Université de Strasbourg

Unsupervised Learning

Maja Temerinac-Ott
temerinacott@unistra.fr

Slides adopted from Germain Forestier

Introduction

- Purpose of automatic classification:
  - obtain a simplified representation (structuring) of the initial data
  - organize a set of objects into a set of homogeneous and / or natural groupings
  - Two approaches:
    - classification / clustering / grouping: discovery in extension of these sets-> notion of classes or clusters
    - formation of concepts: discovery in intention of these clusters ->notion of concepts

Clustering

- Different methods:
  - **partitional clustering**
  - clustering by modeling
  - hierarchical clustering

- Furthermore:
  - density-based clustering
  - grid-based clustering

Partioning

- Partitional clustering:
  - Place the different objects in clusters (groups)
  - **Hard clustering**: each object is in one and only one class
  - **Soft clustering**: An object can be in several classes
  - **Fuzzy clustering**: An object belongs partly to all classes

Partitioning

- Partitional clustering:
  - Matrix of classification of elements in K classes

$$c(D) = \begin{pmatrix} c_{1,1} & \dots & c_{1,K} \\ & \dots & \\ c_{n,1} & \dots & c_{n,K} \end{pmatrix}$$

  - Hard partitioning:
    - $\nexists k$, $c_{i,k}=1$, $c_{i,j} = 0$ for $j \neq k$
  - Soft partitioning:
    - $\exists k_1$, k_t , ... s.t. , $c_{i,ki}$, $c_{i,j} = 0$ for $j \neq k_i$

Partitioning

- Fuzzy clustering:
  - The number of objects in a class is less than the total number objects

$$\forall k = 1 \dots K, \forall x_i \in D, c_{i,k} \in [0,1]$$

with $\forall k = 1 \dots K, 0 < \sum_{i=1}^{n} c_{i,k} < n$

  - An object belongs to different degrees, to all classes:

$$\forall x_i \in D, \sum_{k=1}^{K} c_{i,k} = 1$$

Partitioning

- Partitional clustering:
  - Find the organization in homogeneous classes such that **two objects of the same class are more similar than two objects from different classes**
  - Find the organization in homogeneous classes such that the **classes are as different as possible**
  - How to evaluate these similarities/dissimilarities?

Dissimilarity and Distance

- We denote the dissimilarity of any mapping

$$d : M^2 \rightarrow R^+$$

  - by

$$\forall (x,y) \in M^2, d(x,y) = 0 \leftrightarrow x = y$$

  - It is symmetric:

$$\forall (x,y) \in M^2, d(x,y) = d(y,x)$$

  - We call distance, any symmetric mapping which follows also the triangular inequality:

$$\forall (x,y,z) \in M^3, d(x,z) \leq d(x,y) + d(y,z)$$

Dissimilarity and Distance

- Example:
  - Distance fora p-dimensional vector **x**:

$$d_q = \sqrt[q]{|x_{i,1} - x_{j,1}|^q + |x_{i,2} - x_{j,2}|^q + \ldots + |x_{i,p} - x_{j,p}|^q}$$

Dissimilarity and Distance

- Point-to-point measurement:
  - The data is a vector from $[T,F]^p$
  - Definition of a contingency table with **p** binary attributes and two objects i_1 and i_2:

i1/i2	True	False	Sum
True	N1	N2	N1+N2
False	N3	N4	N3+N4
Sum	N1+N3	N2+N4	p

  - N1 = number of times where an attribute is true for both objects

Dissimilarity and Distance

- Point-to-point measurement:
  - The data is a vector from $[T,F]^p$
  - **symmetric measure**:

$$d_{(i,j)} = \frac{N2 + N3}{N1 + N2 + N3 + N4}$$

  - **asymmetric measure**:

$$d_{(i,j)} = \frac{N2 + N3}{N1 + N2 + N3}$$

Dissimilarity and Distance

- Example:

Region	Elongated	Textured	Big	Isolated	Frequent
City	NO	YES	YES	YES	NO
District	NO	YES	NO	NO	YES
Lake	NO	NO	YES	YES	NO
Field	YES	NO	NO	NO	YES

Dissimilarity and Distance

- Example city/district:

Region	Elongated	Textured	Big	Isolated	Frequent
City	NO	YES	YES	YES	NO
District	NO	YES	NO	NO	YES

	YES	NO
YES	1	2
NO	1	1

- symmetric measure: 3/5
- asymetric measure: 3/4

$$d_{(i,j)} = \frac{N2 + N3}{N1 + N2 + N3 + N4}$$

$$d_{(i,j)} = \frac{N2 + N3}{N1 + N2 + N3}$$

Dissimilarity and Distance

- For the other pairs:

Region	Elongated	Textured	Big	Isolated	Frequent
City	NO	YES	YES	YES	NO
District	NO	YES	NO	NO	YES
Lake	NO	NO	YES	YES	NO
Field	YES	NO	NO	NO	YES

- city/lake : ?
- city/field : ?
- district/field : ?
- district/lake : ?
- lake/field : ?

Dissimilarity and Distance

- For the other pairs:

Region	Elongated	Textured	Big	Isolated	Frequent
City	NO	YES	YES	YES	NO
District	NO	YES	NO	NO	YES
Lake	NO	NO	YES	YES	NO
Field	YES	NO	NO	NO	YES

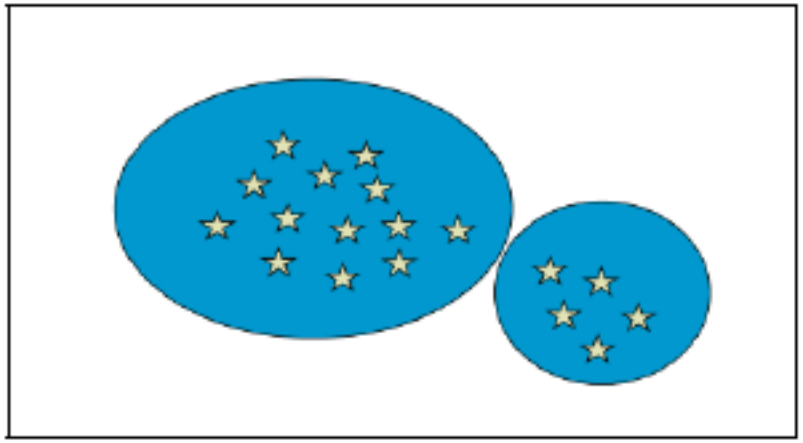
- city/lake : 1/5 and 1/3
- city/field : 1 and 1
- district/field : 2/5 and 2/3
- district/lake : 4/5 and 1
- lake/field : 4/5 and 1

Partitioning measures

- The data is a nominal variable:
  - take finite values from a finite set
  - without natural order:
    - color ∈ {red,blue,green}
  - with order:
    - level ∈ {beginer,experienced,expert}
    - or beginer<experienced<expert
- The data is complex:
  - histogram
  - heterogeneous structure
  - image, video, …

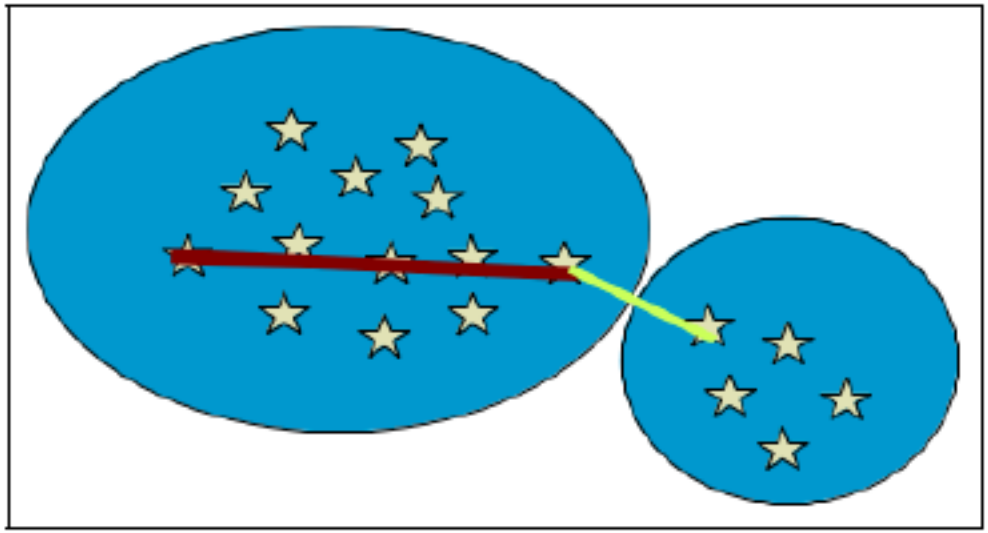
Partitioning

- Goal:
  - find the organization in homogeneous classes such that two objects of the same class resemble more than two objects of different classes



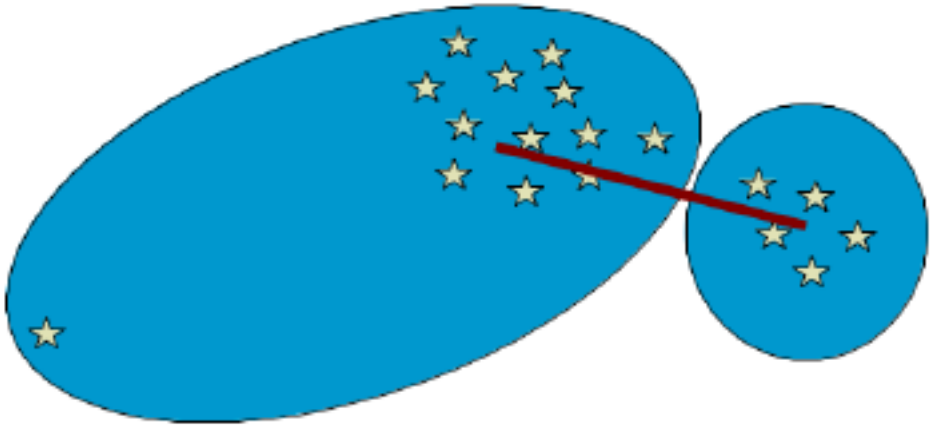
Partitioning

- x1 and x2 of the same class, x3 of a different class then d(x1, x2) <d(x1, x3) and d(x1, x2) <d(x2, x3)



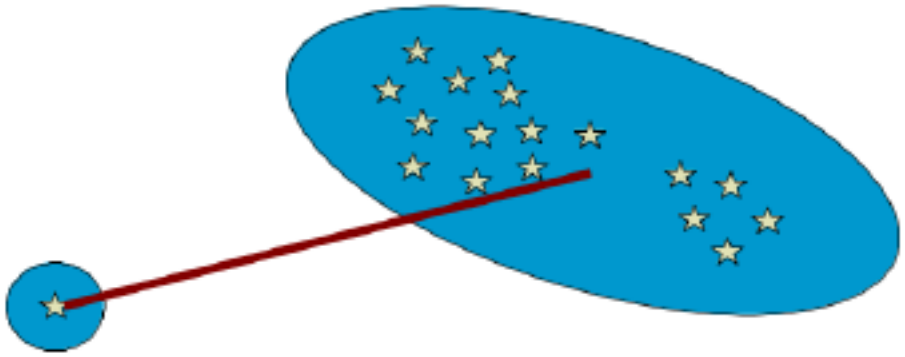
Partitioning-Quality

- Goal: Find the organization in homogeneous classes such that the distances between classes are maximal
  - we would like to have:



Partitioning-Quality

- Goal: Find the organization in homogeneous classes such that the distances between classes are maximal
  - this would increase the distance even more:



Partitioning

- Metric approach
  - minimize **intra-class** inertia I_{intra} (internal to classes):

$$I_{intra}(C_k) = \sum_{x_i \in C_k} p_i \times dist(x_i, g_k)^2$$

$$I_{intra} = \sum_{k=1...K} I_{intra}(C_k)$$

Partitioning

- Metric approach
  - minimize **inter-class** inertia I_{inter} (internal to classes):

$$I_{inter} = \sum_{k=1...K} P_k \times dist(g, g_k)^2$$

Partitioning

- Metric approach
  - The inertia of the whole system is:

