$ (metric)

Common distances

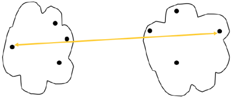
- Minkowski: $\mathcal{D}(x, y) = \left(\sum_{j=1}^{d} |x_j - y_j|^q\right)^{\frac{1}{q}}$
  ($q = 2 \ \to$ Euclidian distance, $q = 1 \ \to$: Manhattan distance)
- Mahalanobis: $\mathcal{D}(x, y) = \left[(x - y)^T \Sigma^{-1} (x - y)\right]^{\frac{1}{2}}$
- Hamming: number of indexes where the 2 vectors differ

Dissimilarity Between Clusters (1/2)

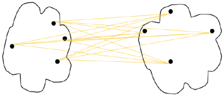
Minimum : $\mathcal{D}(\mathcal{C}_i, \mathcal{C}_j) = \min_{\mathbf{x}\in\mathcal{C}_i, \mathbf{y}\in\mathcal{C}_j} \mathcal{D}(\mathbf{x}, \mathbf{y})$



Maximum : $\mathcal{D}(\mathcal{C}_i, \mathcal{C}_j) = \max_{\mathbf{x}\in\mathcal{C}_i, \mathbf{y}\in\mathcal{C}_j} \mathcal{D}(\mathbf{x}, \mathbf{y})$

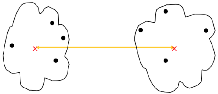


Group Average : $\mathcal{D}(\mathcal{C}_i, \mathcal{C}_j) = \dfrac{1}{n_i\, n_j} \sum_{\mathbf{x}\in\mathcal{C}_i} \sum_{\mathbf{y}\in\mathcal{C}_j} \mathcal{D}(\mathbf{x}, \mathbf{y})$



Between Centroids : $\mathcal{D}(\mathcal{C}_i, \mathcal{C}_j) = \mathcal{D}(m_i, m_j)$,

with $m_i = \dfrac{1}{n_i} \sum_{\mathbf{x}\in\mathcal{C}_i} \mathbf{x}$



Dissimilarity Between Clusters (2/2)

Objective function distances

- Ward distance: $\mathcal{D}(\mathcal{C}_i, \mathcal{C}_j) = \sqrt{\frac{2\, n_i\, n_j}{n_i + n_j}}\mathcal{D}(m_i, m_j)$
- WPGMA (Weighted Pair Group Method with Arithmetic Mean) recursive distance

$$\mathcal{D}(\mathcal{C}_i, \mathcal{C}_j) == \frac{\mathcal{D}(\mathcal{C}_i^1, \mathcal{C}_j) + \mathcal{D}(\mathcal{C}_i^2, \mathcal{C}_j)}{2}$$

where $\mathcal{C}_i^1, \mathcal{C}_i^2$ are the child clusters of $\mathcal{C}_i$

Today's Lecture

1. Introduction

2. Reminders on Clustering
    1. Types of methods and clusters
    2. Distance and Dissimilarity
    3. Clustering Quality

3. From Partitional to Hierarchical Clustering
    1. K-means
    2. Hierarchical Clustering
    3. DBSCAN
    4. HDBSCAN

What makes a good clustering ?

- Centroid: $\boldsymbol{m}_i = \dfrac{1}{n_i} \sum_{\mathbf{x} \in \mathcal{C}_i} \mathbf{x}$
- Inertia: $J_i = \sum_{\mathbf{x} \in \mathcal{C}_i} \mathcal{D}^2(xg, \boldsymbol{m}_i)$
  (low J_i corresponds to a smaller dispersion of points around $\boldsymbol{m}_i$.)
- Within distance: $J_w = \sum_i \sum_{\mathbf{x} \in \mathcal{C}_i} \mathcal{D}^2(xg, \boldsymbol{m}_i) = \sum_i J_i$
- Between distance: $J_b = \sum_i n_i \mathcal{D}^2(\boldsymbol{m}_i, \boldsymbol{m})$
  where $\boldsymbol{m}$ is the sample mean $\boldsymbol{m} = \dfrac{1}{n} \sum \mathbf{x}$

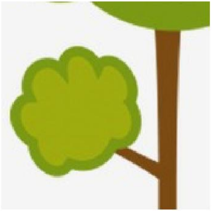
A good clustering...

Minimizes the within distance J_w and maximizes the between distance J_b

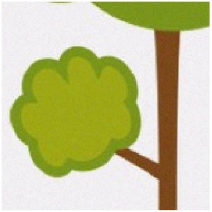
Illustrative Example

Objective

Cluster noisy data for a segmentation application in image processing



(a) Tree data



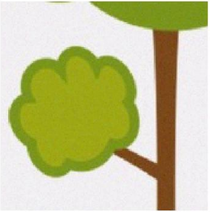
(b) Noisy tree data

Illustrative Example

Objective

Cluster noisy data for a segmentation application in image processing

(c) Tree data



(d) Noisy tree data

Looks easy! But...

