Advanced Machine Learning
Course IV - (Hierarchical) Clustering

L. Omar Chehab[(1)] and Frédéric Pascal[(2)]

[(1)] Parietal Team, Inria
[(2)] Laboratory of Signals and Systems (L2S), CentraleSupélec, University Paris-Saclay

l-emir-omar.chehab@inria.fr, frederic.pascal@centralesupelec.fr,

http://fredericpascal.blogspot.fr

Dominante MDS (Mathématiques, Data Sciences)
Sept. - Dec., 2020

CentraleSupélec

Contents

Key references for this course

- 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

Course 4

(Hierarchical) Clustering

I. Introduction to clustering

II. Clustering algorithms

III. Clustering algorithm performance

What is Clustering?

Divide data into groups (clusters) that are meaningful and / or useful, i.e. that capture the natural structure.

Purposes of the clustering is either understanding or utility:

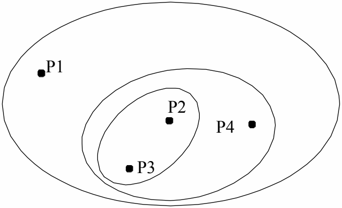
- Clustering for understanding e.g., in Biology, Information retrieval (web...), Climate, Psychology and Medicine, Business...
- Clustering for utility:
  - Summarization : dimension reduction → PCA, regression on high dimensional data. Work on clusters characteristics instead of all data
  - Compression, a.k.a vector quantization
  - Efficiently finding nearest neighbors.

It is an unsupervised learning contrary to (supervised) classification!

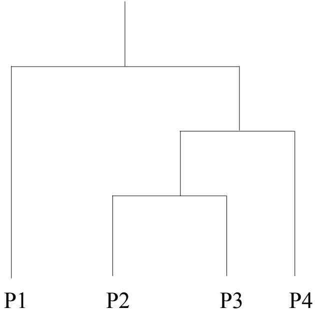
Hierarchical vs Partitional

Partitional clustering: Division of the sets of data objects into non-overlapping subsets (clusters) s.t. each data is in exactly one subset.

If clusters can have sub-clusters ⇒ **Hierarchical clustering:** set of nested clusters, organized as a tree. 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.



(a) Hierarchical Clusters



(b) Dendrogram

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 $\mathscr{C}_k$ 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...

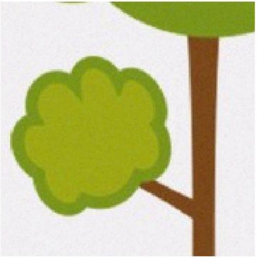
Type of clusters

- 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. 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.
- Others... graph-based...

Data set

The objective is to cluster the noisy data for a segmentation application in image processing.

(c) Tree data



(d) Noisy tree data

Figure: Data on which the clustering algorithms are evaluated

Should be easy...

I. Introduction to clustering

II. Clustering algorithms

- K-means
- Hierarchical clustering
- DBSCAN
- HDBSCAN

III. Clustering algorithm performance

Clustering algorithms

K-means

K-means

It is a prototype-based clustering technique.

Notations: n unlabelled data vectors of $\mathbb{R}^p$ denoted as $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ which should be split into K classes $\mathscr{C}_1, \ldots, \mathscr{C}_K$, with $\text{Card}(\mathscr{C}_k) = n_k$, $\sum_{k=1}^{K} n_k = n$.

Centroid of $\mathscr{C}_k$ is denoted m_k.

Optimal solution

Number of partitions of $\mathbf{x}$ into K subsets:

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

where $C_K^k = \dfrac{K!}{k!\,(K-k)!}$.

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

K-means algorithm

- Partitional clustering approach where K of clusters must be specified
- Each observation is assigned to the cluster with the closest centroid
- Minimizes the intra-cluster variance $V = \sum_k \sum_{i|\mathbf{x}_i \in \mathscr{C}_k} \frac{1}{n_k} ||\mathbf{x}_i - m_k||^2$
- The basic algorithm is very simple

Algorithm 1 K-means algorithm

Input : $\mathbf{x}$ observation vectors and the number K of clusters
Output : $\mathbf{z} = (z_1, \ldots, z_N)$, the labels of $(\mathbf{x}_1, \ldots, \mathbf{x}_N)$
Initialization : Randomly select K points as the initial centroids
Until convergence (define a criterion, e.g. error, changes, centroids estimation...) **Repeat**

1. Form K clusters by assigning $\mathbf{x}_i$ to the closest centroid m_k
   $C_k = \{\mathbf{x}_i,\ \forall i \in \{1, ..., n\} \ |\ \ d(\mathbf{x}_i, m_k) \leq d(\mathbf{x}_i - m_j)\ , \forall j \in \{1, ..., K\}\ \}$
2. Recompute the centroids
   $\forall k \in \{1, ..., K\} : m_k = \frac{1}{n_k} \sum_{\mathbf{x}_i \in \mathscr{C}_k} \mathbf{x}_i.$