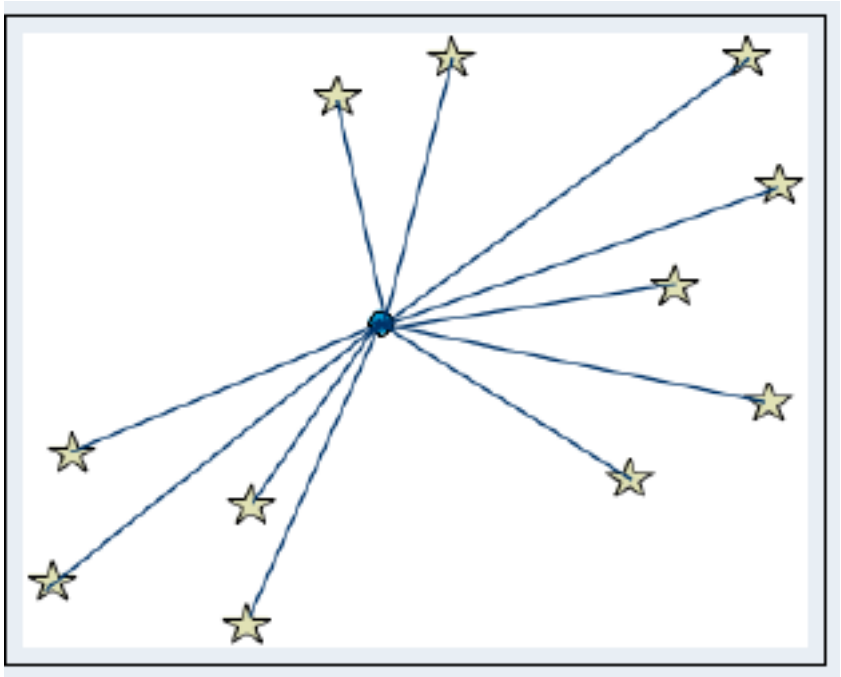
$$I_{totale} = I_{intra} + I_{inter}$$

$$I_{totale} = \sum_{i=1\ldots n} p_i \times dist(x_i, g)^2$$

  - where **g** is the center of gravity

Partitioning

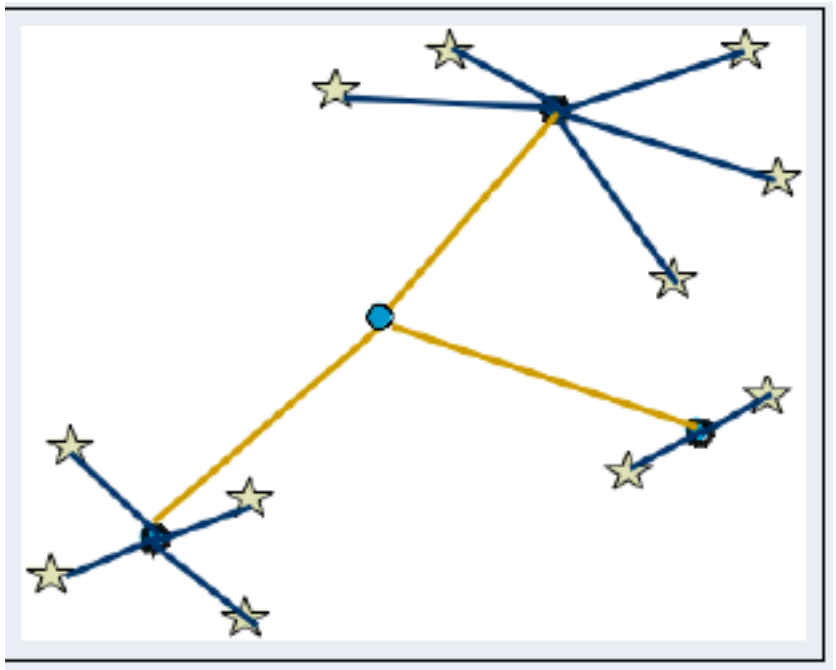
- Metric approach
  - Example for K=3
  - Compute the total inertia



- First round:
  - Compute the center of gravity
- Second round:
  - Compute the distances of all points to the center of gravity

Partitioning

- Metric approach
  - Intra-class and inter-class inertia



- First round:
  - Calculate g_k
- Second round:
  - Calculate the inter-class and intra-class inertia

Partitioning

- Metric approach
  - In most cases, the quality criterion used is the minimization of the intra-class inertia (and hence the maximization of inter-class inertia)
  - The principle of algorithms is therefore to find partitions such that this inertia is minimal

Partitioning

- Construction:
  - How to build the classes?
  - Number of partitions of **k classes** from **n data** defined on **D**:

$$S(n,k) = \frac{1}{k!} \sum_{i=0\ldots k} (-1)^{i-1} C_i^k \, i^n$$

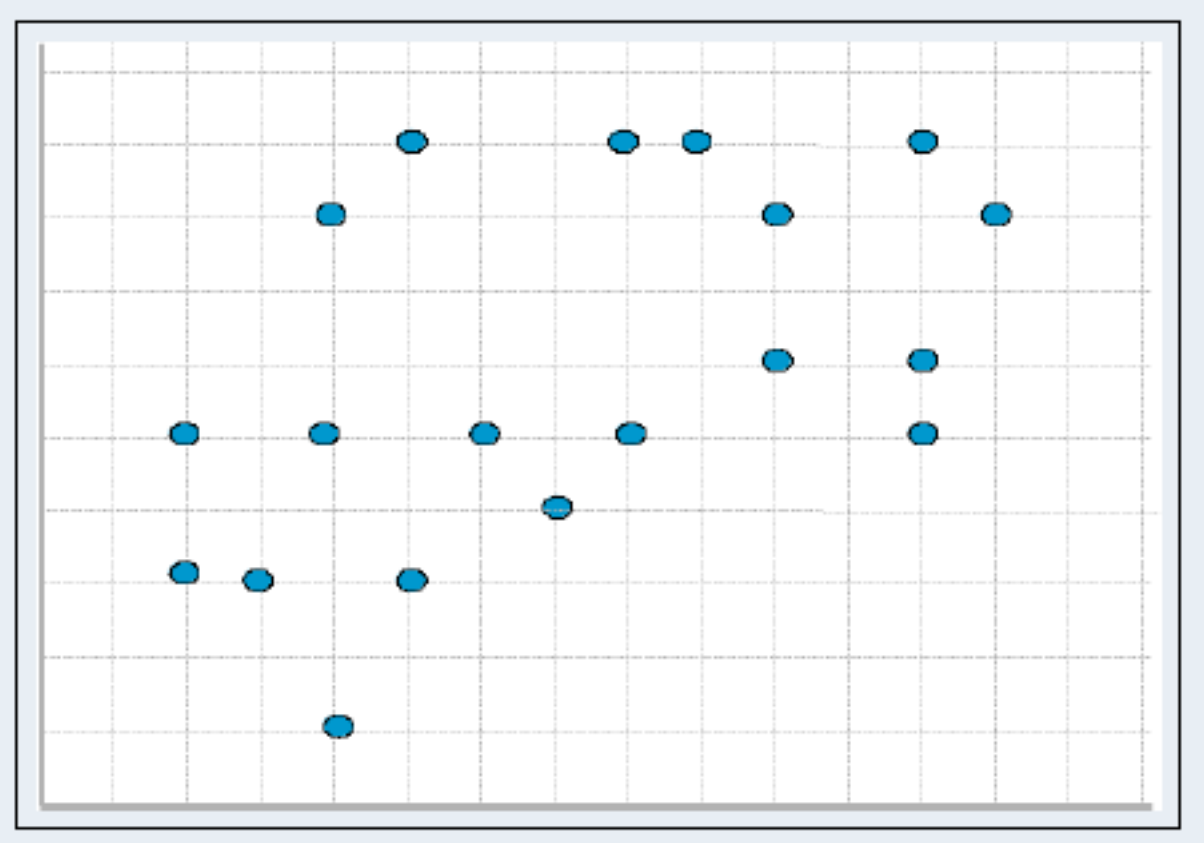
  - goes towards S(n,k)≈ $\frac{k^n}{k!}$ for large n
  - n=8,k=4 → S(n,k)≈1701
  - n=12,k=3 → S(n,k)≈ 86526

Partitioning

- Construction:
  - Criteria:
    - Minimize the intra-class inertia,maximize the inter-class inertia
  - K-Means algorithm:
    1. Choose the number of classes K
    2. Choose K random initial data cluster centers
    3. Assign each data point to its closest cluster center
    4. Compute the new cluster centers (average of data points assigned to the cluster)
    5. Repeat 3.-4. until convergence

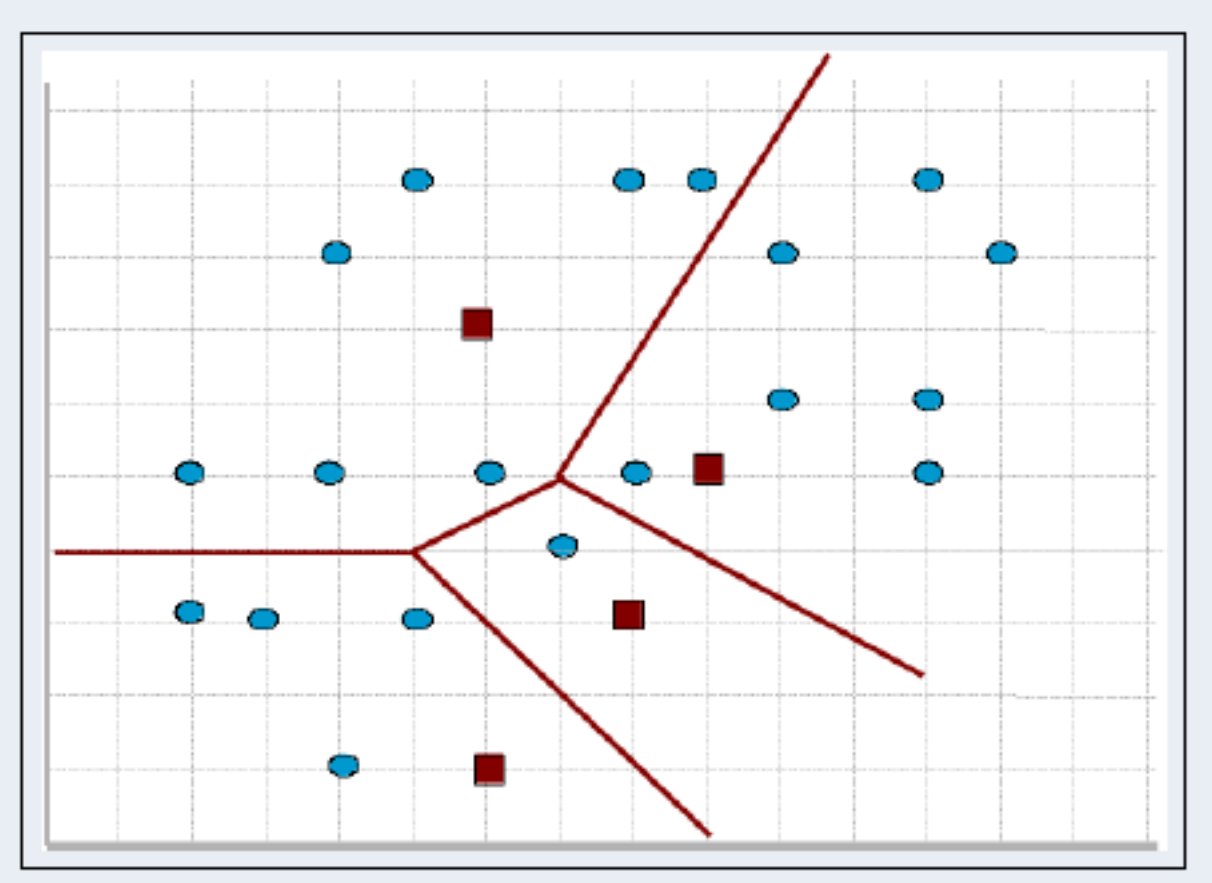
Partitioning

- With K=4:



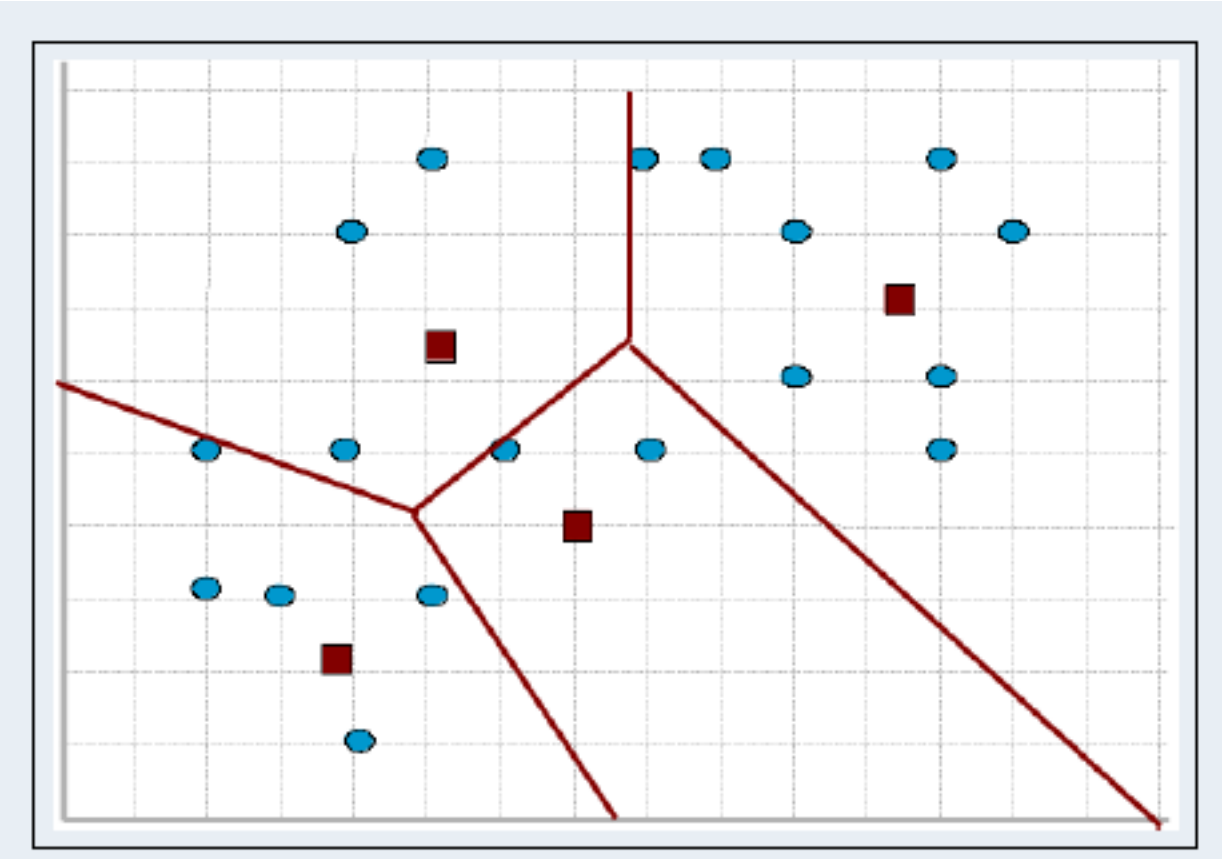
Partitioning

- With K=4 random initialization and allocation:



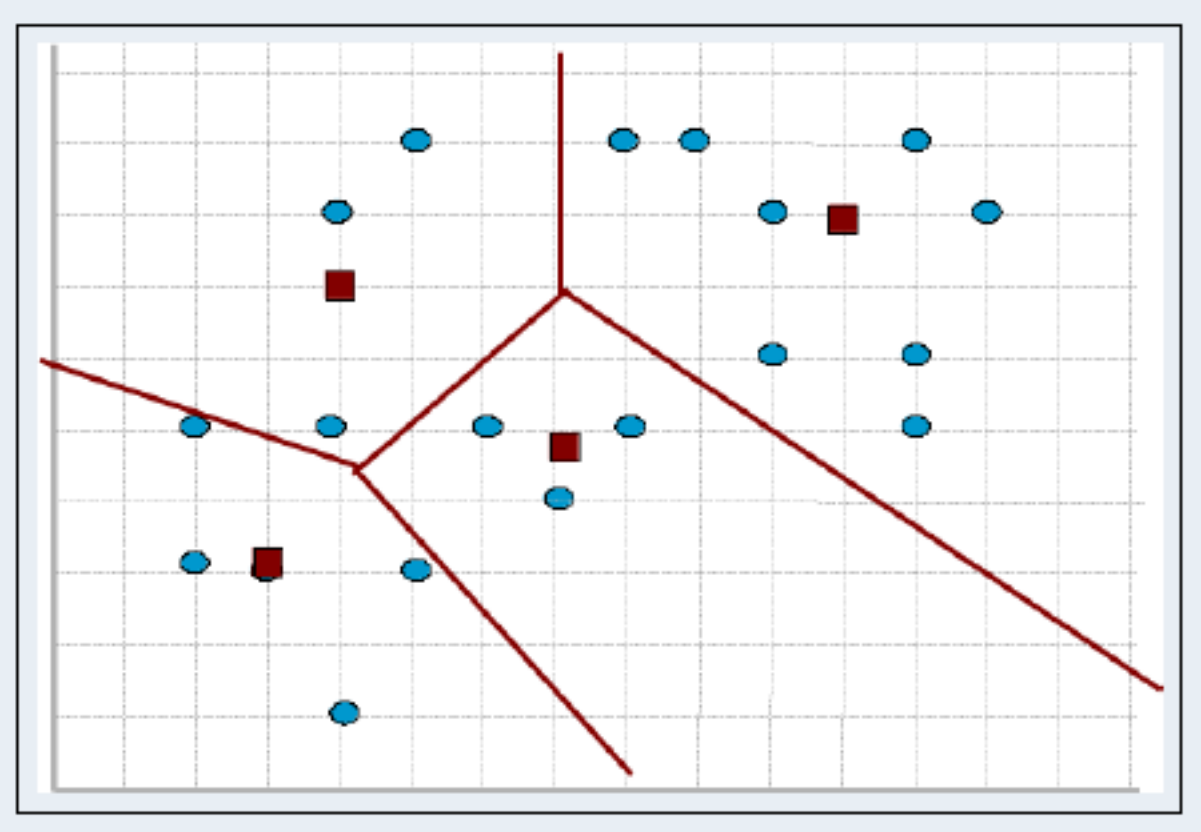
Partitioning

- With K=4 after updating the centers:



Partitioning

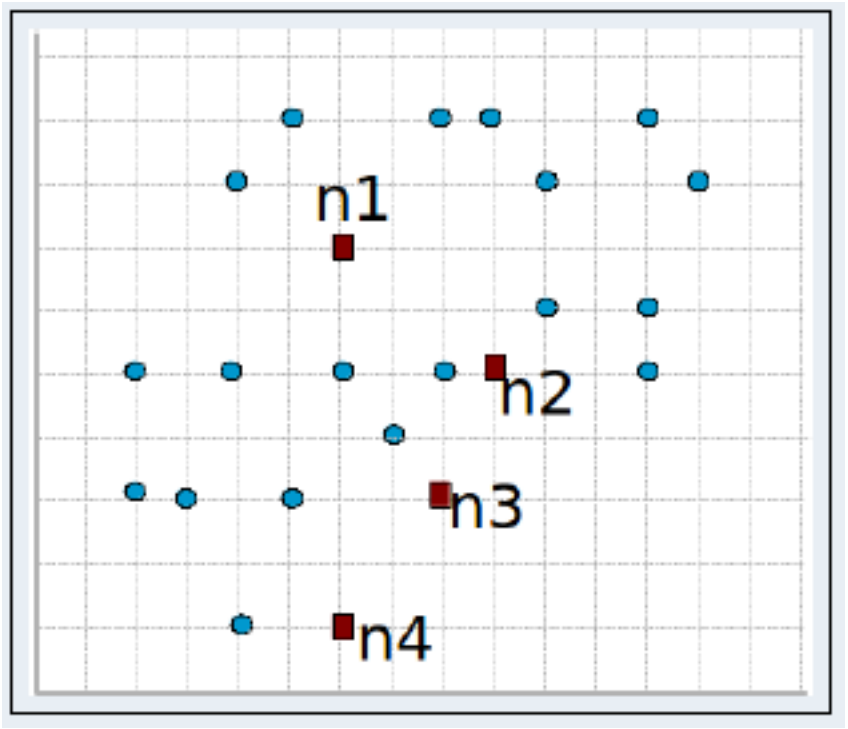
- With K=4 final result:



Partitioning

- K-means properties:

Consider a time point **t, P** is the partition, **L** are the center of gravities



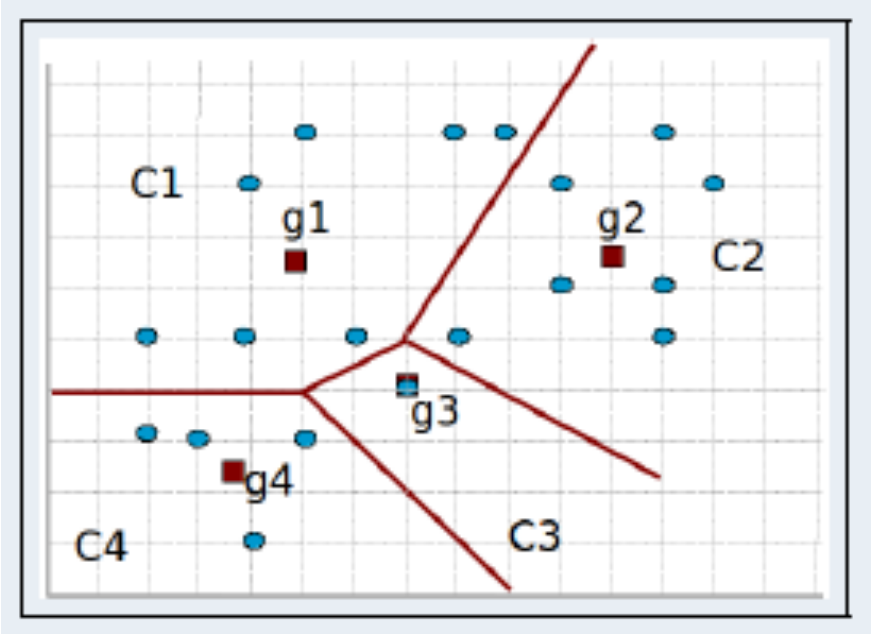
- $P_K(t) = \{C1, C2, ...Ck\}$
- $L_K(t) = \{g1, g2, ..., gk\}$

$$P4(0) = D,$$
$$L4(0) = \{n1, n2, n3, n4\}$$

Partitioning

- K-means properties:

Consider a time point **t, P** is the partition, **L** are the center of gravities



- $P_K(t) = \{C1, C2, ...Ck\}$
- $L_K(t) = \{g1, g2, ..., gk\}$

$$P4(1) = \{C1, C2, C3, C4\},$$
$$L4(1) = \{g1, g2, g3, g4\}$$

Partitioning

- K-means properties:
  - Let Q (P_K (t), L_K (t)) be the total internal inertia of the classes for the partition $P_K(t) = \{C_1, C_2, \ldots, C_k\}$ and all the centers of gravity are L_K (t) = {g1, g2, …, gk}

$$Q(P_K(t), L_K(t)) = \sum_{k=1}^{K} \sum_{x \in C_k} dist(x, g_k)^2$$

  - Let us show that at each stage, this criterion improves (decreases)

Partitioning

- K-means properties:
  - Lets show that $Q(P_K(t+1), L_K(t)) < Q(P_K(t), L_K(t))$
  - Does the reallocation of objects decrease the inertia?

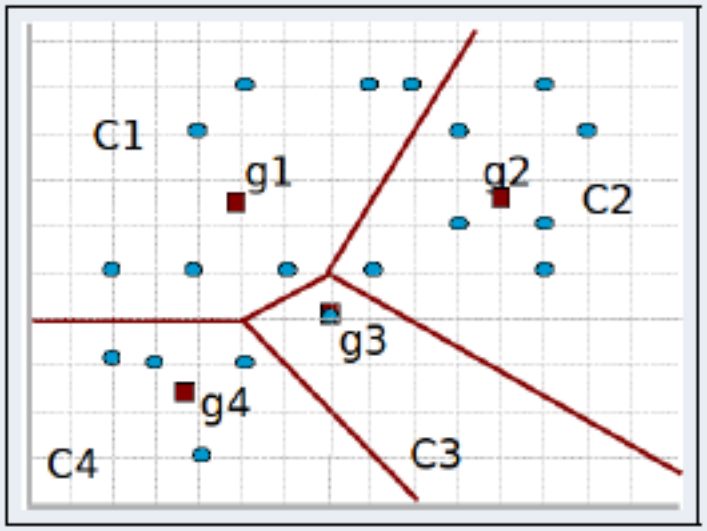
$$P_4(1) = \{C1, C2, C3, C4\}, L_4(1) = \{g1, g2, g3, g4\}$$

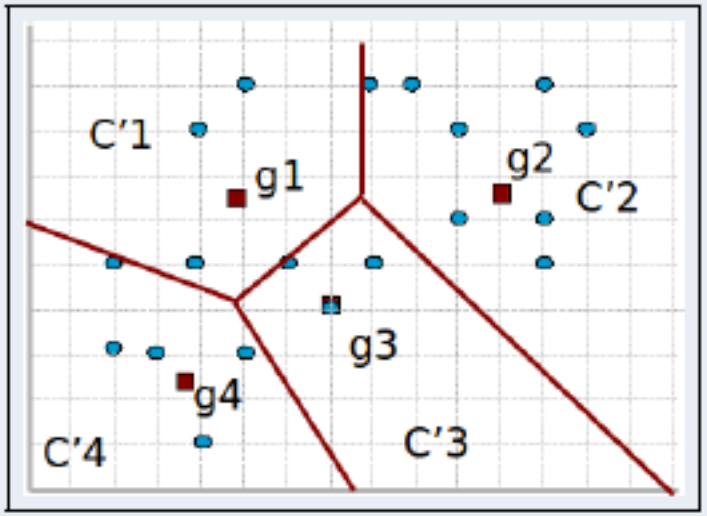
to

$$P_4(2) = \{C'1, C'2, C'3, C'4\}, L_4(1) = \{g1, g2, g3, g4\}$$

Partitioning

- K-means properties:





$P_4(1) = \{C1, C2, C3, C4\}, L_4(1) = \{g1, g2, g3, g4\}$

$P_4(2) = \{C'1, C'2, C'3, C'4\}, L_4(1) = \{g1, g2, g3, g4\}$

Partitioning

- K-means properties:
  - When constructing $P_K(t + 1)$, for a data point x, of class k(x):
    - If x was already associated with this class:
      - neither the center of gravity nor x have changed
      - dist(x, g_k (x)) is the same as in the previous partition
    - If x is in a different class $k^c(x)$, it can only change the class if dist(x, $g_k(x)$) < dist(x,$g_{kc}(x)$)

$$\rightarrow Q(P_K(t+1), L_K(t)) \leq Q(P_K(t), L_K(t))$$

Partitioning

- K-means properties:
  - When constructing P_K (t + 1), for a data point x, of class k(x):
    - If x was already associated with this class:
      - neither the center of gravity nor x have changed
      - dist(x, g_k (x)) is the same as in the previous partition
    - If x is in a different class k^c(x), it can only change the class if dist(x, g_k(x)) < dist(x, g_{kc}(x))
    - Therefore:

      - $\sum$ dist(x, gk(x)) < $\sum$ dist(x, gk_c(x))

Partitioning

- K-means properties:
  - We know from previous slide:
    - $\sum \text{dist}(x, gk(x)) < \sum \text{dist}(x, gk_c(x))$
    - Since: $$Q(P_K(t), L_K(t)) = \sum_{k=1}^{K} \sum_{x \in C_k} dist(x, g_k)^2$$
    - Can be rewritten to: $Q(P_K(t), L_K(t)) = \sum_x dist(x, g_{k(x)})^2$
    - We obtain:

$$Q(P_K(t+1), L_K(t)) \leq Q(P_K(t), L_K(t))$$

Partitioning

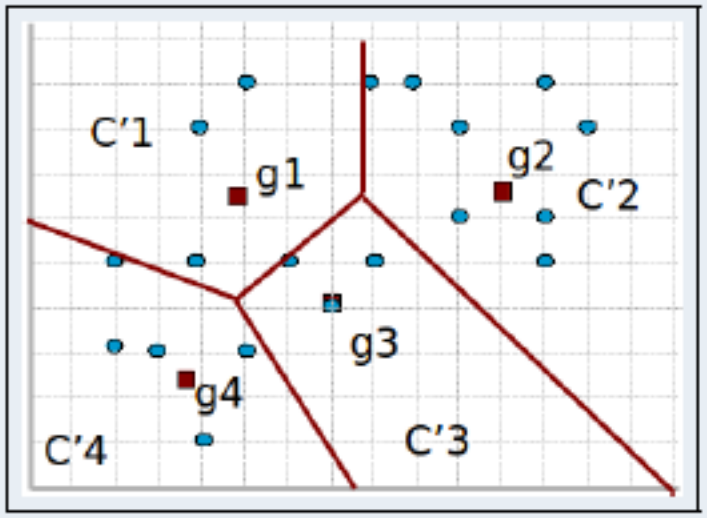
- K-means properties:
  - Now we show that the inertia does not increase if we recompute the gravity centers:

$$Q(P_K(t+1), L_K(t+1)) \leq Q(P_K(t+1), L_K(t))$$

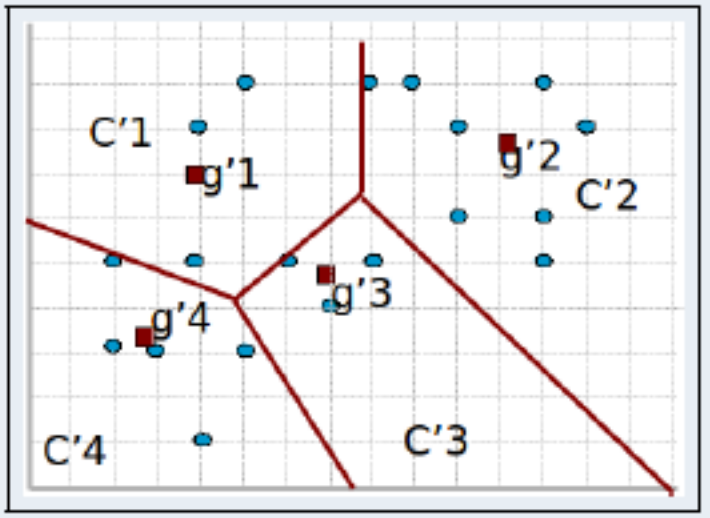
    - For example:
      - $P_4(2)$ = {C1', C2', C3', C'4} $L_4(1)$ = {g1, g2, g3, g4}
      - $P_4(2)$ = {C1', C2', C3', C'4} $L_4(2)$ = {g1', g2', g3', g4'}

Partitioning

- K-means properties:



$P_4(2) = \{C'1, C'2, C'3, C'4\}, L_4(1) = \{g1, g2, g3, g4\}$



$P_4(2) = \{C'1, C'2, C'3, C'4\}, L_4(2) = \{g'1, g'2, g'3, g'4\}$

Partitioning

- K-means properties:
  - According to definition g_k are the center of classes
  - However, the optimality property of the center of gravity (theorem of Huygens) induces inequality on inertia:
    - For each class C_k, the inertia of C_k with respect to the center of gravity is less than inertia relative to any other point
    - In particular, the inertia C with respect to g_k is less than the inertia of C with respect to g_{kc}

Partitioning

- K-means properties:
  - According to definition, g_k are the center of classes
  - However, the optimality property of the center of gravity (theorem of Huygens) induces inequality on inertia:
    - For each class C_k, the inertia of C_k with respect to the center of gravity is less than inertia relative to any other point
    - In particular, the inertia of C with respect to g_k is less than the inertia of C with respect to g_{kc}

$$\sum_{k=1}^{K} Intertie(C_k, g_k) \leq \sum_{k=1}^{K} Intertie(C_k, @g_k)$$

Partitioning

- K-means properties:
  - As

$$\sum_{k=1}^{K} Intertie(C_k, g_k) \leq \sum_{k=1}^{K} Intertie(C_k, @g_k)$$

  - It follows:

$$\sum_{k=1}^{K} \sum_{x \in C_k} dist(x, g_k)^2 \leq \sum_{k=1}^{K} \sum_{x \in C_k} dist(x, @g_k)^2$$

  - And finally:

$$Q(P_K(t+1), L_K(t+1)) \leq Q(P_K(t+1), L_K(t))$$

Partitioning

- K-means properties:
  - These two properties lead to the decay of $Q(P_K(t), L_K(t))$
  - In fact when the number of partitions P is finished, the number of centers L is also finished.
  - Therefore the sequence $Q(P_K(t+1), L_K(t))$ is :
    - descending
    - can only take one final value
    - and is therefore **stationary**

$$\exists n_0 \in \mathbb{N},\ n \geq n_0 \implies u_n = u_{n_0}.$$

Partitioning

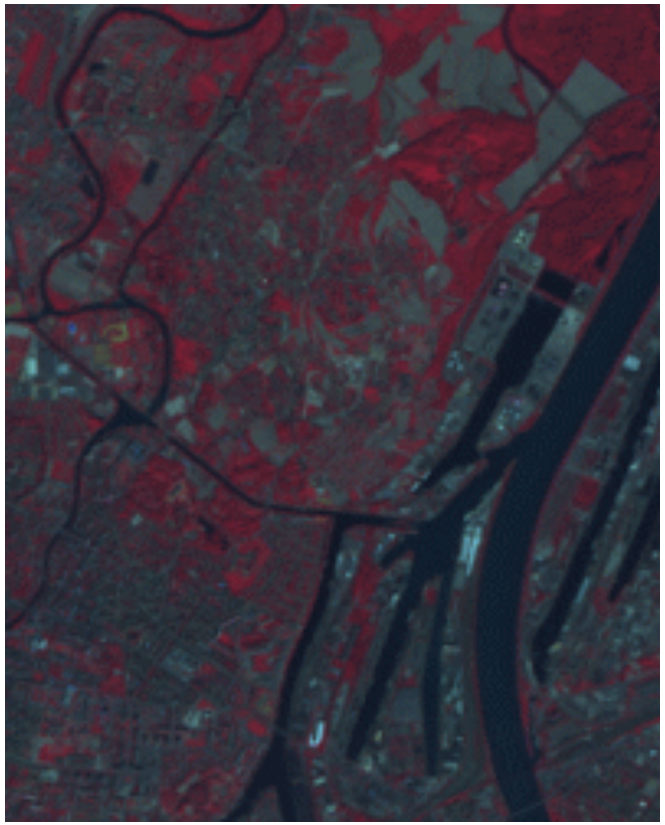
- K-means properties:
  - K-means is an algorithm that minimizes inter-class inertia
  - Nevertheless, the algorithm does not necessarily provide the best result but simply a sequence of pairs with decreasing function
    - **local optimization** only!

Partitioning

- K-means properties:
  - More precisely, it is an algorithm of alternation in two steps:
    - **Find the partitioning**: minimization of Q(P,L) with fixed L
    - **Find the centers**: minimization of Q(P,L) with fixed P
  - Experience shows that it converges quite fast (less than 10 iteration), even with large volumes of data and “bad” initialization

Partitioning

- Example on a SPOT image:

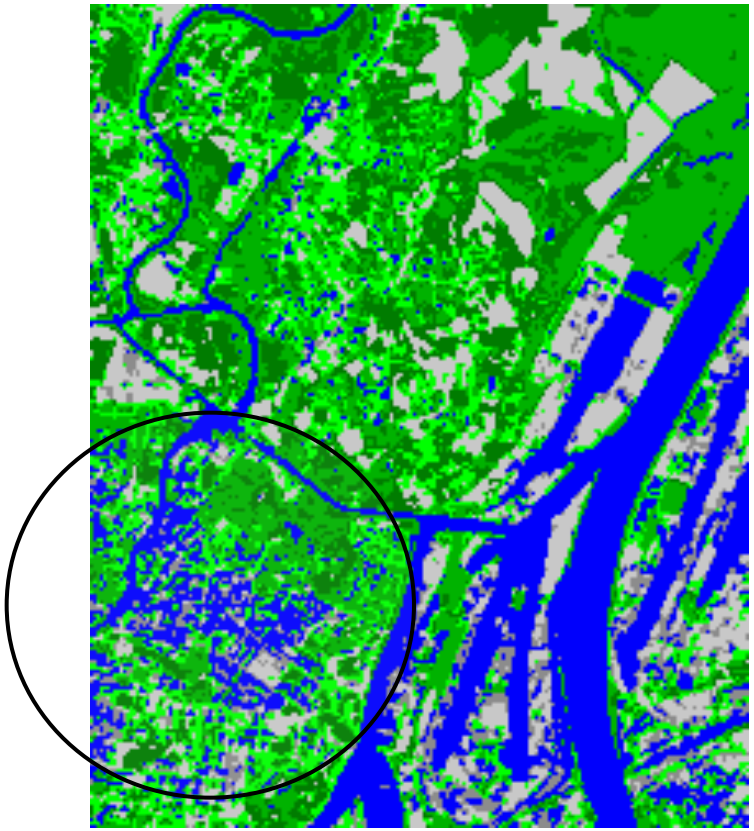


262 000 pixels
3 bandes

Partitioning

- Example on a SPOT image:

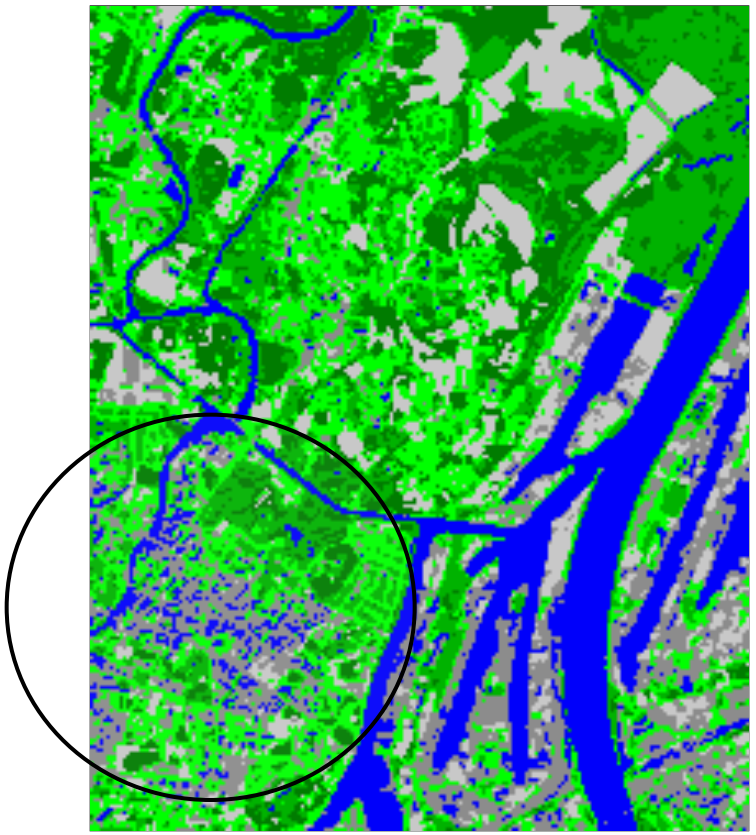
iteration1



Partitioning

- Example on a SPOT image:

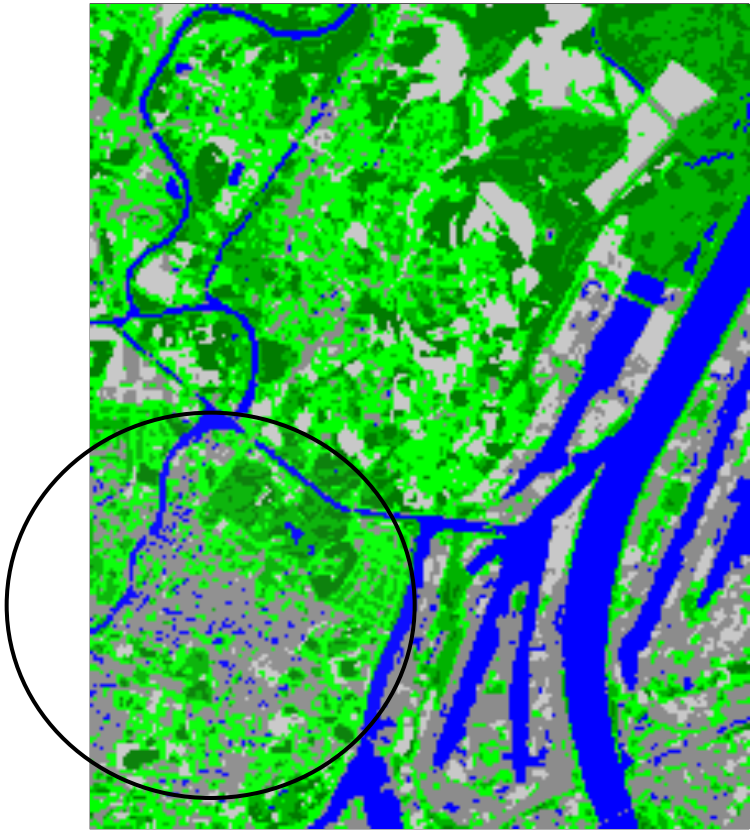
iteration2



Partitioning

- Example on a SPOT image:

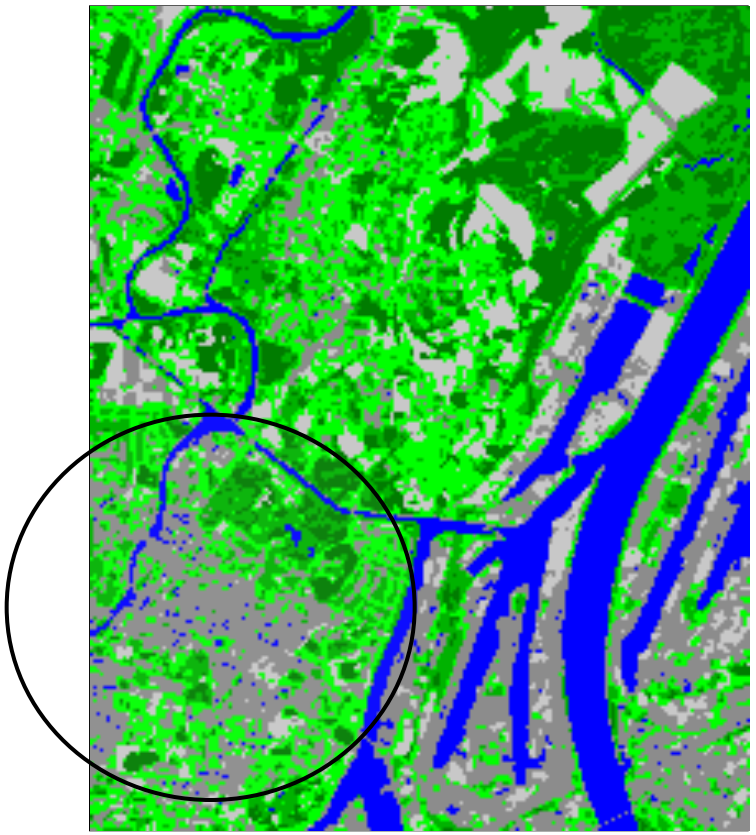
iteration3



Partitioning

- Example on a SPOT image:

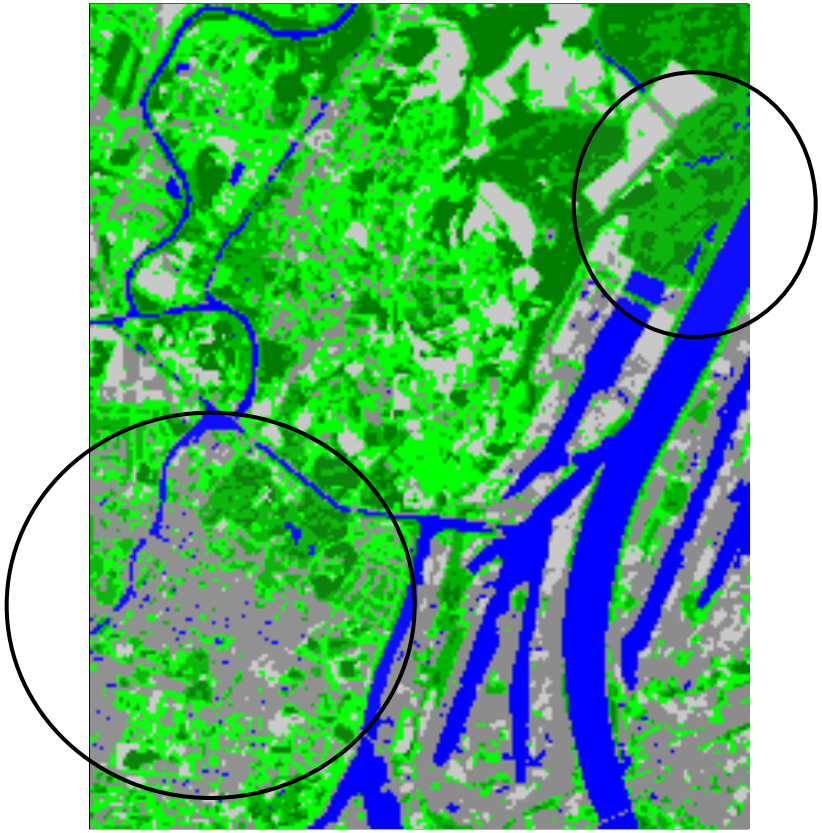
iteration4



Partitioning

- Example on a SPOT image:

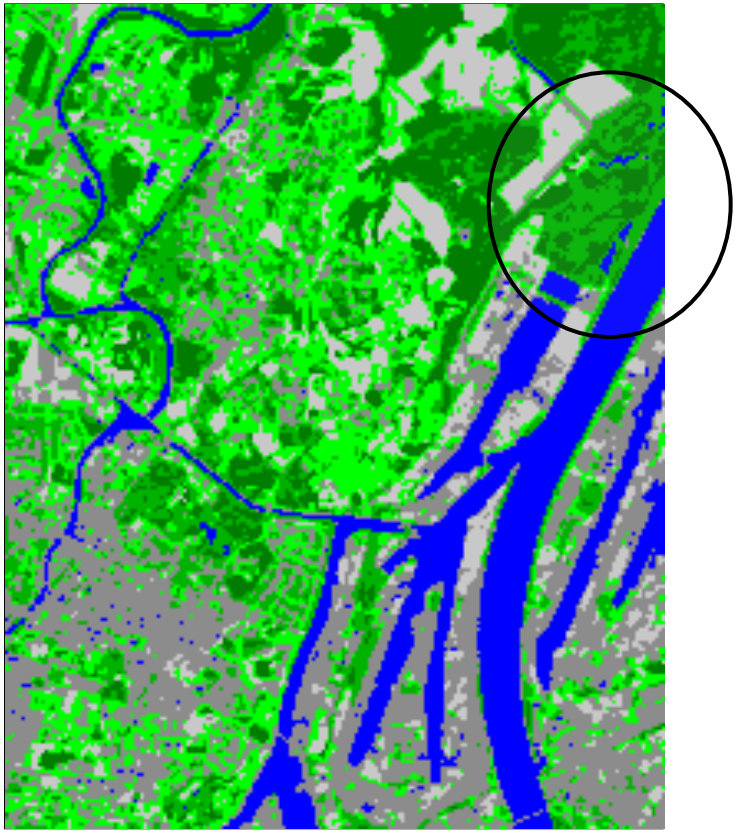
iteration5



Partitioning

- Example on a SPOT image:

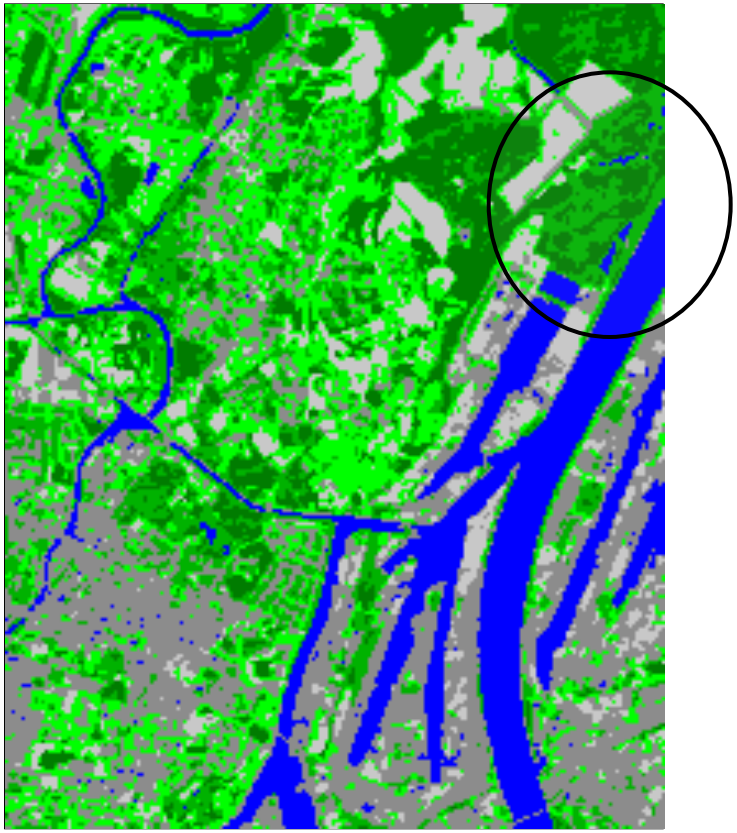
iteration6



Partitioning

- Example on a SPOT image:

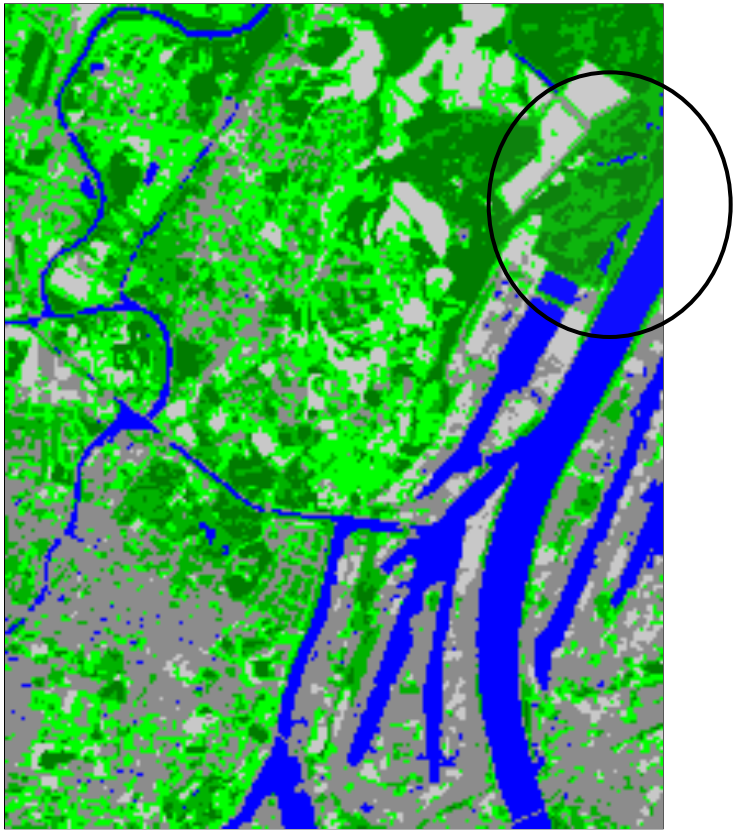
iteration7



Partitioning

- Example on a SPOT image:

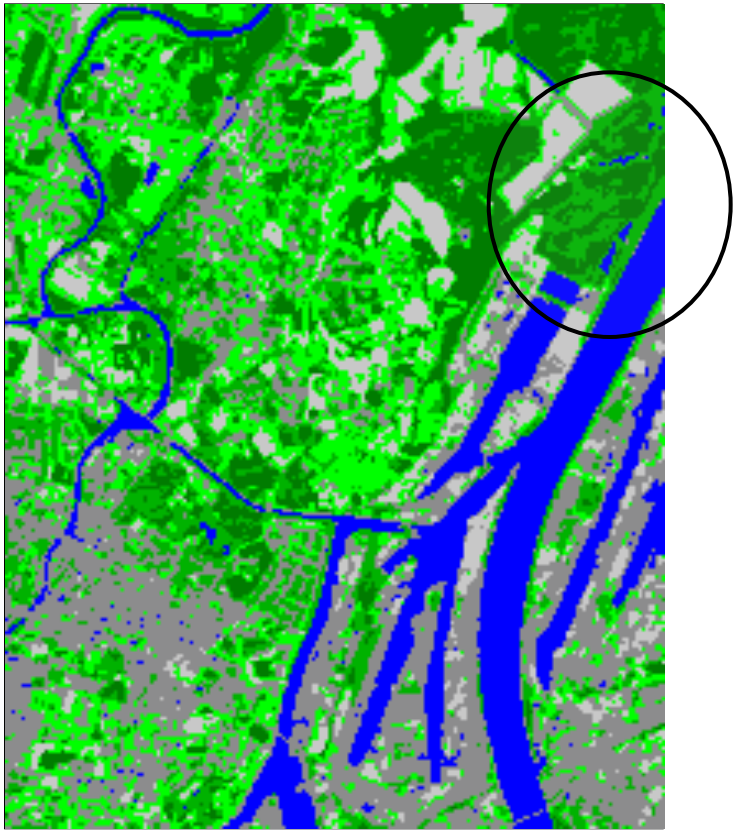
iteration8



Partitioning

- Example on a SPOT image:

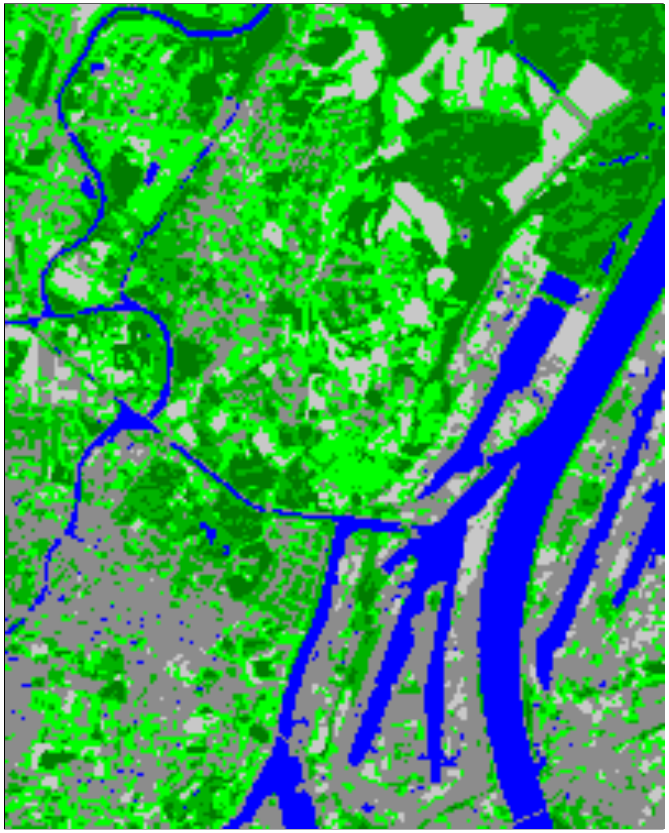
iteration9



Partitioning

- Example on a SPOT image:

iteration10



Partitioning

- Sequential method (MacQueen 1967):
  - K centers are randomly drawn from D
  - In each iteration q, an individual data point x_i is selected at random:
    - Find the center N_k closest to x_i
    - Associate x_i with class k
    - Replace the center N_k with: $N_k' = \frac{x_i + nk * Nk}{n_k + 1}$
    - Where n_k is the cardianality of N_k
  - Dynamique partitions:
    - Adopts the centers of gravity to new data related to the problem we are interested in
    - Similar principle as k-means

Partitioning - Problems

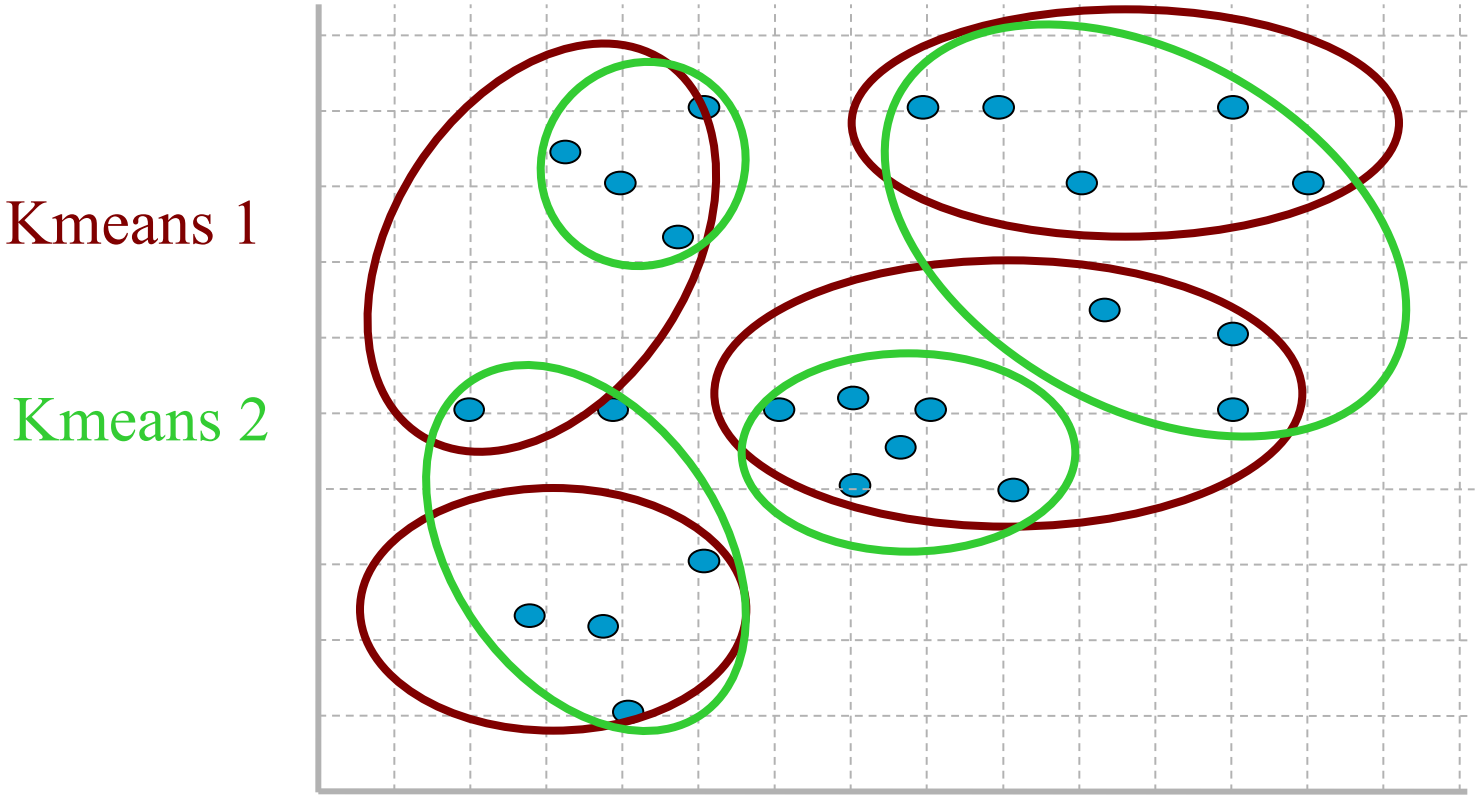
- Partitioning with reallocations: two major problems
  - **Local minimum**:
    - K-means gives different results according to the choice of initial centers
  - **Choice of the number of classes**:
    - how to combine / compare two partitions using different number of classes

Partitioning - Initialization

- First problem **local minimum**
  - for a given K: how should we choose the centers?
    - Use additional information
    - Study the structure of the data beforehand (i.e. histogram)
    - We test for different initializations and keep the best one
    - We use the intermediate result of many partitions

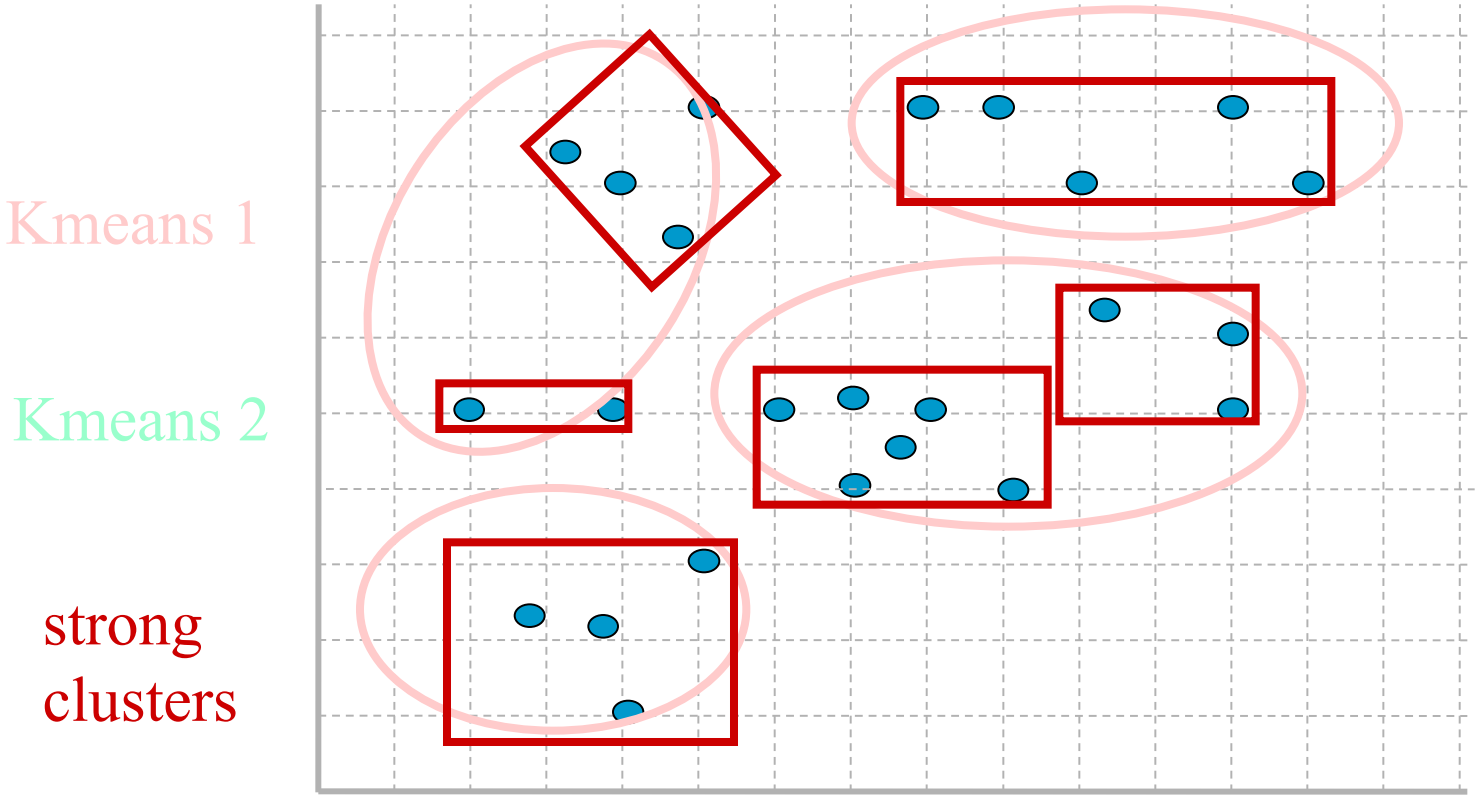
Partitioning - Initialization

- First problem **local minimum**
  - Strong clusters are obtained by superposition (or intersection of the different partitions)