Today's Lecture

Today’s Lecture

1. Introduction
2. Reminders on Clustering
    1. Types of methods and clusters
    2. Distance and Dissimilarity
    3. Clustering Quality
3. From Partitional to Hierarchical Clustering
    1. K-means
    2. Hierarchical Clustering
    3. DBSCAN
    4. HDBSCAN

K-means

- Partition the data into K clusters
- Find K clusters and their center μ_k that minimize the cluster within distance J_w
- J_w can be defined as

$$J_w = \Sigma_{k=1}^{K} \Sigma_{x^i \in C_k}^{K} \|\boldsymbol{x}^i - \mu_k\|^2$$

How can we solve this problem?

- NP-hard problem: number of partitions of $\mathbf{x}$ into K subsets

$$P(n, K) = \frac{1}{K!} \sum_{k=0}^{K} k^n \, (-1)^{K-k} \, \frac{K!}{k! \, (K-k)!} \text{ for } K < n$$

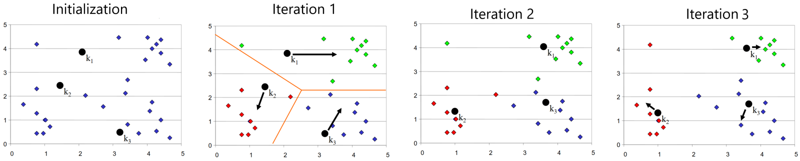
Example: $P(100, 5) \approx 10^{68}$!!!!

- Local solution is obtained with k-means ($O(tKN)$)

K-means algorithm

Iterative minimization

1. Initialize K cluster centers μ_k
2. Assign each $\boldsymbol{x}^i$ to the nearest cluster (nearest center)
   e.g., $\boldsymbol{s}_i \leftarrow \arg\min_k \|\boldsymbol{x}^i - \mu_k\|^2$
3. Re-estimate K cluster centers
   e.g., $\mu_k = \frac{1}{K} \sum_{x^i \in C_k} \boldsymbol{x}^i$
4. Repeat until stopping criterion is reached



K-means Drawbacks and Alternatives

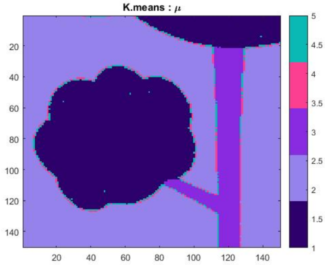
K-means is simple but ...

- Solution depends on initialization
- Need to know K in advance
- Can't handle noise or outliers : *non-robust*
- Fails with clusters of *non-convex* shapes

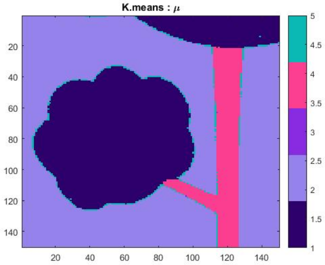
Several alternatives

- K-means++: seeding algorithm $\rightarrow$ initialize clusters with centroids "spread-out" throughout the data
- K-medoids $\rightarrow$ address the robustness aspects
- Kernel K-means $\rightarrow$ overcoming the convex shape limitation
- Many others ...

Results on the tree example



(e) K-means++



(f) “Clusters”

Figure: Clustering obtained with two different initializations

Today's Lecture

1. Introduction
2. Reminders on Clustering
    1. Types of methods and clusters
    2. Distance and Dissimilarity
    3. Clustering Quality
3. From Partitional to Hierarchical Clustering
    1. K-means
    2. Hierarchical Clustering
    3. DBSCAN
    4. HDBSCAN

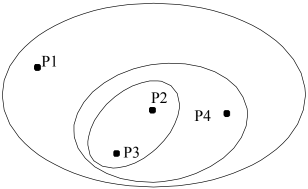
Hierarchical clustering Principles

- Produces a set of nested clusters organized as a hierarchical tree $\rightarrow$ bypass choice of K
- Can be visualized as a dendrogram: a tree like diagram that records the sequences of merges or splits with branch length corresponding to cluster distance

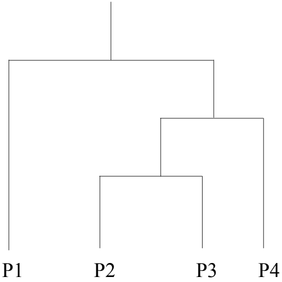
Two approaches

1. Agglomerative: *Bottom-up* - Start with as much clusters as observations and iteratively *aggregate* observations using a given *distance*
2. Divise: *Top-down* - Start with one cluster containing all observations and iteratively *split* into smaller clusters

Hierarchical Clustering: The tree



(a) Hierarchical Clusters



(b) Dendrogram

We can see that ...

- Each node (cluster) in the tree (except the leaf nodes) is the union of its children (subclusters)
- The root of the tree is the cluster containing all objects.