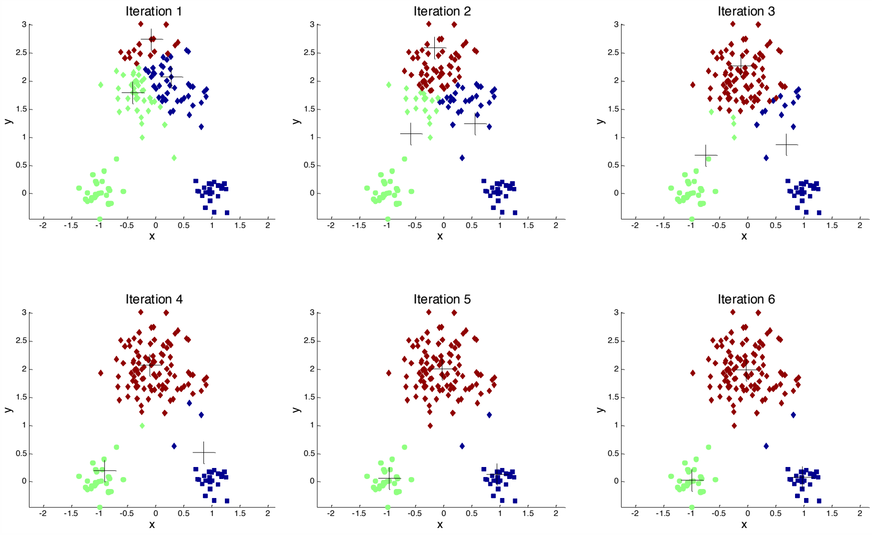
K-means drawbacks...

- Random initialization
- Empty clusters
- Used for clusters with convex shape
- sensitive to noise and outliers
- Computational cost
- ...

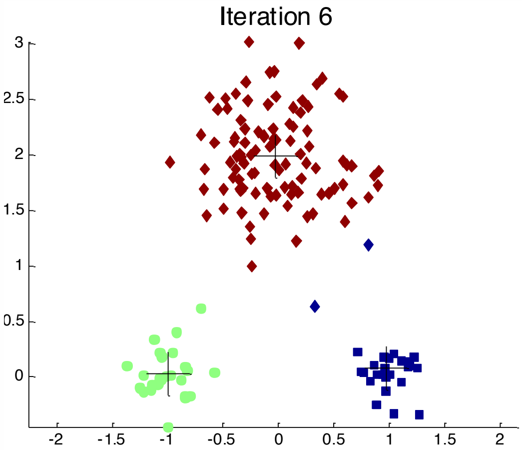
Several alternatives

- K-means++: Seeding algorithm to initialize clusters with centroids "spread-out" throughout the data
- K-medoids: To address the robustness aspects
- Kernel K-means: For overcoming the convex shape
- Many others ...

