Advanced Machine Learning
Course 2 - (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

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

http://fredericpascal.blogspot.fr

Dominante MDS (Mathématiques, Data Sciences)

Sept. - Nov., 2021

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

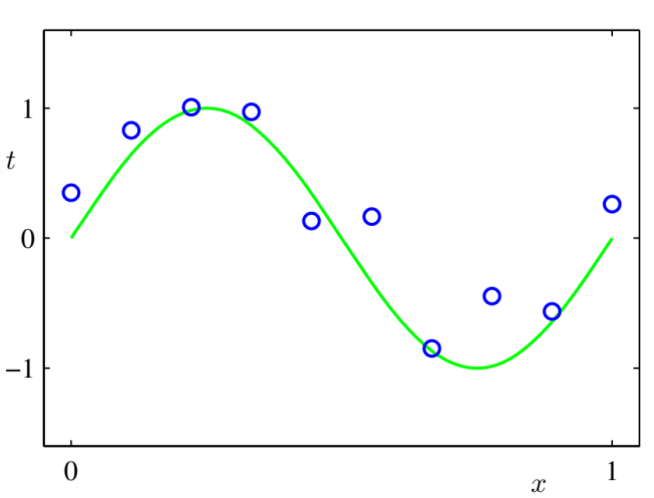
Course 2

Classification and (hierarchical) Clustering

Regression vs Classification

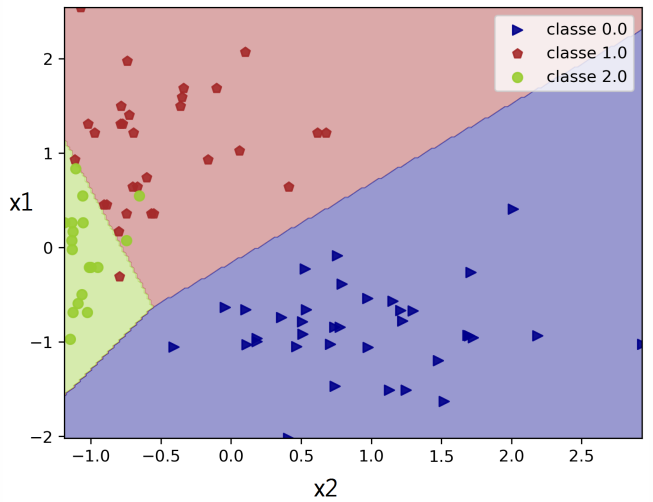
Regression

- $y \in \mathbb{R}$ is a continuous variable
- Predict a numerical value



Classification

- labels are discrete variables
- Binary Classification $y \in \{0, 1\}$, $y \in \{-1, 1\}$, ...
- Multiclass $y \in \{1, ..., K\}$



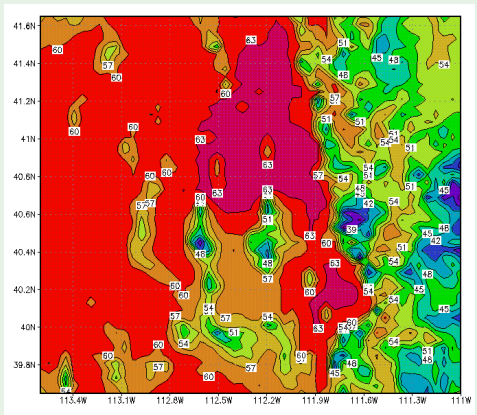
Regression Applications

Financial data

- x: economical, social, political variables
- y: stock price

Weather prediction

- x: location, ...
- y: temperature value



Today's Lecture

I. Classification
- Reminders on linear SVMs
- SVM - Handling non-linear boundaries: Kernel Machines

II. Introduction to clustering

III. Reminders on Clustering
- Types of methods and clusters
- Distance and Dissimilarity
- Clustering Quality

IV. Clustering algorithms
- K-means
- Hierarchical clustering
- DBSCAN
- HDBSCAN

Today's Lecture

I. Classification
- Reminders on linear SVMs
- SVM - Handling non-linear boundaries: Kernel Machines

II. Introduction to clustering

III. Reminders on Clustering
- Types of methods and clusters
- Distance and Dissimilarity
- Clustering Quality

IV. Clustering algorithms
- K-means
- Hierarchical clustering
- DBSCAN
- HDBSCAN

Linear SVM: Problem Formulation

- Training set of pairs $(x_i, y_i), i = 1, .., n$
- $x_i \in \mathbb{R}^d$ and $y \in \{-1, 1\}$

Objective

Find a linear function $f(x) = w^T x + b,\ \ w \in \mathbb{R}^d, b \in \mathbb{R}$ that classifies input samples such that

$$f(x) > 0 \ \ x \text{ is assigned to class 1}$$
$$f(x) < 0 \ \ x \text{ is assigned to class -1}$$

- Classification rule is $\text{sign}(f(x))$

Linear SVM: Problem Formulation

- Training set of pairs $(x_i, y_i), i = 1, .., n$
- $x_i \in \mathbb{R}^d$ and $y \in \{-1, 1\}$

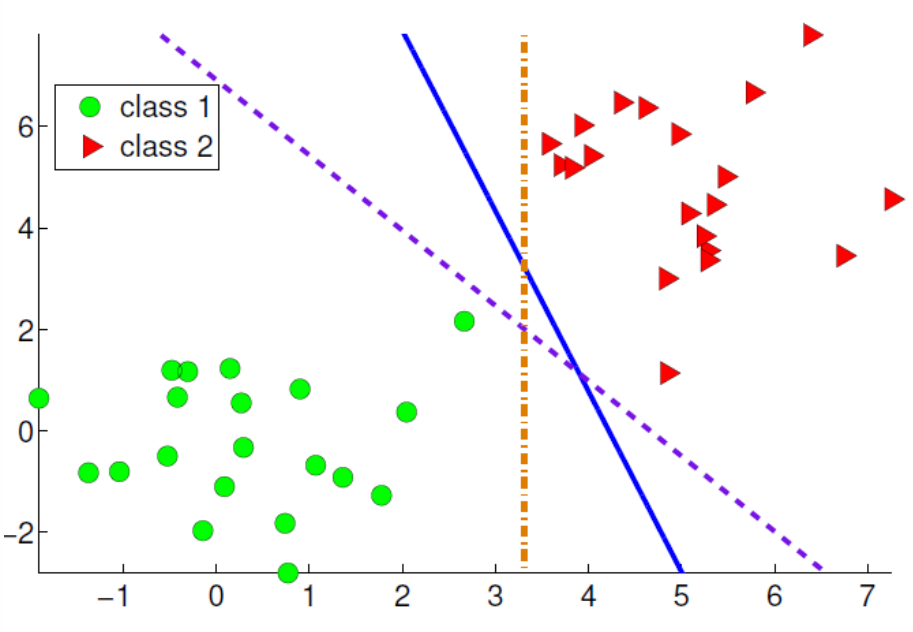
Objective

Find a linear function $f(x) = w^T x + b,\ \ w \in \mathbb{R}^d, b \in \mathbb{R}$ that classifies input samples such that

$$f(x) > 0 \ \ x \text{ is assigned to class 1}$$
$$f(x) < 0 \ \ x \text{ is assigned to class -1}$$

- Classification rule is $\text{sign}(f(x))$

Max Margin Classifier



Best classifier?

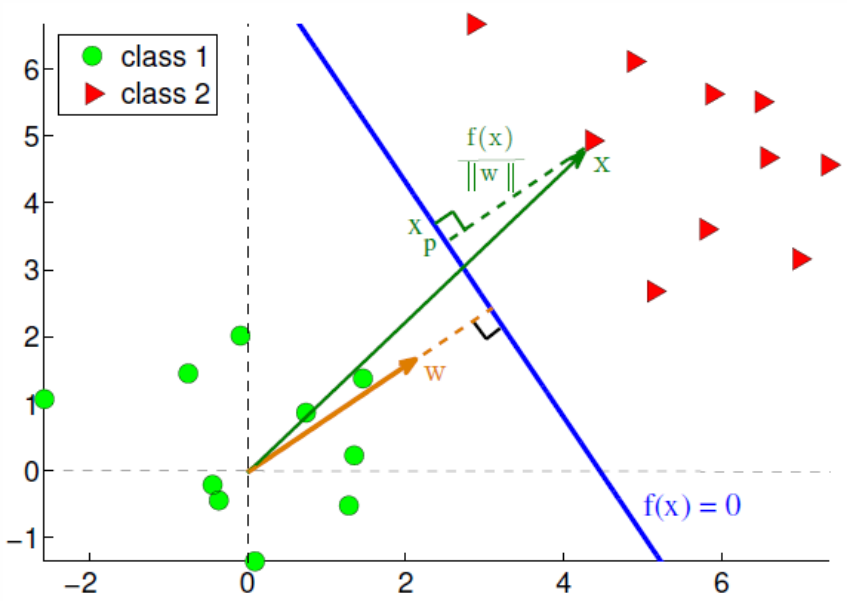
- Decision boundary that is more "stable", we are confident in all decisions
- We want observations to be as far from the decision boundary as possible

$\rightsquigarrow$ large margin

Max Margin Classifier

The margin is the smallest distance $d(H, x)$ between the boundary (H) and any of the observations

$$d(x_i, H) = \frac{y_i(w^T x_i + b)}{\|w\|} = \frac{|f(x_i)|}{\|w\|}$$

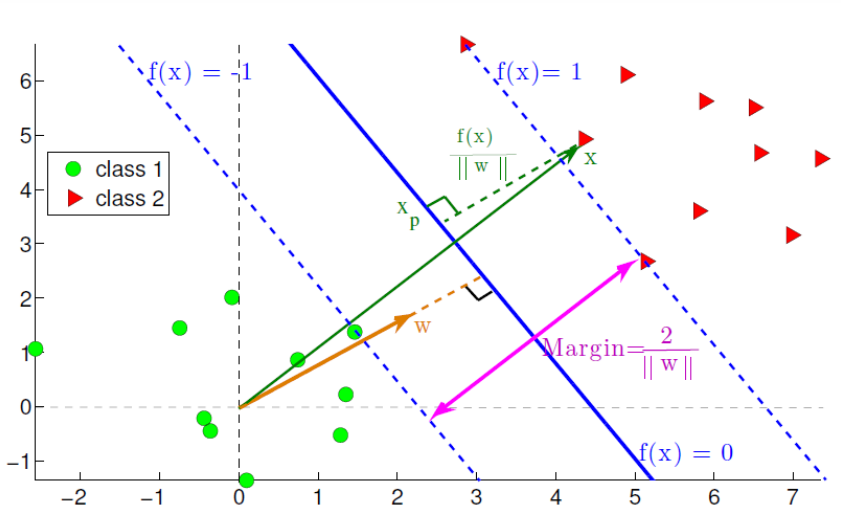


Max Margin Classifier: Canonical Hyperplane

Constraints for the hyperplane: one forces the training samples that are the closest to the boundary to satisfy

$$y_i(w^T x_i + b) = 1 \implies \min_{x_i} |w^T x + b| = 1$$

- The x_i satisfying $y_i(w^T x_i + b) = 1$ are the support vectors



The geometrical margin $M = \frac{2}{\|w\|}$

Linear SVM: Optimization Problem

Goal: Maximize the margin while correctly classifying each sample $\rightsquigarrow$ constrained optimization problem

Primal problem

$$\min_{w,b} \frac{1}{2}\|w\|^2 \quad \text{s.t} \quad y_i(w^T x_i + b) \geq 1, \ \forall i = 1, ..., n$$

Simple problem since the cost function to optimize is quadratic and the constraints are linear!

Linear SVM: Dual problem

Lagrangian formulation

$$L(w,\alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i [y_i(w^T x_i + b) - 1]$$

- α_i are the Lagrange multipliers, dual variables
- Set derivatives wrt w and b to zero

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \quad \text{and} \quad w = \sum_{i=1}^{n} \alpha_i y_i x_i$$

- Substitute the latter in L

Maximization problem

$$\max_{\alpha} \ \sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \ \text{ s.t } \alpha_i \geq 0, \forall i \ \text{ and } \ \sum_{i=1}^{n} \alpha_i y_i = 0$$

Linear SVM: solution

Once we have the dual problem...

1. Find the solution $\hat{\alpha}$ (quadratic function to optimize and linear constraints)
2. Compute the weights according to $\hat{w} = \sum_{i=1}^{n} \hat{\alpha}_i y_i x_i$
3. Two scenarios

$$\begin{cases} x_i \text{ is on the margin } \rightarrow \alpha_i > 0 \\ y_i(w^T x_i + b) > 1 \text{ and } \alpha_i = 0 \end{cases}$$

   Only the support vectors play a role in prediction !!!

4. Compute b knowing that $\hat{\alpha}_i > 0$ satisfy $y_i(\hat{w}^T x_i + b) = 1$

Classification function:

$$f(x) = \hat{w}^T x + b = \sum_{i=1}^{n} \hat{\alpha}_i y_i x_i^T x + b$$

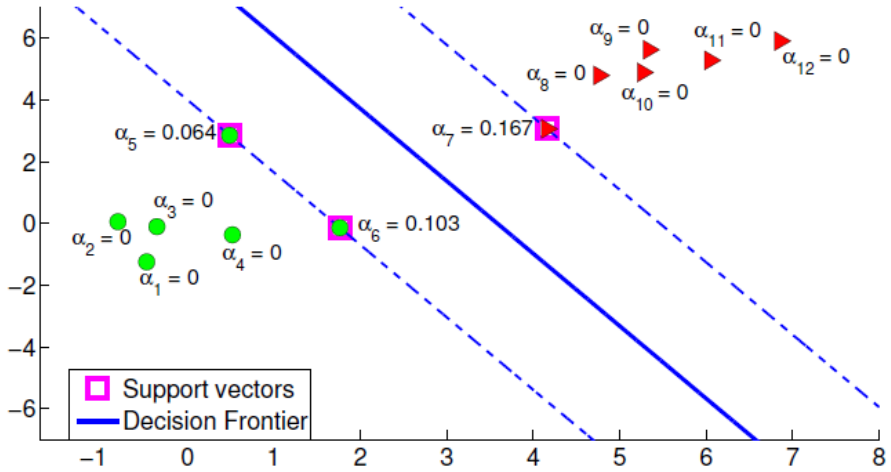
Linear SVMs: summary

In the primal problem

- Predictions are based on the learnt $(n+1)$ values of w and b
- Parametric approach

In the dual formulation...

- Only the support vectors play a role in prediction
- Central in practice because once the model is trained, a significant proportion of datapoints can be discarded



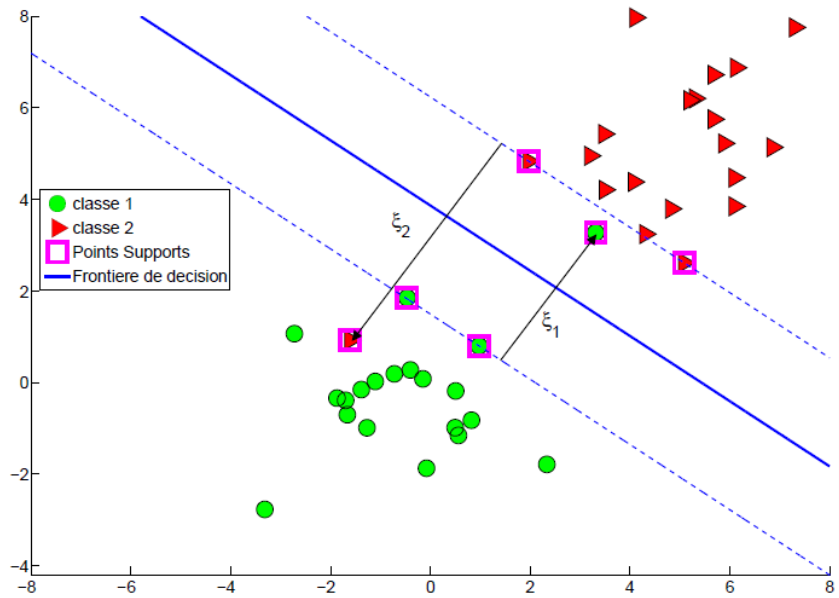
Soft SVM: The overlapping case

Key principle: If the classes are overlapping, we can't learn a perfect linear classifier

- Allow for some error or *slack* : $\xi_i \geq 0$
- The slack relaxes the classification constraint

$$y_i(w^T x_i + b) \geq 1 - \xi_i$$

- Minimize the sum of slacks $\sum_{i=1}^{n} \xi_i$



Soft SVM: Optimization problem

New optimization problem

$$\min_{w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \quad \text{s.t} \quad y_i(w^T x_i + b) \geq 1 - \xi_i,\ \xi_i \geq 0\ \forall i$$

- C controls the trade-off between slack errors and margin maximization
  $\rightarrow$ user defined

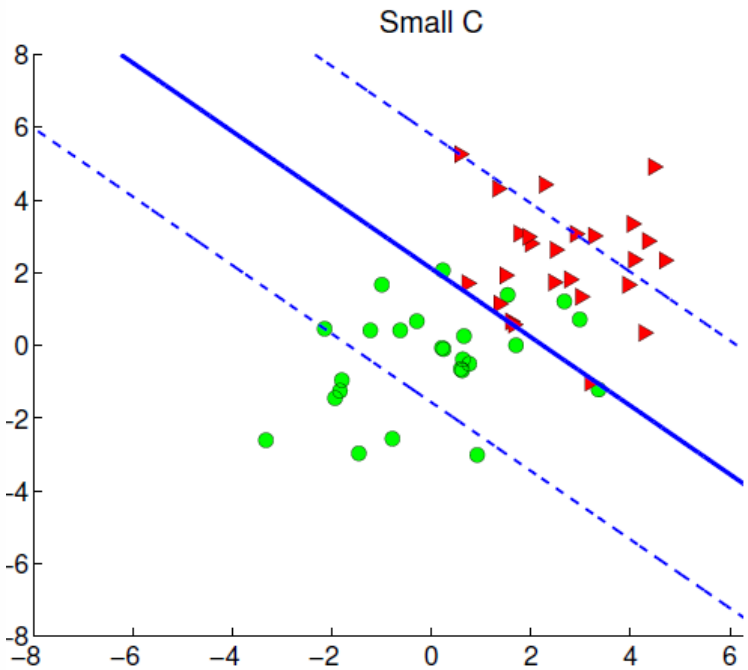
Dual problem (after similar computations...)

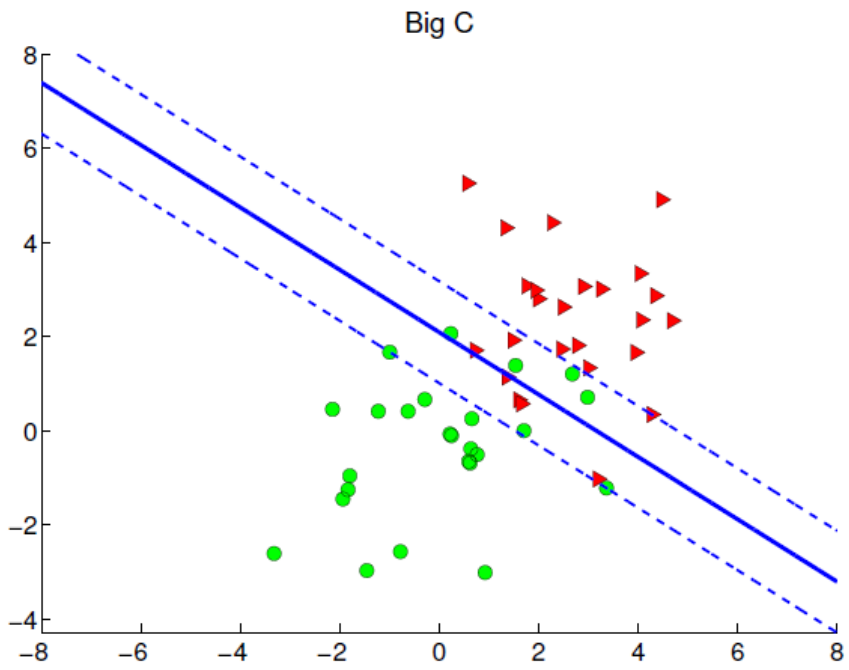
$$\max_{\alpha} \sum_{i=1}^{n}\alpha_i - \sum_{i=1}^{n}\alpha_i\alpha_j y_i y_j x_i^T x_j$$
$$\text{s.t } 0 \leq \alpha_i \leq C, \forall i \quad \text{and} \quad \sum_{i=1}^{n}\alpha_i y_i = 0$$

Soft SVM : Examples

Influence of C

Larger C values penalize the slack more $\rightsquigarrow$ narrow margin





Today's Lecture

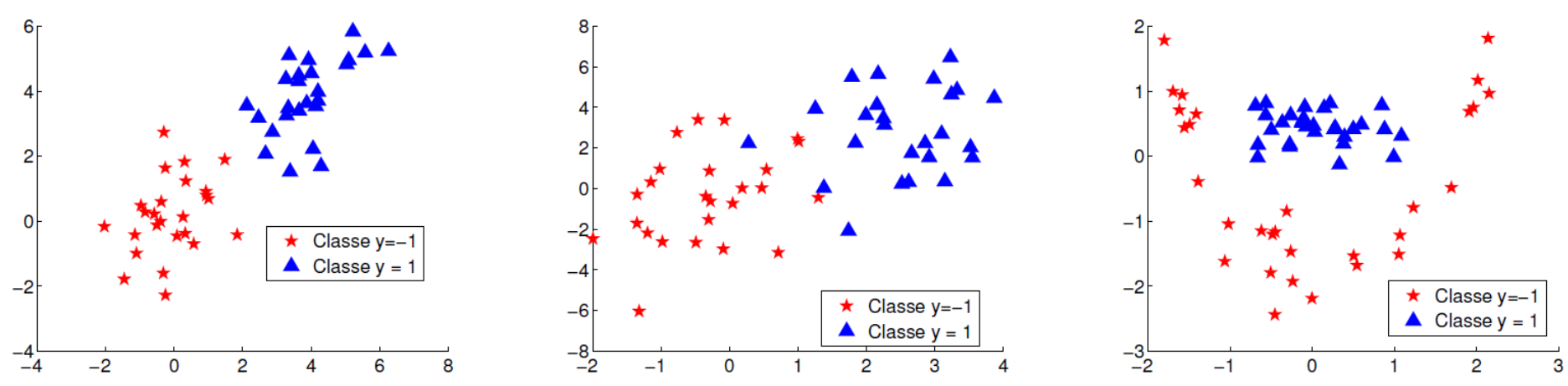
I. Classification
- Reminders on linear SVMs
- SVM - Handling non-linear boundaries: Kernel Machines

II. Introduction to clustering

III. Reminders on Clustering
- Types of methods and clusters
- Distance and Dissimilarity
- Clustering Quality

IV. Clustering algorithms
- K-means
- Hierarchical clustering
- DBSCAN
- HDBSCAN

Non-linear Boundaries



Linear SVM limitations

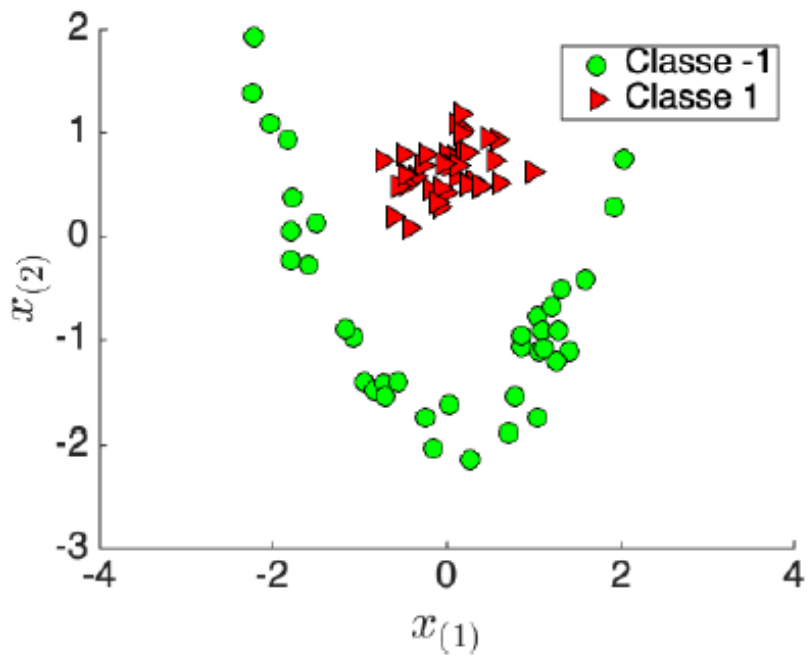
- The decision boundary is not always linear
- Data are not always vectors (e.g., string, time series, graphs, images ...)

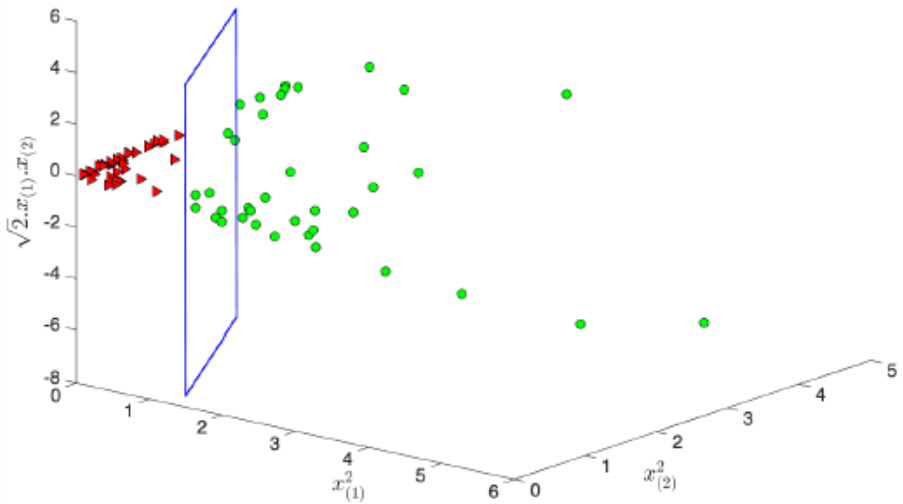
Higher Dimensional Embedding

Key Idea: Data might be linearly separable in a higher dimensional space

- Use a non-linear embedding $\Phi(x) : \mathbb{R}^p \mapsto \mathbb{R}^q$
- Train the SVM using pairs $(\Phi(x_i), y_i)$

(Non-linear transformation $\rightsquigarrow$ linear separabilty)





Example

Consider the binary case

The classes $\mathscr{C}_1 = \{(1,1),(-1,-1)\}$ and $\mathscr{C}_2 = \{(1,-1),(-1,1)\}$ are not linearly separable. Consider the application Φ defined by

$$\Phi : \begin{cases} \mathbb{R}^2 \mapsto \mathbb{R}^6 \\ (x_1, x_2) \mapsto (\sqrt{2}x_1, \sqrt{2}x_1x_2, 1, \sqrt{2}x_2, x_1^2, x_2^2) \end{cases}$$

The data are separable in the plane (Φ_1, Φ_2)

Non-linear SVM: Kernels

The decision function is now

$$f(x) = w^T \Phi(x) + b = \sum_{SV} \alpha_i y_i \Phi(x_i)^T \Phi(x)$$

The kernel trick

- Exploit the inner product in the dual formulation of SVM
- Define a function $k(.,.) : \chi \times \chi \mapsto \mathbb{R}$ (similarity in implicit higher dimensional space)
- Replace the inner product between samples by the kernel k
- Independent of the implicit feature dimension!
- $\rightsquigarrow$ reduce computational cost from $O(n^3), O(n^2)$ to $O(n)$ using $k(x,y) = \Phi(x)^T \Phi(y)$

Non-linear SVMs: Kernels

A kernel k is a function $k(.,.) : \chi \times \chi \mapsto \mathbb{R}$ such that

$$k(x, y) = \langle \Phi(x), \Phi(y) \rangle$$

What are the conditions on k?

The kernel must be positive-definite to ensure a well-defined dual problem

1. Symmetric $k(x, y) = k(y, x)$
2. And for any positive integer n

$$\forall \alpha_i \sum_i \sum_j \alpha_i^n \alpha_j^n k(x_i, x_j) \geq 0$$

- The associated Gram matrix $G \in \mathbb{R}^{n \times n}$ $G_{ij} = k(x_i, x_j)$ is positive definite

Common Kernels

Type	Name	$k(\mathbf{x}, \mathbf{z})$
radial	Gaussian	$\exp\left(-\frac{\|\mathbf{x}-\mathbf{z}\|^2}{2\sigma^2}\right)$
radial	Laplacian	$\exp(-\|\mathbf{x}-\mathbf{z}\|/\sigma)$
non stat.	χ^2	$\exp(-r/\sigma),\ r = \sum_k \frac{(\mathbf{x}_k-\mathbf{z}_k)^2}{\mathbf{x}_k+\mathbf{z}_k}$
projectif	polynomial	$(\mathbf{x}^\top\mathbf{z}+\sigma)^p$
projectif	cosinus	$\mathbf{x}^\top\mathbf{z}/\|\mathbf{x}\|\|\mathbf{z}\|$
projectif	correlation	$\exp\left(\frac{\mathbf{x}^\top\mathbf{z}}{\|\mathbf{x}\|\|\mathbf{z}\|}-\sigma\right)$

How to choose the right kernel?

Short answer: test it !

- Use cross-validation for the hyperparameters (polynomial order p, bandwidth σ)

Non Linear SVM: kernel formulation

With similar computations of the Lagrangian we obtain...

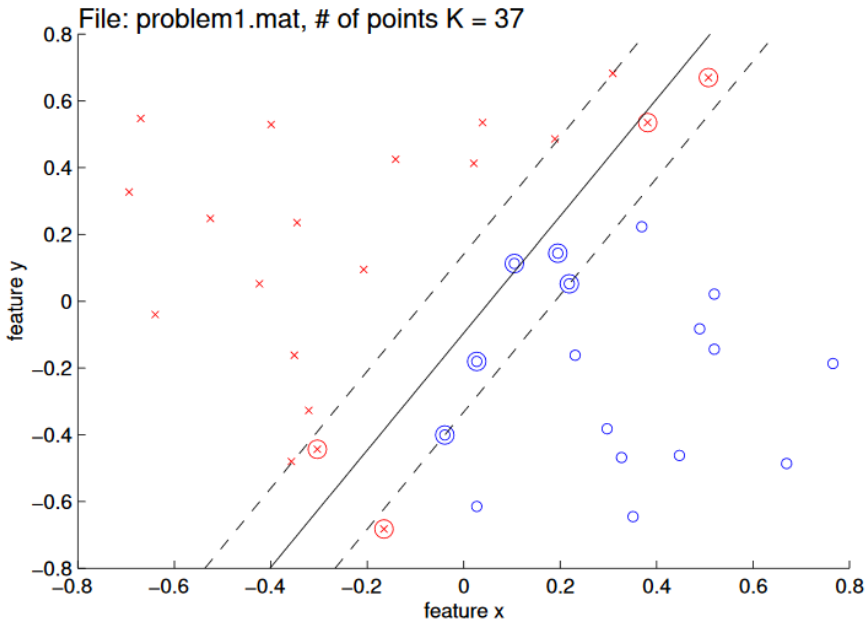
Dual problem

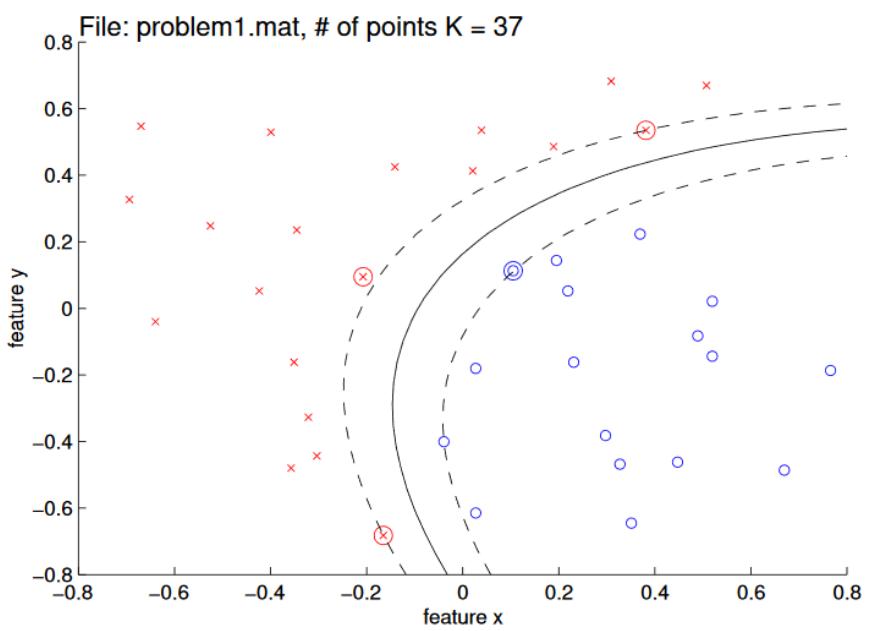
$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n} \alpha_i \alpha_j y_i y_j k(x_i, x_j)$$
$$\text{s.t } 0 \leq \alpha_i \leq C, \forall i \quad \text{and} \quad \sum_{i=1}^{n} \alpha_i y_i = 0$$

Classification function

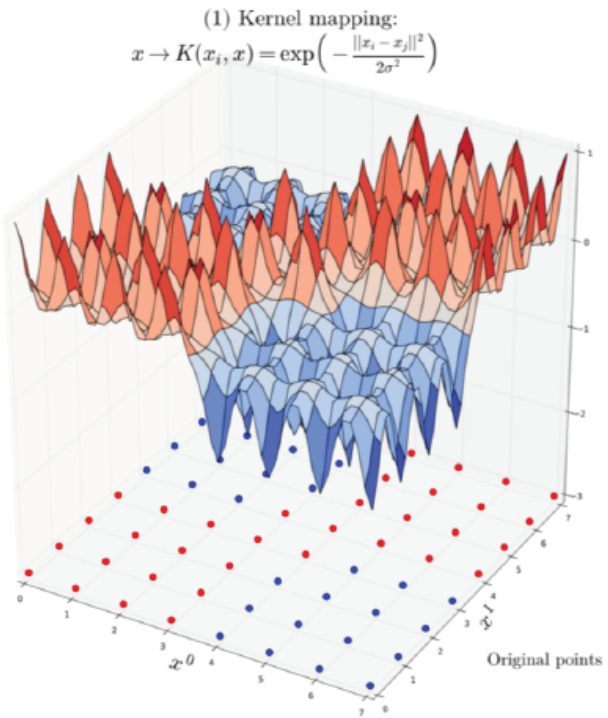
$$f(x) = \sum_{SV} \alpha_i y_i k(x_i, x)$$

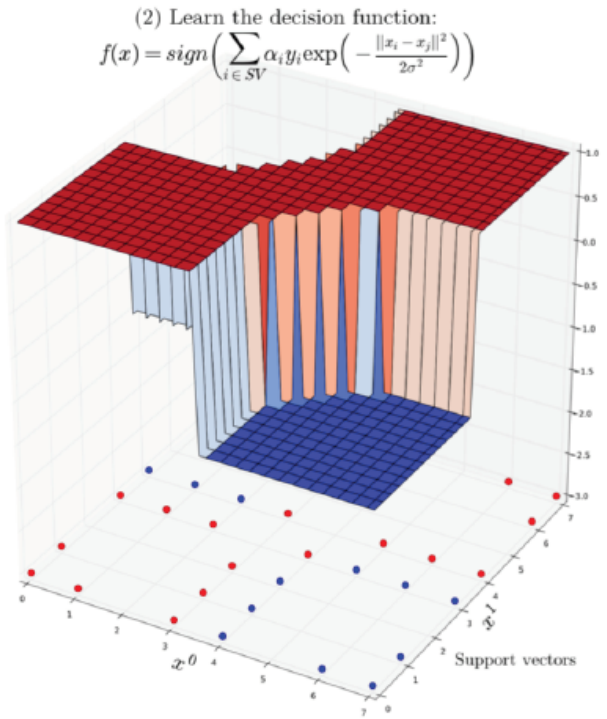
Non-linear SVM: Example

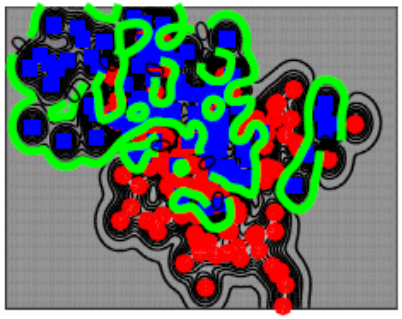




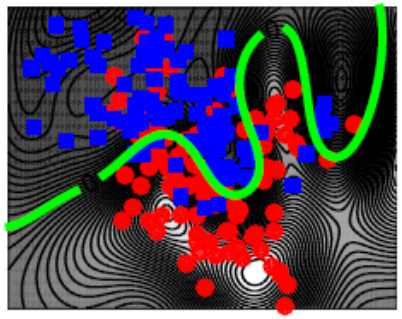
Example with Gaussian Kernel



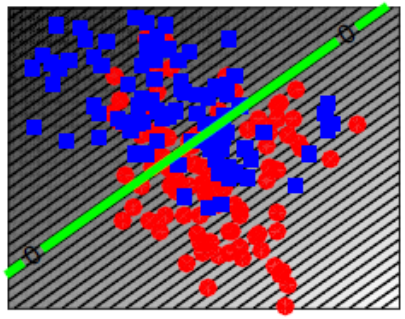




σ too small

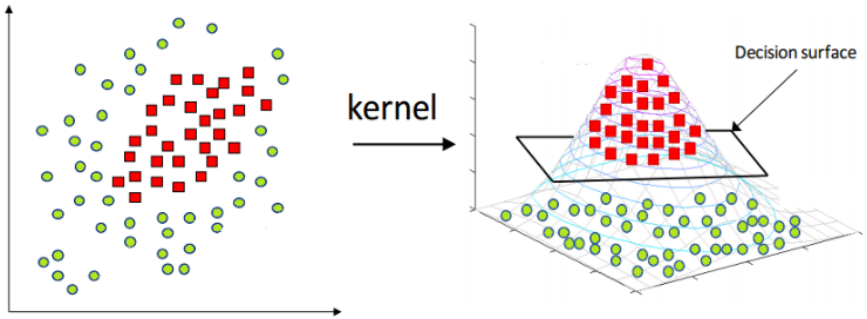


nice σ



σ too large

Non Linear SVM: Summary



- Exploit inner product in dual formulation
- No explicit representation of the non-linear embedding Φ
- Can be defined on any kind of data provided we are able to define a measure of similarity
- Need to save the support vectors : instance based approach (save data rather than parameters)
- In practice: no right way to choose the kernel, cross-validation for the hyperparameters
- In practice: small to moderate datasets

Today's Lecture

I. Classification
- Reminders on linear SVMs
- SVM - Handling non-linear boundaries: Kernel Machines

II. Introduction to clustering

III. Reminders on Clustering
- Types of methods and clusters
- Distance and Dissimilarity
- Clustering Quality

IV. Clustering algorithms
- K-means
- Hierarchical clustering
- DBSCAN
- 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, ...)

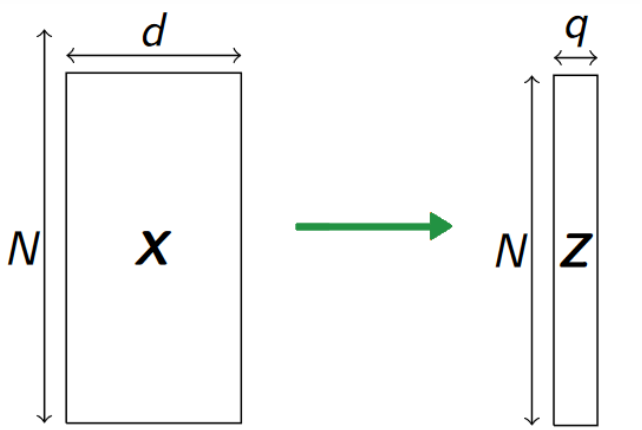
The labels are unknown!

Dimension reduction vs Clustering

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

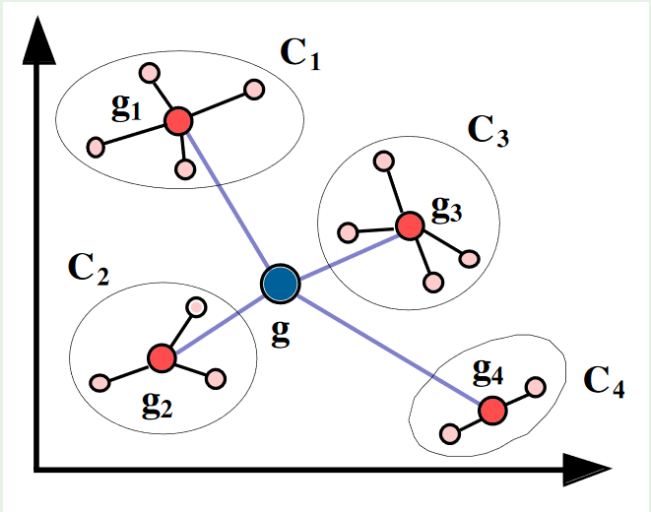
Dimension reduction

- 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 $\mathscr{D}(C_1, C_2)$



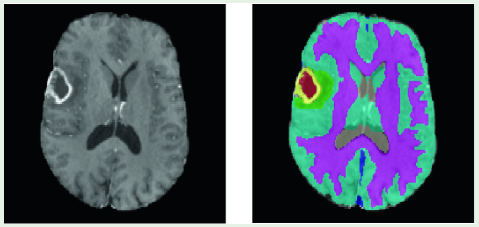
Clustering Applications

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

I. Classification
- Reminders on linear SVMs
- SVM - Handling non-linear boundaries: Kernel Machines

II. Introduction to clustering

III. Reminders on Clustering
- Types of methods and clusters
- Distance and Dissimilarity
- Clustering Quality

IV. Clustering algorithms
- K-means
- Hierarchical clustering
- DBSCAN
- HDBSCAN

Today's Lecture

I. Classification
- Reminders on linear SVMs
- SVM - Handling non-linear boundaries: Kernel Machines

II. Introduction to clustering

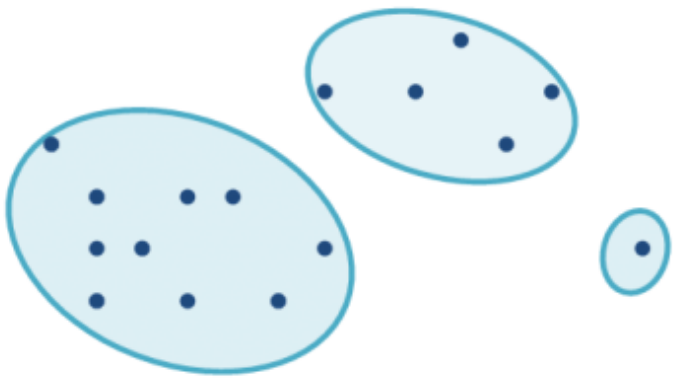
III. Reminders on Clustering
- Types of methods and clusters
- Distance and Dissimilarity
- Clustering Quality

IV. Clustering algorithms
- K-means
- Hierarchical clustering
- DBSCAN
- HDBSCAN

Types of clustering: Partitional vs Hierarchical

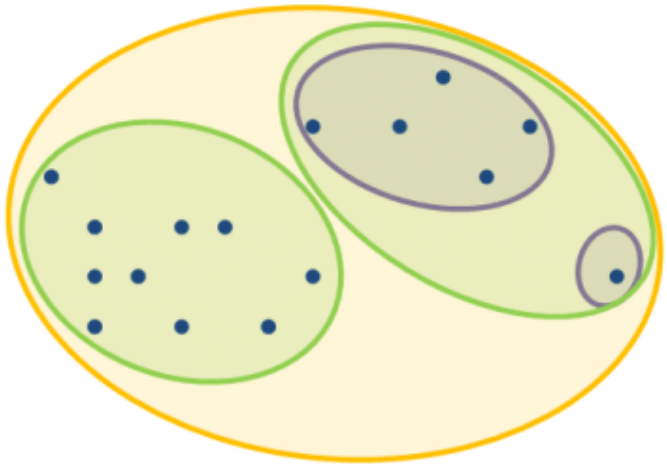
Partitional

- 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 $\mathscr{C}_k$ 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...

Today's Lecture

I. Classification
- Reminders on linear SVMs
- SVM - Handling non-linear boundaries: Kernel Machines

II. Introduction to clustering

III. Reminders on Clustering
- Types of methods and clusters
- Distance and Dissimilarity
- Clustering Quality

IV. Clustering algorithms
- K-means
- Hierarchical clustering
- DBSCAN
- HDBSCAN

Dissimilarity Measures

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

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

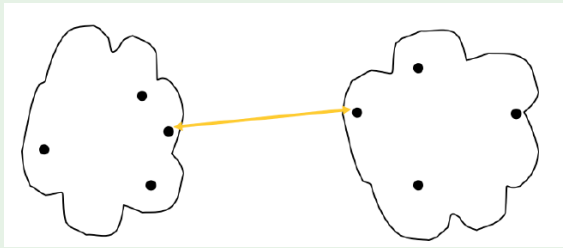
Distance is a dissimilarity measure that satisfies also

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

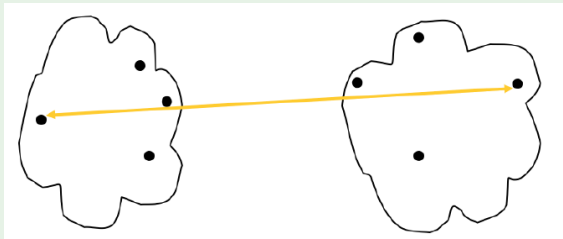
Common distances

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

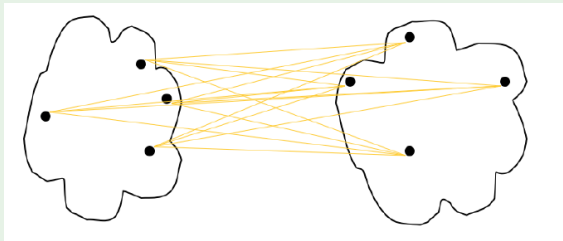
Minimum : $\mathscr{D}(\mathscr{C}_i, \mathscr{C}_j) = \min_{\mathbf{x}\in\mathscr{C}_i, \mathbf{y}\in\mathscr{C}_j} \mathscr{D}(\mathbf{x}, \mathbf{y})$



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

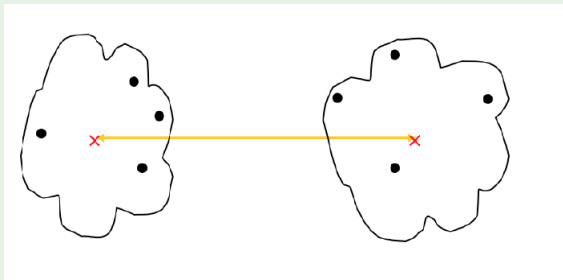


Group Average : $\mathscr{D}(\mathscr{C}_i, \mathscr{C}_j) = \frac{1}{n_i n_j} \sum_{\mathbf{x}\in\mathscr{C}_i} \sum_{\mathbf{y}\in\mathscr{C}_j} \mathscr{D}(\mathbf{x}, \mathbf{y})$



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

with $m_i = \frac{1}{n_i} \sum_{\mathbf{x}\in\mathscr{C}_i} \mathbf{x}$



Dissimilarity Between Clusters (2/2)

Objective function distances

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

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

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

Today's Lecture

I. Classification
- Reminders on linear SVMs
- SVM - Handling non-linear boundaries: Kernel Machines

II. Introduction to clustering

III. Reminders on Clustering
- Types of methods and clusters
- Distance and Dissimilarity
- Clustering Quality

IV. Clustering algorithms
- K-means
- Hierarchical clustering
- DBSCAN
- HDBSCAN

What makes a good clustering ?

- Centroid: $m_i = \frac{1}{n_i} \sum_{\mathbf{x} \in \mathcal{C}_i} \mathbf{x}$
- Inertia: $J_i = \sum_{\mathbf{x} \in \mathcal{C}_i} \mathcal{D}^2(xg, m_i)$

  (low J_i corresponds to a smaller dispersion of points around m_i.)
- Within distance: $J_w = \sum_i \sum_{\mathbf{x} \in \mathcal{C}_i} \mathcal{D}^2(xg, m_i) = \sum_i J_i$
- Between distance: $J_b = \sum_i n_i \mathcal{D}^2(m_i, m)$

  where m is the sample mean $m = \frac{1}{n} \sum \mathbf{x}$
- Performance measures: accuracy (when ground truth is known), ARI (Adjusted Rand Index), AMI (Adjusted Mutual Information)...

A good clustering...

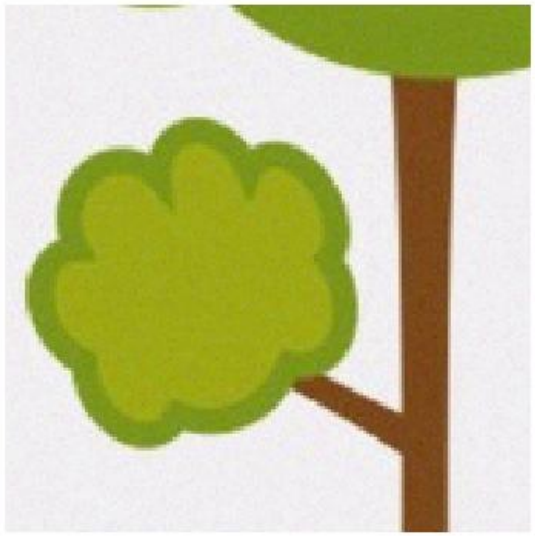
Minimizes the within distance J_w and maximizes the between distance J_b

Illustrative example

Objective

Cluster noisy data for a segmentation application in image processing

(a) Tree data



(b) Noisy tree data

Figure: Data on which the clustering algorithms are evaluated

Should be easy...

Today's Lecture

I. Classification
- Reminders on linear SVMs
- SVM - Handling non-linear boundaries: Kernel Machines

II. Introduction to clustering

III. Reminders on Clustering
- Types of methods and clusters
- Distance and Dissimilarity
- Clustering Quality

IV. Clustering algorithms
- K-means
- Hierarchical clustering
- DBSCAN
- HDBSCAN

Today's Lecture

I. Classification
- Reminders on linear SVMs
- SVM - Handling non-linear boundaries: Kernel Machines

II. Introduction to clustering

III. Reminders on Clustering
- Types of methods and clusters
- Distance and Dissimilarity
- Clustering Quality

IV. Clustering algorithms
- K-means
- Hierarchical clustering
- DBSCAN
- HDBSCAN

K-means

It is a prototype-based clustering technique.

Notations: n unlabelled data vectors of $\mathbb{R}^p$ denoted as $\mathbf{x} = (\mathbf{x}_1, ..., \mathbf{x}_n)$ which should be split into K classes $\mathscr{C}_1, ..., \mathscr{C}_K$, 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
   $C_k = \{\mathbf{x}_i, \ \forall i \in \{1, ..., n\} \mid \ d(\mathbf{x}_i, m_k) \leq d(\mathbf{x}_i - m_j) \ , \forall j \in \{1, ..., K\} \ \}$
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 and alternatives

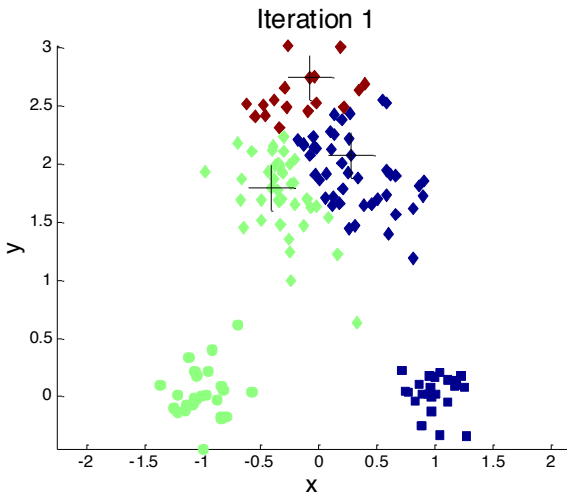
K-means is simple but ...

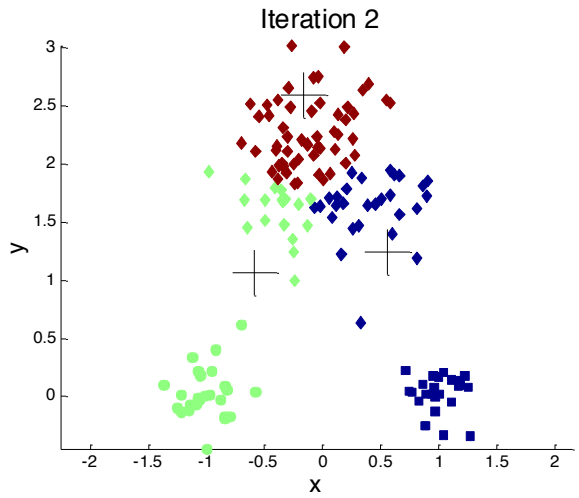
- Solution depends on initialization
- Need to know K in advance
- Can't handle noise or outliers : *non-robust*
- Fails with clusters of *non-convex* shapes

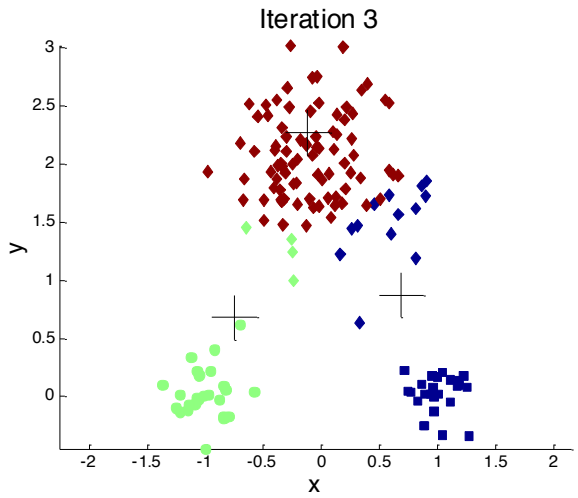
Several alternatives

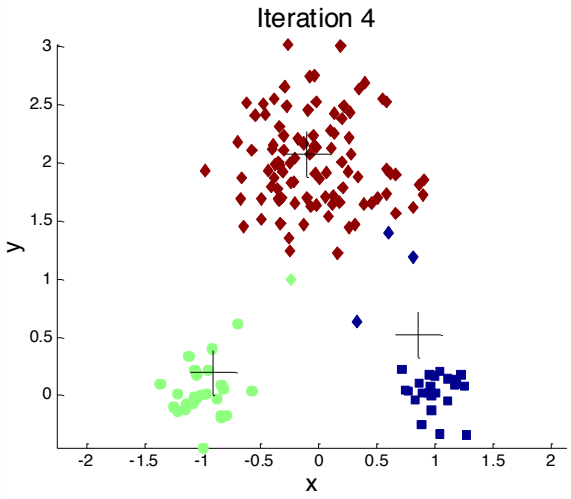
- K-means++: Seeding algorithm to initialize clusters with centroids "spread-out" throughout the data
- K-medoids: To address the robustness aspects
- Kernel K-means: For overcoming the convex shape
- Many others ...

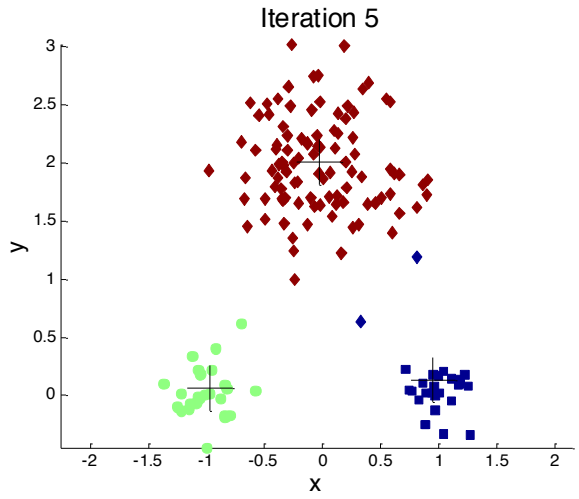
Correct initilization

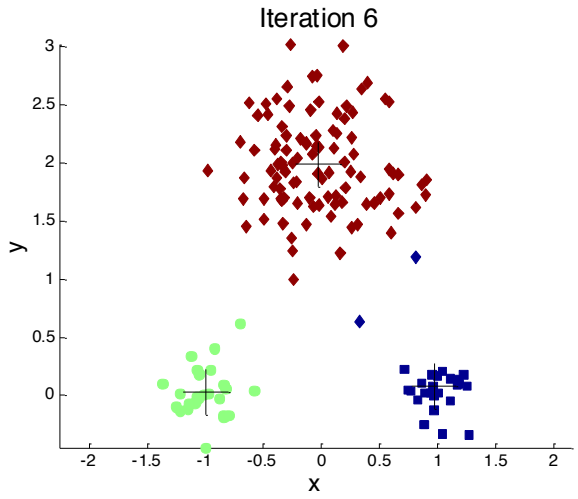




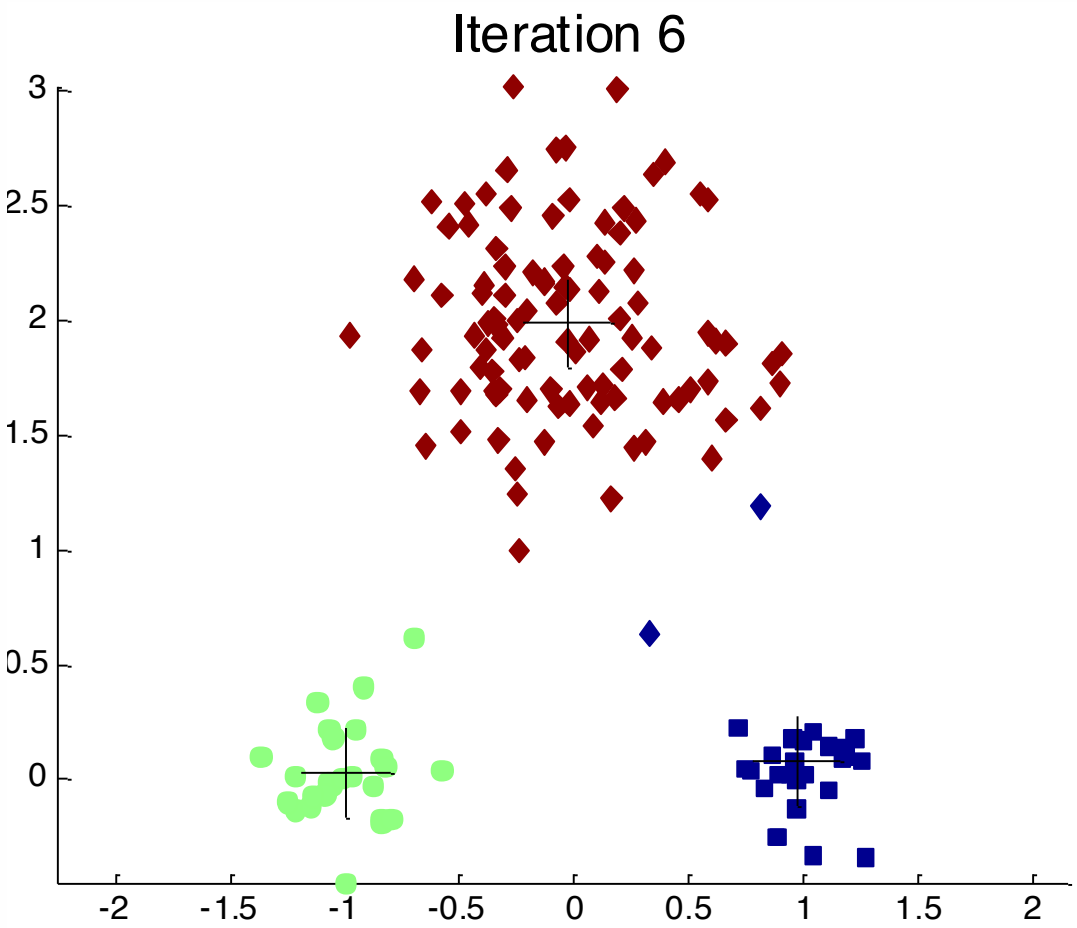




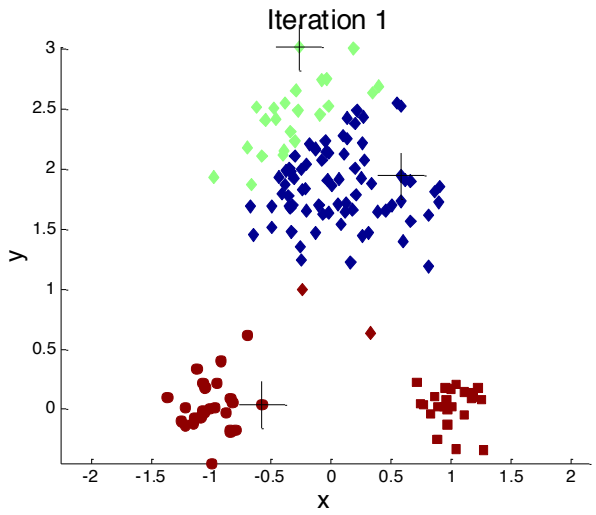


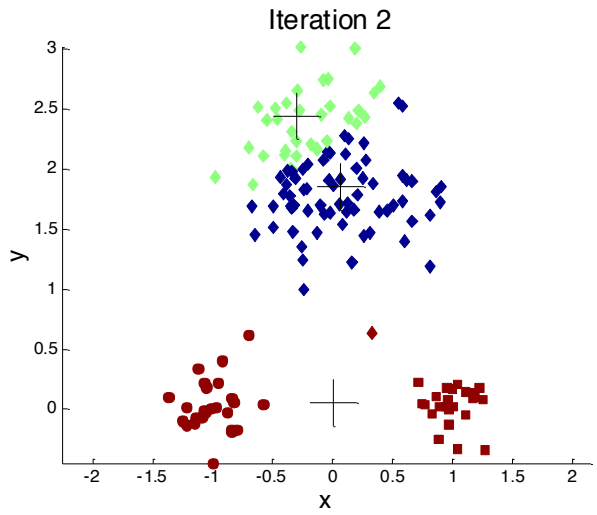


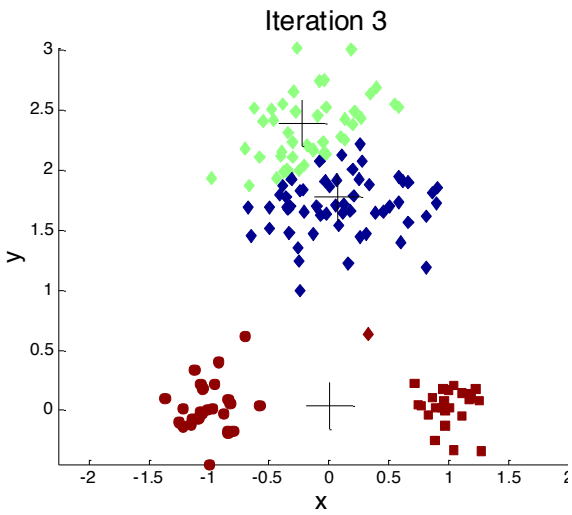
Correct initilization

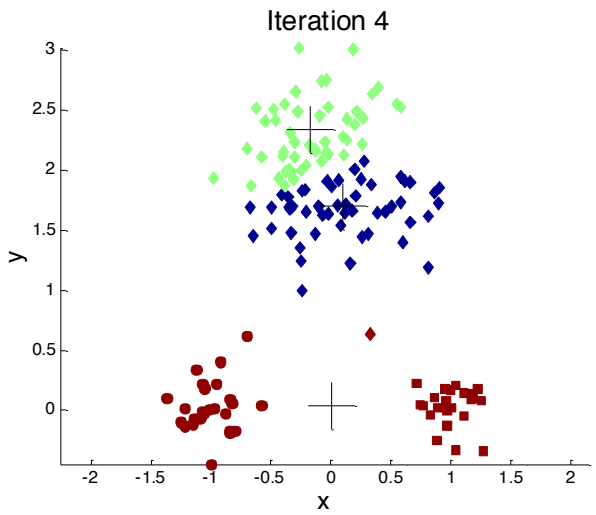


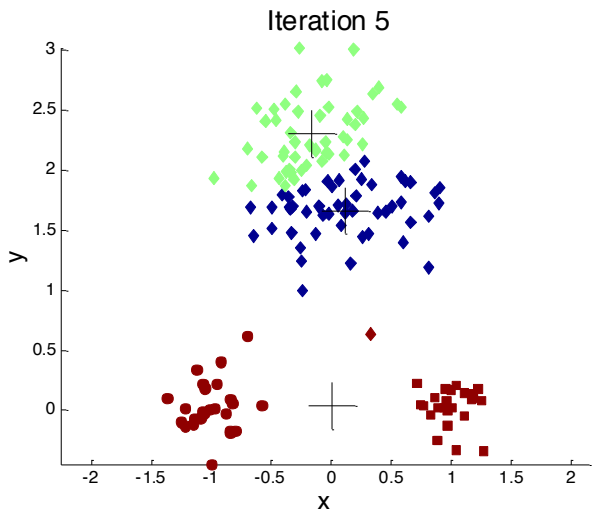
Bad initialization



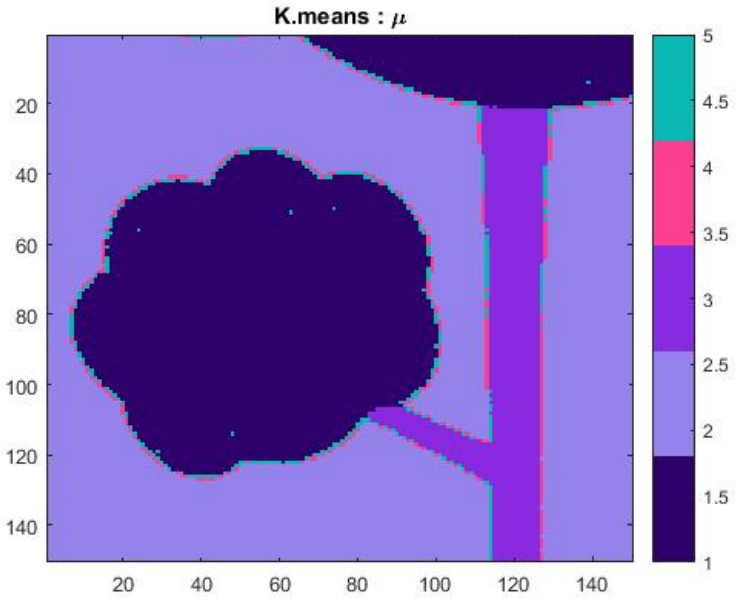




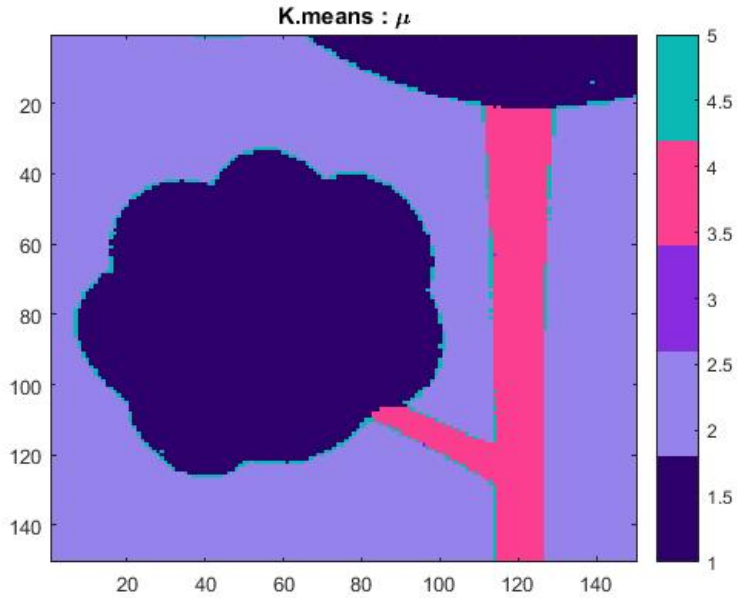




Results on the data set



(a) K-means++



(b) "Clusters"

Figure: Clustering obtained with two different initialization techniques

Comments...

Today's Lecture

I. Classification
- Reminders on linear SVMs
- SVM - Handling non-linear boundaries: Kernel Machines

II. Introduction to clustering

III. Reminders on Clustering
- Types of methods and clusters
- Distance and Dissimilarity
- Clustering Quality

IV. Clustering algorithms
- K-means
- Hierarchical clustering
- DBSCAN
- HDBSCAN

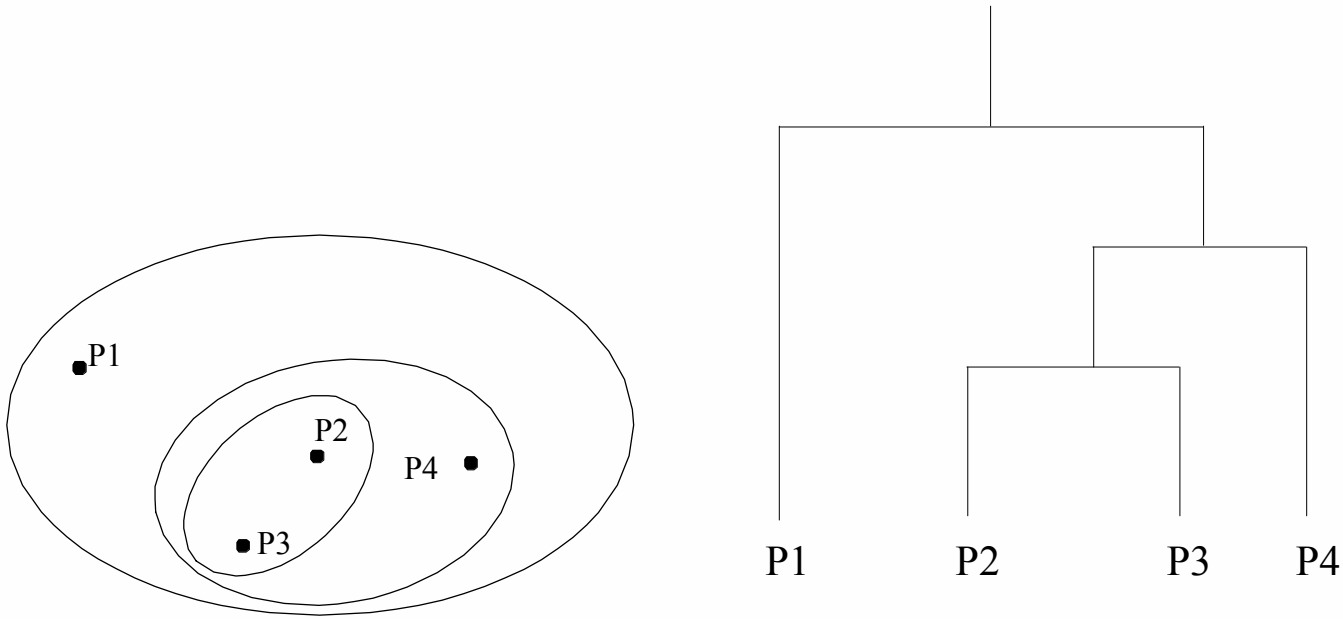
Hierarchical clustering Principles

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

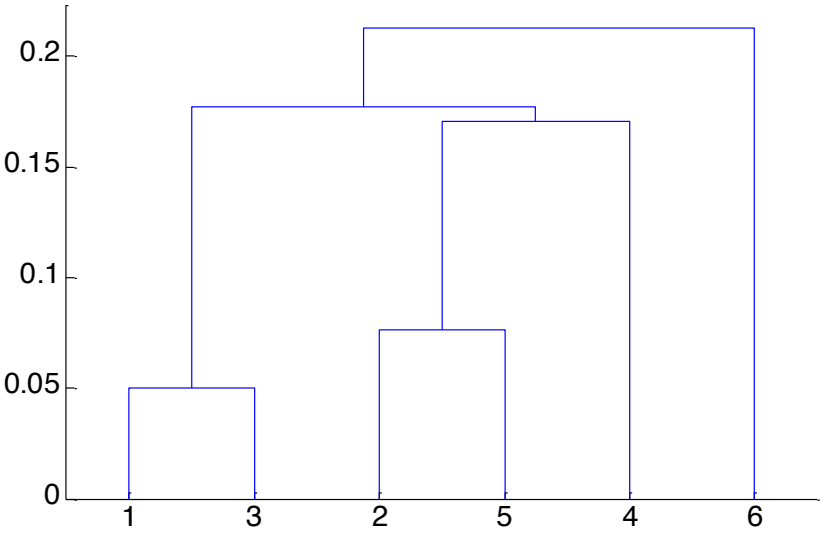


(a) Hierarchical Clusters (b) Dendrogram

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



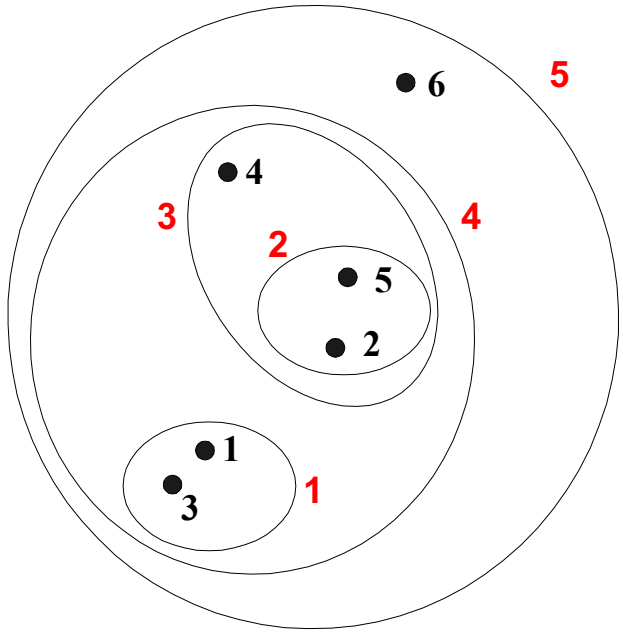


Figure: General principles

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.

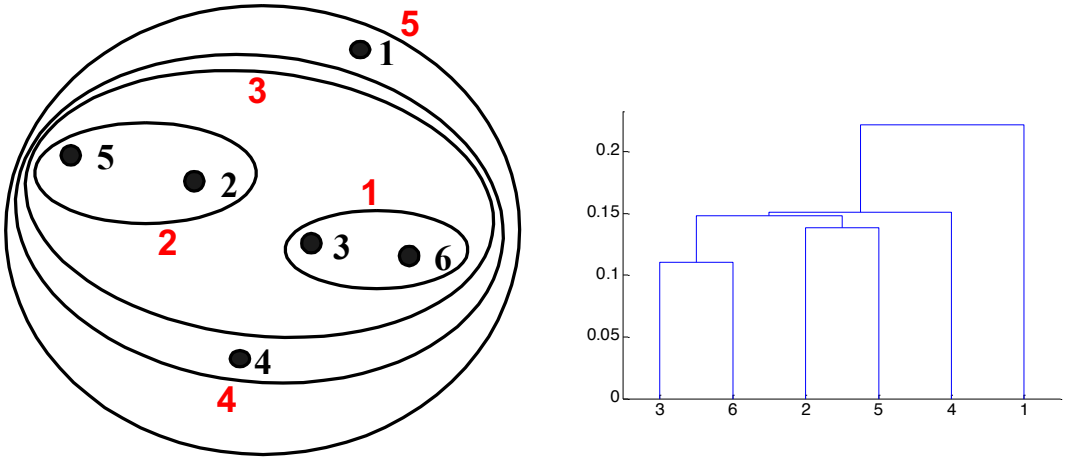
While Number of clusters > 1

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

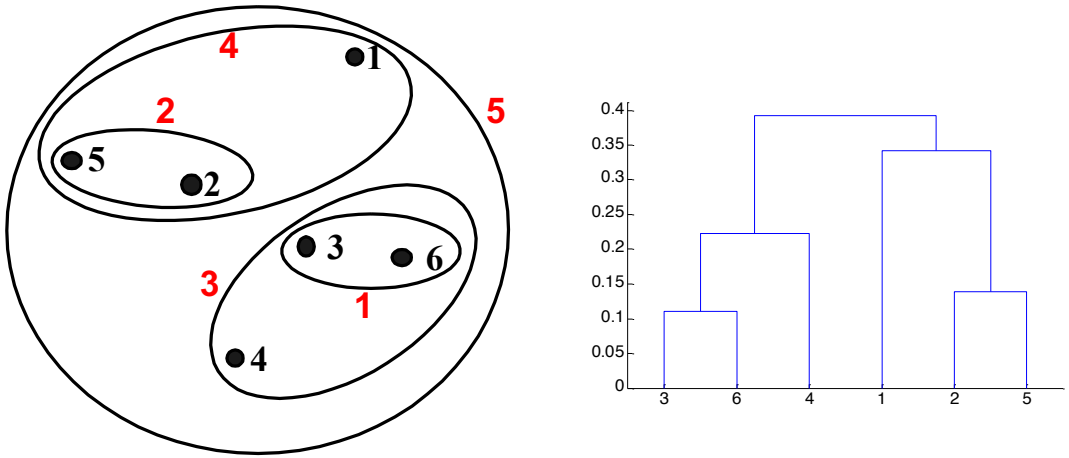
Inter-Cluster distances

- 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: $d(\mathscr{C}_i, \mathscr{C}_j) = \frac{1}{n_i n_j} \sum_{\mathbf{x}\in\mathscr{C}_i} \sum_{\mathbf{y}\in\mathscr{C}_j} d(\mathbf{x}, \mathbf{y})$
- Between centroid → Centroid Linkage: $d(\mathscr{C}_i, \mathscr{C}_j) = d(m_i, m_j)$, with $m_i = \frac{1}{n_i} \sum_{\mathbf{x}\in\mathscr{C}_i} \mathbf{x}$
- Objective function → Objective Linkage:
  - Ward distance $d(\mathscr{C}_i, \mathscr{C}_j) = \sqrt{\frac{2\, n_i\, n_j}{n_i + n_j}}\, d(m_i, m_j)$
  - WPGMA (Weighted Pair Group Method with Arithmetic Mean) recursive distance $d(\mathscr{C}_i, \mathscr{C}_j) == \frac{d(\mathscr{C}_i^1, \mathscr{C}_j) + d(\mathscr{C}_i^2, \mathscr{C}_j)}{2}$ 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

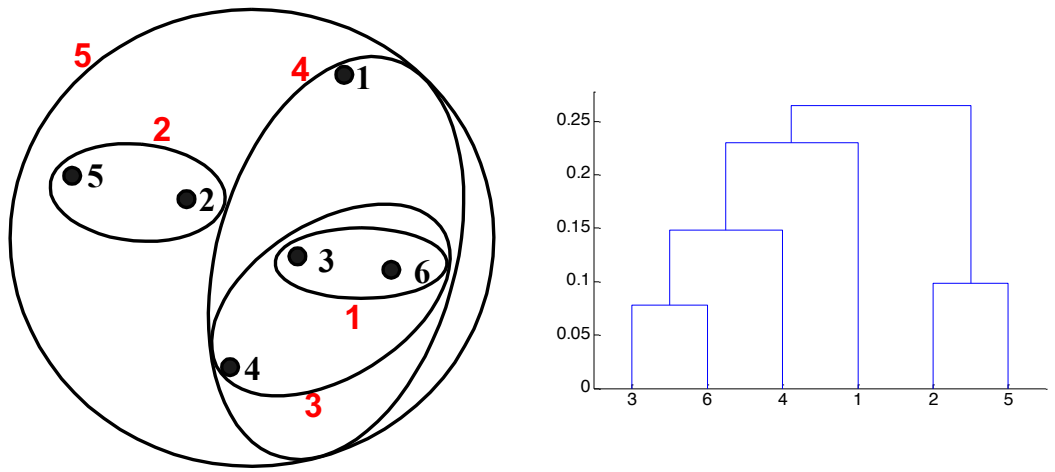
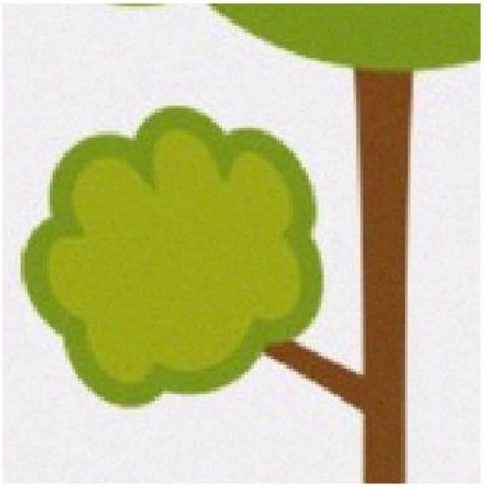


Figure: Group average

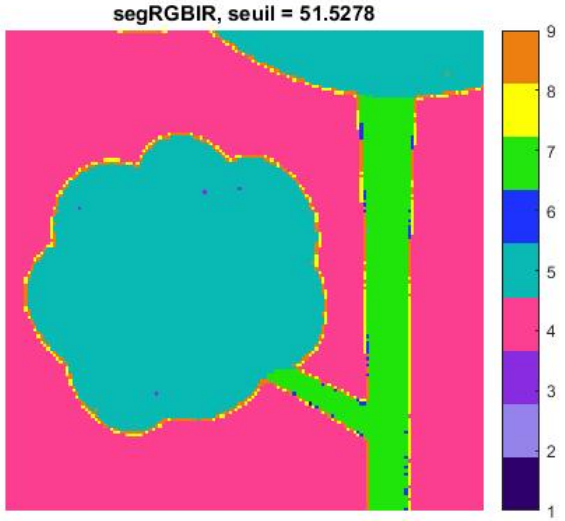
Ward: very similar results.

- 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
- Average: don't break large clusters BUT biased towards globular clusters
- Ward: Hierarchical analogue of K-means

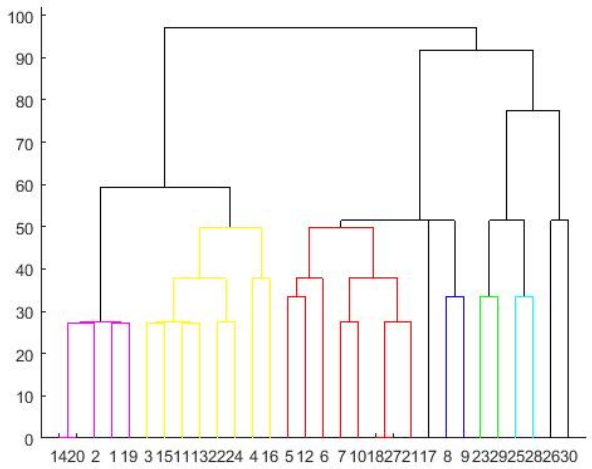
Results on the data set - Single Linkage



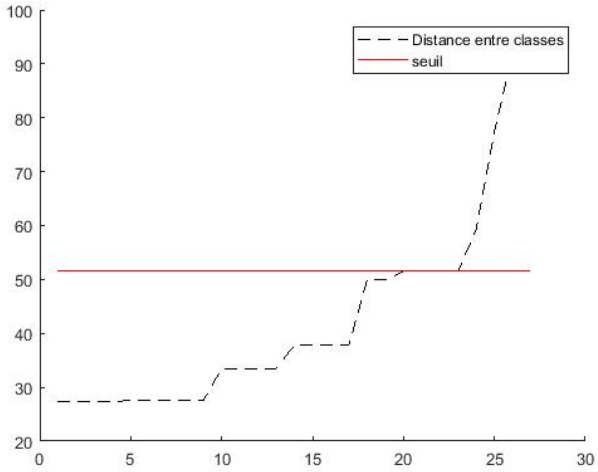
(a) Noisy Tree



(b) Single Linkage

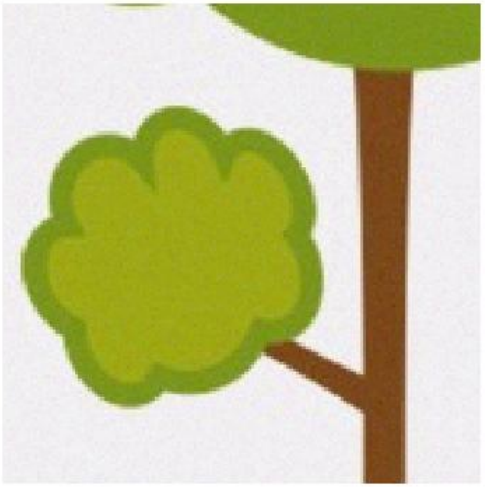


(c) Dendrogram

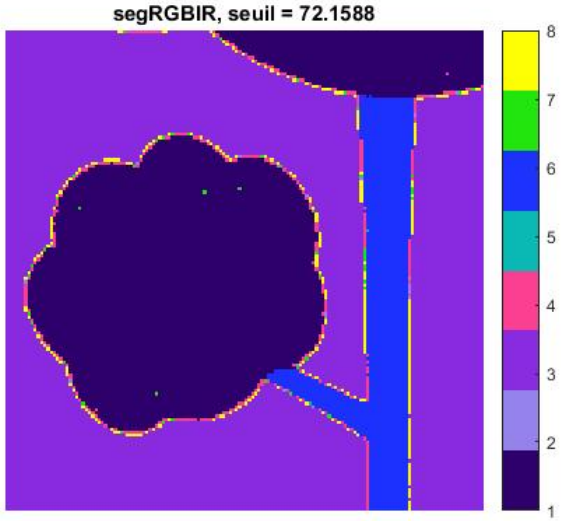


(d) Cutting Threshold

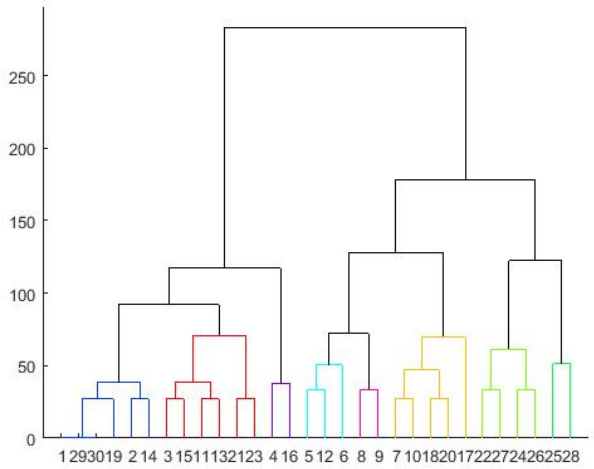
Results on the data set - Complete Linkage



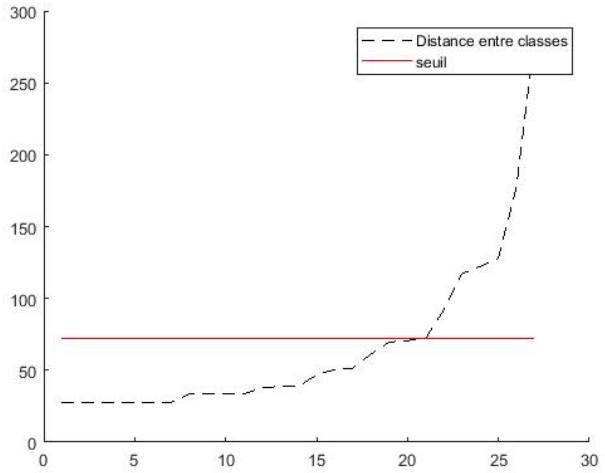
(e) Noisy Tree



(f) Complete Linkage

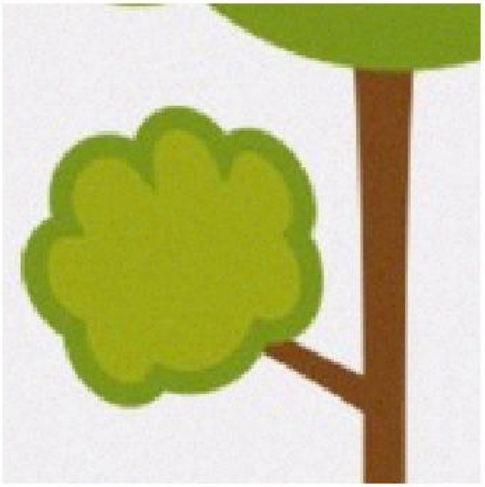


(g) Dendrogram

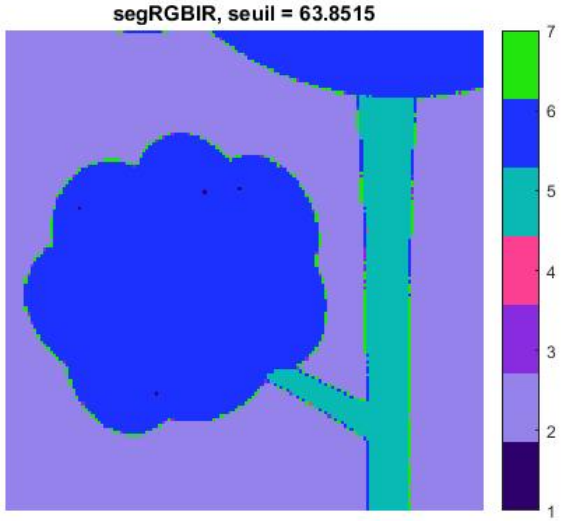


(h) Cutting Threshold

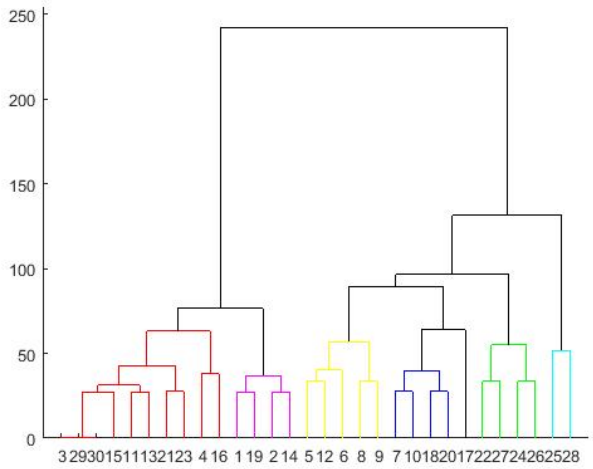
Results on the data set - Average Linkage



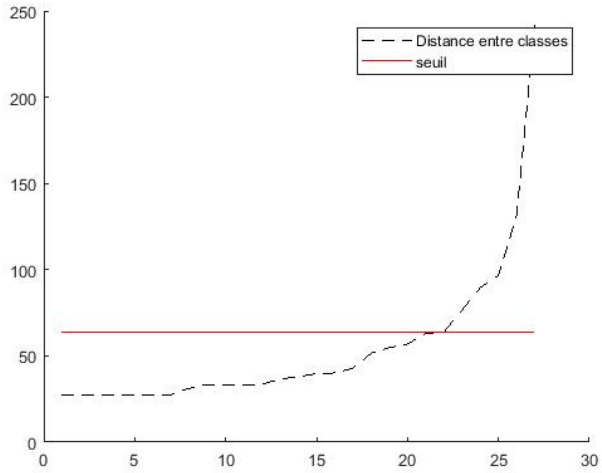
(i) Noisy Tree



(j) Average Linkage

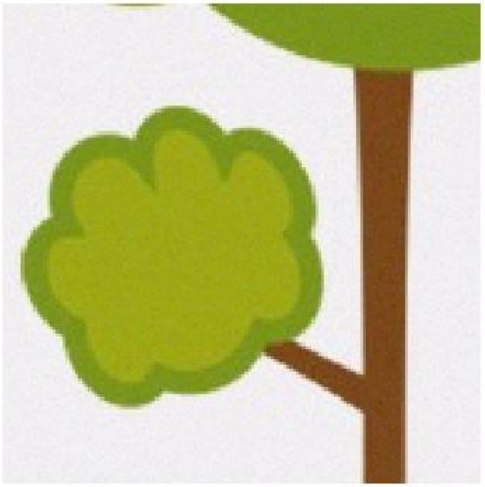


(k) Dendrogram

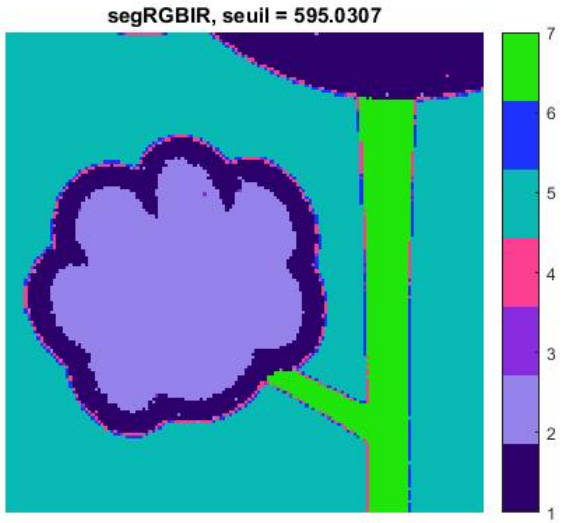


(l) Cutting Threshold

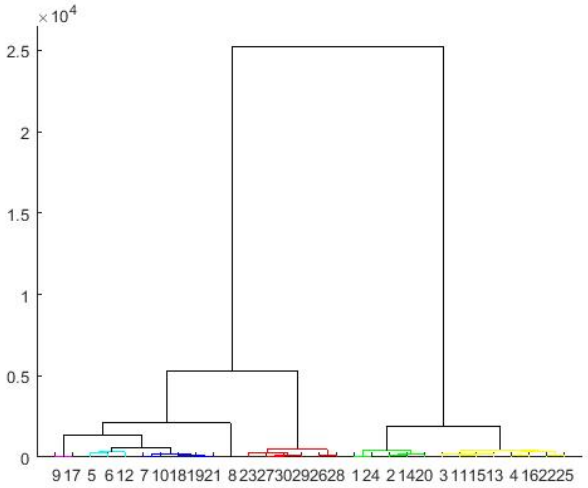
Results on the data set - Ward Linkage



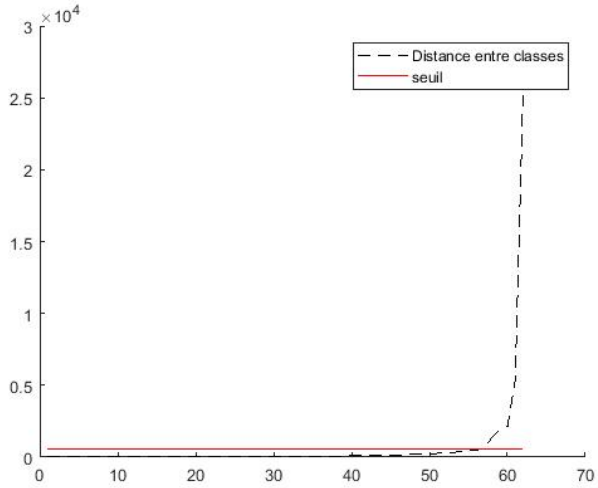
(m) Noisy Tree



(n) Average Linkage

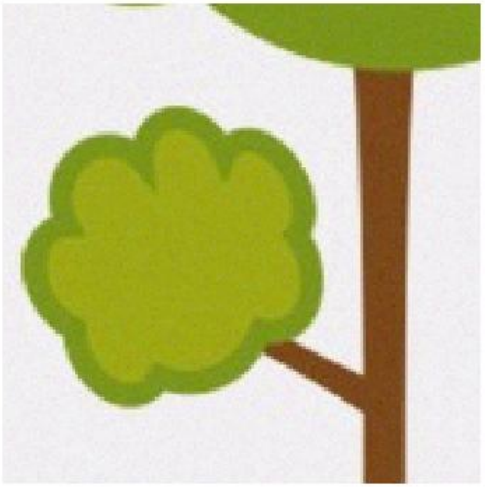


(o) Dendrogram



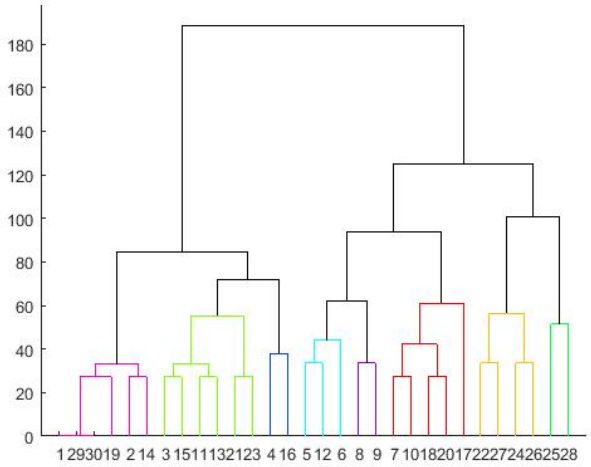
(p) Cutting Threshold

Results on the data set - WPGMA Linkage

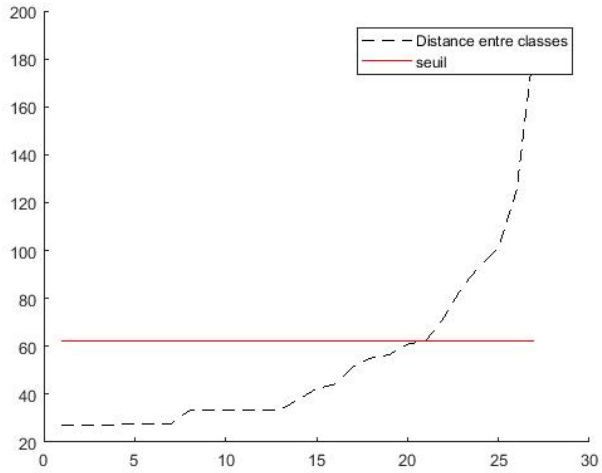


(q) Noisy Tree

(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...)

Today's Lecture

I. Classification
- Reminders on linear SVMs
- SVM - Handling non-linear boundaries: Kernel Machines

II. Introduction to clustering

III. Reminders on Clustering
- Types of methods and clusters
- Distance and Dissimilarity
- Clustering Quality

IV. Clustering algorithms
- K-means
- Hierarchical clustering
- DBSCAN
- 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 $\mathscr{C}_k$ of $\mathbf{x}_i$,
- else $\mathbf{x}_i$ is an isolated observation (outlier).

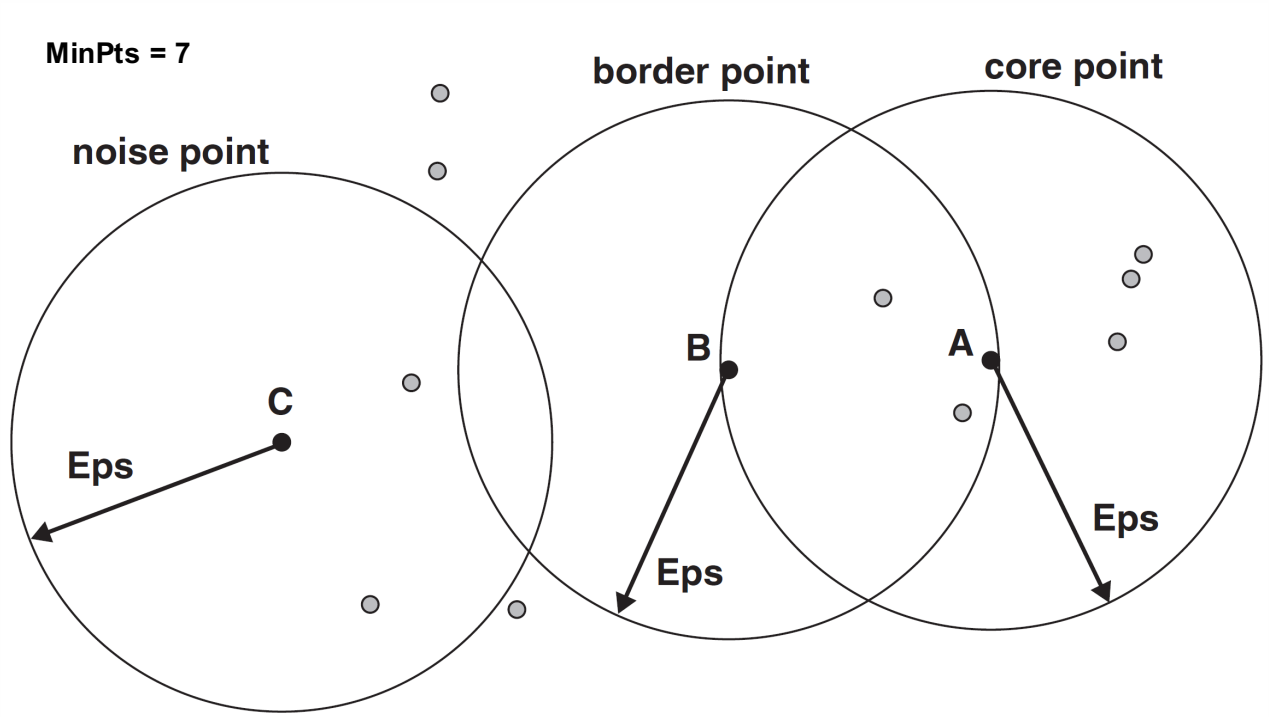
This results in three types of points called core, border, or noise points.

Key parameters

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

DBSCAN: Three Types of Points

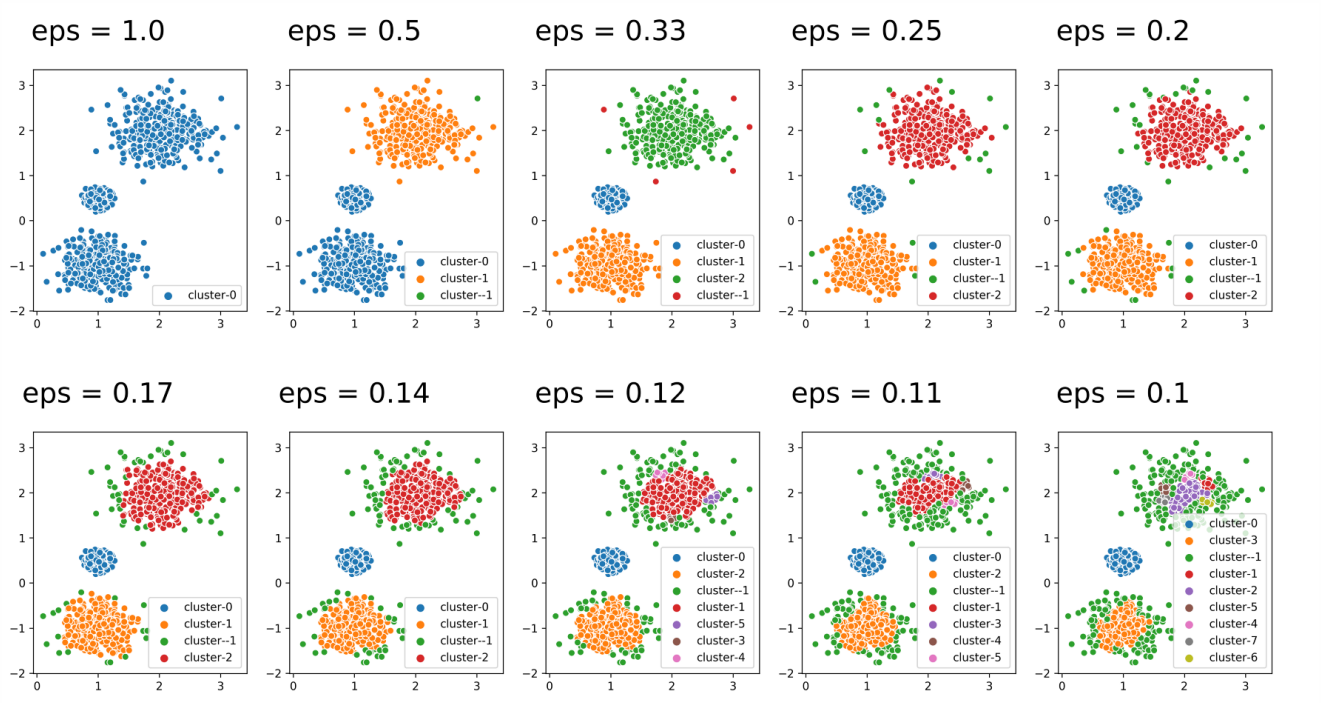
1. Core point: is near the center of a cluster/has MinPts neighbors
2. Border point: is not a core point, but is in the neighborhood of a core point
3. Noise point: is any point that is neither a core nor a border point



DBSCAN: Influence of ϵ

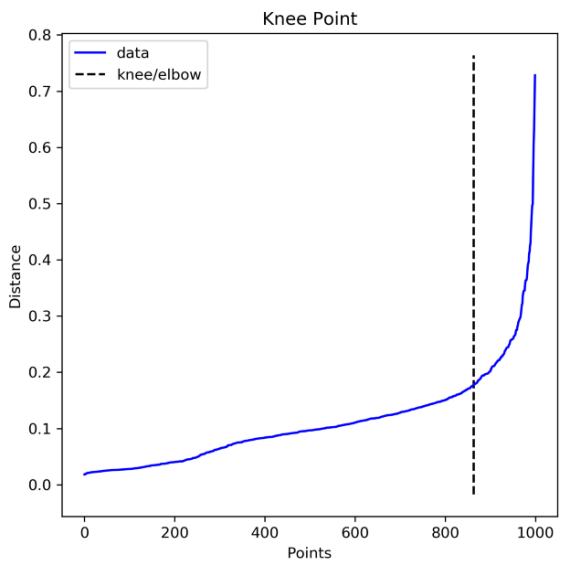
The parameter ϵ represents the minimum distance between two non-neighboring points:

- → 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!):
  - → x-axis all the points
  - → y-axis the average distance of each point to their its k-NN
- Remains a difficult choice!



DBSCAN algorithm

Algorithm 3 DBSCAN algorithm

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** $\text{card}(\mathcal{N}_\varepsilon(\mathbf{x}_i)) \leq n_{min}$, 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** $\text{card}(\mathscr{N}_\varepsilon(\mathbf{x}_j)) \geq n_{min}$
   $\mathscr{N}_\varepsilon(\mathbf{x}_i) = \mathscr{N}_\varepsilon(\mathbf{x}_i) + \mathscr{N}_\varepsilon(\mathbf{x}_j)$
4. **If** $\mathbf{x}_j$ is not clustered, add to $\mathscr{C}_k$.

Illustration of DBSCAN principles

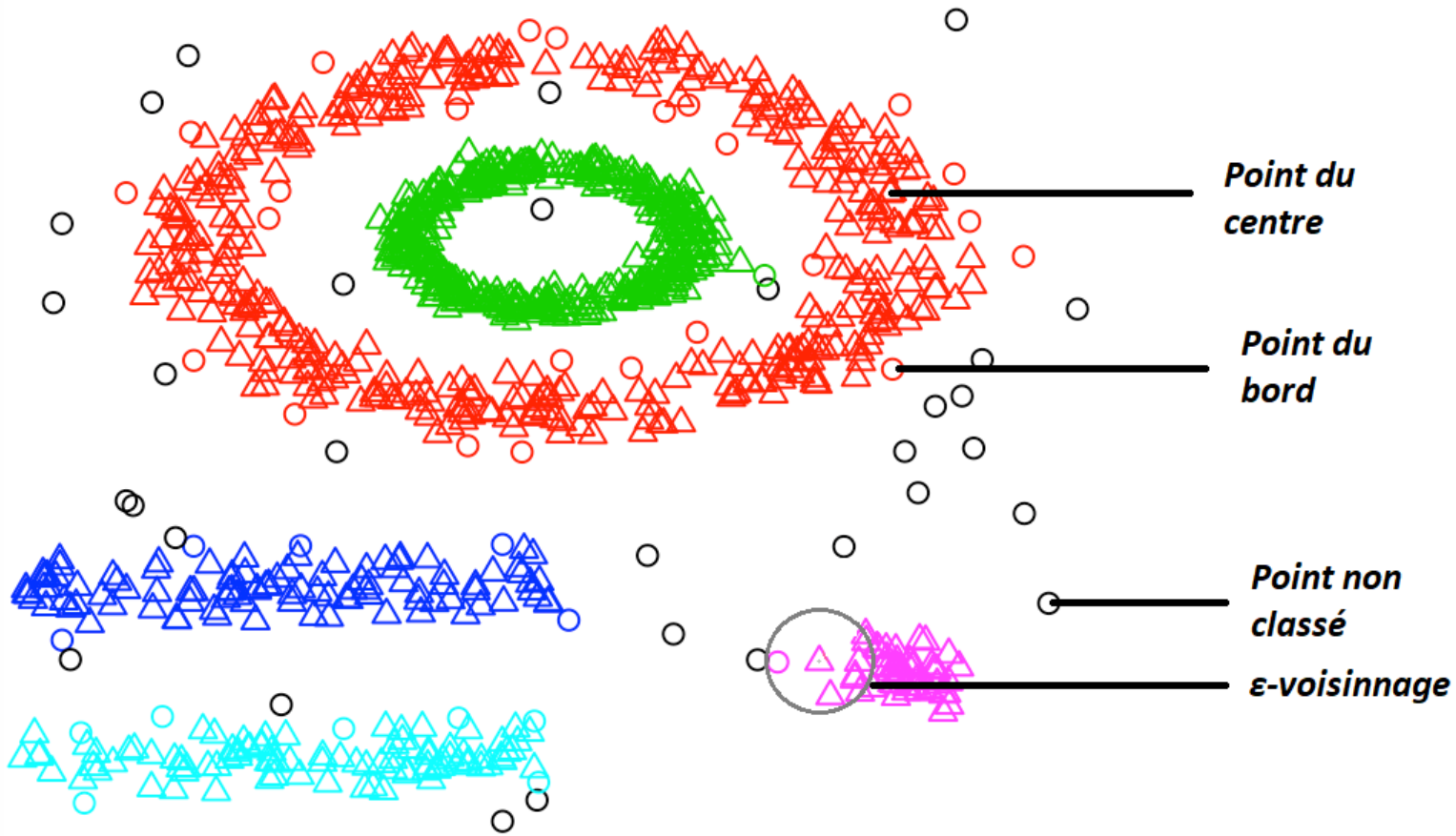
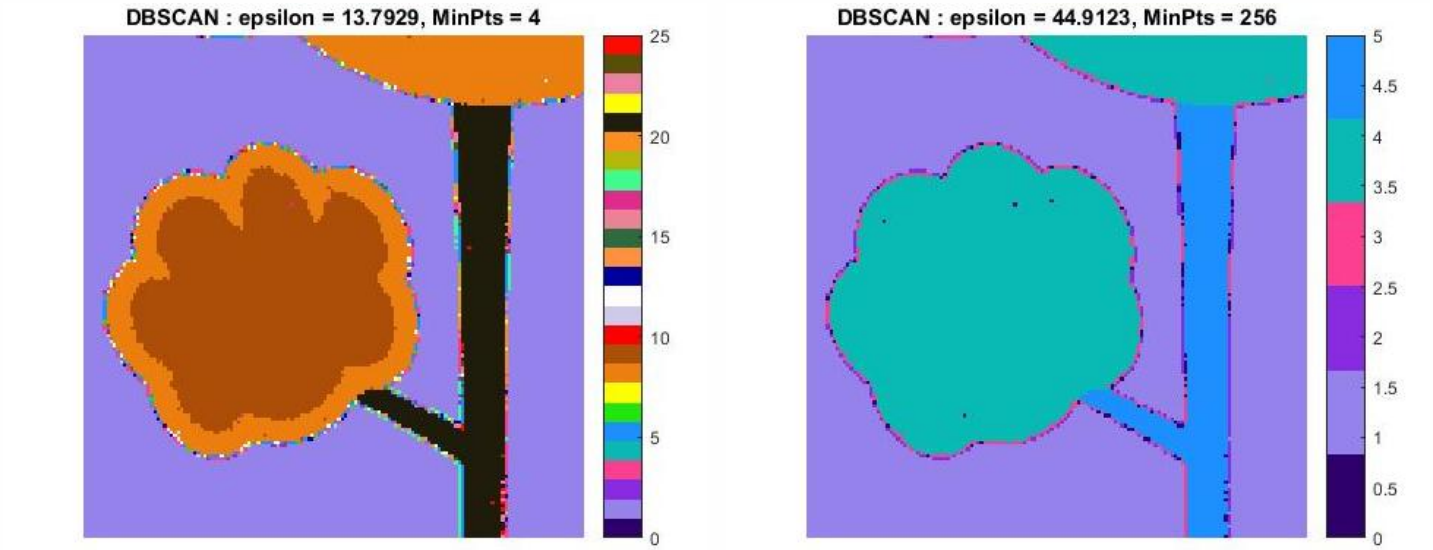


Figure: Clustering results obtained with DBSCAN algorithm.

Results on the data set - DBSCAN



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

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

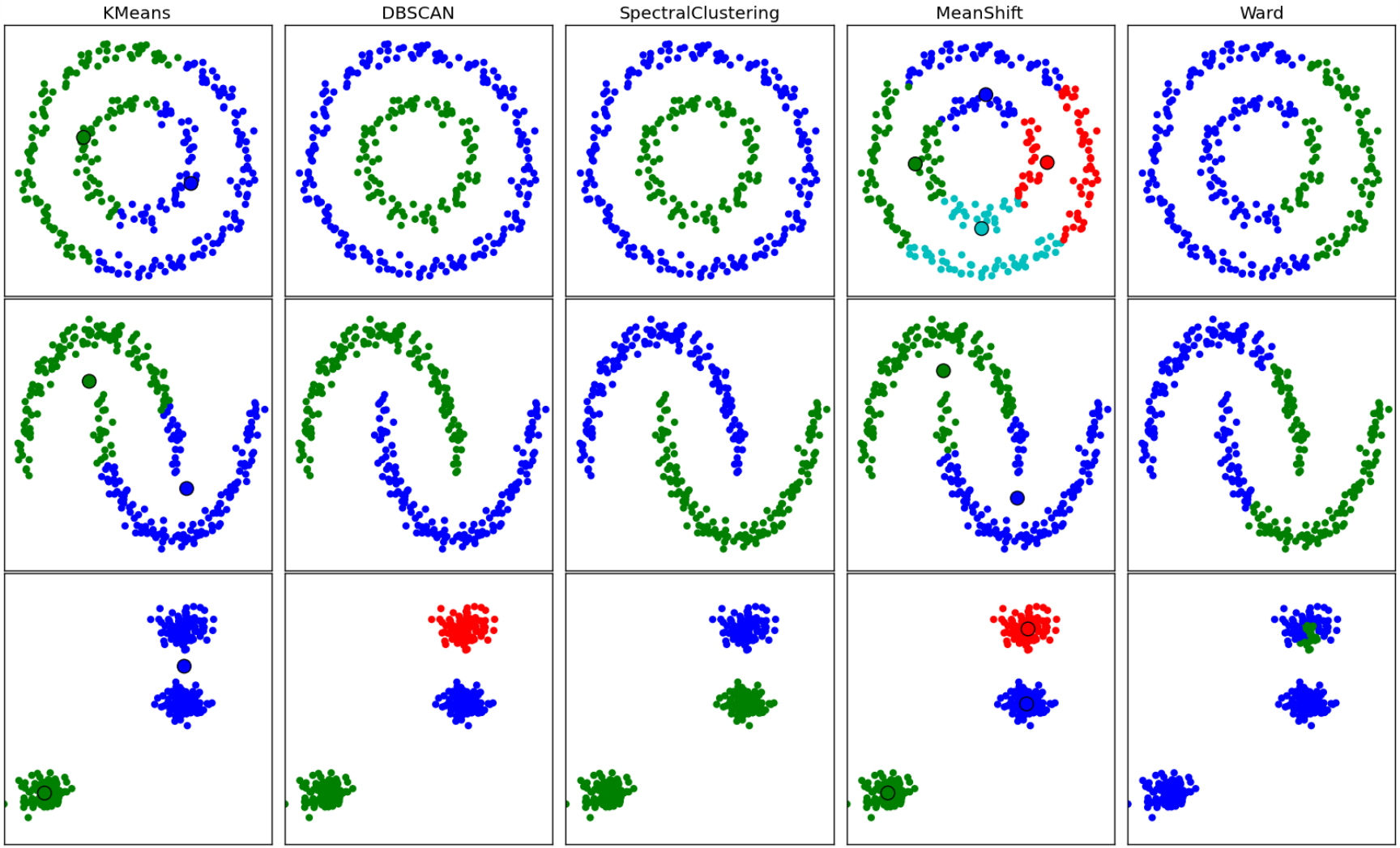


Figure: From Scikits learn: `https://ogrisel.github.io/scikit-learn.org/sklearn-tutorial/modules/clustering.html`

Today's Lecture

I. Classification
- Reminders on linear SVMs
- SVM - Handling non-linear boundaries: Kernel Machines

II. Introduction to clustering

III. Reminders on Clustering
- Types of methods and clusters
- Distance and Dissimilarity
- Clustering Quality

IV. Clustering algorithms
- K-means
- Hierarchical clustering
- DBSCAN
- 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

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!

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

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

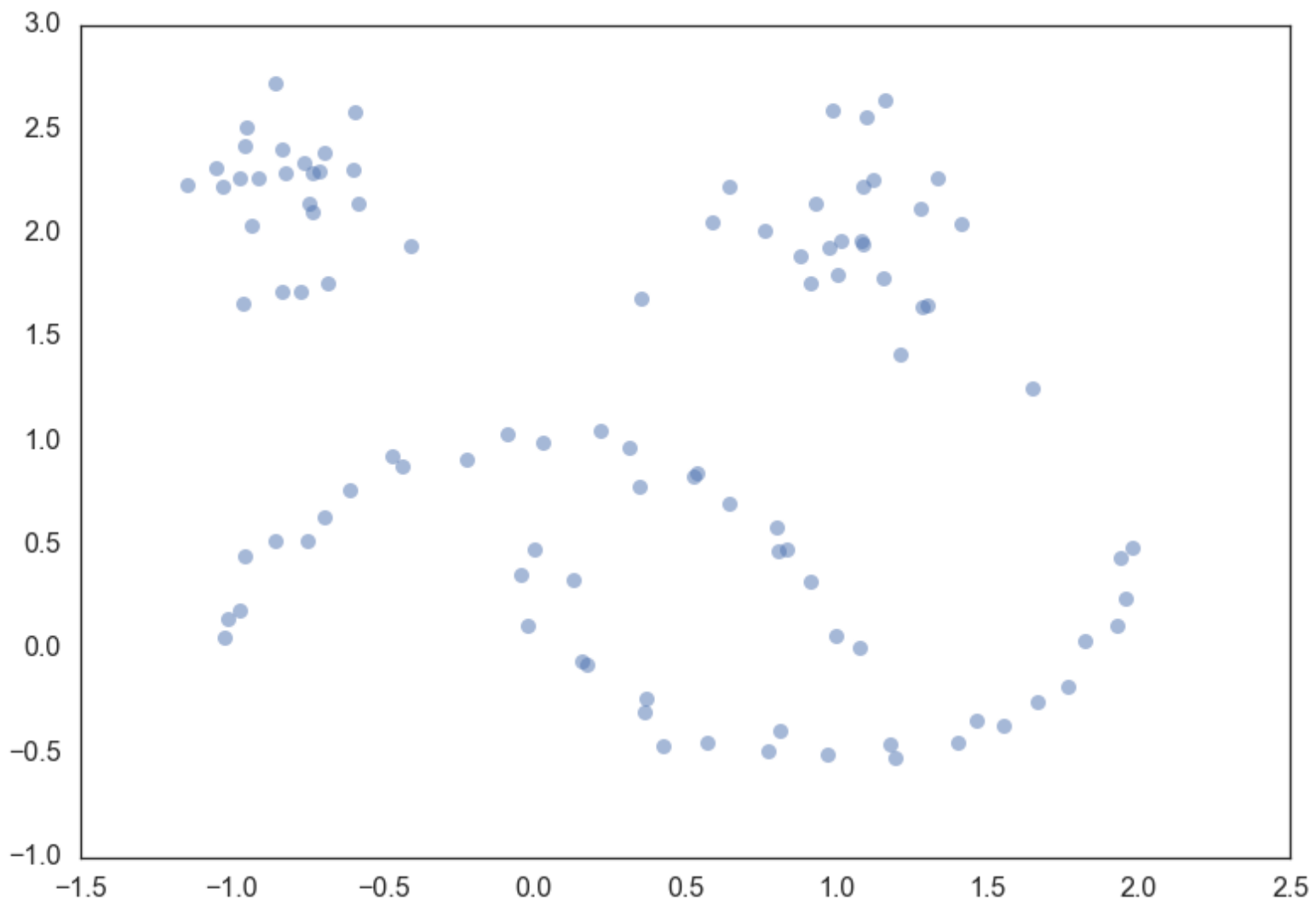
Main steps:

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!

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: Illustrative example



©https://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.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 → Make *sea* points more distant from each other and from the *land*

How do we evaluate density ?

- Need an inexpensive density estimate ⇒ 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: 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 → Make *sea* points more distant from each other and from the *land*

How do we evaluate density ?

- Need an inexpensive density estimate ⇒ 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 ?

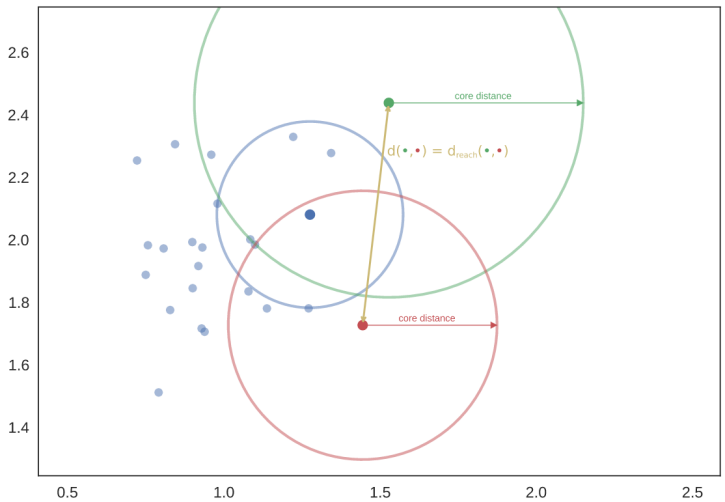
Step 1 : Mutual Reachability Distance

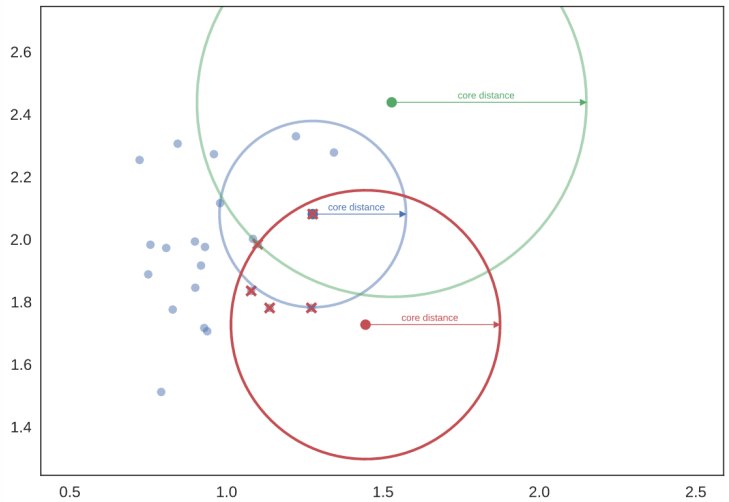
A new distance metric is defined as

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

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 : $\text{core}_k(\mathbf{x}_i)$



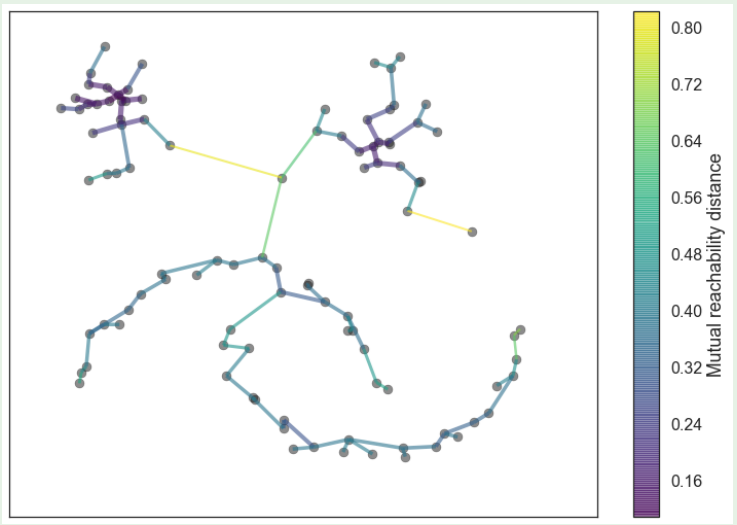


Step 2 : The Minimum Spanning Tree

- 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 → the minimum spanning tree

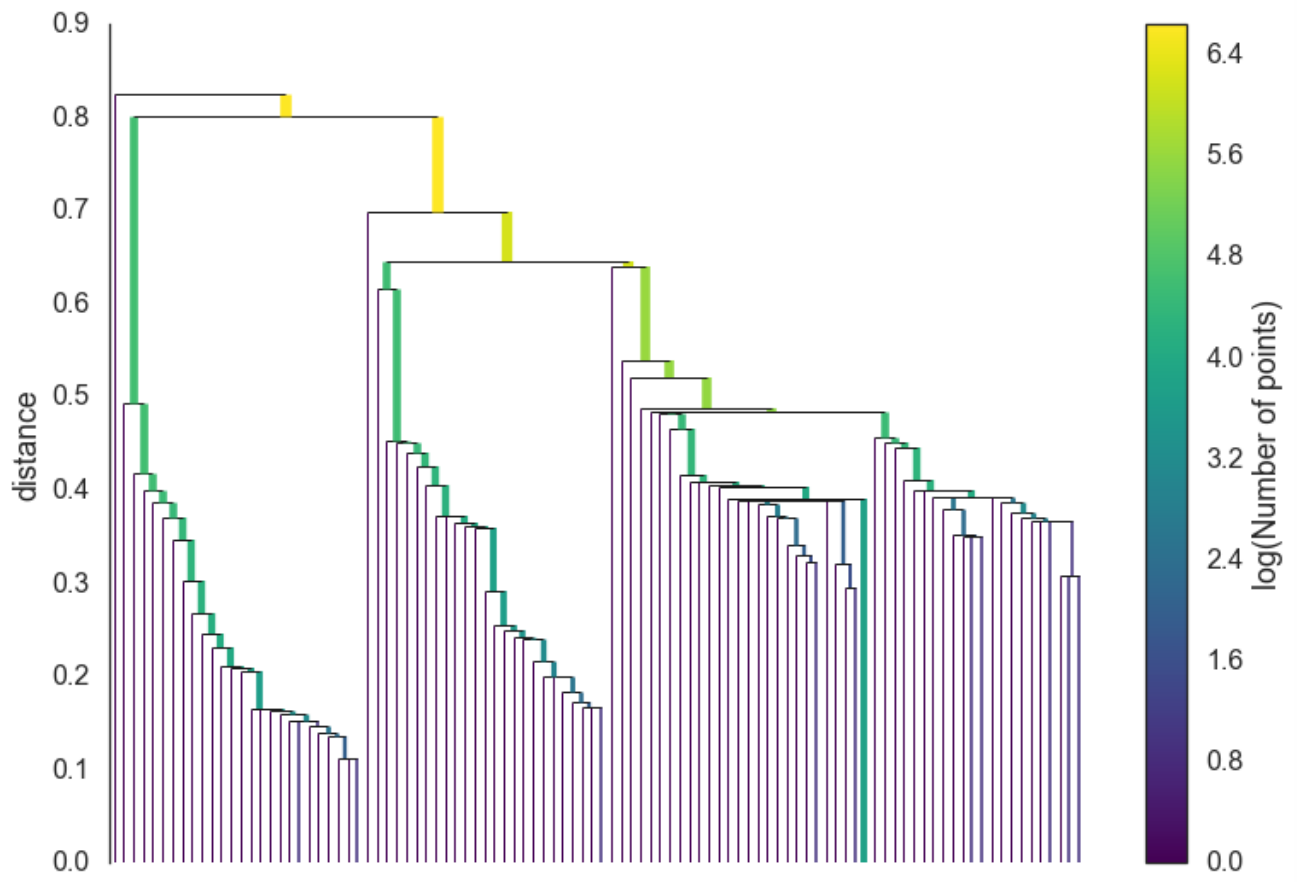
Algorithms from graph theory

- Prim's algorithm
- Dual Tree Boruvka



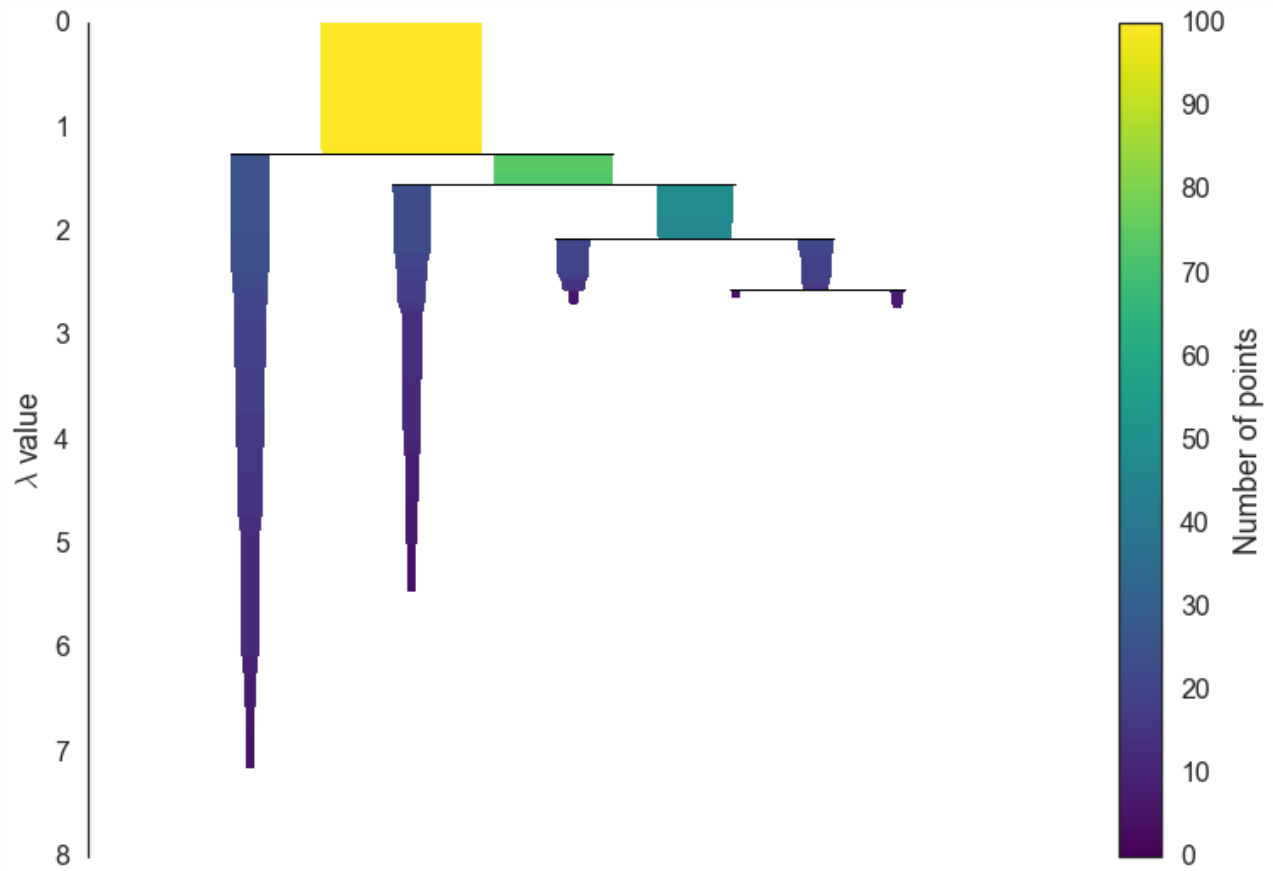
Step 3: Build the cluster hierarchy

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



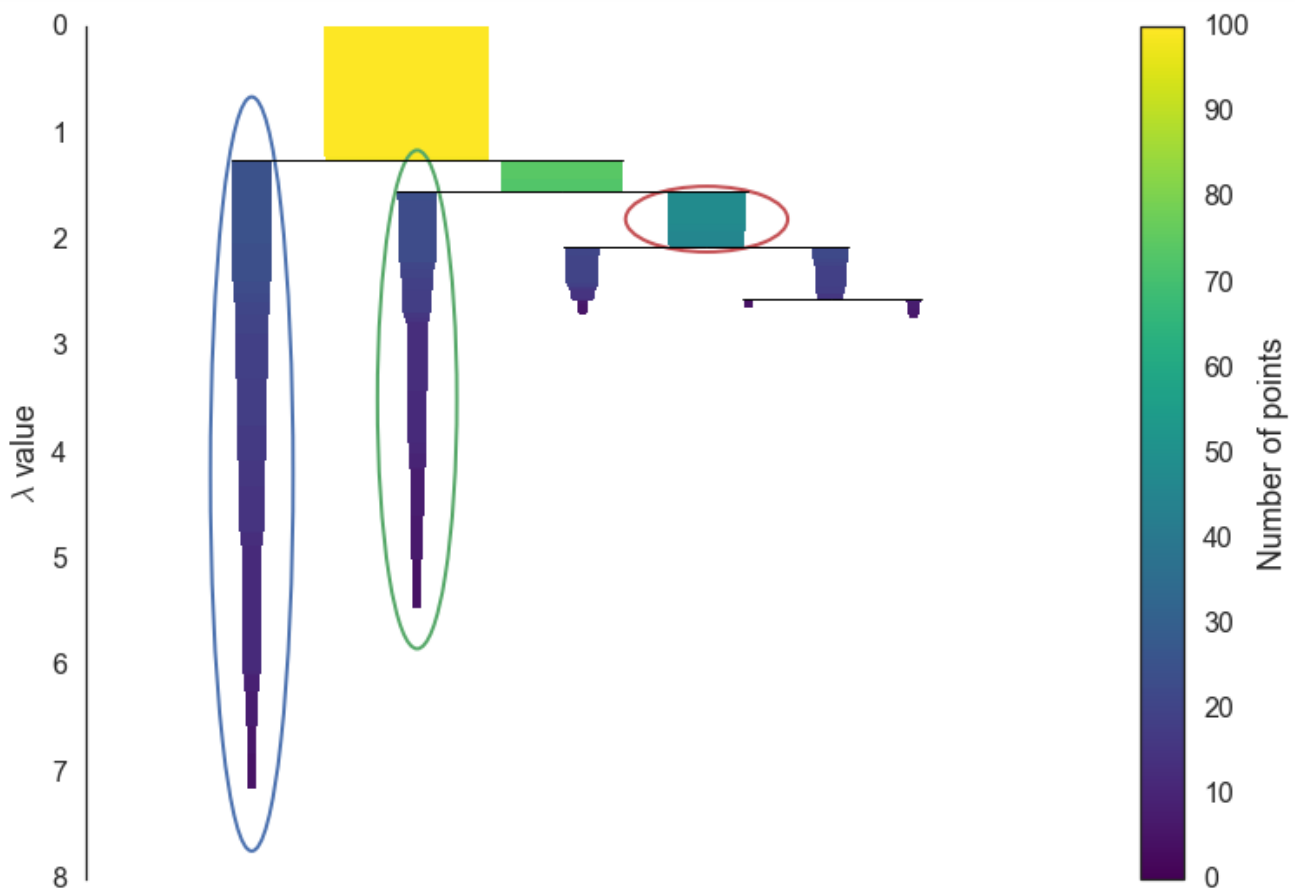
Step 4 : Condense the cluster tree

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

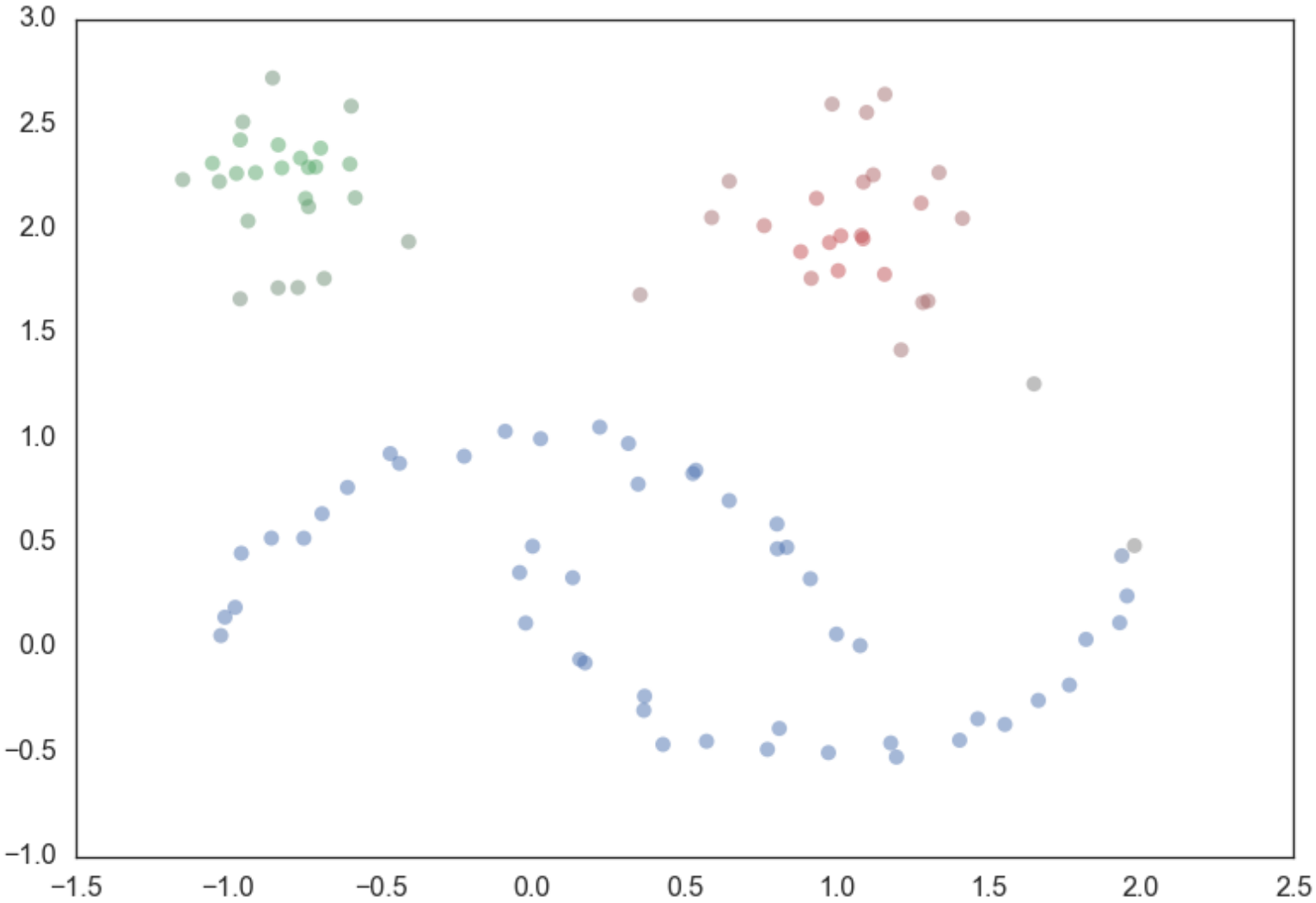


Extract the clusters

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 $\text{core}_k(\mathbf{x}_i)$ using $MinPts$ → Measure density
2. Transform the space: use new metric d_{mreach}
   → Robustness to noise!
3. Construct a minimum spanning tree
   → Lower computational cost
4. Simplify/condense the tree using C_{min}
   → Preprocessing for the next step
5. Extract final clustering results
   → Maximize cluster stability

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

HDBSCAN: Summary

Implementation: The 5 main steps

1. Compute $\text{core}_k(\mathbf{x}_i)$ using $MinPts$ → Measure density
2. Transform the space: use new metric d_{mreach}
   → Robustness to noise!
3. Construct a minimum spanning tree
   → Lower computational cost
4. Simplify/condense the tree using C_{min}
   → Preprocessing for the next step
5. Extract final clustering results
   → Maximize cluster stability

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

HDBSCAN: Summary

Implementation: The 5 main steps

1. Compute $\text{core}_k(\mathbf{x}_i)$ using $MinPts$ → Measure density
2. Transform the space: use new metric d_{mreach}
   → Robustness to noise!
3. Construct a minimum spanning tree
   → Lower computational cost
4. Simplify/condense the tree using C_{min}
   → Preprocessing for the next step
5. Extract final clustering results
   → Maximize cluster stability

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

HDBSCAN: Summary

Implementation: The 5 main steps

1. Compute $\text{core}_k(\mathbf{x}_i)$ using $MinPts$ → Measure density
2. Transform the space: use new metric d_{mreach}
   → Robustness to noise!
3. Construct a minimum spanning tree
   → Lower computational cost
4. Simplify/condense the tree using C_{min}
   → Preprocessing for the next step
5. Extract final clustering results
   → Maximize cluster stability

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

HDBSCAN: Summary

Implementation: The 5 main steps

1. Compute $\text{core}_k(\mathbf{x}_i)$ 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...