Hierarchical Clustering: Example

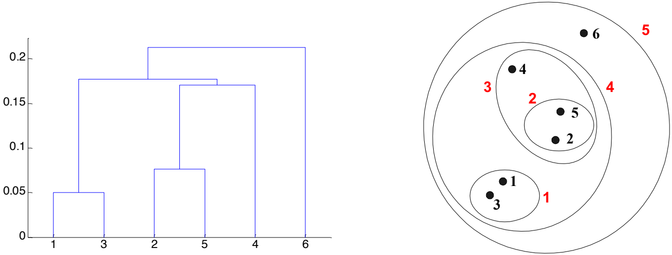


Figure: General principles

Agglomerative Hierarchical Clustering

Algorithm 1 : Agglomerative Hierarchical Clustering

- **Input** : **x** observation vectors and "cutting" threshold λ
- **Output** : all merged clusters set (at each iteration) and "inter-cluster" distances (between clusters)
- **Initialization** : n = sample size = number of clusters.

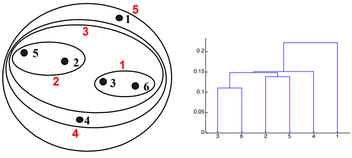
While Number of clusters > 1

1. Compute distances between clusters
2. Merged the two nearest clusters

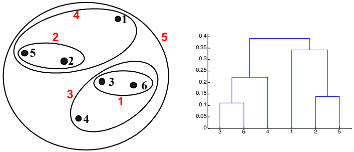
Recall: Inter-Cluster distances

- MIN $\rightarrow$ Single Linkage: $d(\mathcal{C}_i, \mathcal{C}_j) = \min_{\mathbf{x} \in \mathcal{C}_i, \mathbf{y} \in \mathcal{C}_j} d(\mathbf{x}, \mathbf{y})$
- MAX $\rightarrow$ Complete Linkage: $d(\mathcal{C}_i, \mathcal{C}_j) = \max_{\mathbf{x} \in \mathcal{C}_i, \mathbf{y} \in \mathcal{C}_j} d(\mathbf{x}, \mathbf{y})$
- Group Average $\rightarrow$ Average Linkage: $d(\mathcal{C}_i, \mathcal{C}_j) = \frac{1}{n_i n_j} \sum_{\mathbf{x} \in \mathcal{C}_i} \sum_{\mathbf{y} \in \mathcal{C}_j} d(\mathbf{x}, \mathbf{y})$
- Between centroids $\rightarrow$ Centroid Linkage: $d(\mathcal{C}_i, \mathcal{C}_j) = d(m_i, m_j)$
- Objective function $\rightarrow$ Objective Linkage:
  - Ward distance $d(\mathcal{C}_i, \mathcal{C}_j) = \sqrt{\frac{2\, n_i\, n_j}{n_i + n_j}} d(m_i, m_j)$
  - WPGMA recursive distance $d(\mathcal{C}_i, \mathcal{C}_j) == \frac{d(\mathcal{C}_i^1, \mathcal{C}_j) + d(\mathcal{C}_i^2, \mathcal{C}_j)}{2}$ where $\mathcal{C}_i^1, \mathcal{C}_i^2$ are the child clusters of $\mathcal{C}_i$

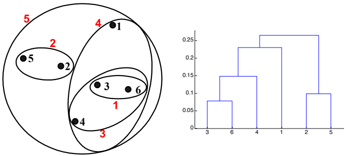
Different distances $\Rightarrow$ Different results



(a) MIN



(b) MAX



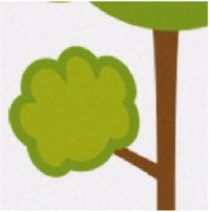
(c) Group average
(Ward provides very similar results)

Different distances $\Rightarrow$ Different results

Pros and cons of different distances

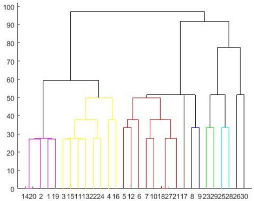
- MIN: can handle non-elliptical shape BUT sensitive to outliers, noise...
- MAX: less sensitive to outliers BUT can break large clusters and biased towards globular clusters
- Average: don't break large clusters BUT biased towards globular clusters
- Ward: Hierarchical analogue of K-means