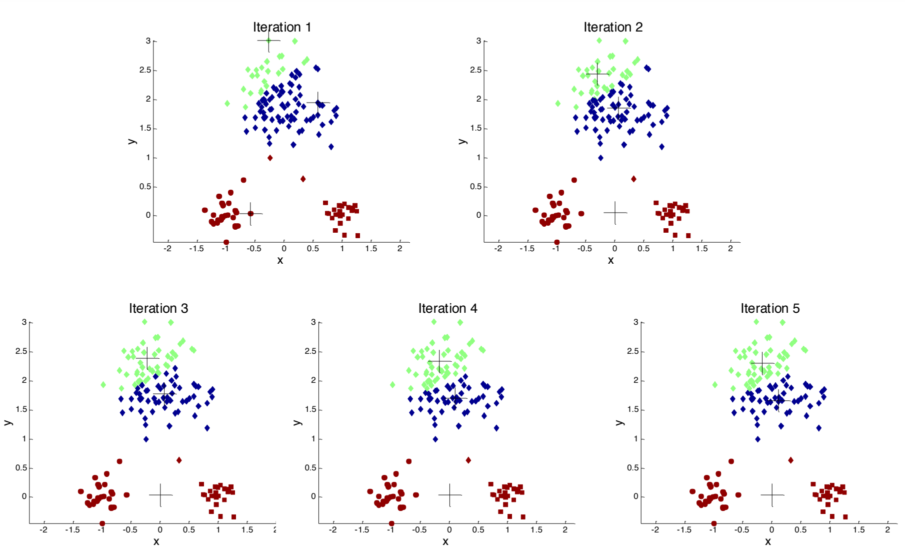
Correct initilization



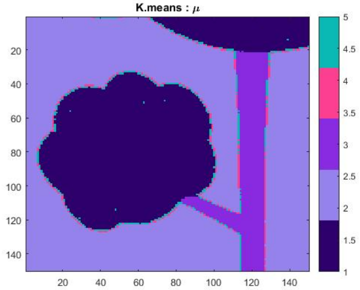
Correct initilization



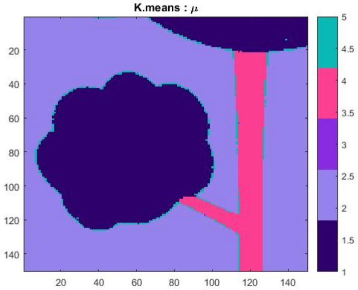
Bad initialization



Results on the data set



(a) K-means++



(b) "Clusters"

Figure: Clustering obtained with two different initialization techniques

Comments...

Clustering algorithms

Hierarchical clustering

Hierarchical clustering

Two types of Hierarchical clustering:

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

Principles:

- Produces a set of nested clusters organized as a hierarchical tree
- 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

Hierarchical clustering

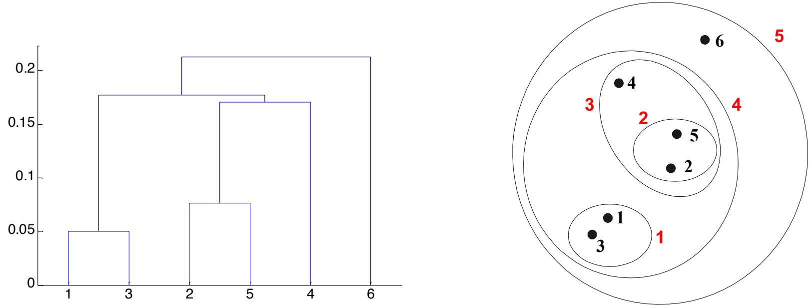


Figure: General principles

Inter-Cluster distance

Most popular clustering techniques

Algorithm 2 Agglomerative hierarchical clustering

Input : $\mathbf{x}$ 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.

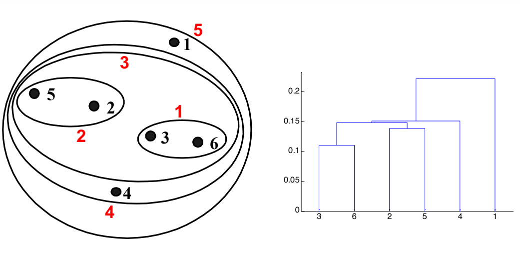
While Number of clusters > 1

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

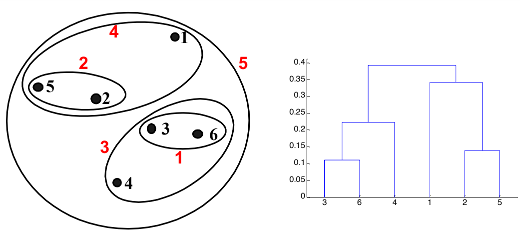
Inter-Cluster distances

- MIN → Single Linkage: $d(\mathscr{C}_i, \mathscr{C}_j) = \min_{\mathbf{x}\in\mathscr{C}_i, \mathbf{y}\in\mathscr{C}_j} d(\mathbf{x}, \mathbf{y})$
- MAX → Complete Linkage: $d(\mathscr{C}_i, \mathscr{C}_j) = \max_{\mathbf{x}\in\mathscr{C}_i, \mathbf{y}\in\mathscr{C}_j} d(\mathbf{x}, \mathbf{y})$
- Group Average → Average Linkage: $d(\mathscr{C}_i, \mathscr{C}_j) = \dfrac{1}{n_i n_j} \sum_{\mathbf{x}\in\mathscr{C}_i} \sum_{\mathbf{y}\in\mathscr{C}_j} d(\mathbf{x}, \mathbf{y})$
- Between centroid → Centroid Linkage: $d(\mathscr{C}_i, \mathscr{C}_j) = d(m_i, m_j)$, with $m_i = \dfrac{1}{n_i} \sum_{\mathbf{x}\in\mathscr{C}_i} \mathbf{x}$
- Objective function → Objective Linkage:
  - Ward distance $d(\mathscr{C}_i, \mathscr{C}_j) = \sqrt{\dfrac{2\, n_i\, n_j}{n_i + n_j}}\, d(m_i, m_j)$
  - WPGMA (Weighted Pair Group Method with Arithmetic Mean) recursive distance $d(\mathscr{C}_i, \mathscr{C}_j) == \dfrac{d(\mathscr{C}_i^1, \mathscr{C}_j) + d(\mathscr{C}_i^2, \mathscr{C}_j)}{2}$ where $\mathscr{C}_i^1, \mathscr{C}_i^2$ are the child clusters of $\mathscr{C}_i$
  - ...

Different distances ⇒ different results



(a) MIN



(b) MAX

Different distances ⇒ different results

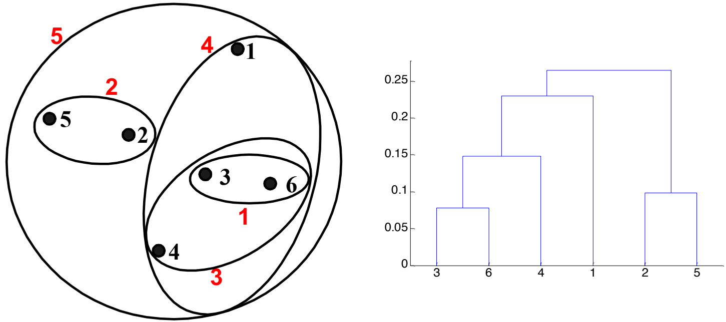
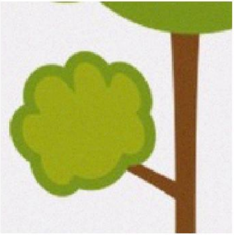


Figure: Group average

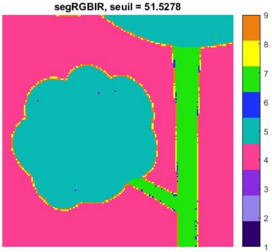
Ward: very similar results.

- 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

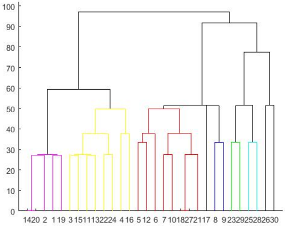
Results on the data set - Single Linkage



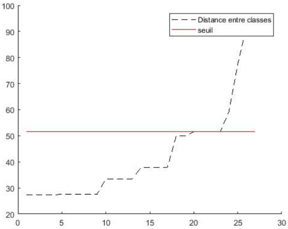
(a) Noisy Tree



(b) Single Linkage

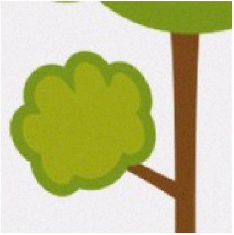


(c) Dendrogram

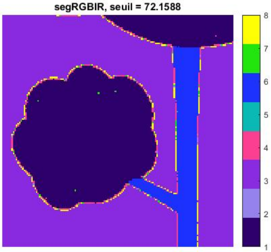


(d) Cutting Threshold

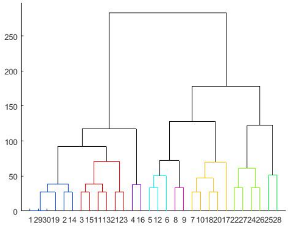
Results on the data set - Complete Linkage



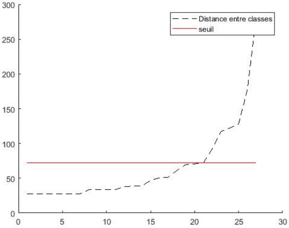
(e) Noisy Tree



(f) Complete Linkage

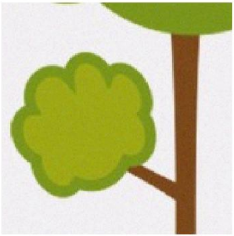


(g) Dendrogram

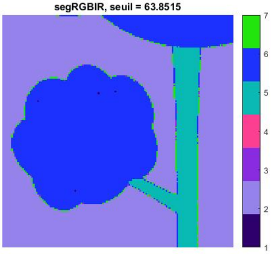


(h) Cutting Threshold

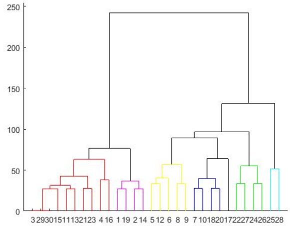
Results on the data set - Average Linkage