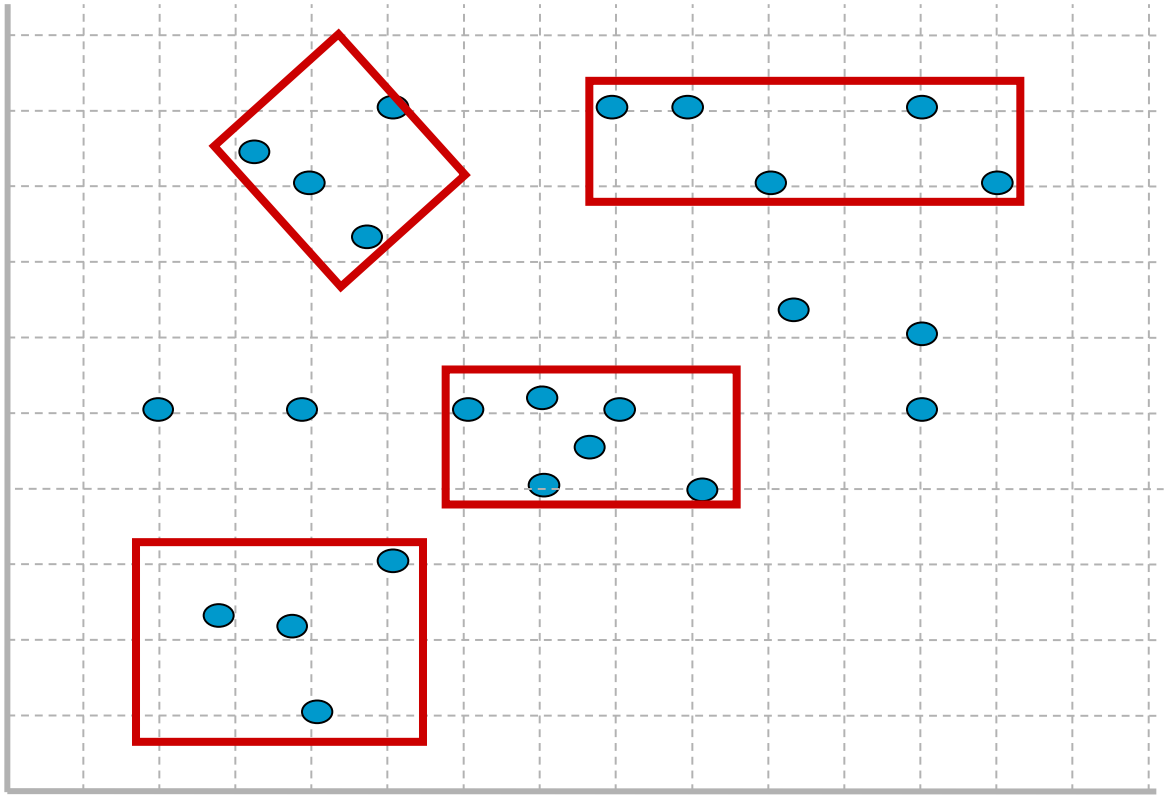
Partitioning - Initialization

- First problem **local minimum**
  - Strong clusters are obtained by superposition (or intersection of the different partitions)



Partitioning - Initialization

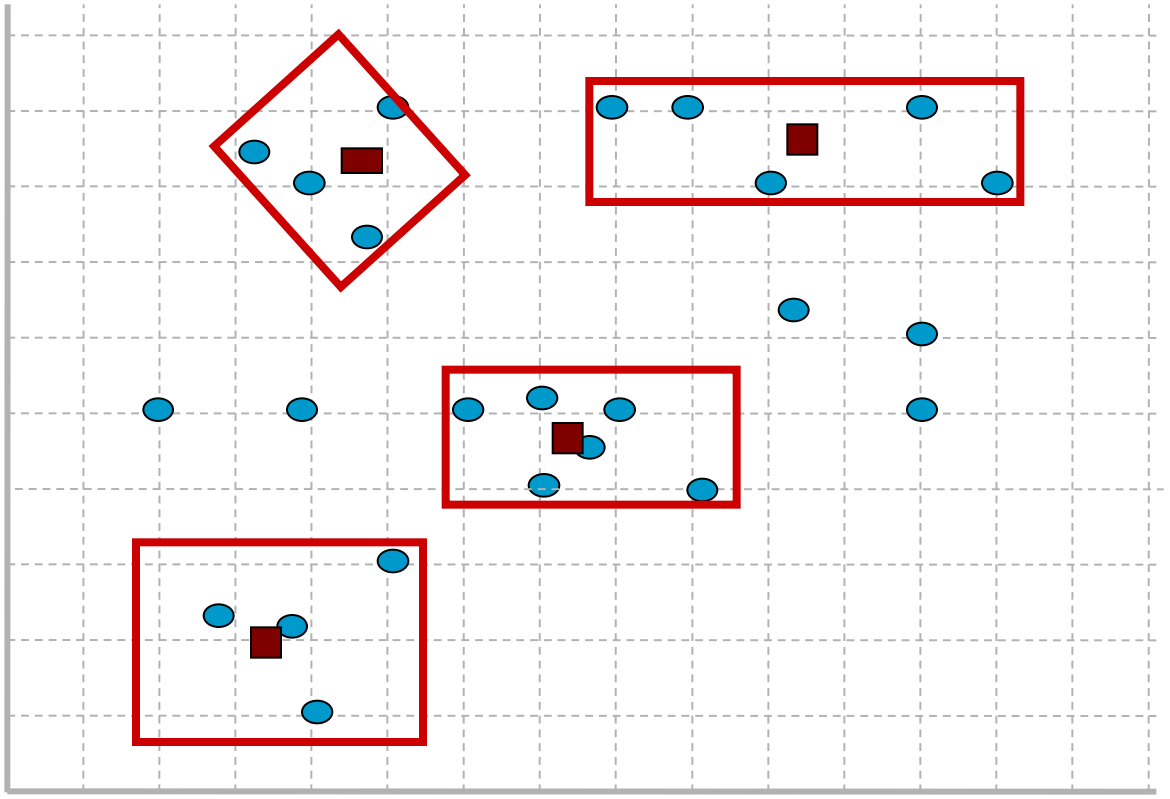
- Strong forms:
  - Keep the 4 strongest forms



strong clusters

Partitioning - Initialization

- Strong forms:
  - Take the center of gravity from the 4 strongest forms

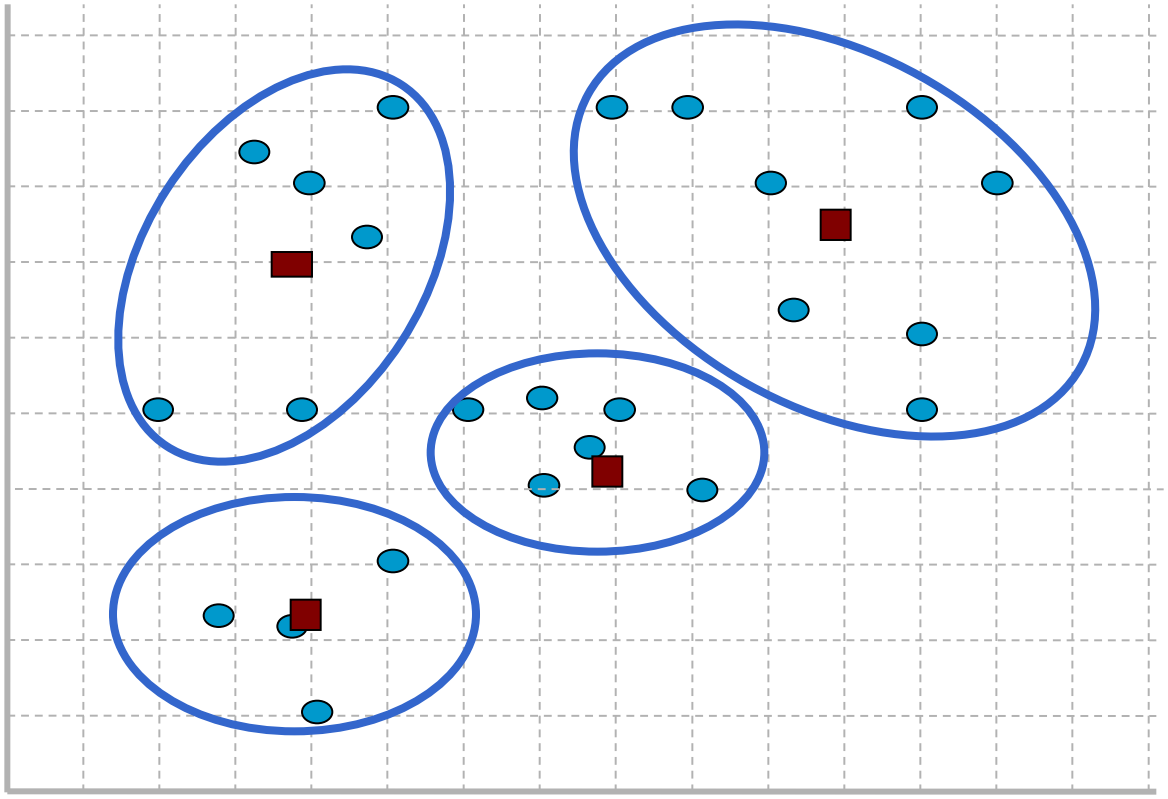


strong clusters

Partitioning - Initialization

- Strong forms:
  - Then take the centers of the strong forms as the initial points for k-means

Kmeans 3



Partitioning – Choosing K

- **Second problem**: how should we choose K?

1. **We choose K a priori**:
   - We have supplementary information
   - We study the histogram of the data
2. **We change K during the classification**
   - interactive unsupervised classification: the user creates / deletes classes
   - A statistical measure is used to update K

Partitioning – Choosing K

- A measure can be used that takes into account the variance of data in each class.

$$T = \frac{1}{n} \sum_{k=1}^{K} \sum_{x_i \in C_k} \left| x_i - g_k \right|^2$$

Partitioning – Choosing K

- Use the ISODATA algorithm:

  - **I**terative **S**elf-**O**rganizing **D**ata **A**nalysis **T**echnique **A**lgorithm
  - allows the number of clusters to be adjusted automatically during the iteration by **merging similar clusters** and **splitting clusters** with large standard deviations.

Partitioning – Choosing K

- Use the ISODATA algorithm:

- **Repeat**:
  - Perform a clustering by k-means
  - Remove too small clusters
  - Split dispersed clusters
  - Fuse clusters that are too close
- **Until convergence**

Fluid Partitioning

- Remember: $\forall k = 1 \ldots K, \forall x_i \in D, c_{i,k} \in [0,1]$

$$\forall k = 1 \ldots K, 0 < \sum_{i=1}^{n} c_{i,k} < n$$

- The number of objects in a class is less than the total number of objects

$$\forall x_i \in D, \sum_{k=1}^{K} c_{i,k} = 1$$

- An object belongs to different degrees, to all classes

Fuzzy C-means

- We will minimize a heuristic criterion:

$$J_{fuzzy} = \sum_{j=1}^{C} \sum_{i=1}^{n} c_{i,j}^{b} ||x_i - \text{centre}_j||^2$$

$$b \geq 1$$

- where b regulates the degree of „fuzzyness“ (in general b=2)

Fuzzy C-means

- **First step**:
  - fix a (random) number of clusters K, and a membership matrix C
- **Second step**:
  - Calculate the centroids $centre_k = \frac{\sum_{i=1}^{n} \left(c_{i,k}^{b}\right).x_i}{\sum_{i=1}^{n} \left(c_{i,k}^{b}\right)}$
- **Third step**:
  - Calculate the new membership matrix with the new clusters

$$c_{i,j} = \frac{\left(\frac{1}{d_{i,k}^2}\right)^{1/(b-1)}}{\sum_{j=1}^{C} \left(\frac{1}{d_{i,j}^2}\right)^{1/(b-1)}}$$

Fuzzy C-means

- **Fourth step**:
  - Recalculate J_{fuzzy}
    - we resume in step 1 if J_{fuzzy} has increased
    - it has been proved that J_{fuzzy} increases, except pathological case, and therefore that the algorithm converges to a local maximum

Fuzzy ISODATA

- If K is not known, we seek as in the "classic" ISODATA with criteria:
  - The fuzzy coefficient of partioning

$$F(M) = \frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{K} \left(c_{i,k}^{b} \right)$$

  - The entropy:

$$H(M) = \frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{K} c_{i,k} . \log(c_{i,k})$$

Fuzzy ISODATA

- Reminder on entropy:

$$H(M) = \sum_{k=1}^{K} p_k . \log\left(p_k\right) \text{ où } p_k = \frac{|C_k|}{|M|}$$

- H(M) is minimal (equal to 0) if the data are distributed in a single class
- H(M) is maximal (depends on the number of classes) if the data is distributed uniformly in all classes

Fuzzy ISODATA

- Reminder on entropy:

$$H(M) = \sum_{k=1}^{K} p_k . \log\left(p_k\right) \text{ où } p_k = \frac{|C_k|}{|M|}$$