The Tree Example : Single Linkage

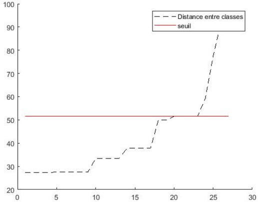


(c) Noisy Tree

(d) Single Linkage

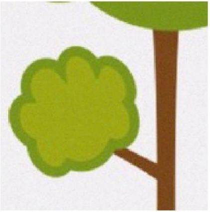


(e) Dendrogram

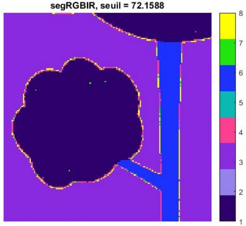


(f) Cutting Threshold

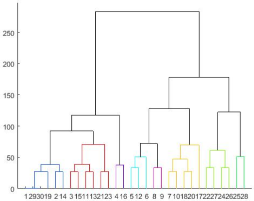
The Tree Example : Complete Linkage



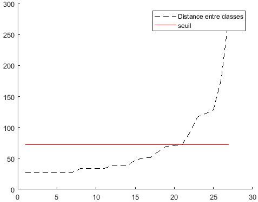
(g) Noisy Tree



(h) Complete Linkage

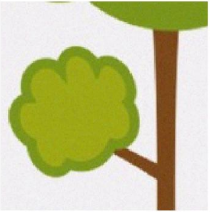


(i) Dendrogram

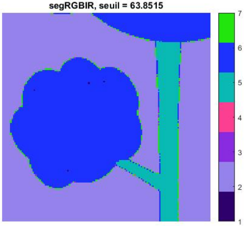


(j) Cutting Threshold

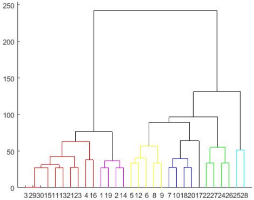
The Tree Example : Average Linkage



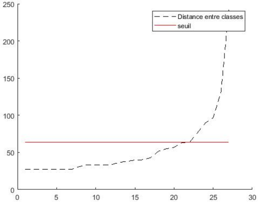
(k) Noisy Tree



(l) Average Linkage

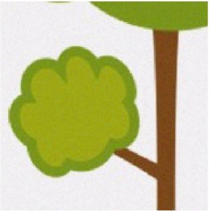


(m) Dendrogram

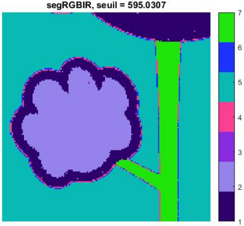


(n) Cutting Threshold

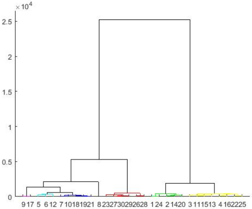
The Tree Example : Ward Linkage



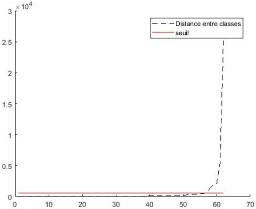
(o) Noisy Tree



(p) Average Linkage

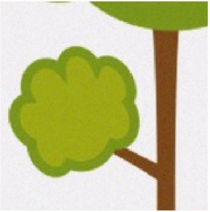


(q) Dendrogram

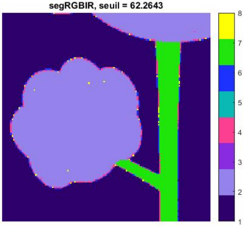


(r) Cutting Threshold

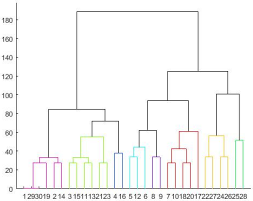
The Tree Example : WPGMA Linkage



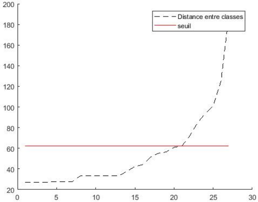
(s) Noisy Tree



(t) Average Linkage



(u) Dendrogram



(v) Cutting Threshold

Hierarchical clustering : Pros and cons

Pros

- Simple and intuitive
- Unsupervised: no *a priori* assumptions
- Interpretable: number of clusters, used distance...

Cons

- Computational cost: single linkage (O(n^3),O(n^2) or O(n)), complete linkage (O(n^3) or O(n^2)), average (O(n^3)), Ward's method (O(n^3)), ...
- Cutting threshold: challenging choice!
- Lack of robustness: sensitivity to outliers and noise
- No global objective function to optimize
- Handle heterogeneous data (clusters of $\neq$ size, non-globular shapes...)

Today's Lecture

1. Introduction

2. Reminders on Clustering
   1. Types of methods and clusters
   2. Distance and Dissimilarity
   3. Clustering Quality

3. From Partitional to Hierarchical Clustering
   1. K-means
   2. Hierarchical Clustering
   3. DBSCAN
   4. HDBSCAN

DBSCAN : A Density-based Algorithm

For an observation $\mathbf{x}_i$, find a sufficiently (MinPts) large neighborhood (ε), then

- aggregate the new observations (neighbors) to the cluster $\mathcal{C}_k$ of $\mathbf{x}_i$,
- else $\mathbf{x}_i$ is an isolated observation (outlier).

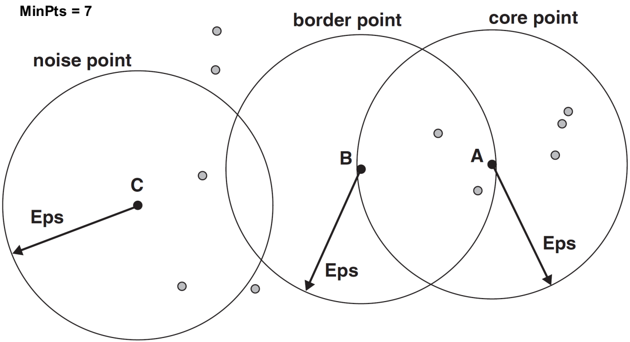
This results in three types of points called core, border, or noise points.

Key parameters

- ε and ε-neighborhood: $\mathcal{N}_\varepsilon(\mathbf{x}_i) = \{\mathbf{z} | d(\mathbf{x}_i, \mathbf{z}) < \varepsilon\}$
- MinPts: n_{min} for defining core points $\mathbf{x}_i$ s.t. $\mathrm{card}(\mathcal{N}_\varepsilon(\mathbf{x}_i)) \geq n_{min}$

DBSCAN: Three Types of Points

1. Core point: is near the center of a cluster/has MinPts neighbors
2. Border point: is not a core point, but is in the neighborhood of a core point
3. Noise point: is any point that is neither a core nor a border point

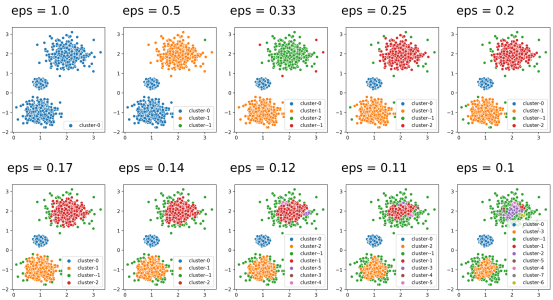


DBSCAN: Influence of ϵ

The parameter ϵ represents the minimum distance between two non-neighboring points:

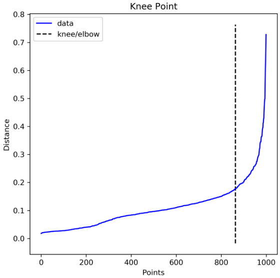
→ A very large ϵ causes all possible clusters to merge into one cluster

→ A very small ϵ leads to a lot of noise, points are not assigned to clusters



DBSCAN: So how do we choose ϵ ?

- Depends on the distance between the data points
- The Elbow trick on the k-NN plot is commonly used in practice (***k*** is MinPts!):
  - $\rightarrow$ x-axis all the points
  - $\rightarrow$ y-axis the average distance of each point to their its k-NN
- Remains a difficult choice!



DBSCAN Algorithm

Algorithm 2-a: DBSCAN

- **Input**: $\mathbf{x}$ observations, ε, MinPts
- **Output**: $\mathcal{Z}$, labels of $\mathbf{x}$

For all $\mathbf{x}_i$

1. Verify that $\mathbf{x}_i$ has not been visited by the algo, else $\mathbf{x}_i$ is marked "as visited"
2. Identify the ε-neighborhood of $\mathbf{x}_i$, $\mathcal{N}_\varepsilon(\mathbf{x}_i)$.
3. **If** $\mathrm{card}(\mathcal{N}_\varepsilon(\mathbf{x}_i)) \leq n_{min}$, then mark P as an isolated point.
   Else Create a cluster $\mathcal{C}_k$ containing $\mathbf{x}_i$ and run class_extension$(\mathcal{C}_k, \mathbf{x}_i, \varepsilon, n_{min})$

DBSCAN Algorithm : Cluster Extension

Algorithm 2-b: DBSCAN Class extension

- **Input**: Cluster $\mathcal{C}_k$ to increase, observation $\mathbf{x}_i$ of $\mathcal{C}_k$, n_{min}, ε.
- **Output** : $\mathcal{Z}$ labels of observations in $\mathcal{N}_\varepsilon(\mathbf{x}_i)$

For all $\mathbf{x}_j, i \neq j$ of $\mathcal{N}_\varepsilon(\mathbf{x}_i)$

1. Verify that $\mathbf{x}_j$ has not been visited by the algo, else $\mathbf{x}_j$ is marked “as visited”
2. Identify the ε-neighborhood of $\mathbf{x}_j$, $\mathcal{N}_\varepsilon(\mathbf{x}_j)$.
3. **If** $\mathrm{card}(\mathcal{N}_\varepsilon(\mathbf{x}_j)) \geq n_{min}$
   $\mathcal{N}_\varepsilon(\mathbf{x}_i) = \mathcal{N}_\varepsilon(\mathbf{x}_i) + \mathcal{N}_\varepsilon(\mathbf{x}_j)$
4. **If** $\mathbf{x}_j$ is not clustered, add to $\mathcal{C}_k$.

Illustration of DBSCAN Principles

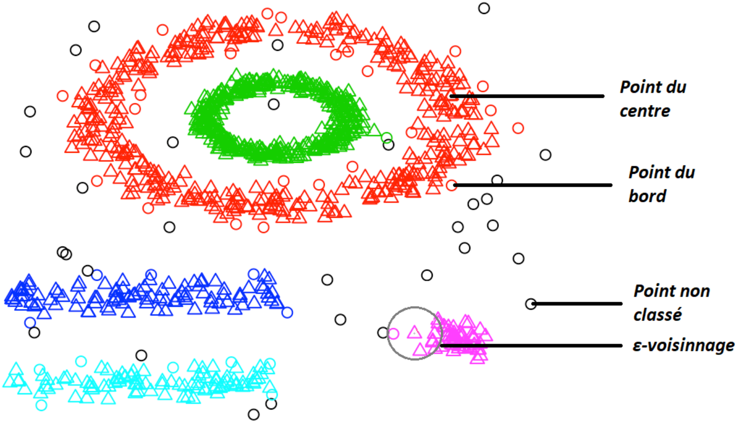


Figure: Clustering results obtained with DBSCAN algorithm.

Algorithms comparison

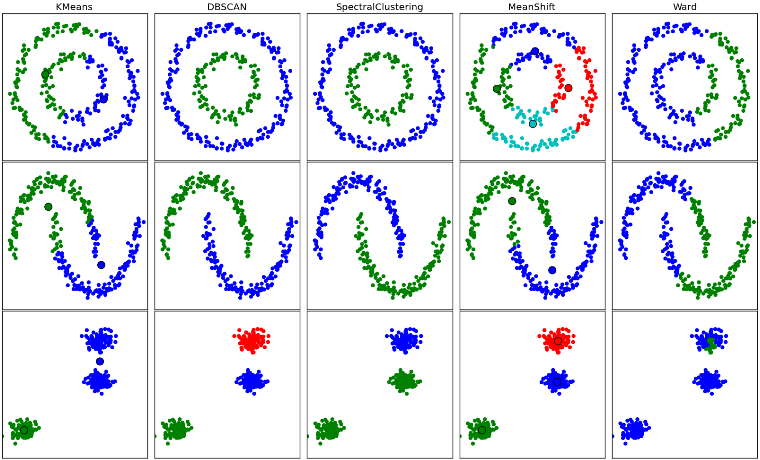
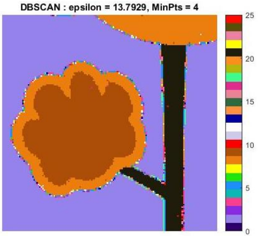
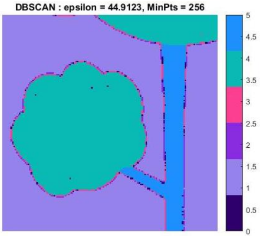


Figure: From Scikits learn: `https://ogrisel.github.io/scikit-learn.org/sklearn-tutorial/modules/clustering.html`

The Tree Data - DBSCAN



(a) MinPts = 4



(b) MinPts = 256

- Pros: Resistant to Noise, can handle clusters of different shapes and sizes
- Cons: Lack of interpretable parameters (estimation), Varying densities, High-dimensional data

Today's Lecture

1. Introduction

2. Reminders on Clustering
   1. Types of methods and clusters
   2. Distance and Dissimilarity
   3. Clustering Quality

3. From Partitional to Hierarchical Clustering
   1. K-means
   2. Hierarchical Clustering
   3. DBSCAN
   4. HDBSCAN

HDBSCAN

Key Idea: Convert DBSCAN into a hierarchical clustering algorithm and

→ bypass the choice of the ϵ-parameter!

→ scan all possible solutions with all values of ϵ

Five main steps

1. Transform the space according to the density/sparsity
2. Build the minimum spanning tree of the distance weighted graph
3. Construct a cluster hierarchy of connected components
4. Condense the cluster hierarchy based on minimum cluster size
5. Extract the stable clusters from the condensed tree

HDBSCAN

Key Idea: Convert DBSCAN into a hierarchical clustering algorithm and

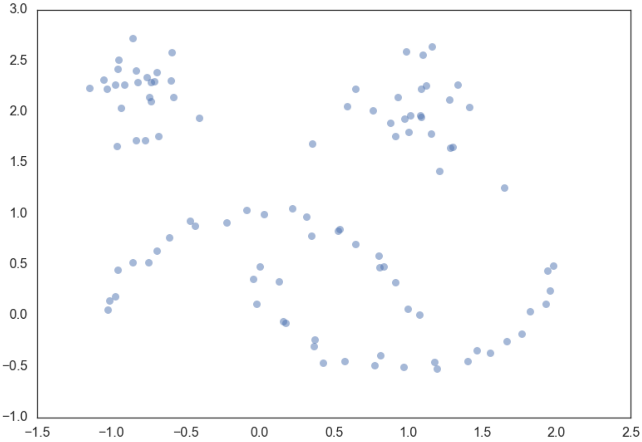
- → bypass the choice of the ϵ-parameter!
- → scan all possible solutions with all values of ϵ

Five main steps

1. Transform the space according to the density/sparsity
2. Build the minimum spanning tree of the distance weighted graph
3. Construct a cluster hierarchy of connected components
4. Condense the cluster hierarchy based on minimum cluster size
5. Extract the stable clusters from the condensed tree

Easier to understand with an example!

HDBSCAN : Illustrative Example



©https://hdbscan.readthedocs.io/en/latest/how$_h$*dbscan$_w$orks.html*

Step 1: Transform The Space

- Goal: Prepare the data for a single linkage clustering (real data is noisy and single linkage is not robust!)
- Key idea: Push sparse points away from the rest of the data before clustering
- The *islands/sea* analogy $\rightarrow$ Make *sea* points more distant from each other and from the *land*

How do we evaluate density ?

- Need an inexpensive density estimate $\Rightarrow$ k-NN is the simplest
- Call it the core distance for parameters k and point $\mathbf{x}_i$, $\mathrm{core}_k(\mathbf{x}_i)$

Step 1: Transform The Space

- Goal: Prepare the data for a single linkage clustering (real data is noisy and single linkage is not robust!)
- Key idea: Push sparse points away from the rest of the data before clustering
- The *islands/sea* analogy $\rightarrow$ Make *sea* points more distant from each other and from the *land*

How do we evaluate density ?

- Need an inexpensive density estimate $\Rightarrow$ k-NN is the simplest
- Call it the core distance for parameters k and point $\mathbf{x}_i$, $\text{core}_k(\mathbf{x}_i)$

And how do we connect points now ?

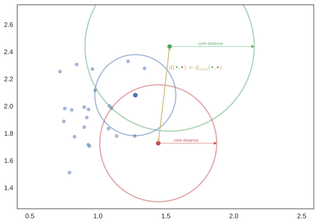
Step 1 : Mutual Reachability Distance

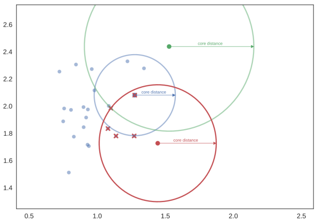
A new distance metric is defined as

$$d_{mreach-k}(\mathbf{x}_i, \mathbf{x}_j) = \max(\mathrm{core}_k(\mathbf{x}_i), \mathrm{core}_k(\mathbf{x}_j), d(\mathbf{x}_i, \mathbf{x}_j)),$$

Meaning that we want to connect points that are

1. Close enough to each other : $d(\mathbf{x}_i, \mathbf{x}_j)$
2. In a dense enough region : $\mathrm{core}_k(\mathbf{x}_i)$



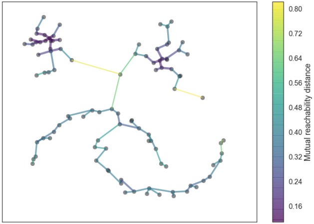


Step 2 : The Minimum Spanning Tree

- Goal: Prepare the data for clustering using d_{mreach}
- Key ideas:
  - Construct a graph that connects all points
  - Start disconnecting them by lowering a threshold (sea level drops)
  - Points are the vertices and the *edges* are weighted by d_{mreach}
  - n^2 possible edges $\rightarrow$ the minimum spanning tree

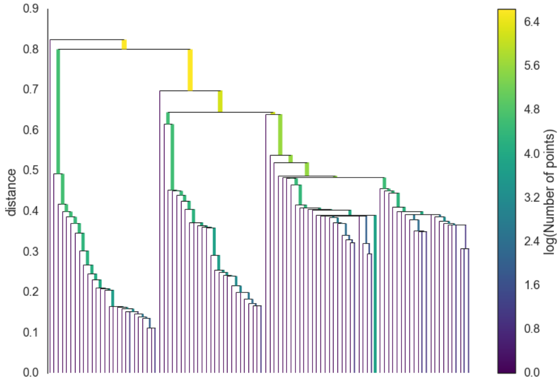
Algorithms from graph theory

- Prim's algorithm
- Dual Tree Boruvka



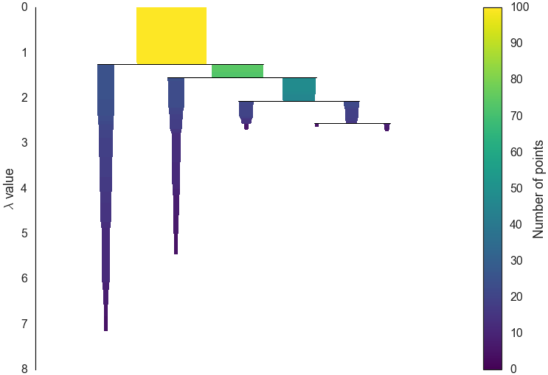
Step 3: Build the cluster hierarchy

Clusters emerge progressively as we lower the d_{mreach} threshold ($\rightarrow$ sort the edges and start single linkage)



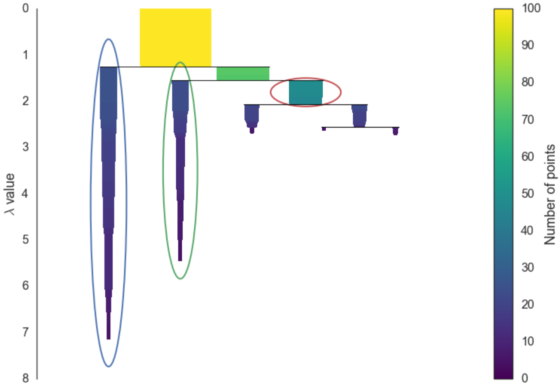
Step 4 : Condense the cluster tree

Get rid of levels that resulted in noise : nbr of points $\leq C_{\min}$
(clusters are shrinking $\neq$ splitting)

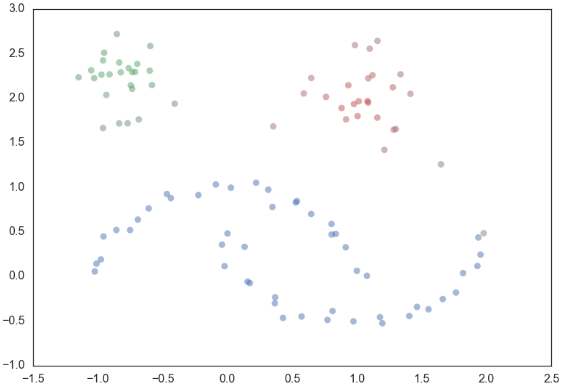


Extract the clusters

Key idea: Choose clusters that persist (live for a long time) and that are large → maximize a stability criterion
(flat clustering: can't select descendance of a selected cluster!)



Results



HDBSCAN: Summary

Implementation: The 5 main steps

1. Compute $\mathrm{core}_k(\mathbf{x}_i)$ using *MinPts* $\rightarrow$ Measure density
2. Transform the space: use new metric d_{mreach}
3. Construct a minimum spanning tree
4. Simplify/condense the tree using $C_{\min}$
5. Extract final clustering results

In conclusion: Two parameters (*MinPts* and $C_{\min}$), varying densities, robust to outliers, interpretability...

HDBSCAN: Summary

Implementation: The 5 main steps

1. Compute $\mathrm{core}_k(\mathbf{x}_i)$ using *MinPts* $\rightarrow$ Measure density
2. Transform the space: use new metric d_{mreach}

   $\rightarrow$ Robustness to noise!
3. Construct a minimum spanning tree
4. Simplify/condense the tree using C_{min}
5. Extract final clustering results

In conclusion: Two parameters (*MinPts* and C_{min}), varying densities, robust to outliers, interpretability...

HDBSCAN: Summary

Implementation: The 5 main steps

1. Compute $\mathrm{core}_k(\mathbf{x}_i)$ using *MinPts* $\rightarrow$ Measure density
2. Transform the space: use new metric d_{mreach}

   $\rightarrow$ Robustness to noise!
3. Construct a minimum spanning tree

   $\rightarrow$ Lower computational cost
4. Simplify/condense the tree using $C_{\min}$
5. Extract final clustering results

In conclusion: Two parameters (*MinPts* and $C_{\min}$), varying densities, robust to outliers, interpretability...

HDBSCAN: Summary

Implementation: The 5 main steps

1. Compute $\mathrm{core}_k(\mathbf{x}_i)$ using *MinPts* $\rightarrow$ Measure density
2. Transform the space: use new metric d_{mreach}

   $\rightarrow$ Robustness to noise!
3. Construct a minimum spanning tree

   $\rightarrow$ Lower computational cost
4. Simplify/condense the tree using C_{min}

   $\rightarrow$ Preprocessing for the next step
5. Extract final clustering results

In conclusion: Two parameters (*MinPts* and C_{min}), varying densities, robust to outliers, interpretability...

HDBSCAN: Summary

Implementation: The 5 main steps

1. Compute $\mathrm{core}_k(\mathbf{x}_i)$ using *MinPts* $\rightarrow$ Measure density
2. Transform the space: use new metric d_{mreach}

   $\rightarrow$ Robustness to noise!
3. Construct a minimum spanning tree

   $\rightarrow$ Lower computational cost
4. Simplify/condense the tree using $C_{\min}$

   $\rightarrow$ Preprocessing for the next step
5. Extract final clustering results

   $\rightarrow$ Maximize cluster stability

In conclusion: Two parameters (*MinPts* and $C_{\min}$), varying densities, robust to outliers, interpretability...