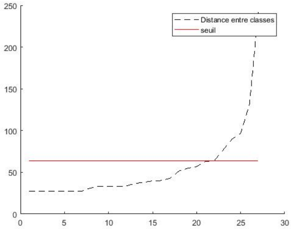
(i) Noisy Tree



(j) Average Linkage

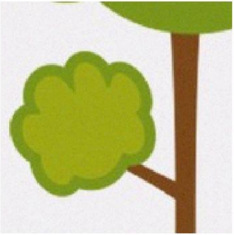


(k) Dendrogram

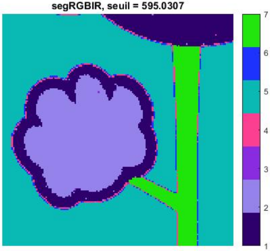


(l) Cutting Threshold

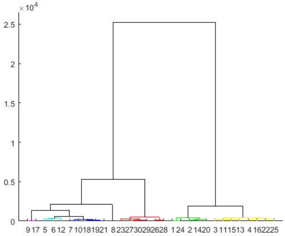
Results on the data set - Ward Linkage



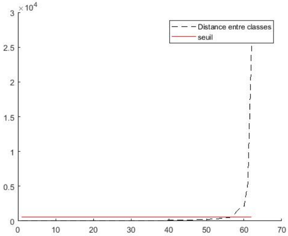
(m) Noisy Tree



(n) Average Linkage

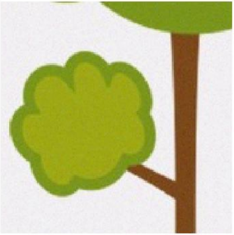


(o) Dendrogram

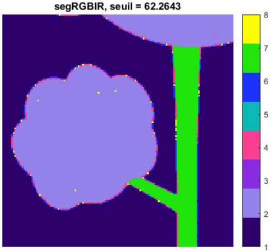


(p) Cutting Threshold

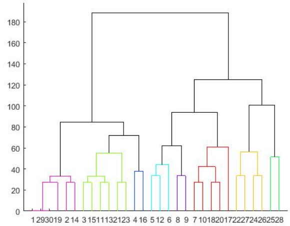
Results on the data set - WPGMA Linkage



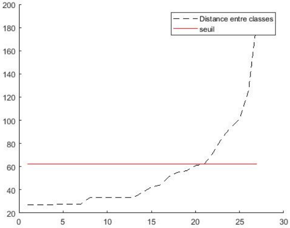
(q) Noisy Tree



(r) Average Linkage



(s) Dendrogram



(t) 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...)

Clustering algorithms

DBSCAN

DBSCAN

Principles: Density-based algorithm: for an observation $\mathbf{x}_i$, find a sufficiently (MinPts) large neighborhood (ε) and aggregate the new observations (neighbors) to the cluster $\mathscr{C}_k$ of $\mathbf{x}_i$. Else $\mathbf{x}_i$ is an isolated observation (outlier).

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. $\text{card}(\mathcal{N}_\varepsilon(\mathbf{x}_i)) \geq n_{min}$

Also, a border points is not a core point, but is in the neighborhood of a core point and a noise point is any point that is not a core or a border point.

DBSCAN

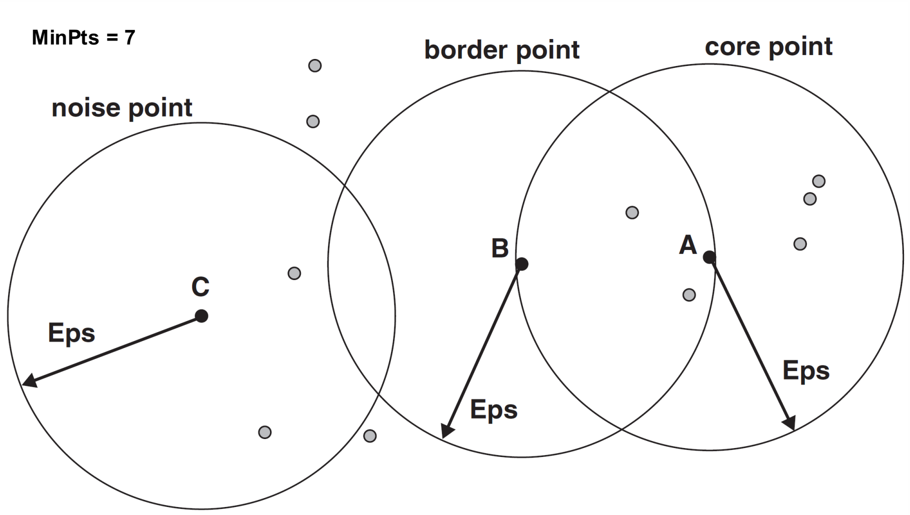


Figure: Different points

DBSCAN algorithm

Algorithm 3 DBSCAN algorithm

Input: $\mathbf{x}$ observations, ε, MinPts
Output: $\mathscr{Z}$, 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 $\mathbf{x}_i$, $\mathscr{N}_\varepsilon(\mathbf{x}_i)$.
3. **If** $\mathrm{card}(\mathscr{N}_\varepsilon(\mathbf{x}_i)) \leq n_{min}$, then mark P as an isolated point.
   Else Create a cluster $\mathscr{C}_k$ containing $\mathbf{x}_i$ and run class_extension($\mathscr{C}_k, \mathbf{x}_i, \varepsilon, n_{min}$)

Cluster extension

Algorithm 4 Extension class function

Input: Cluster $\mathscr{C}_k$ to increase, observation $\mathbf{x}_i$ of $\mathscr{C}_k$, n_{min}, ε.

Output : $\mathscr{Z}$ labels of observations in $\mathscr{N}_\varepsilon(\mathbf{x}_i)$

Forall $\mathbf{x}_j, i \neq j$ of $\mathscr{N}_\varepsilon(\mathbf{x}_i)$

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

Illustration of DBSCAN principles

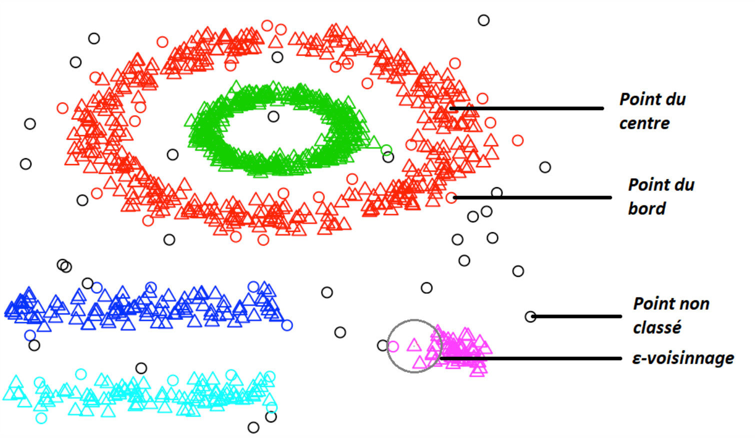
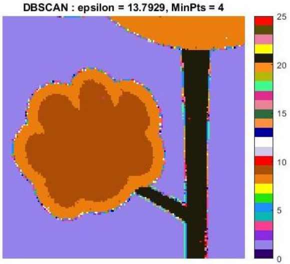
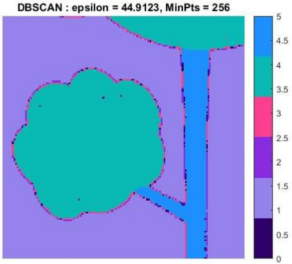


Figure: Clustering results obtained with DBSCAN algorithm.

Results on the data set - DBSCAN





(a) MinPts = 256 (b) MinPts = 4

Figure: Influence of MinPts and ε

Discussion: ε, number of clusters, MinPts...

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

Algorithms comparison

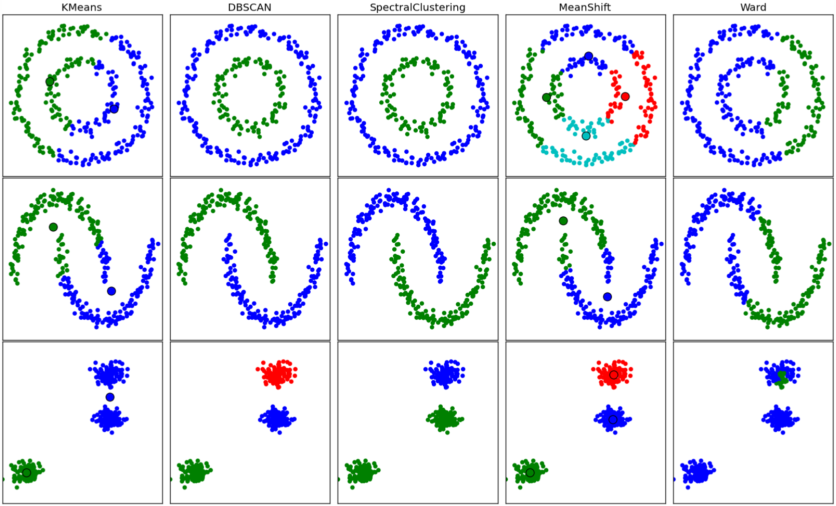


Figure: From Scikits learn: `https://ogrisel.github.io/scikit-learn.org/sklearn-tutorial/modules/clustering.html`

Clustering algorithms

Hierarchical DBSCAN

Campello, R.J., Moulavi, D. and Sander, J., *"Density-based clustering based on hierarchical density estimates"*. In Pacific-Asia conference on knowledge discovery and data mining (pp. 160-172). Springer, Berlin, Heidelberg, April 2013.

HDBSCAN

General (Intuitive) Idea: Convert DBSCAN into a hierarchical clustering algorithm.

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.

Data example

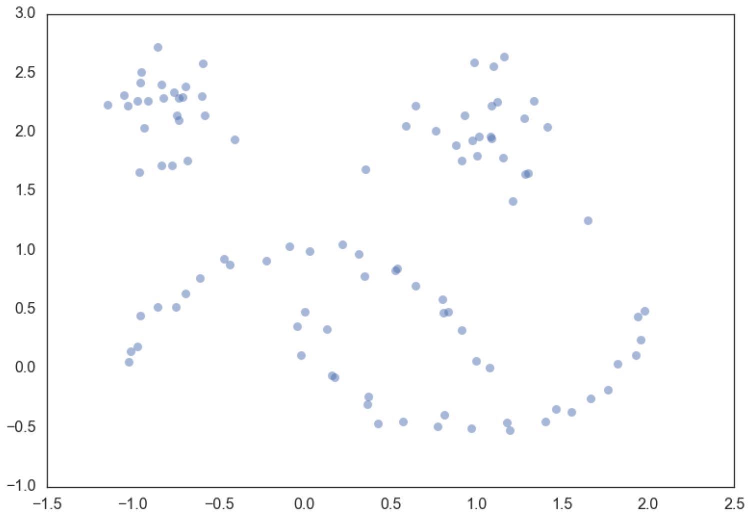


Figure: Data

Transform the space

Goal: Finds "islands" of higher density amid a sea of sparser noise (important for real data!).
Behind there is a single linkage algorithm Remember: not robust to outliers, SO identify/evaluate the outliers, "sea" points, initial step.

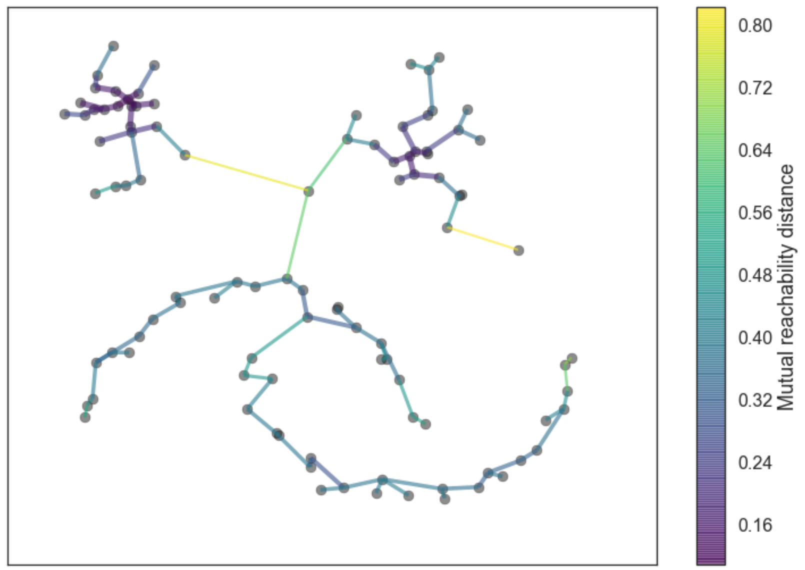
Intuition: *Make "sea" points more distant from each other and from the "land".*

Practically (theoretically): need inexpensive density estimate ⇒ distance of the kNN is the simplest. Call it the core distance for parameters k and point $\mathbf{x}_i$, $\text{core}_k(\mathbf{x}_i)$. Now to spread apart points with low density, new distance metric, called the mutual reachability distance:

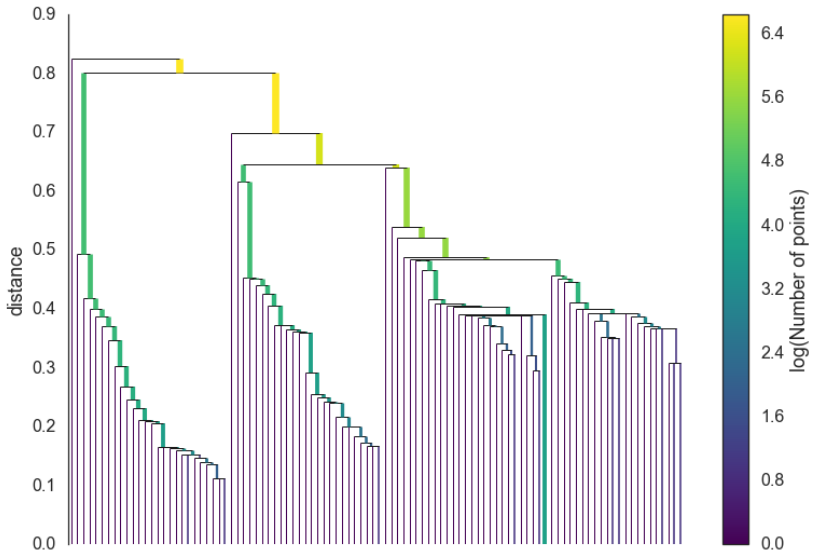
$$d_{mreach-k}(\mathbf{x}_i, \mathbf{x}_j) = \max(\text{core}_k(\mathbf{x}_i), \text{core}_k(\mathbf{x}_j), d(\mathbf{x}_i, \mathbf{x}_j))$$

where $d(.,.)$ is the original metric.

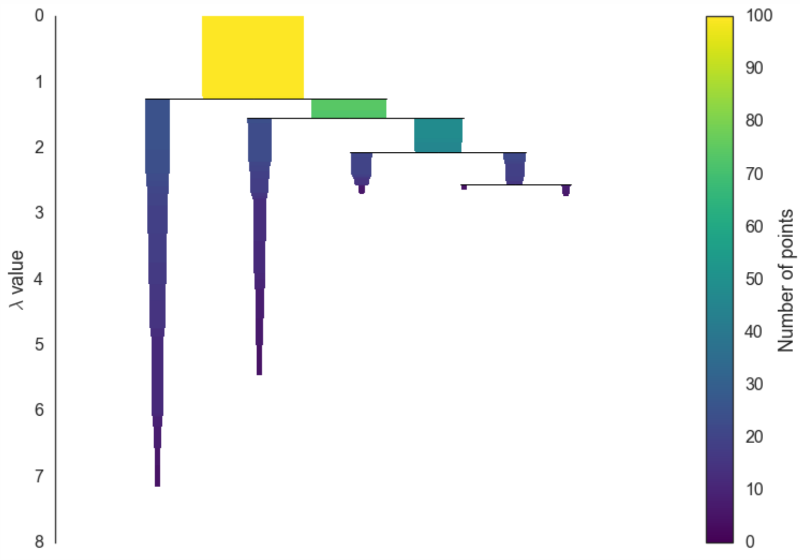
Build the minimum spanning tree



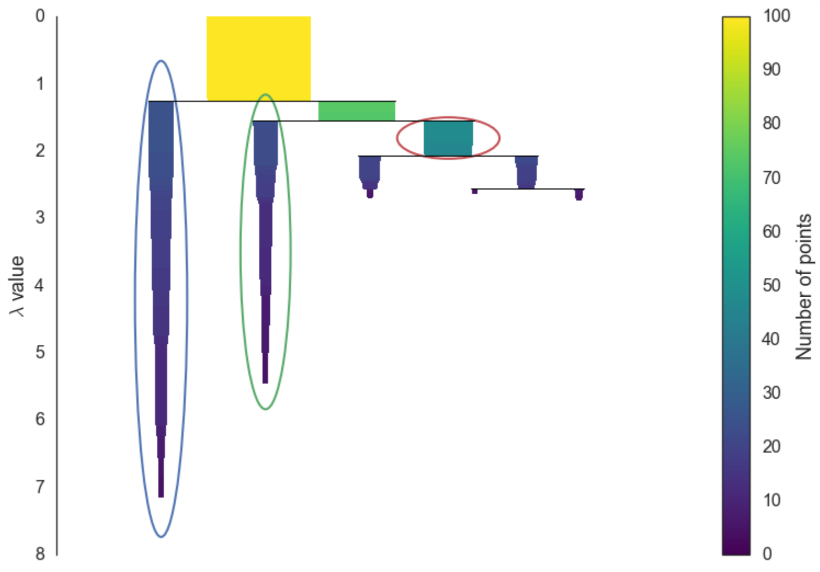
Build the cluster hierarchy



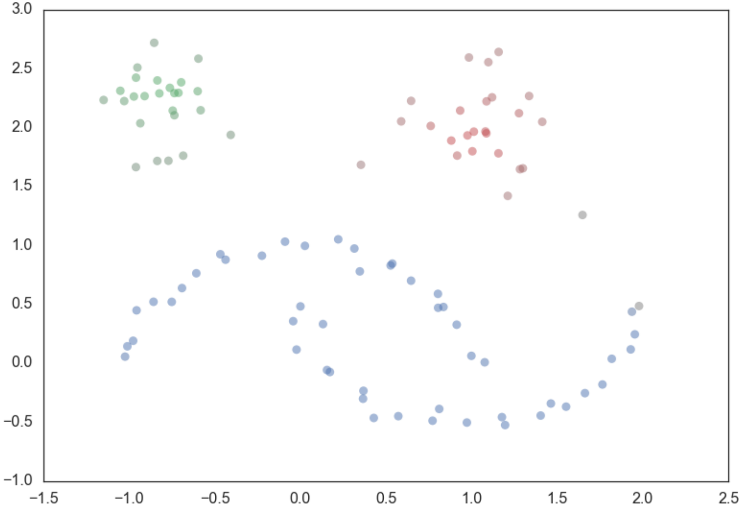
Condense the cluster tree



Extract the clusters



Results



Interests: Varying densities, confidence information on the observation cluster, robust to outliers, interpretability...

How to evaluate the quality of of clustering results?

To be updated