- H(M) is minimal (equal to 0) if the data are distributed in a single class
- H(M) is maximal (depends on the number of classes) if the data is distributed uniformly in all classes

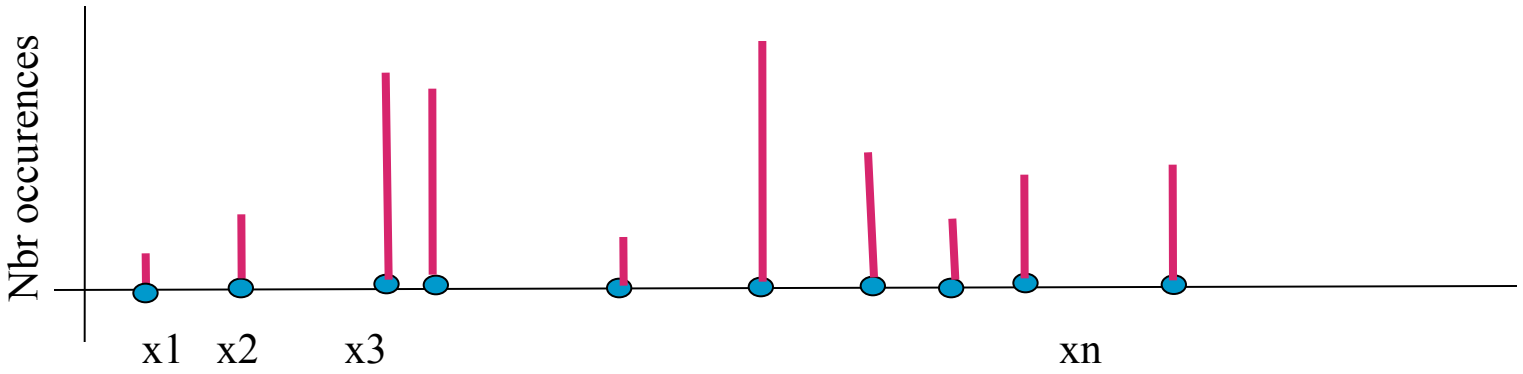
Clustering

- Different methods:
  - partitional clustering
  - **clustering by modeling**
  - hierarchical clustering

- Furthermore:
  - density-based clustering
  - grid-based clustering

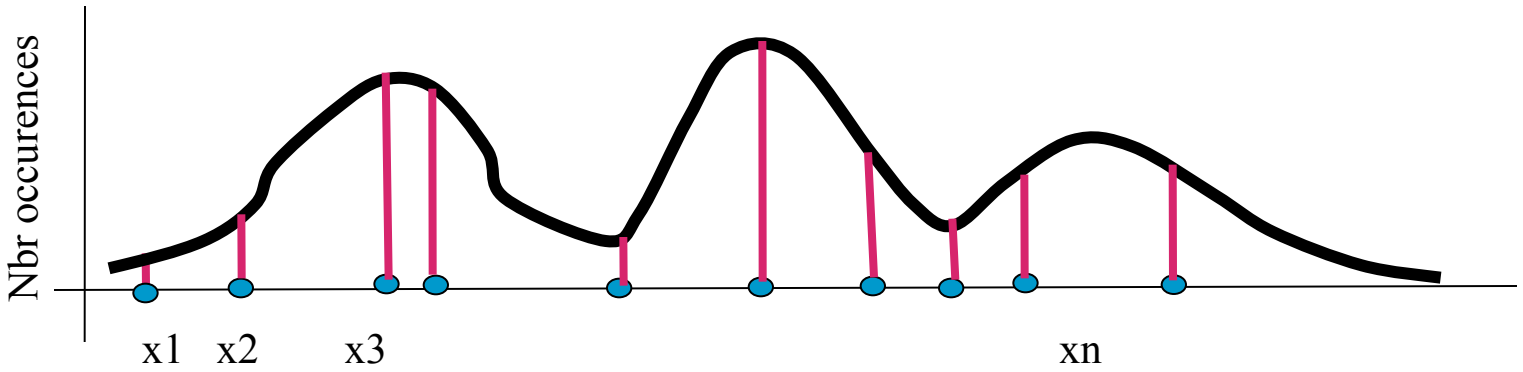
Model-based clustering

- The aim is to model each cluster by a probability distribution



Model-based clustering

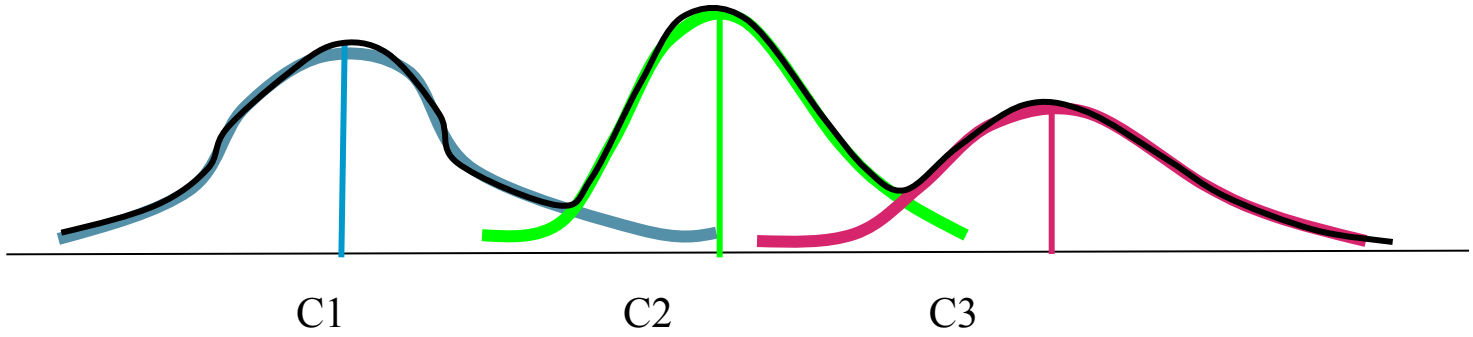
- The aim is to model each cluster by a probability distribution



Model-based clustering

- Assumed that this distribution is a mixture of multiple distributions such that each cluster can be represented using a normal distribution (**Gaussian mixture model**)

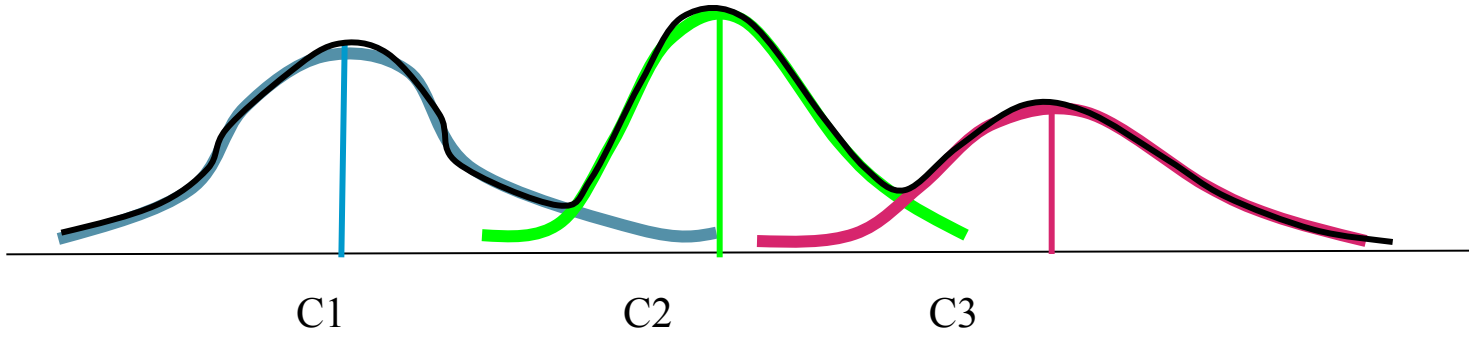
- Example with 3 clusters:



Model-based clustering

- Assumed that this distribution is a mixture of multiple distributions such that each cluster can be represented using a normal distribution (**Gaussian mixture model**)

- Example with 3 clusters, one dimensional data:



Model-based clustering

- Mixtures of Distributions:
  - We need to find the three distributions
  - We assume that each observation is derived from a single source C1, C2 or C3
  - for example, if we assume that the distributions are normal then, we must "find":

$$v = (\alpha_1, \mu_1, \sigma_1, \alpha_2, \mu_2, \sigma_2, \alpha_3, \mu_3, \sigma_3)$$

  - with:

$$p(x \mid v) = \sum_{k=1}^{3} \alpha_k N(x, \mu_k, \sigma_k) \quad \text{avec} \quad \sum_{k=1}^{K} \alpha_k = 1$$

Model-based clustering

- Mixtures of Distributions:
  - In general case we search:

$$v = (\alpha_1, \theta_1, \ldots, \alpha_k, \theta_k, \ldots, \alpha_K, \theta_K)$$

  - To maximize the probability

$$p(x \mid v) = \sum_{k=1}^{K} \alpha_k . p\left(x \mid C_k, \theta_k\right) \quad \text{avec} \quad \sum_{k=1}^{K} \alpha_k = 1$$

Model-based clustering

- Mixtures of Distributions:
  - We can compute the likelihood by:

$$L(v \mid x_1,\ldots,x_n) = \prod_{i=1}^{n} p(x_i \mid v) = \prod_{i=1}^{n}\prod_{k=1}^{K} \alpha_k . p(x_i \mid C_k, \theta_k)$$

- The likelihood quantifies the probability that the observations actually come from the distribution p

Model-based clustering

- Mixtures of Distributions:
  - We work with the natural logarithm to simplify the formulas

$$l(v \mid x_1,\ldots,x_n) = \ln\left(\prod_{i=1}^{n} p(x_i \mid v)\right) = \sum_{i=1}^{n} \ln\left(p(x_i \mid v)\right)$$

$$l(v \mid x_1,\ldots,x_n) = \sum_{i=1}^{n} \ln\left(\sum_{k=1}^{K} \alpha_k p(x_i \mid C_k, \theta_k)\right)$$

Model-based clustering

- Mixtures of Distributions:
  - We must therefore find the values of $\theta = \{\theta_k\}$ which maximize this likelihood.
  - Maximizing l(X,v) is difficult because there is no derivative for the logarithm of a sum
    - We need to find a way to simplify the computation of the likelihood
    - We will use an iterative process for the calculation (or estimation) of the maximum

Model-based clustering

- EM – Algorithm (Gaussian case):
  - We will therefore assume that each object comes from one of the distributions: but it is not known which
  - Let h_i be a variable giving the belonging of the object $o_{i:}$
    - h_k=1 if the object o_i belongs to class C_k
    - An object is therefore described by: $o_i=(x_i,h_1,...,h_k)$
  - The goal is to find:
    - The parameters of the distribution
    - For each object the distribution it came from

Model-based clustering

- EM – Algorithm (Gaussian case):
  - Two steps:
    - **Expectation**: with hypothesis θ_j
      - Compute h_i for all the points
      - Compute the expactation of the likelihood

$$Q = \sum_{i=1}^{n}\sum_{k=1}^{K} \ln\left(p\left(C_k \mid x_i, \theta_j\right)\right).p\left(C_k\right) + \sum_{i=1}^{n}\sum_{k=1}^{K} \ln\left(p\left(x_i \mid \theta_j\right)\right).p\left(C_k \mid x_i, \theta_j\right)$$

    - **Maximization**:
      - We search θ_{j+1} which maxmizes the expectation from the previous step

Model-based clustering

- EM – Algorithm (Gaussian case):
  - **Expectation**: with hypothesis θ_j
    - Compute the memebership function:

$$E\left(h_j^t = 1 \mid x_i, v^t\right) = \frac{\alpha_j^t N\left(x_i, \mu_j^t, \sigma_j^t\right)}{\sum_{k=1}^{K} E\left(h_i \mid x_i, v\right)^{(t)}}$$

- And let:

$$S_k^{(t+1)} = \sum_{i=1}^{n} E\left(h_i \mid x_i, v\right)^{(t)}$$

Model-based clustering

- EM – Algorithm (Gaussian case):
  - **Maximization**:
    - We search θ_{j+1} which maxmizes the expectation from the previous step

$$\alpha_k^{(t+1)} = \frac{1}{n}\sum_{i=1}^{n} S_k^{(t+1)}$$

$$\mu_k^{(t+1)} = \frac{1}{S_k^{(t+1)}}\sum_{i=1}^{n} x_i E\left(h_i \mid x_i, v\right)^{(t)}$$

$$\sigma_k^{2(t+1)} = \frac{1}{S_k^{(t+1)}}\sum_{i=1}^{n}\left(x_i - \mu_k^{(t+1)}\right)^2 E\left(h_i \mid x_i, v\right)^{(t)}$$

Model-based clustering

- EM – Algorithm (Gaussian case):
  - **Example**: