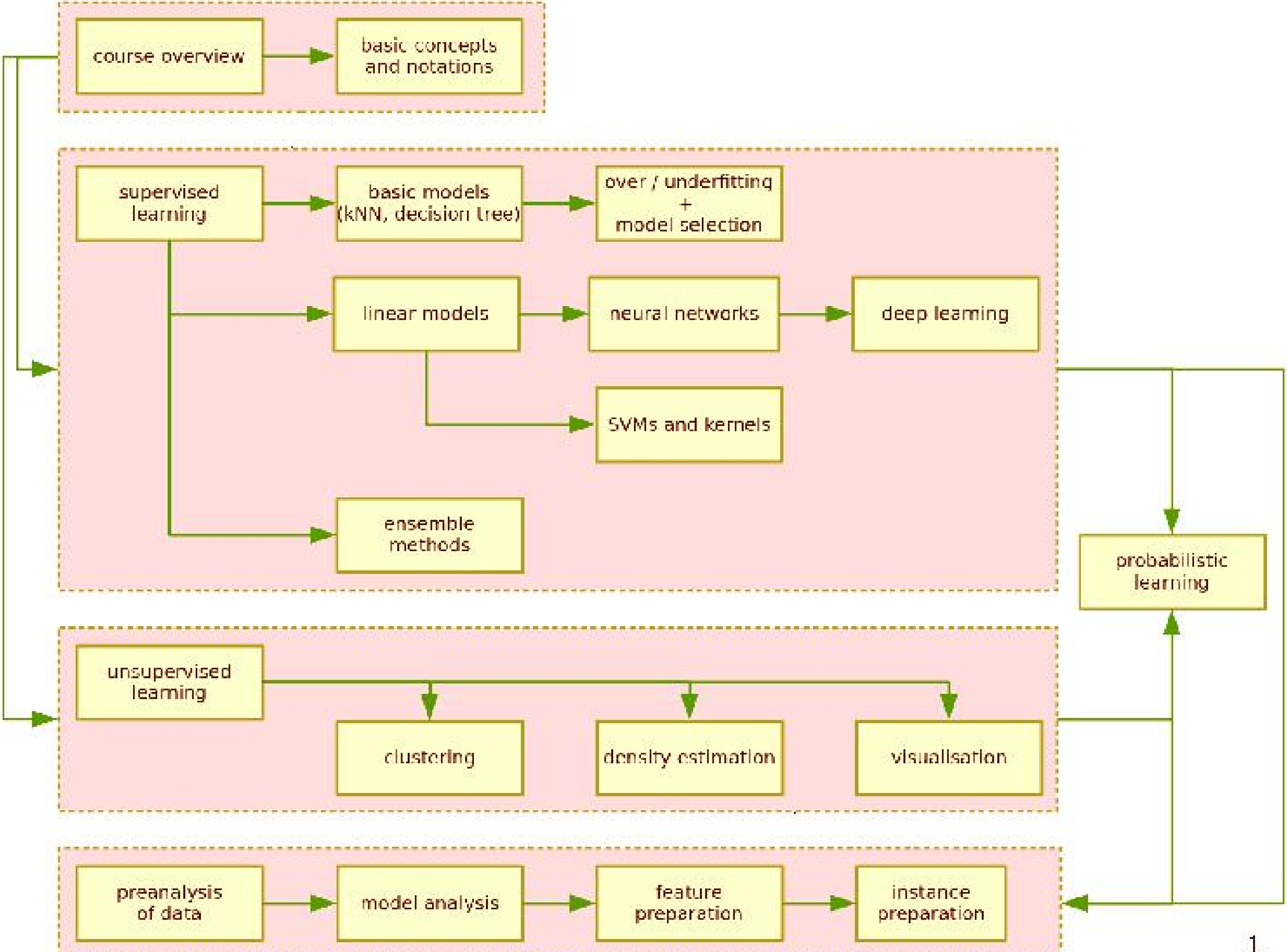
Machine Learning: Lesson 12

Clustering

Benoît Frénay - Faculty of Computer Science

UNIVERSITÉ DE NAMUR


course overview
basic concepts and notations
supervised learning
basic models (kNN, decision tree)
over / underfitting + model selection
linear models
neural networks
deep learning
SVMs and kernels
ensemble methods
probabilistic learning
unsupervised learning
clustering
density estimation
visualisation
preanalysis of data
model analysis
feature preparation
instance preparation

Outline of this Lesson

- clustering
  - problem statement
  - the k-means algorithm
  - choosing the number of clusters
  - the DBSCAN algorithm
- application: clustering of geographical curves

Clustering: Problem Statement

Definition of Clustering

Statistical Pattern Recognition by Web and Copsey

Cluster analysis is the grouping of individuals in a population in order to discover structure in the data. In some sense, we would like the individuals within a group to be close or similar to one another, but dissimilar from individuals in other groups.

Pattern Recognition and Machine Learning by Bishop

Clustering is the problem of identifying groups, or clusters, of data points in a multidimensional space. Intuitively, we might think of a cluster as comprising a group of data points whose inter-point distances are small compared with the distances to points outside of the cluster. We can formalize this notion by first introducing a set of prototypes representing the centres of the clusters.

Definition of Clustering

Statistical Pattern Recognition by Web and Copsey

Cluster analysis is the grouping of individuals in a population in order to discover structure in the data. In some sense, we would like the individuals within a group to be close or similar to one another, but dissimilar from individuals in other groups.

Pattern Recognition and Machine Learning by Bishop

Clustering is the problem of identifying groups, or clusters, of data points in a multidimensional space. Intuitively, we might think of a cluster as comprising a group of data points whose inter-point distances are small compared with the distances to points outside of the cluster. We can formalize this notion by first introducing a set of prototypes representing the centres of the clusters.

Definition of Clusters

No universal definition

- each method implicitly assumes a given structure
- some methods can produce clusters even if there are none

Examples of definition

- groups of instances which are close to prototypes
- regions of high density separated by regions of low density

Clustering: the k-means Algorithm

The k-means Algorithm

Characteristics

- iterative procedure to find k clusters
- summarise each cluster by a centroid/prototype
- find prototypes which are the most representative
- many extensions (fuzzy k-means, k-medoids...)

Alternate names

c-means, iterative relocation, basic ISODATA, generalised Lloyd algorithm

Derivation of the k-means Algorithm

Notations

- $\mathcal{C} = \{\mathbf{z}_j\}$: codebook of centroids
- $y(\mathbf{x})$ = index of the centroid to which is assigned instance $\mathbf{x}$

Objective function

minimise reconstruction error with the codebook of centroids

$$J(\mathcal{C}) = \int_{\mathbf{x}} p(\mathbf{x}) d(\mathbf{x}, \mathbf{z}_{y(\mathbf{x})})^2 d\mathbf{x}$$

which is approximated by the training reconstruction error

$$J(\mathcal{C}) = \frac{1}{n} \sum_{i=1}^{n} d(\mathbf{x}_i, \mathbf{z}_{y(\mathbf{x}_i)})^2$$

Derivation of the k-means Algorithm

Notations

- $\mathcal{C} = \{\mathbf{z}_j\}$: codebook of centroids
- $y(\mathbf{x})$ = index of the centroid to which is assigned instance $\mathbf{x}$

Objective function

minimise reconstruction error with the codebook of centroids

$$J(\mathcal{C}) = \int_{\mathbf{x}} p(\mathbf{x}) d(\mathbf{x}, \mathbf{z}_{y(\mathbf{x})})^2 d\mathbf{x}$$

which is approximated by the training reconstruction error

$$J(\mathcal{C}) = \frac{1}{n} \sum_{i=1}^{n} d(\mathbf{x}_i, \mathbf{z}_{y(\mathbf{x}_i)})^2$$

Derivation of the k-means Algorithm

Encoding-decoding view

$$\mathbf{x}_i \longrightarrow \boxed{\text{encoder}} \longrightarrow \text{index } y(\mathbf{x}_i) \longrightarrow \boxed{\text{decoder}} \longrightarrow \text{centroid } \mathbf{z}_{y(\mathbf{x}_i)}$$

goal : encoder and decoder which minimise the training reconstruction error

$$J(\mathcal{C}) = \frac{1}{n} \sum_{i=1}^{n} d(\mathbf{x}_i, \mathbf{z}_{y(\mathbf{x}_i)})^2$$

Approximate solution with the k-means algorithm

- no analytical solution to the encoder-decoder problem
- k-means algorithm: iterative, greedy algorithm
- start from an initial decoder (codebook), then improve it

k-means Algorithm: Encoding Step

decoder/codebook is known, what is the best encoder/assignment ?

Optimal solution for the encoding step

$$\mathbf{x}_i \longrightarrow \boxed{\textbf{encoder}} \longrightarrow \text{index } y(\mathbf{x}_i) \longrightarrow \boxed{\text{decoder}} \longrightarrow \text{centroid } \mathbf{z}_{y(\mathbf{x}_i)}$$

first step: encoder which minimise the training reconstruction error

$$J(\mathcal{C}) = \frac{1}{n}\sum_{i=1}^{n} d(\mathbf{x}_i, \mathbf{z}_{y(\mathbf{x}_i)})^2$$

solution: assign $\mathbf{x}_i$ to the closest centroid $\mathbf{z}_{y(\mathbf{x}_i)}$

$$y(\mathbf{x}_i) = \arg\min_{j=1\ldots k} d(\mathbf{x}_i, \mathbf{z}_j)$$

k-means Algorithm: Encoding Step

decoder/codebook is known, what is the best encoder/assignment ?

Optimal solution for the encoding step

$$\mathbf{x}_i \longrightarrow \boxed{\text{encoder}} \longrightarrow \text{index } y(\mathbf{x}_i) \longrightarrow \boxed{\text{decoder}} \longrightarrow \text{centroid } \mathbf{z}_{y(\mathbf{x}_i)}$$

first step: encoder which minimise the training reconstruction error

$$J(\mathcal{C}) = \frac{1}{n} \sum_{i=1}^{n} d(\mathbf{x}_i, \mathbf{z}_{y(\mathbf{x}_i)})^2$$

solution: assign $\mathbf{x}_i$ to the closest centroid $\mathbf{z}_{y(\mathbf{x}_i)}$

$$y(\mathbf{x}_i) = \arg \min_{j=1 \ldots k} d(\mathbf{x}_i, \mathbf{z}_j)$$

k-means Algorithm: Decoding Step

encoder/assignment is known, what is the best decoder/codebook ?

Optimal solution for the decoding step

$$\mathbf{x}_i \longrightarrow \boxed{\text{encoder}} \longrightarrow \text{index } y(\mathbf{x}_i) \longrightarrow \boxed{\textbf{decoder}} \longrightarrow \text{centroid } \mathbf{z}_{y(\mathbf{x}_i)}$$

second step: decoder which minimise the training reconstruction error

$$J(\mathcal{C}) = \frac{1}{n}\sum_{i=1}^{n} d(\mathbf{x}_i, \mathbf{z}_{y(\mathbf{x}_i)})^2 = \sum_{j=1}^{k}\left(\frac{|\mathcal{C}_j|}{n}\right)\left(\frac{1}{|\mathcal{C}_j|}\sum_{i\in\mathcal{C}_j} d(\mathbf{x}_i, \mathbf{z}_{y(\mathbf{x}_i)})^2\right)$$

solution: move centroid $\mathbf{z}_j$ to the center of gravity of cluster $\mathcal{C}_j$

$$\mathbf{z}_j = \frac{1}{|\mathcal{C}_j|}\sum_{i\in\mathcal{C}_j} \mathbf{x}_i$$

k-means Algorithm: Decoding Step

encoder/assignment is known, what is the best decoder/codebook ?

Optimal solution for the decoding step

$$\mathbf{x}_i \longrightarrow \boxed{\text{encoder}} \longrightarrow \text{index } y(\mathbf{x}_i) \longrightarrow \boxed{\textbf{decoder}} \longrightarrow \text{centroid } \mathbf{z}_{y(\mathbf{x}_i)}$$

second step: decoder which minimise the training reconstruction error

$$J(\mathcal{C}) = \frac{1}{n}\sum_{i=1}^{n} d(\mathbf{x}_i, \mathbf{z}_{y(\mathbf{x}_i)})^2 = \sum_{j=1}^{k}\left(\frac{|\mathcal{C}_j|}{n}\right)\left(\frac{1}{|\mathcal{C}_j|}\sum_{i\in\mathcal{C}_j} d(\mathbf{x}_i, \mathbf{z}_{y(\mathbf{x}_i)})^2\right)$$

solution: move centroid $\mathbf{z}_j$ to the center of gravity of cluster $\mathcal{C}_j$

$$\mathbf{z}_j = \frac{1}{|\mathcal{C}_j|}\sum_{i\in\mathcal{C}_j}\mathbf{x}_i$$

Details of the k-means Algorithm

k-means algorithm

Input: dataset $\mathcal{D} = \{\mathbf{x}_i\}$ and number k of clusters
Output: codebook $\mathcal{C} = \{\mathbf{z}_j\}$ and assignment function y

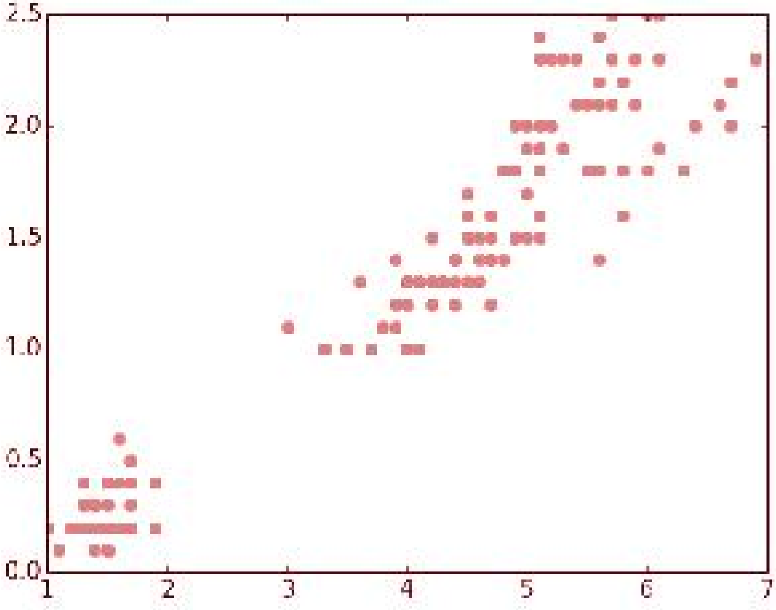
while termination criterion is not met **do**
  // encoding/assignment step
  for each instance $\mathbf{x}_i$ **do**
    $y(\mathbf{x}_i) = \arg\min_{j=1\dots k} d(\mathbf{x}_i, \mathbf{z}_j)$
  end for

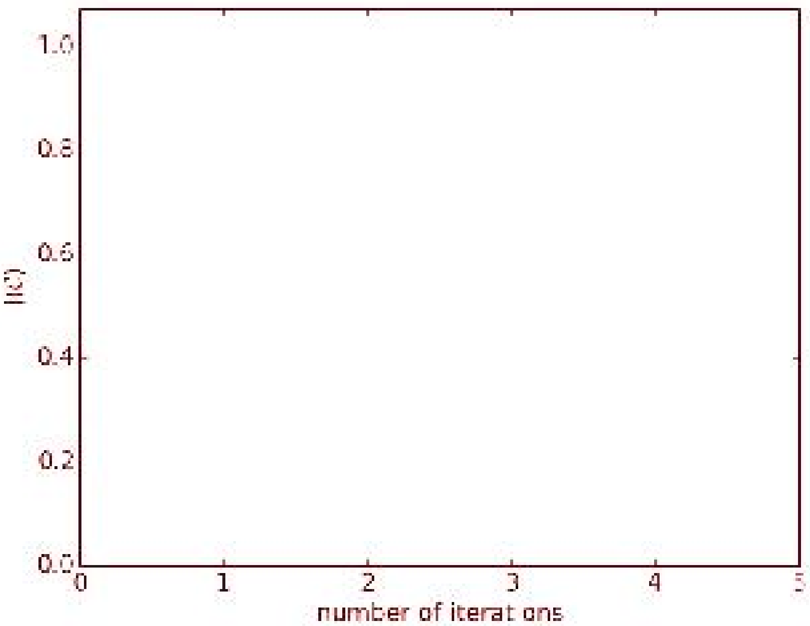
  // decoding/codebook update step
  for each centroid $\mathbf{z}_j$ **do**
    $\mathbf{z}_j = \frac{1}{|\mathcal{C}_j|} \sum_{i \in \mathcal{C}_j} \mathbf{x}_i$
  end for
end while

termination: # of iterations, change of successive codebooks / $J(\mathcal{C})$ values

Example of Clustering with k-means

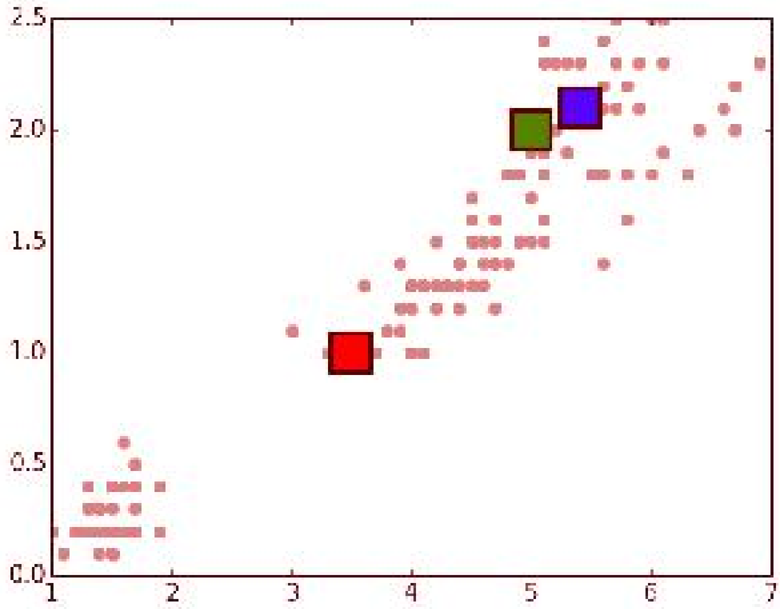
data points (Iris dataset with $n = 150$)

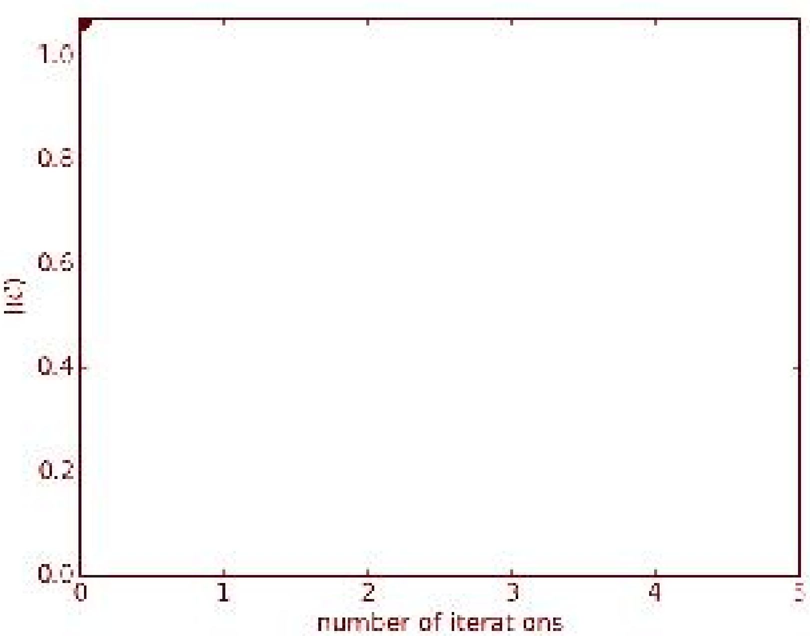




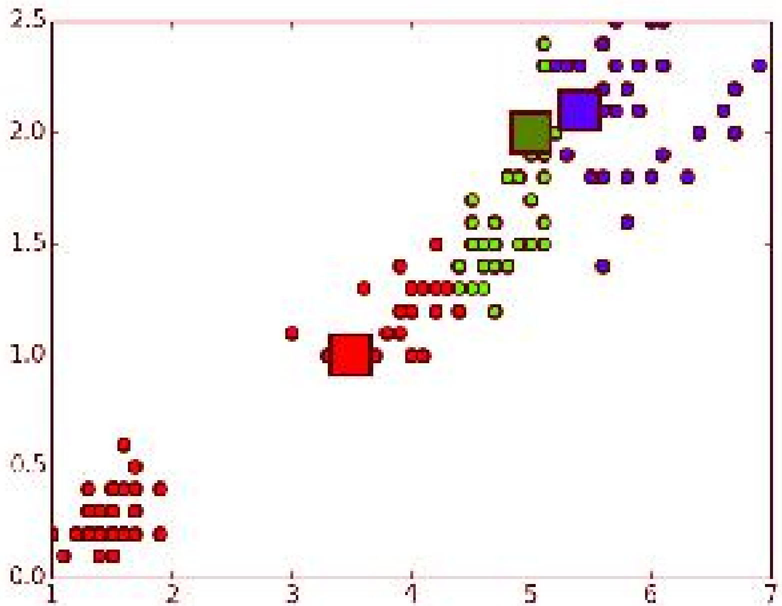
Example of Clustering with k-means

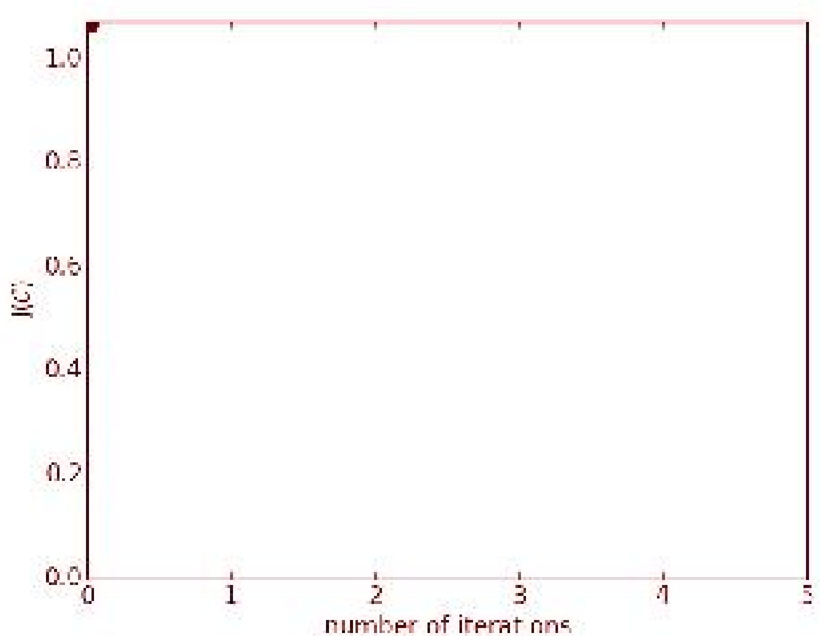
initial prototypes (randomly chosen)





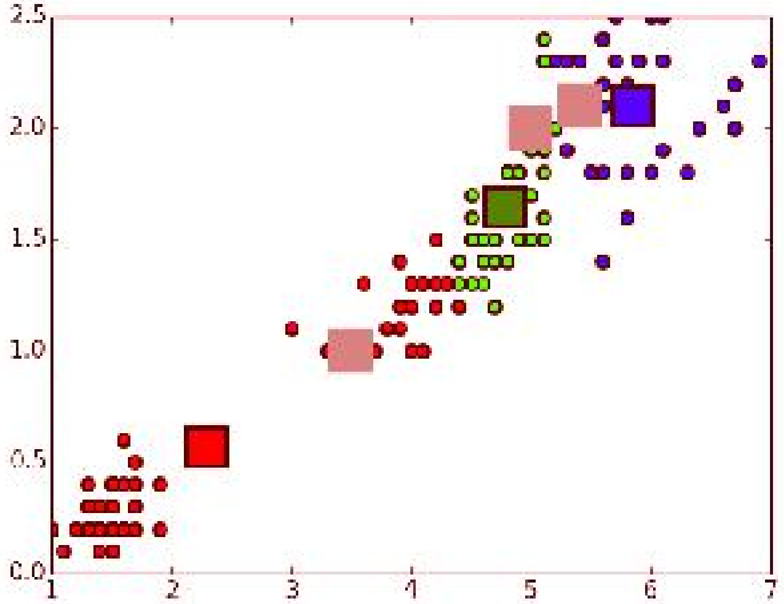
iteration 1: assignment to clusters

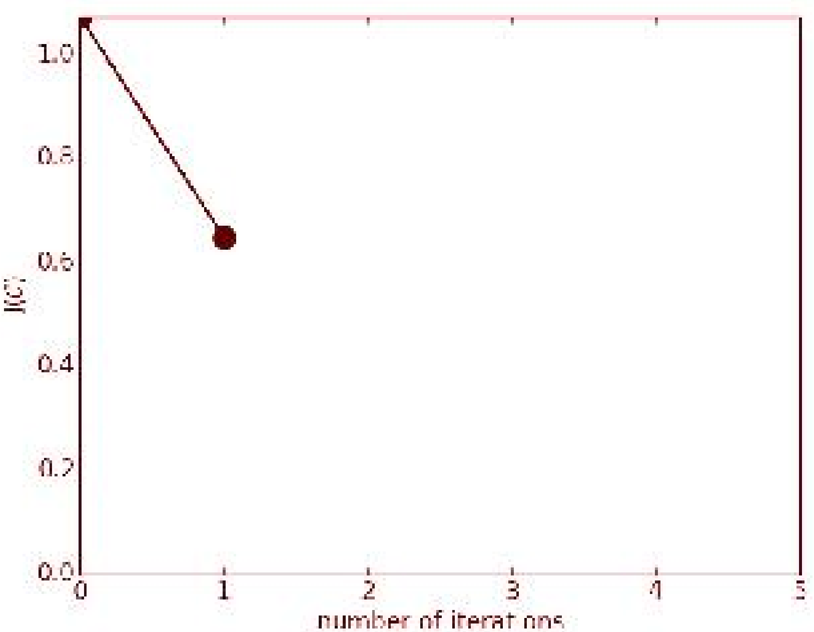




Example of Clustering with k-means

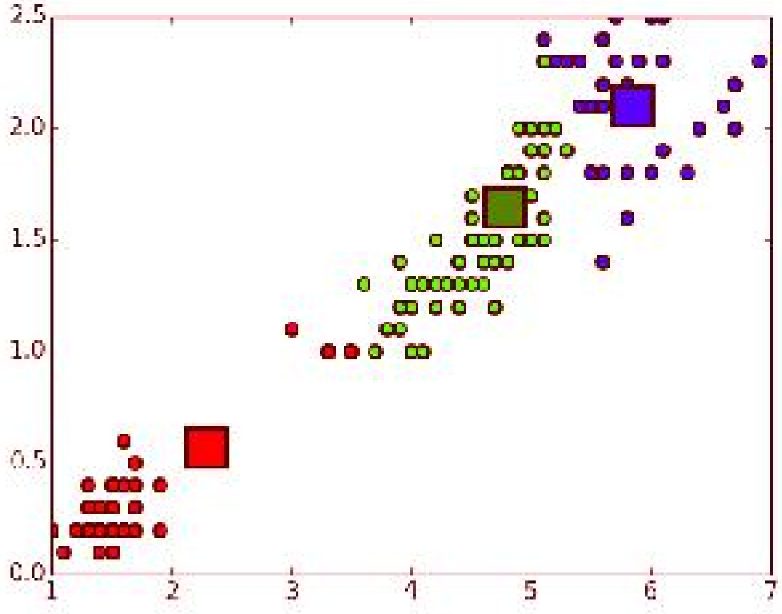
iteration 1: update of the centroids

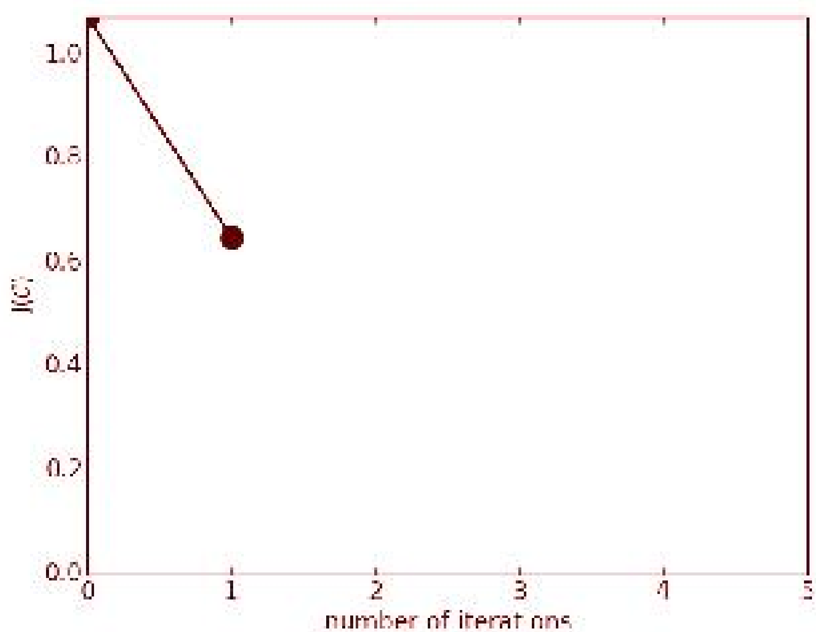




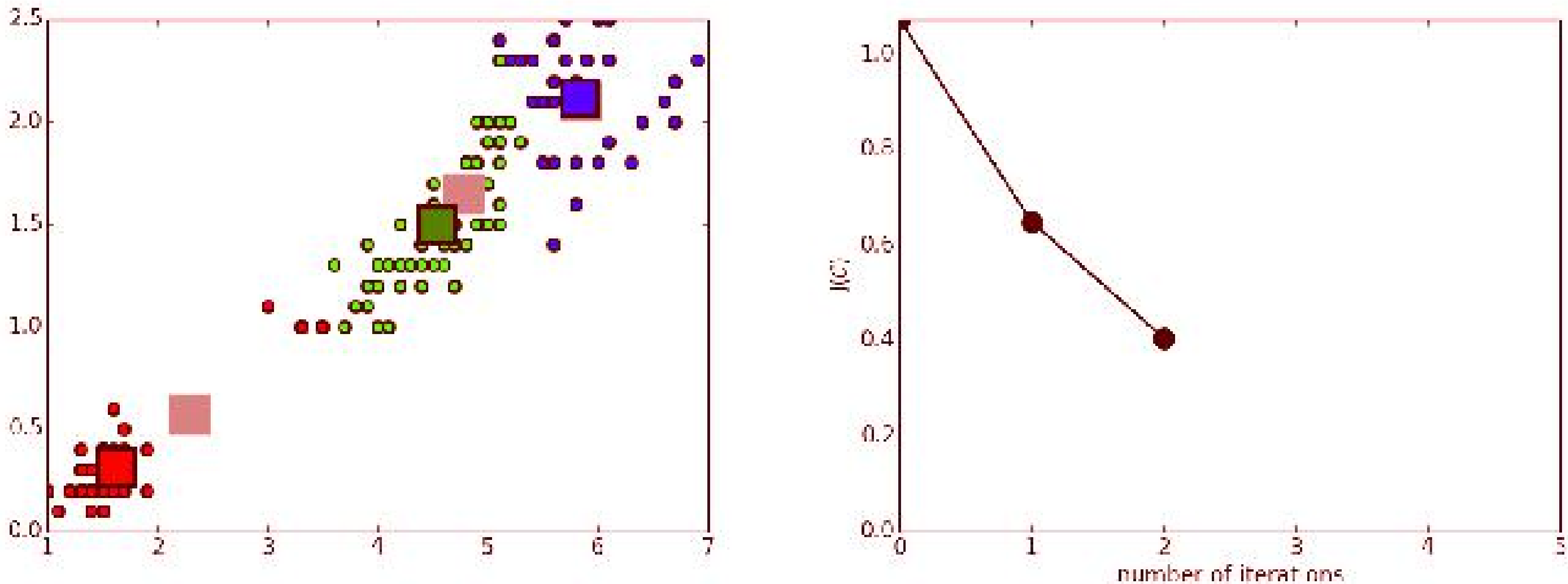
Example of Clustering with k-means

iteration 2: assignment to clusters



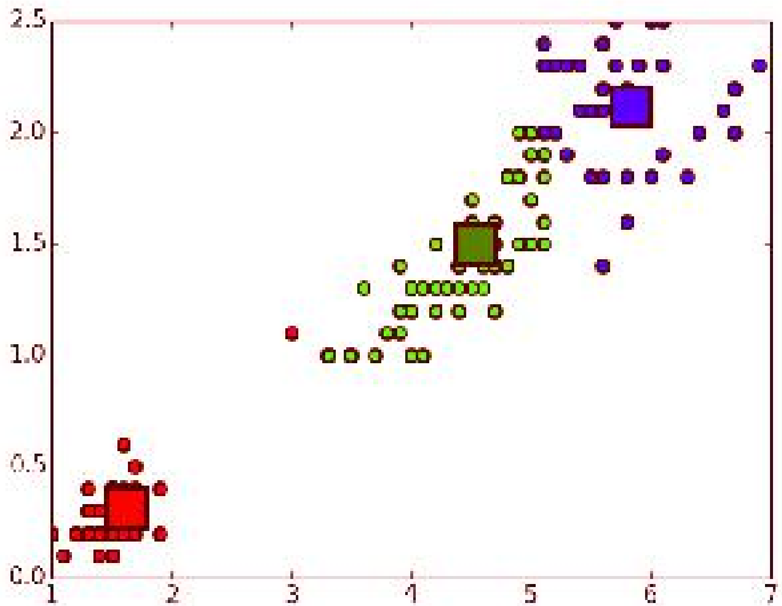


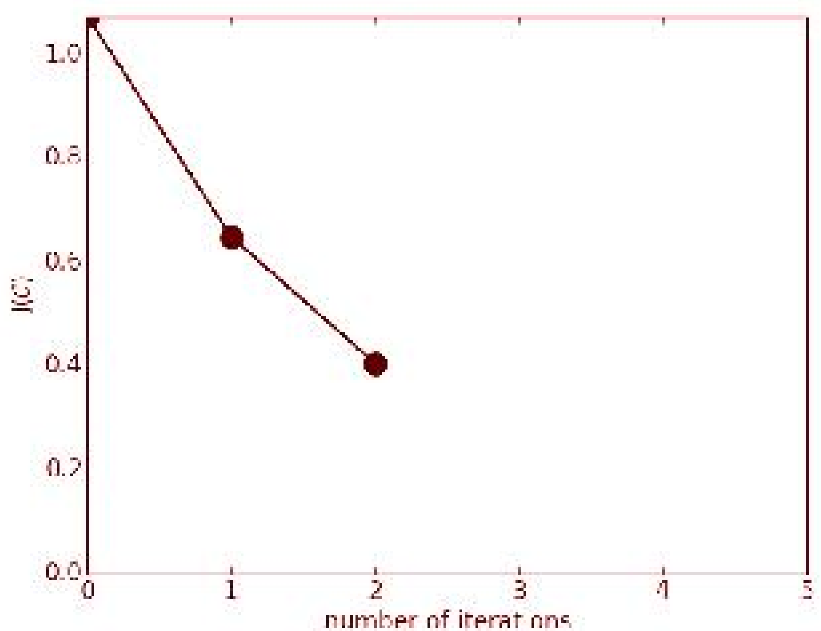
iteration 2: update of the centroids



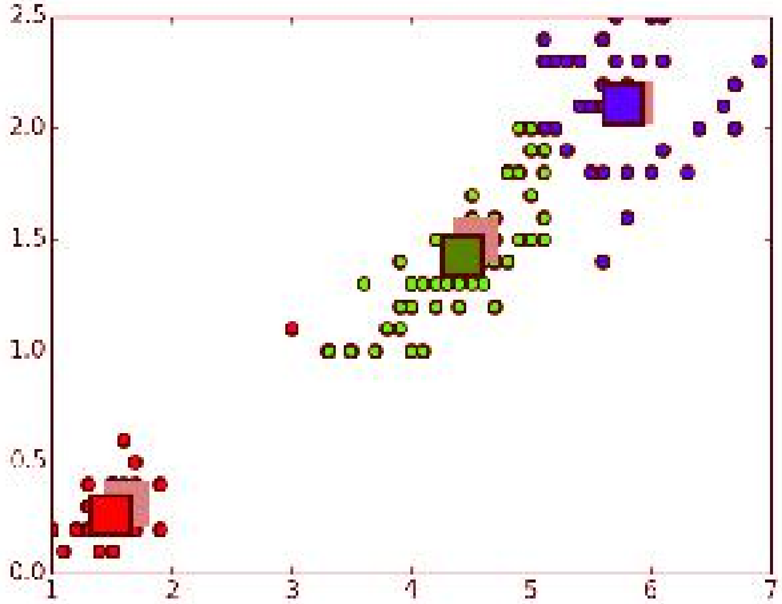
Example of Clustering with k-means

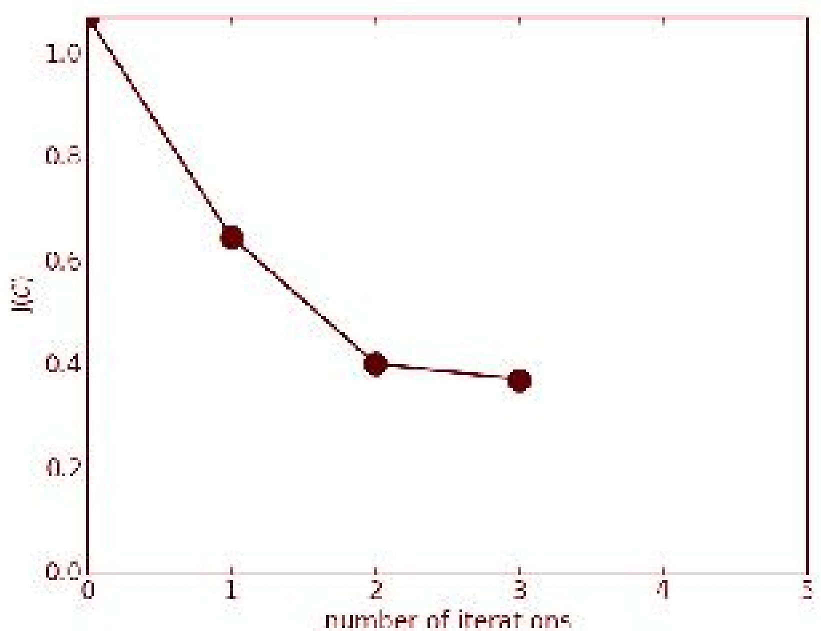
iteration 3: assignment to clusters





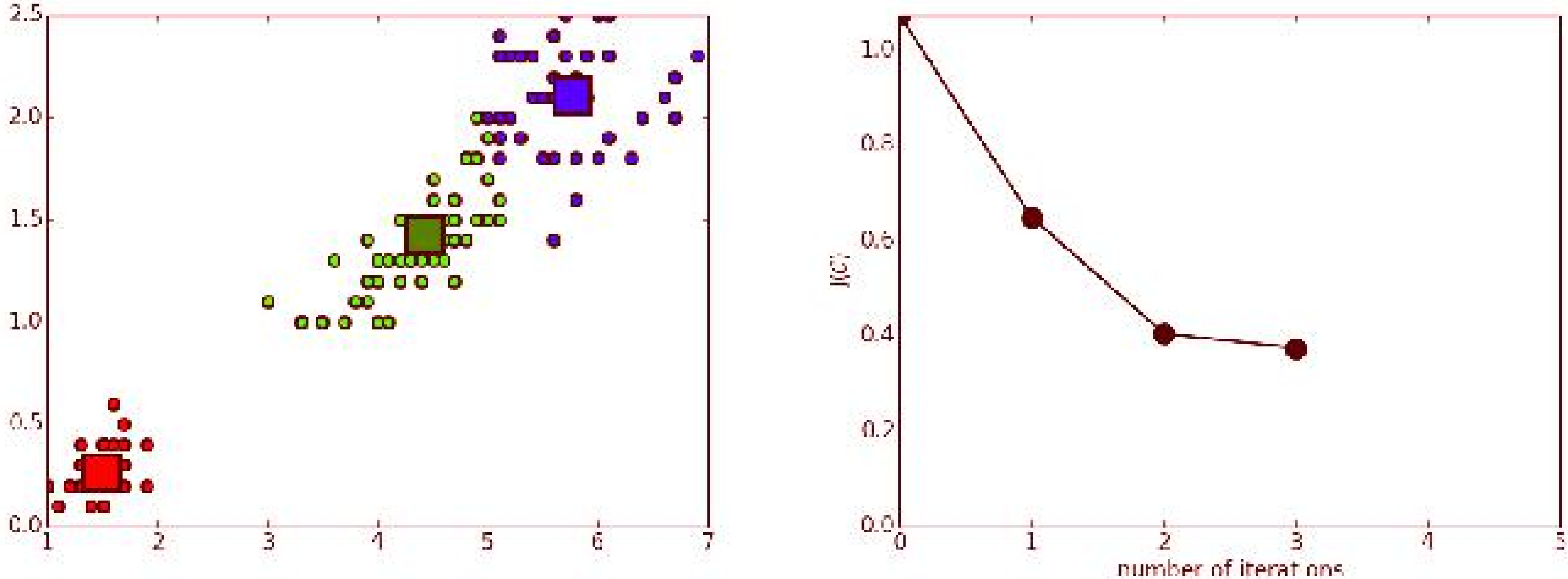
iteration 3: update of the centroids





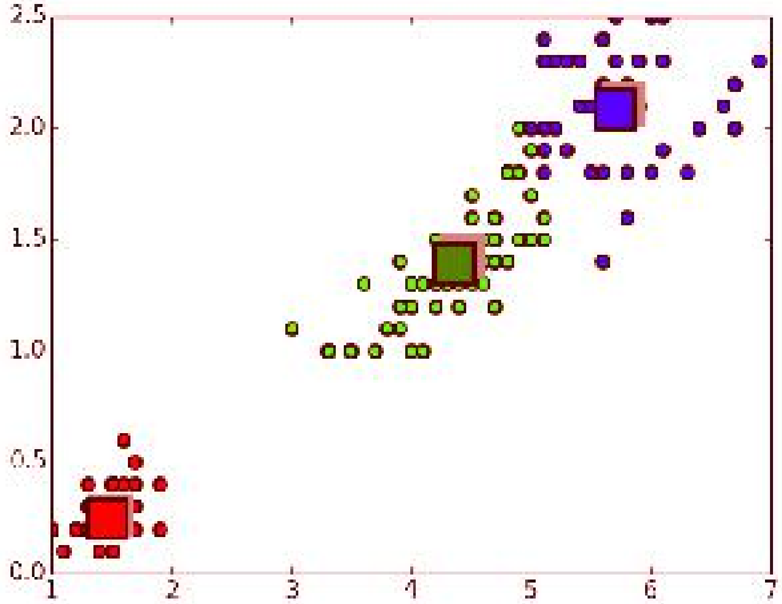
Example of Clustering with k-means

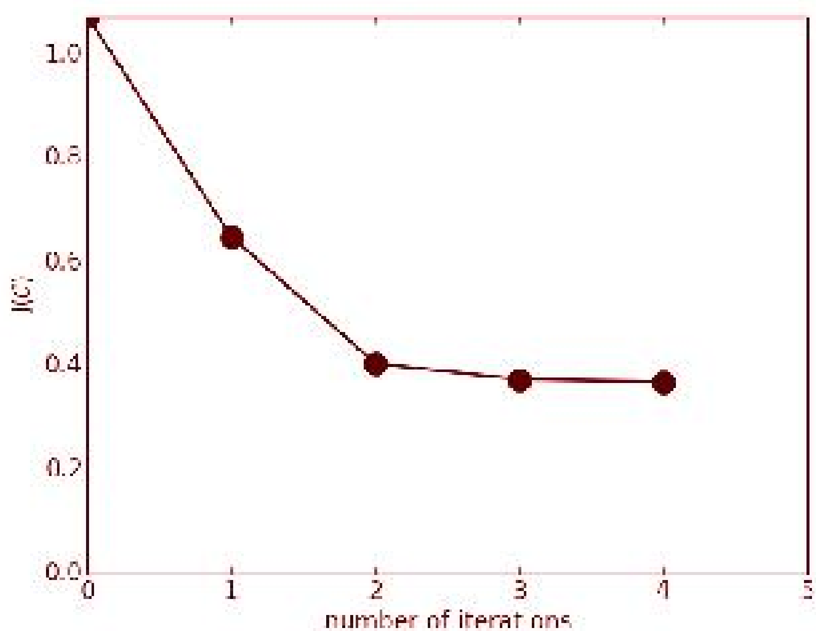
iteration 4: assignment to clusters



Example of Clustering with k-means

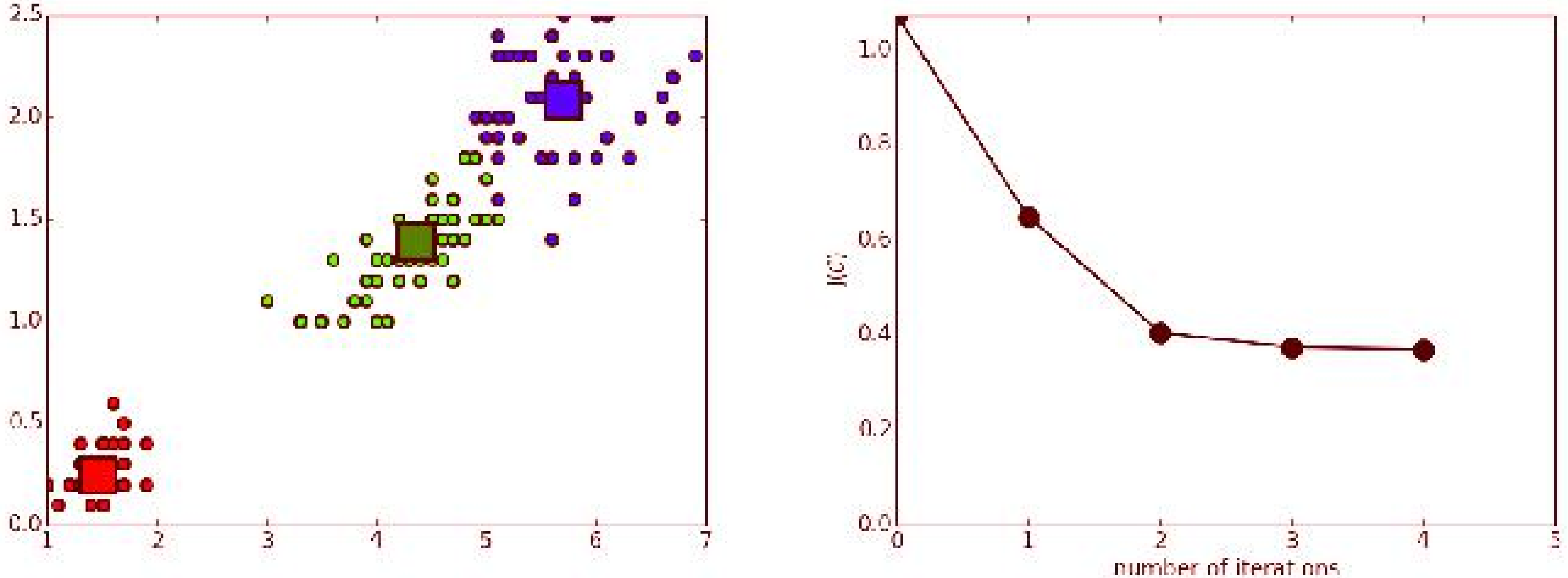
iteration 4: update of the centroids





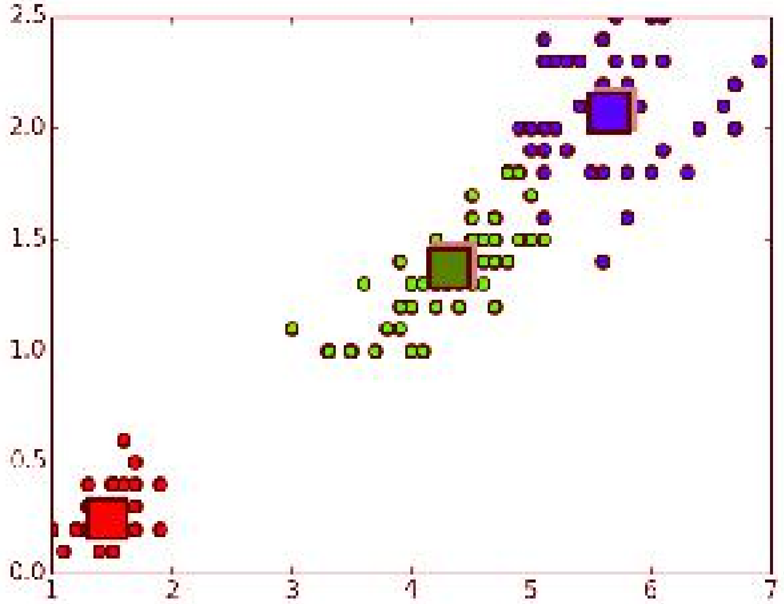
Example of Clustering with k-means

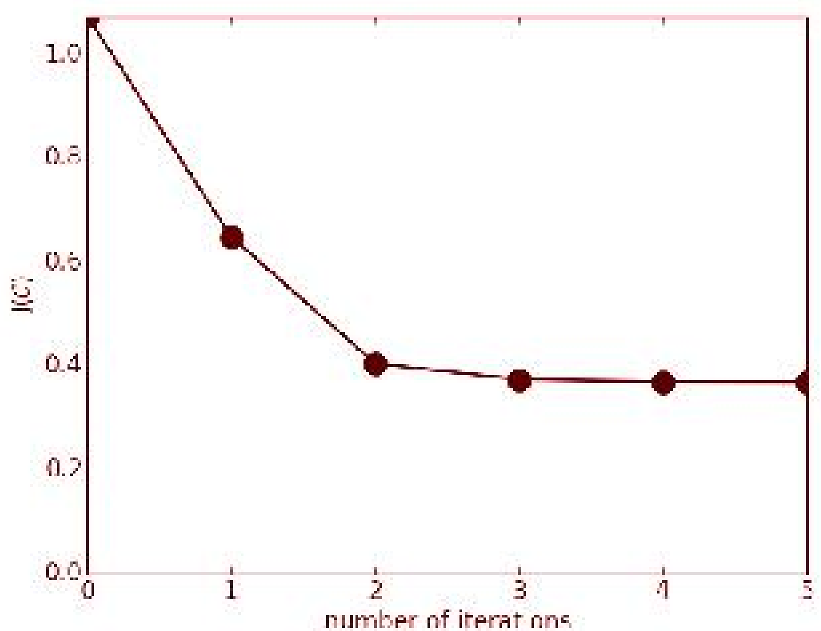
iteration 5: assignment to clusters



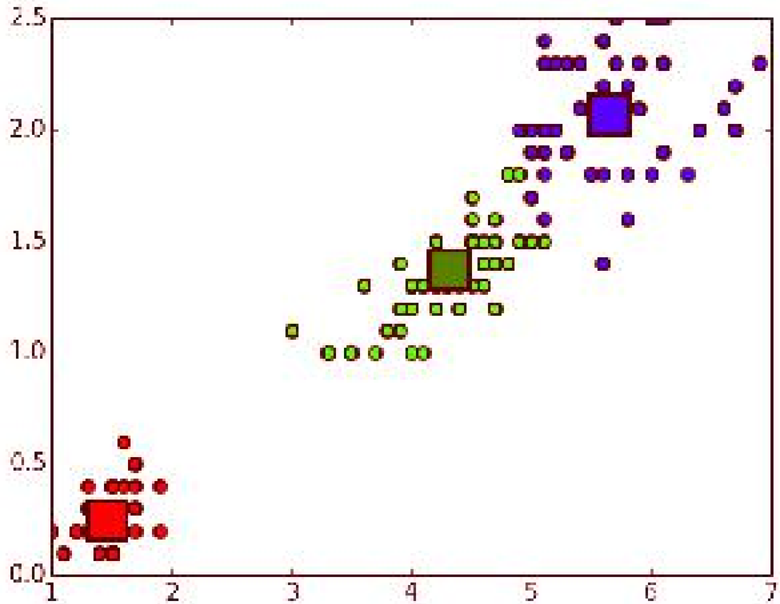
Example of Clustering with k-means

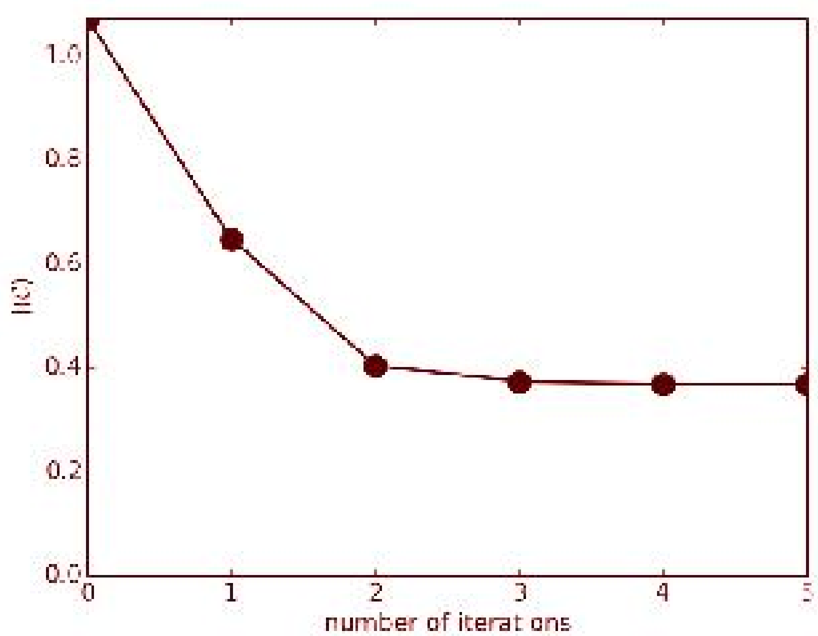
iteration 5: update of the centroids





final solution (5 iterations)





Pros and Cons of the k-means Algorithm

Advantages

- ✓ simple to understand and implement
- ✓ very fast (compute distances + $\arg\min$ and mean operations)

Convergence analysis

- × only converges to a local minimum of $J(\mathcal{C})$
- × many restarts are necessary in practice (thousands)

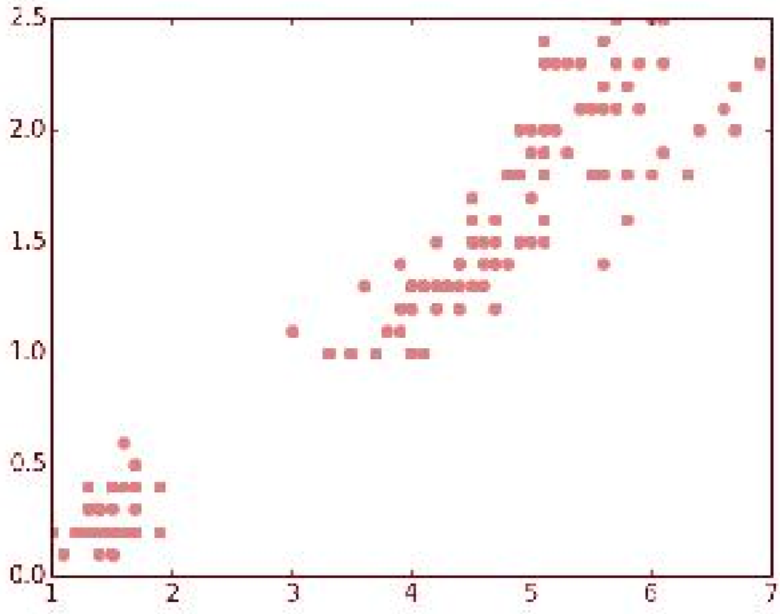
Limitations

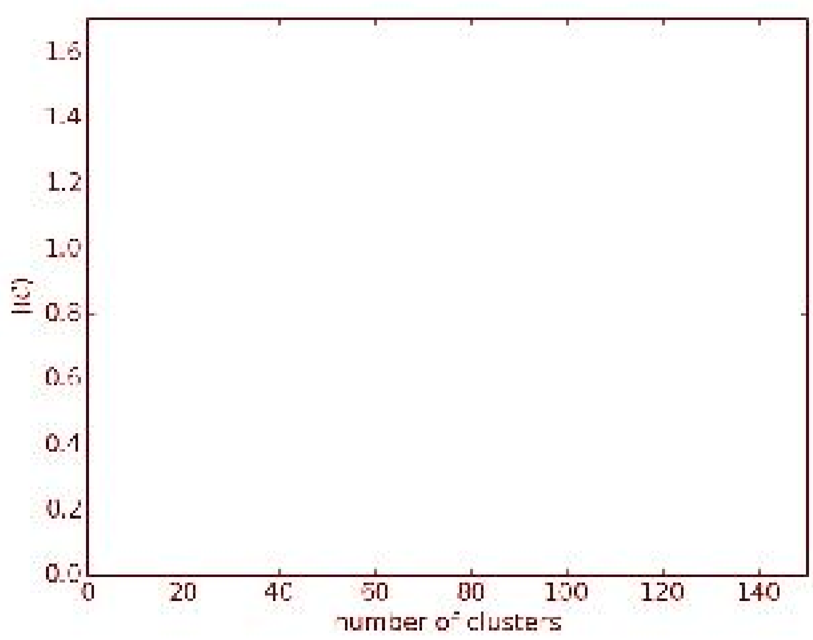
- × each instance belongs to one cluster (crisp assignment)
- × fuzzy extensions compute memberships to each cluster

Clustering: Choosing the Number of Clusters

How Many Clusters should we Use with k-means ?

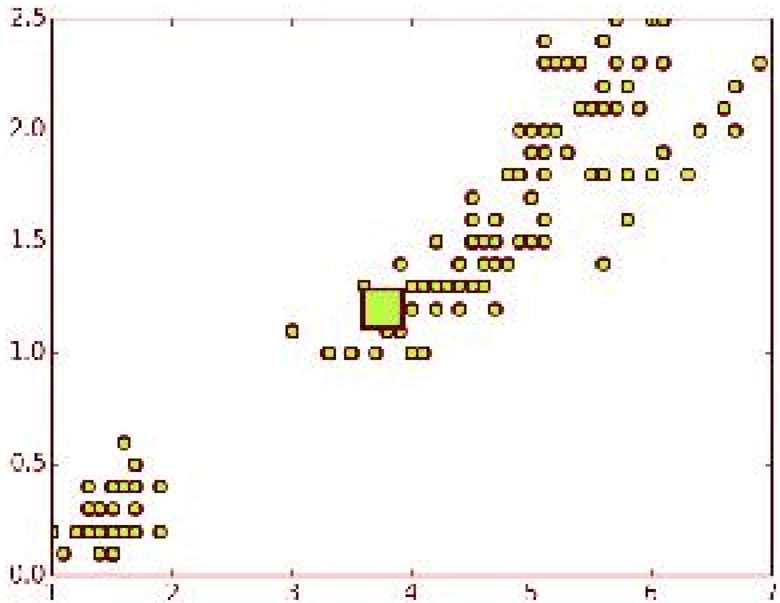
data points (Iris dataset with $n = 150$)

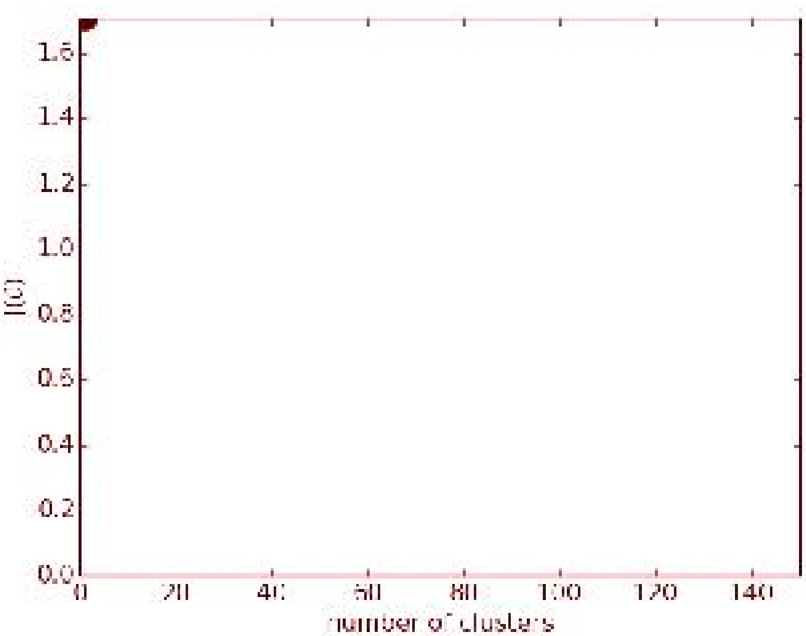




How Many Clusters should we Use with k-means ?

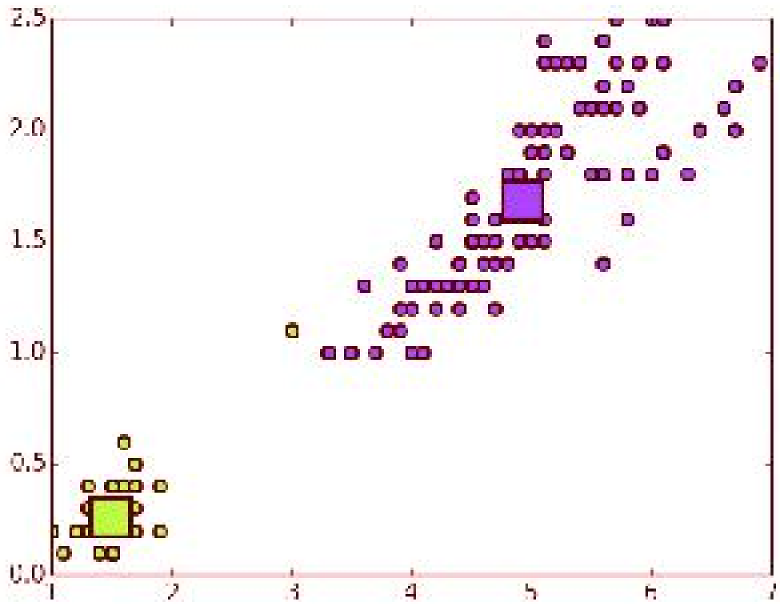
k-means solution with $k = 1$ clusters

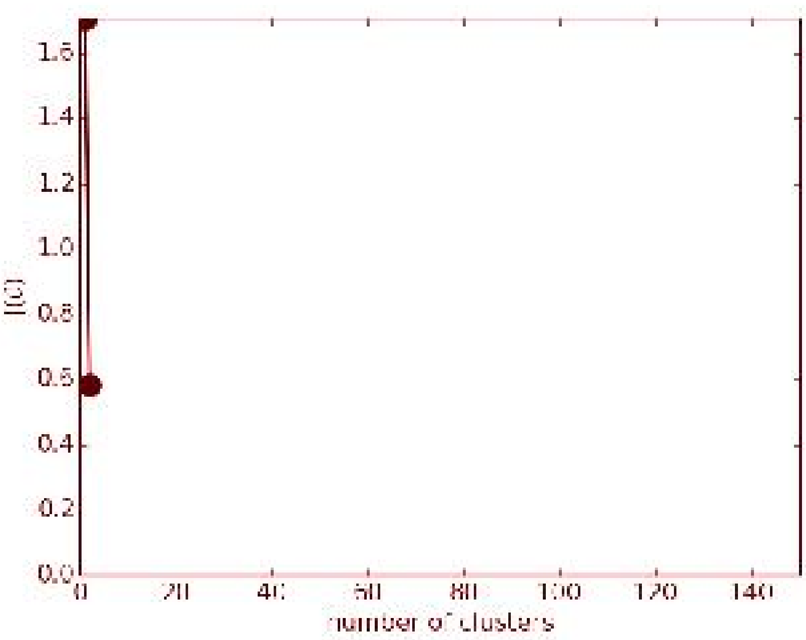




How Many Clusters should we Use with k-means ?

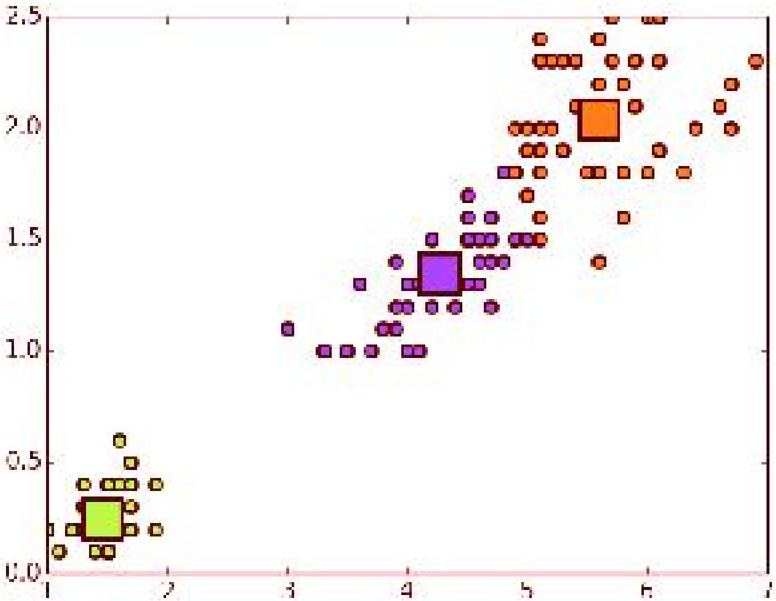
k-means solution with $k = 2$ clusters

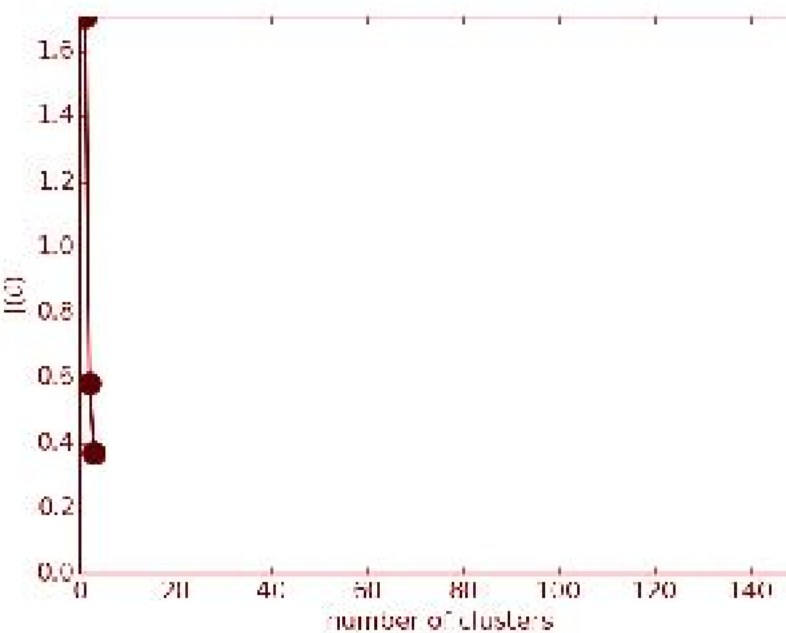




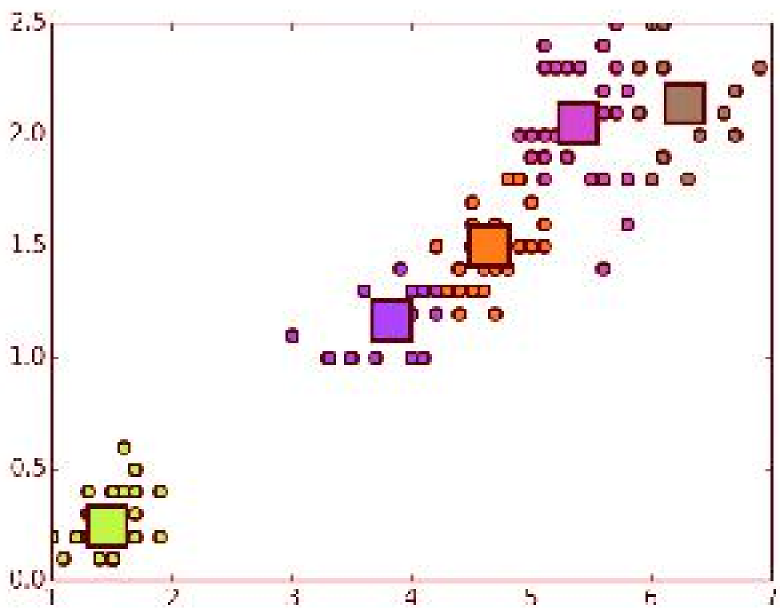
How Many Clusters should we Use with k-means ?

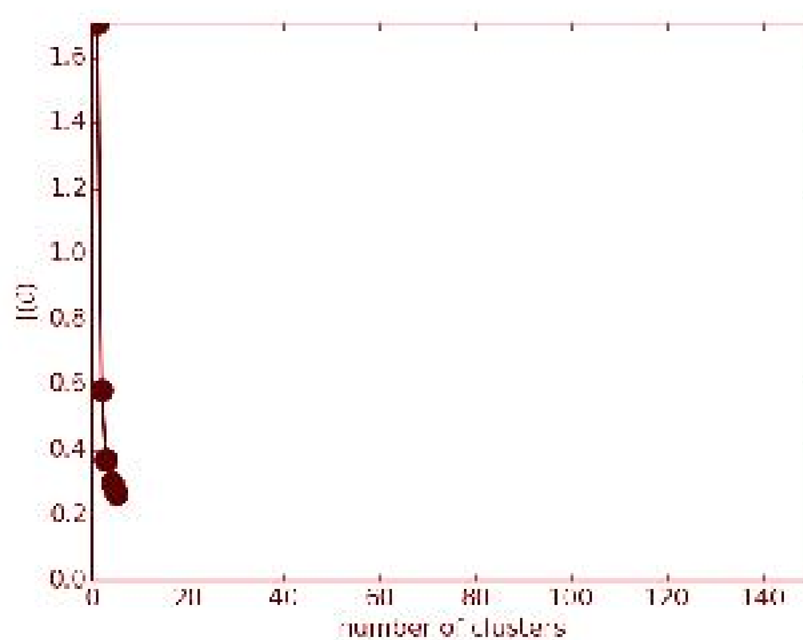
k-means solution with $k = 3$ clusters





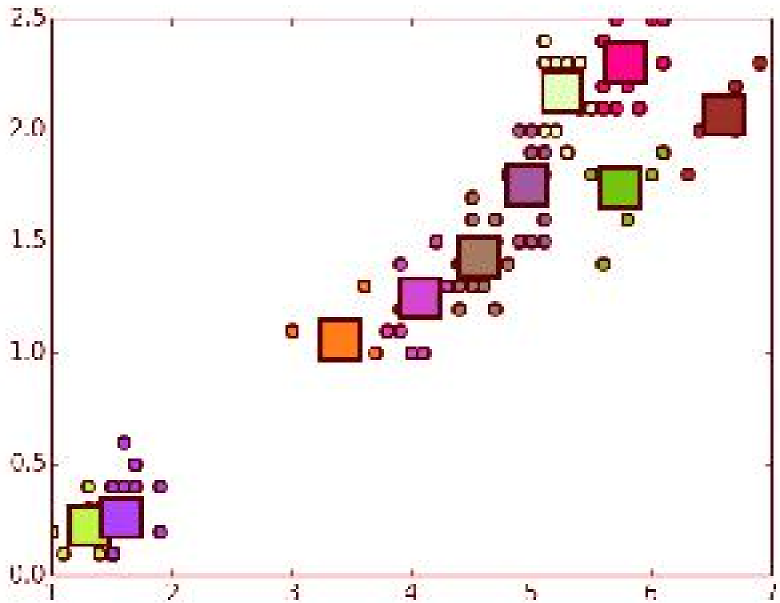
k-means solution with $k = 5$ clusters

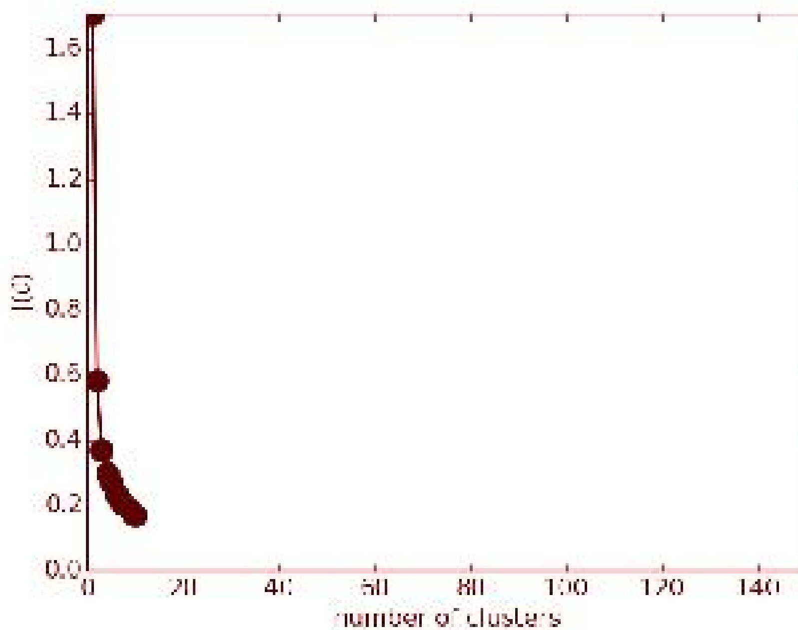




How Many Clusters should we Use with k-means ?

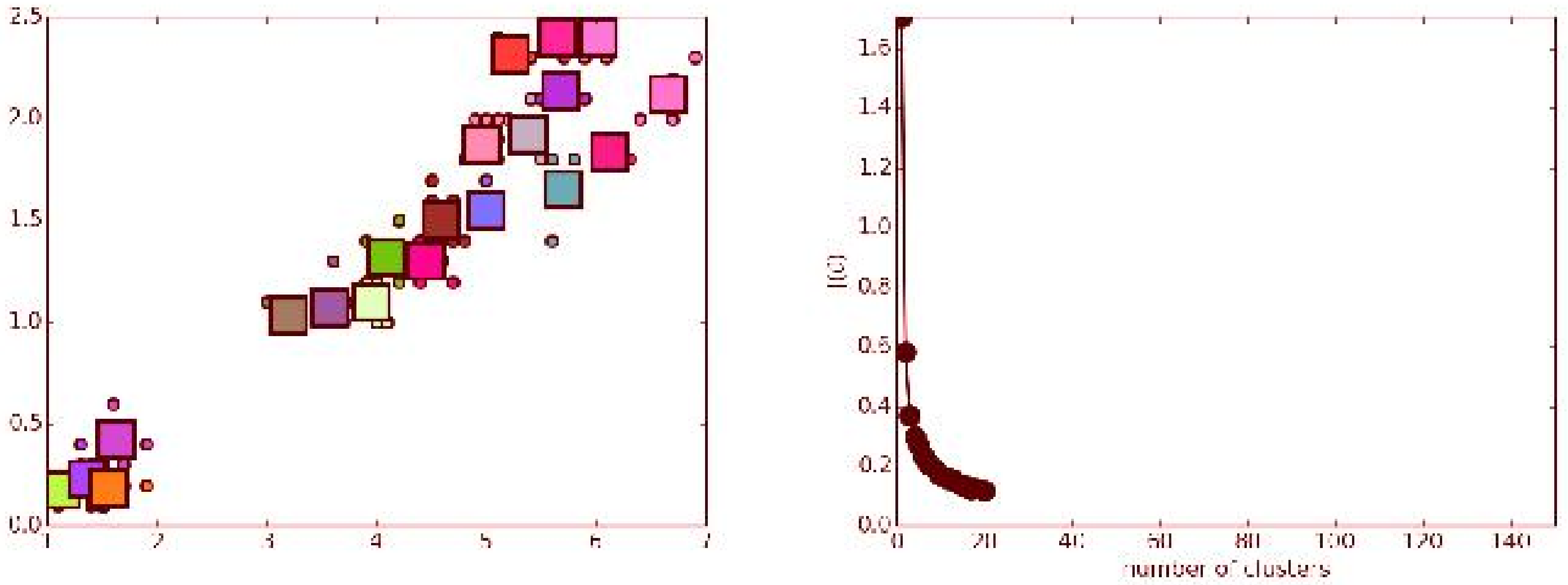
k-means solution with $k = 10$ clusters





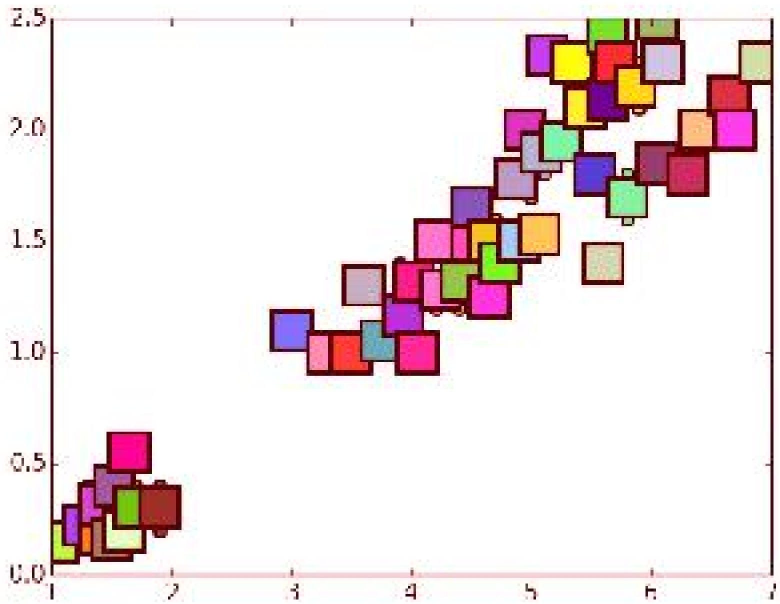
How Many Clusters should we Use with k-means ?

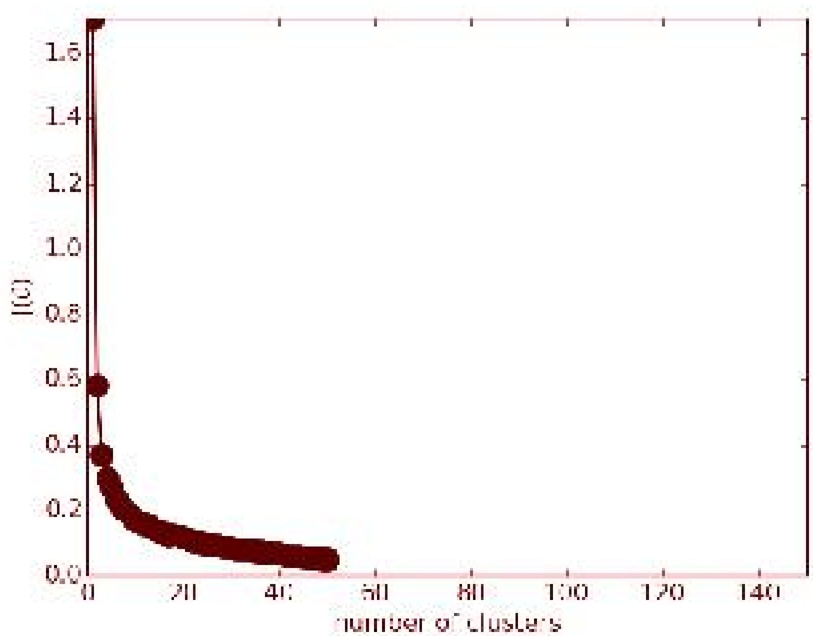
k-means solution with $k = 20$ clusters



How Many Clusters should we Use with k-means ?

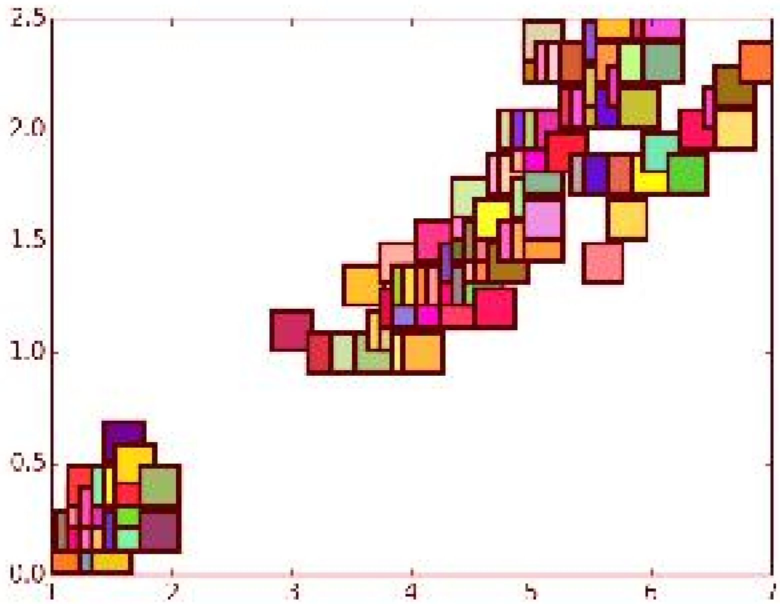
k-means solution with $k = 50$ clusters

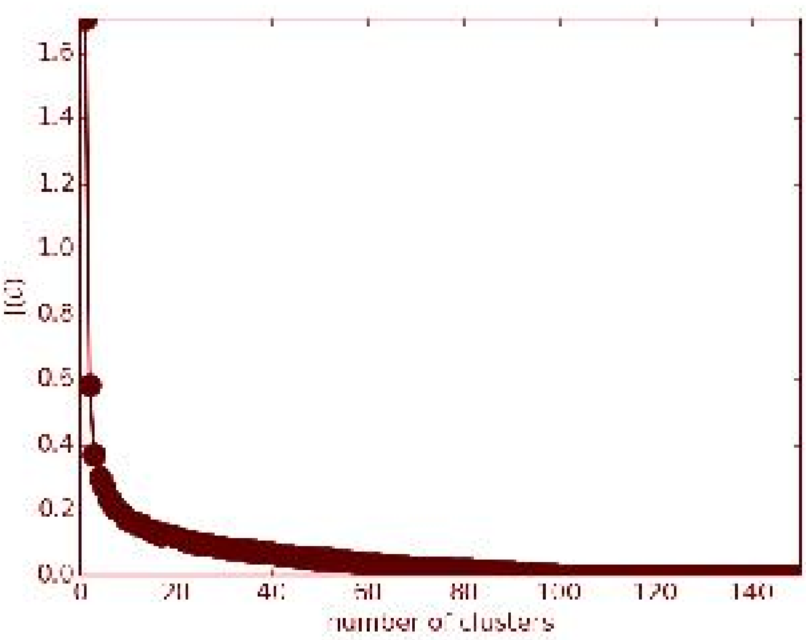




How Many Clusters should we Use with k-means ?

k-means solution with $k = 150$ clusters





How Many Clusters should we Use with k-means ?

In practice

- no supervision from data $\Rightarrow$ no way to automatically choose k
- heuristics and "clustering quality measures" all resort on assumptions
- there is **no** "natural number of clusters" (expect for toy problems)
- the choice of k depends on what the user wants to do with data

Interaction with the user

- clustering is mainly used to "explore" data
- interaction with the user **is necessary**
- in some cases, clustering is only a preprocessing step $\Rightarrow$ supervision ?

Clustering: the DBSCAN Algorithm

Density-Based Spatial Clustering of Applications with Noise

DBSCAN

density-based clustering algorithm: find high-density regions

- one of the most widely used clustering algorithm
- 24th most cited data mining article in 2010
- does not require to choose the number of clusters
- no free lunch: other settings have to be tuned

M. Ester, H.-P. Kriegel, J. Sander, X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. Proc. KDD, 1996, pp. 226–231.

Density-Based Spatial Clustering of Applications with Noise

DBSCAN

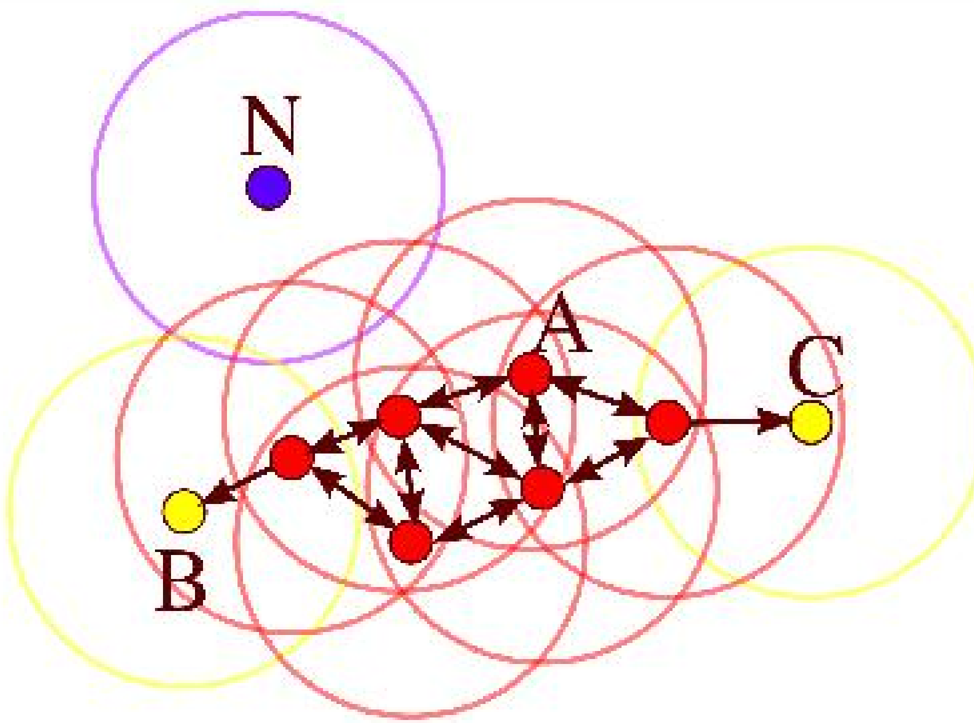
density-based clustering algorithm: find high-density regions

- one of the most widely used clustering algorithm
- 24th most cited data mining article in 2010
- does not require to choose the number of clusters
- no free lunch: other settings have to be tuned

M. Ester, H.-P. Kriegel, J. Sander, X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. Proc. KDD, 1996, pp. 226–231.

DBSCAN: Definitions and Algorithm Outline

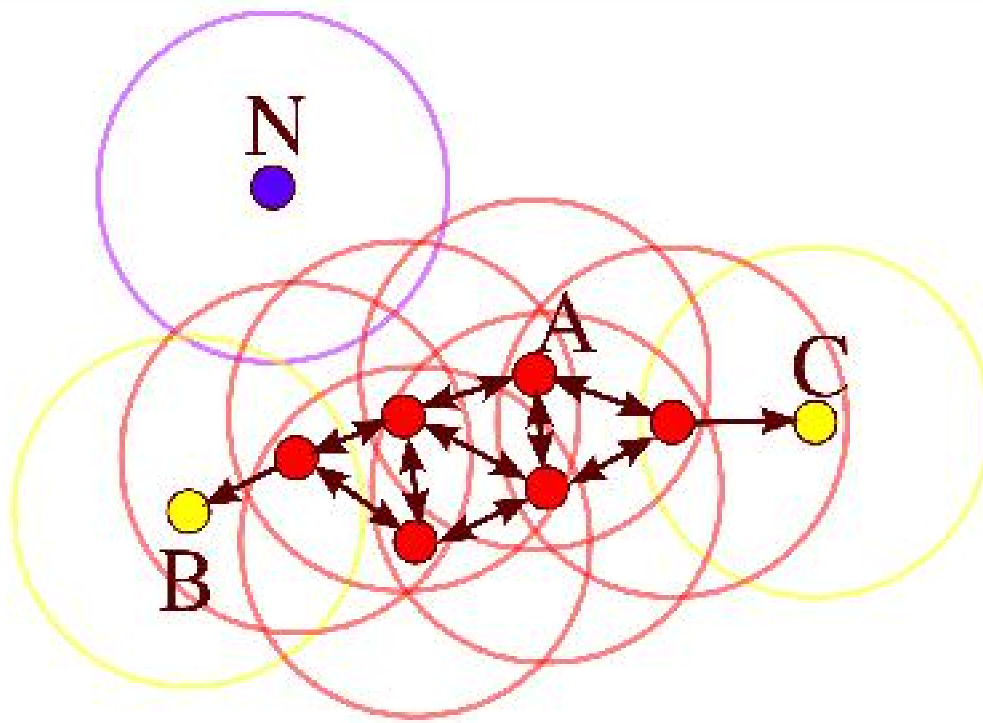
- ϵ-neighbourhood of $\mathbf{x}$ = set of points $\mathbf{x}_i$ such that $d(\mathbf{x}, \mathbf{x}_i) \leq \epsilon$
- $\mathbf{x}$ = core point if its ϵ-neighbourhood contains at least $n_{min} \neq$ points
- $\mathbf{x}_i$ in the ϵ-neighbourhood of core point $\mathbf{x}$ is directly reachable from $\mathbf{x}$
- $\mathbf{x}_n$ is reachable from $\mathbf{x}_1$ if there is a path $\mathbf{x}_1 \to \mathbf{x}_2 \to \cdots \to \mathbf{x}_n$ where $\mathbf{x}_{i+1}$ is directly reachable from $\mathbf{x}_i$ $\Rightarrow$ non-reachable points are outliers



source: https://en.wikipedia.org/wiki/DBSCAN

DBSCAN: Definitions and Algorithm Outline

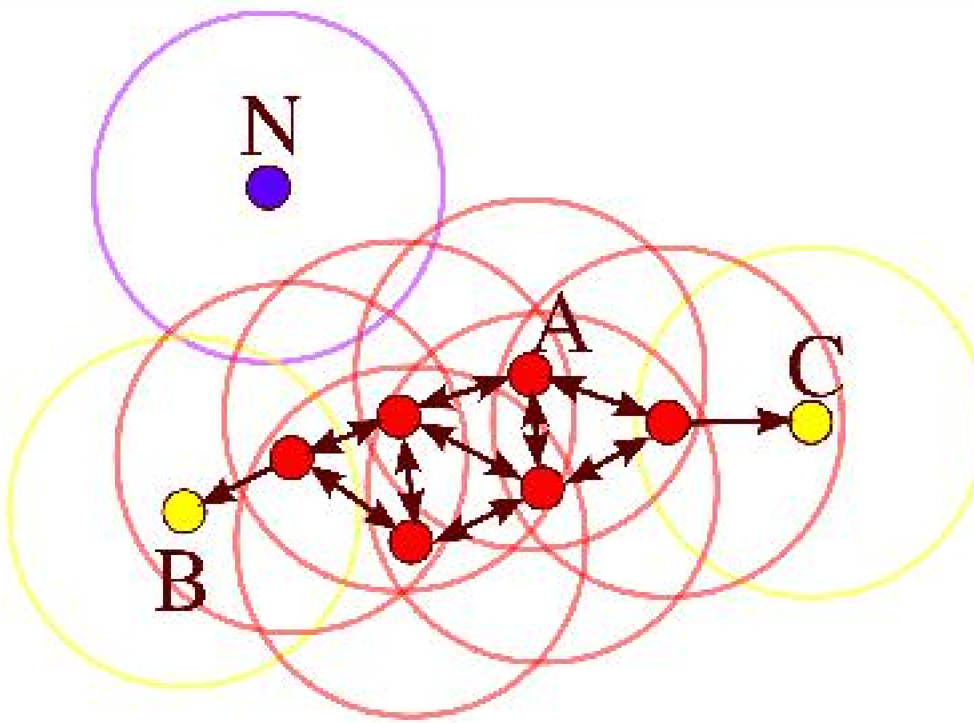
- ϵ-neighbourhood of $\mathbf{x}$ = set of points $\mathbf{x}_i$ such that $d(\mathbf{x}, \mathbf{x}_i) \leq \epsilon$
- $\mathbf{x}$ = core point if its ϵ-neighbourhood contains at least $n_{min} \neq$ points
- $\mathbf{x}_i$ in the ϵ-neighbourhood of core point $\mathbf{x}$ is directly reachable from $\mathbf{x}$
- $\mathbf{x}_n$ is reachable from $\mathbf{x}_1$ if there is a path $\mathbf{x}_1 \to \mathbf{x}_2 \to \cdots \to \mathbf{x}_n$ where $\mathbf{x}_{i+1}$ is directly reachable from $\mathbf{x}_i$ $\Rightarrow$ non-reachable points are outliers



source: https://en.wikipedia.org/wiki/DBSCAN

DBSCAN: Definitions and Algorithm Outline

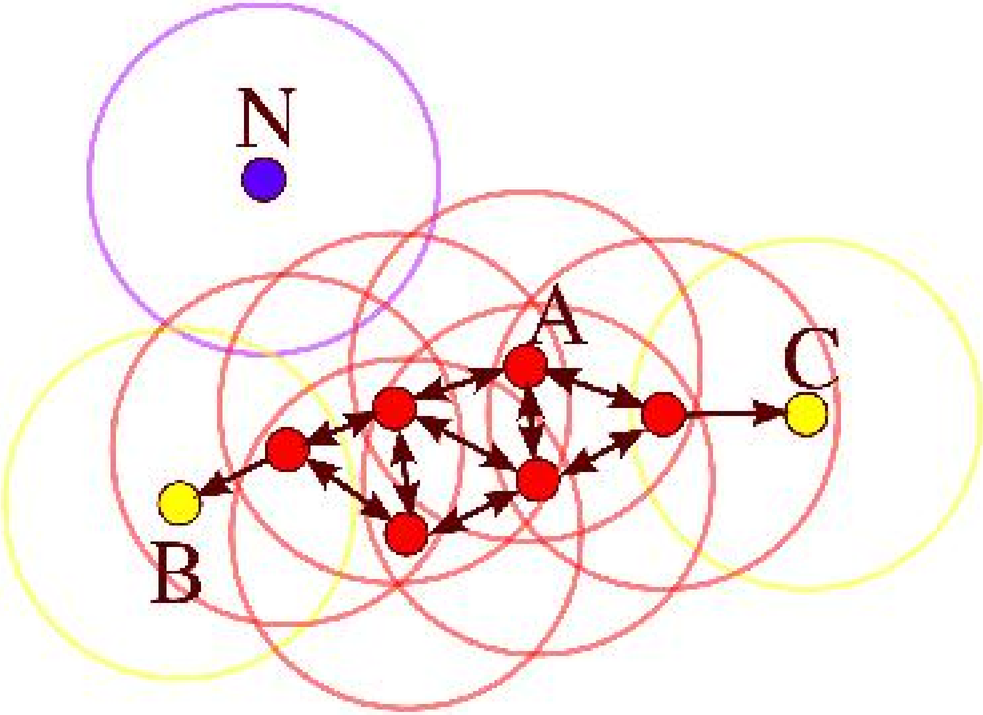
- ϵ-neighbourhood of $\mathbf{x}$ = set of points $\mathbf{x}_i$ such that $d(\mathbf{x}, \mathbf{x}_i) \leq \epsilon$
- $\mathbf{x}$ = core point if its ϵ-neighbourhood contains at least $n_{min} \neq$ points
- $\mathbf{x}_i$ in the ϵ-neighbourhood of core point $\mathbf{x}$ is directly reachable from $\mathbf{x}$
- $\mathbf{x}_n$ is reachable from $\mathbf{x}_1$ if there is a path $\mathbf{x}_1 \rightarrow \mathbf{x}_2 \rightarrow \cdots \rightarrow \mathbf{x}_n$ where $\mathbf{x}_{i+1}$ is directly reachable from $\mathbf{x}_i$ $\Rightarrow$ non-reachable points are outliers



source: https://en.wikipedia.org/wiki/DBSCAN

DBSCAN: Definitions and Algorithm Outline

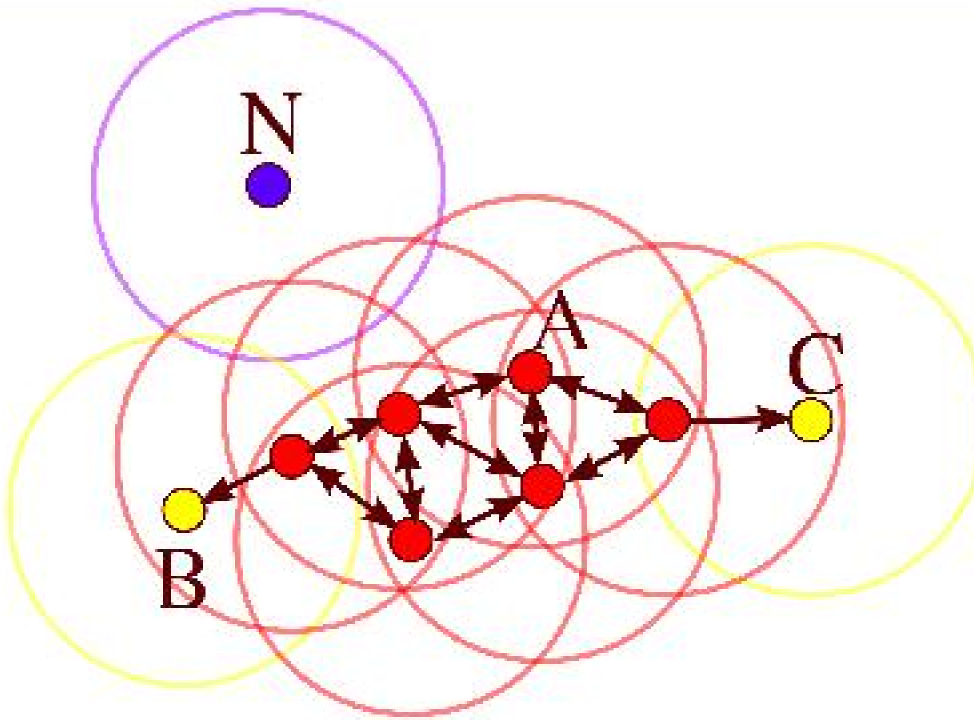
- ϵ-neighbourhood of $\mathbf{x}$ = set of points $\mathbf{x}_i$ such that $d(\mathbf{x}, \mathbf{x}_i) \leq \epsilon$
- $\mathbf{x}$ = core point if its ϵ-neighbourhood contains at least $n_{min} \neq$ points
- $\mathbf{x}_i$ in the ϵ-neighbourhood of core point $\mathbf{x}$ is directly reachable from $\mathbf{x}$
- $\mathbf{x}_n$ is reachable from $\mathbf{x}_1$ if there is a path $\mathbf{x}_1 \rightarrow \mathbf{x}_2 \rightarrow \cdots \rightarrow \mathbf{x}_n$ where $\mathbf{x}_{i+1}$ is directly reachable from $\mathbf{x}_i$ $\Rightarrow$ non-reachable points are outliers



source: https://en.wikipedia.org/wiki/DBSCAN

DBSCAN: Definitions and Algorithm Outline

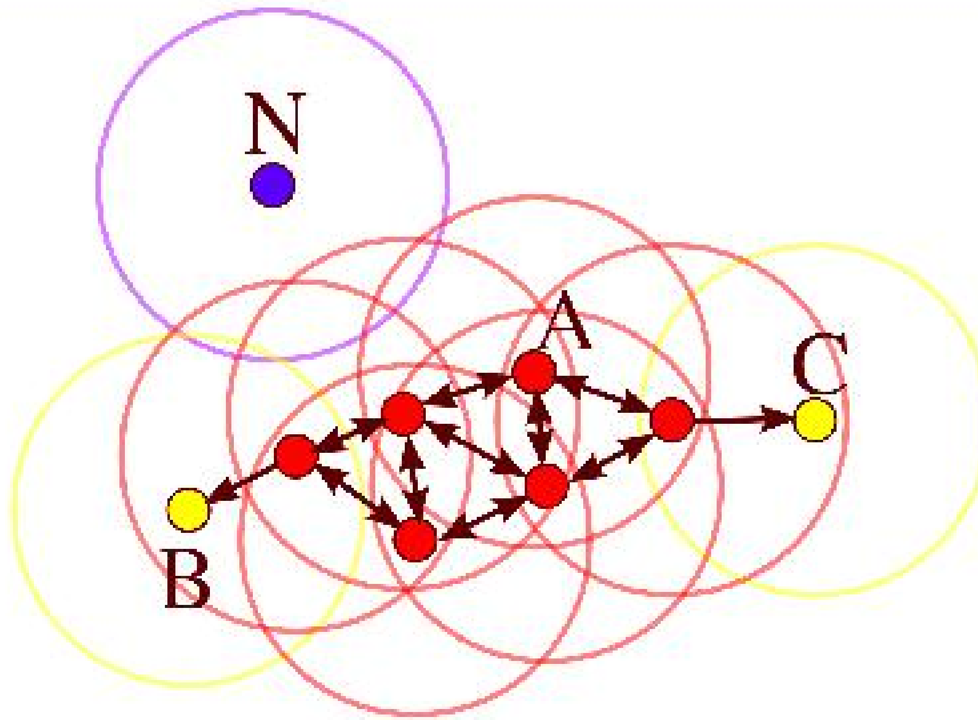
- cluster to which a core point belongs = set of reachable points
- non-core points ≈ edge of the clusters (cannot reach others points)
- reachability is transitive, but not symmetric (except for core points)
- non-core points in the same cluster are not reachable from each other



source: https://en.wikipedia.org/wiki/DBSCAN

DBSCAN: Definitions and Algorithm Outline

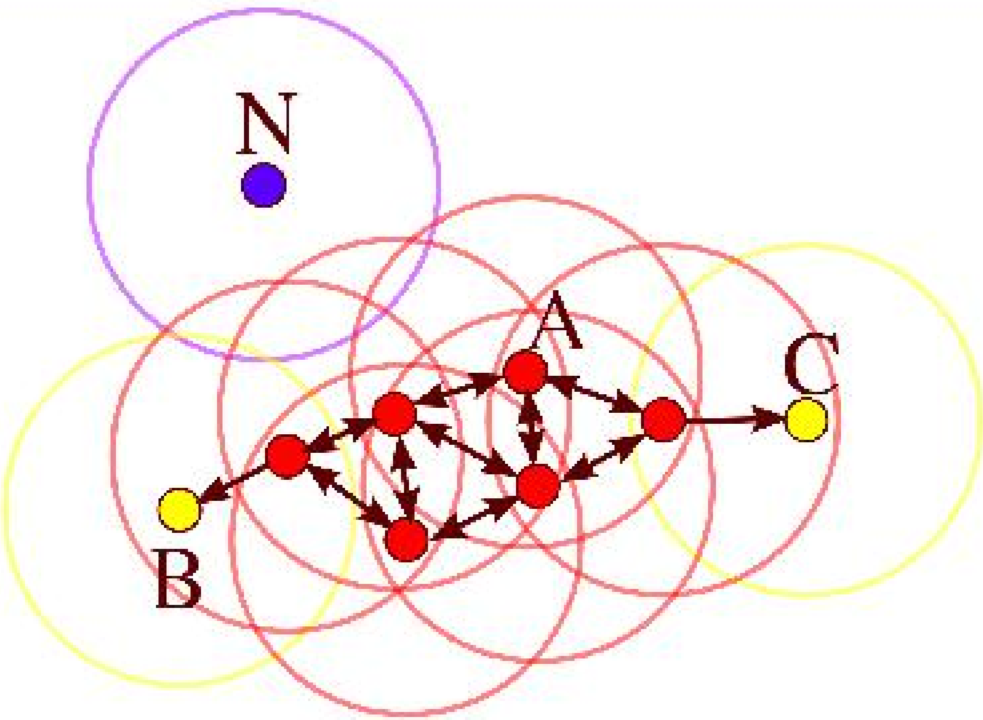
- cluster to which a core point belongs = set of reachable points
- non-core points ≈ edge of the clusters (cannot reach others points)
- reachability is transitive, but not symmetric (except for core points)
- non-core points in the same cluster are not reachable from each other



source: `https://en.wikipedia.org/wiki/DBSCAN`

DBSCAN: Definitions and Algorithm Outline

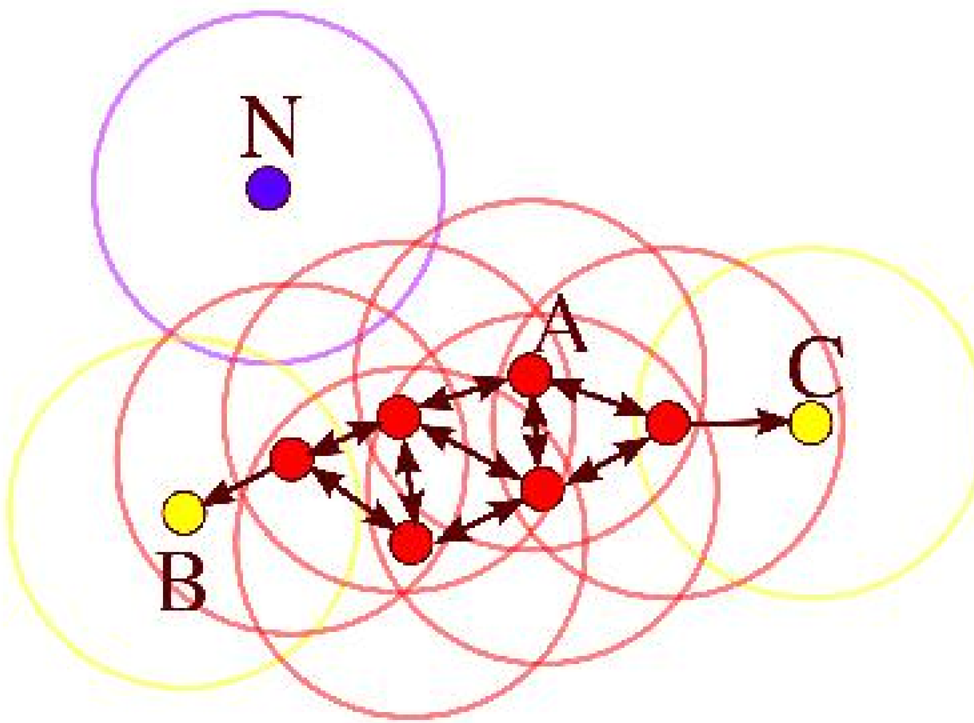
- cluster to which a core point belongs = set of reachable points
- non-core points ≈ edge of the clusters (cannot reach others points)
- reachability is transitive, but not symmetric (except for core points)
- non-core points in the same cluster are not reachable from each other



source: `https://en.wikipedia.org/wiki/DBSCAN`

DBSCAN: Definitions and Algorithm Outline

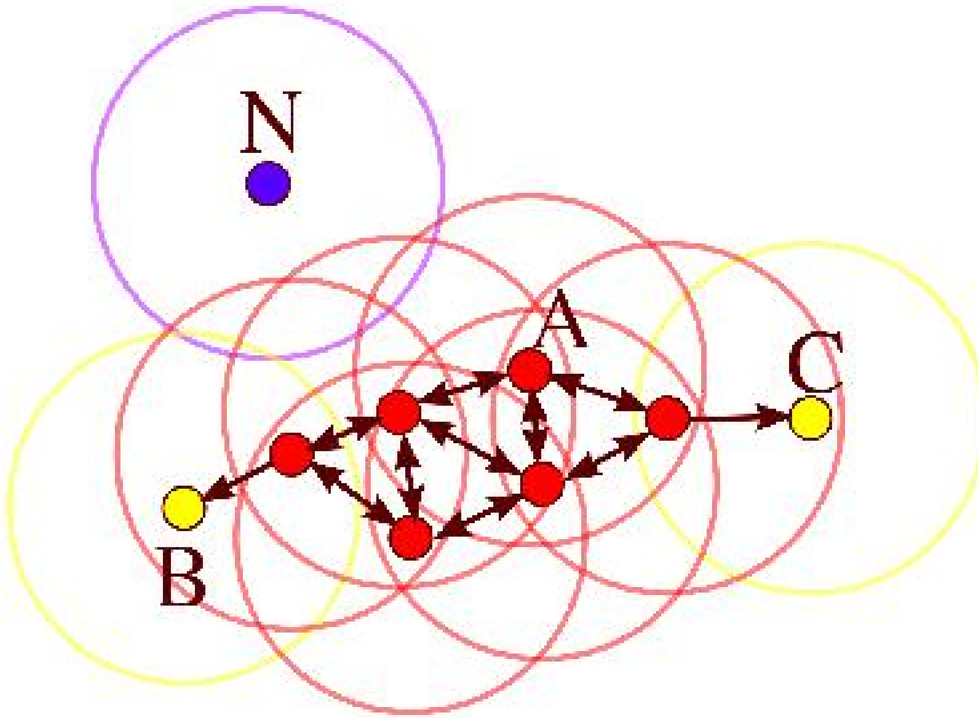
- $\mathbf{x}_i$ and $\mathbf{x}_j$ are connected if $\exists \mathbf{x}_k$ from which $\mathbf{x}_i$ and $\mathbf{x}_j$ are reachable
- any point reachable from a point in the cluster belongs to the cluster
- all points in a cluster are mutually connected (solves the edge problem)



source: `https://en.wikipedia.org/wiki/DBSCAN`

DBSCAN: Definitions and Algorithm Outline

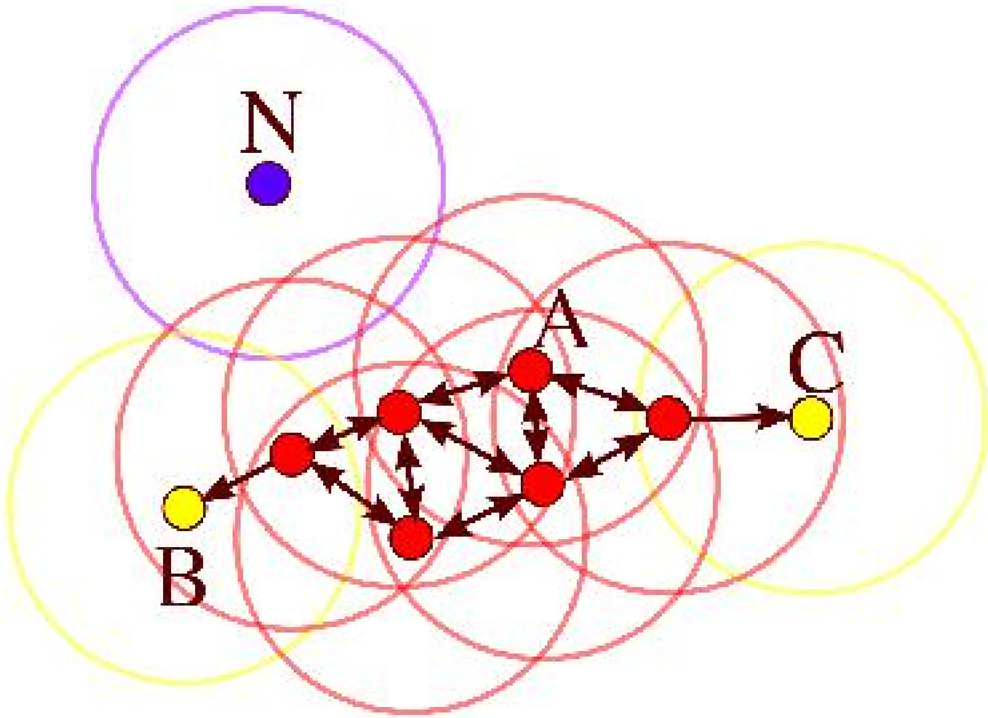
- $\mathbf{x}_i$ and $\mathbf{x}_j$ are connected if $\exists \mathbf{x}_k$ from which $\mathbf{x}_i$ and $\mathbf{x}_j$ are reachable
- any point reachable from a point in the cluster belongs to the cluster
- all points in a cluster are mutually connected (solves the edge problem)



source: https://en.wikipedia.org/wiki/DBSCAN

DBSCAN: Definitions and Algorithm Outline

- $\mathbf{x}_i$ and $\mathbf{x}_j$ are connected if $\exists \mathbf{x}_k$ from which $\mathbf{x}_i$ and $\mathbf{x}_j$ are reachable
- any point reachable from a point in the cluster belongs to the cluster
- all points in a cluster are mutually connected (solves the edge problem)



source: https://en.wikipedia.org/wiki/DBSCAN

DBSCAN: Definitions and Algorithm Outline

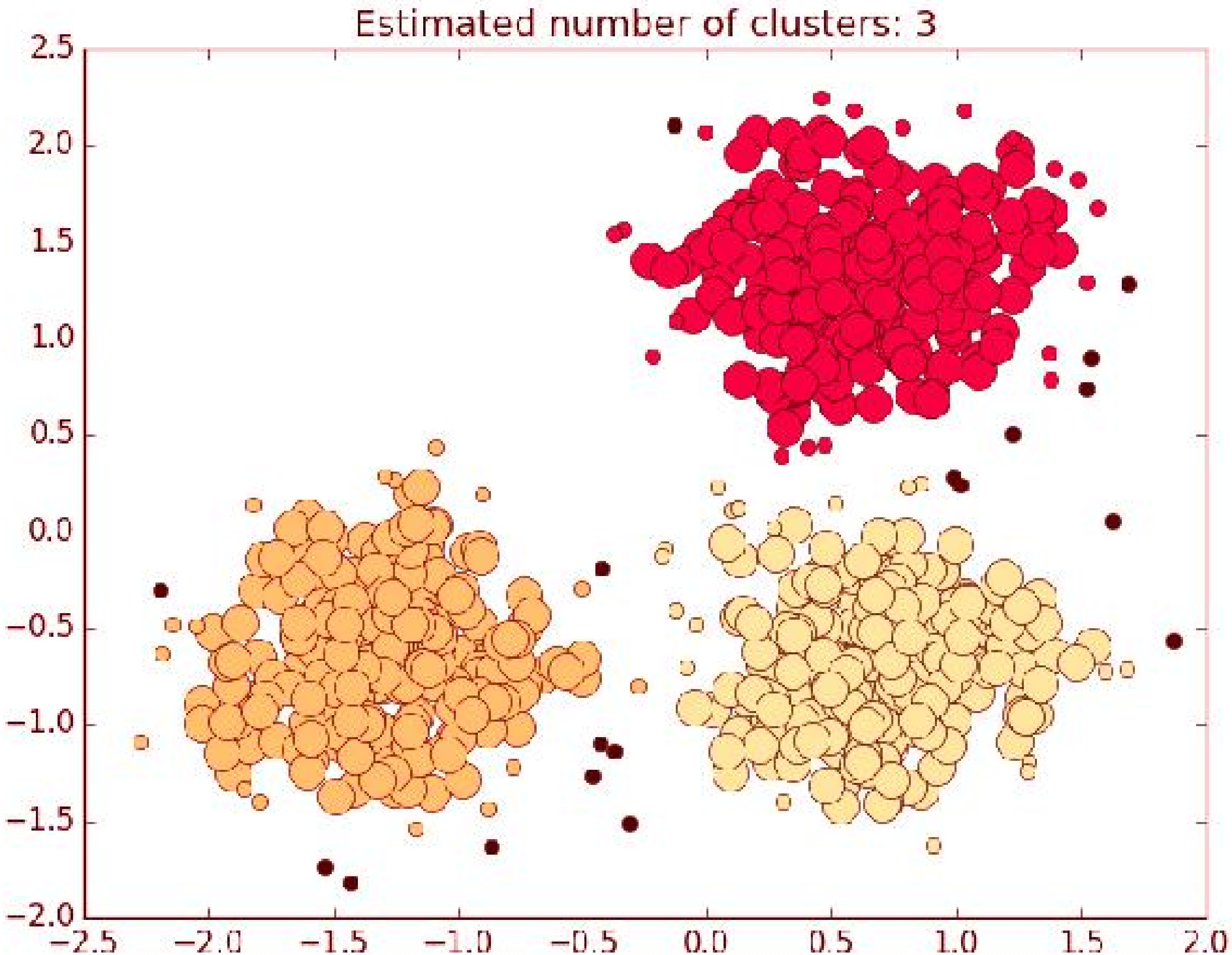
```
DBSCAN(D, eps, MinPts) {
   C = 0
   for each point P in dataset D {
      if P is visited
         continue next point
      mark P as visited
      NeighborPts = regionQuery(P, eps)
      if sizeof(NeighborPts) < MinPts
         mark P as NOISE
      else {
         C = next cluster
         expandCluster(P, NeighborPts, C, eps, MinPts)
      }
   }
}

expandCluster(P, NeighborPts, C, eps, MinPts) {
   add P to cluster C
   for each point P' in NeighborPts {
      if P' is not visited {
         mark P' as visited
         NeighborPts' = regionQuery(P', eps)
         if sizeof(NeighborPts') >= MinPts
            NeighborPts = NeighborPts joined with NeighborPts'
      }
      if P' is not yet member of any cluster
         add P' to cluster C
   }
}

regionQuery(P, eps)
   return all points within P's eps-neighborhood (including P)
```

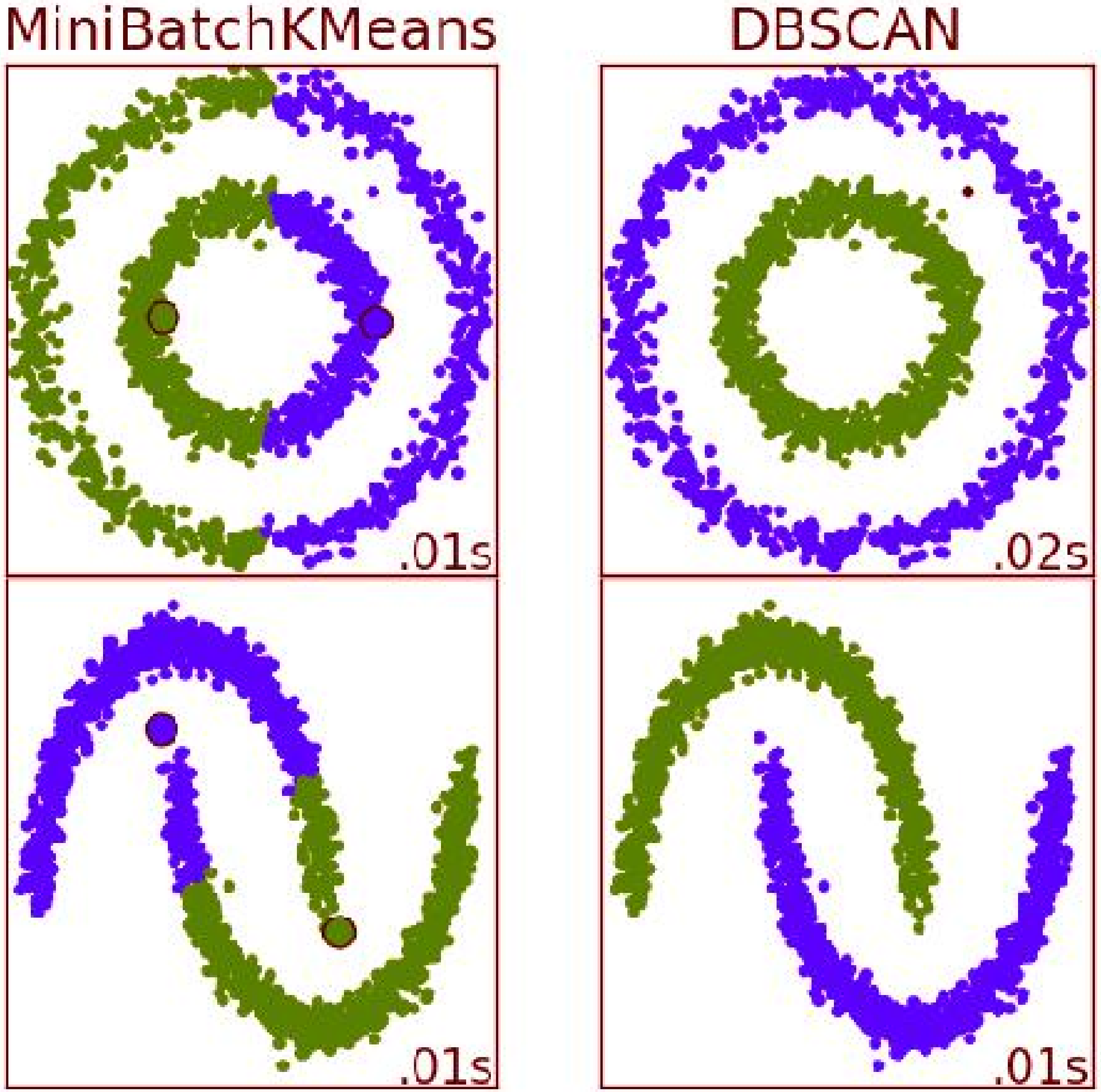
source: https://en.wikipedia.org/wiki/DBSCAN

Examples of Clustering with DBSCAN



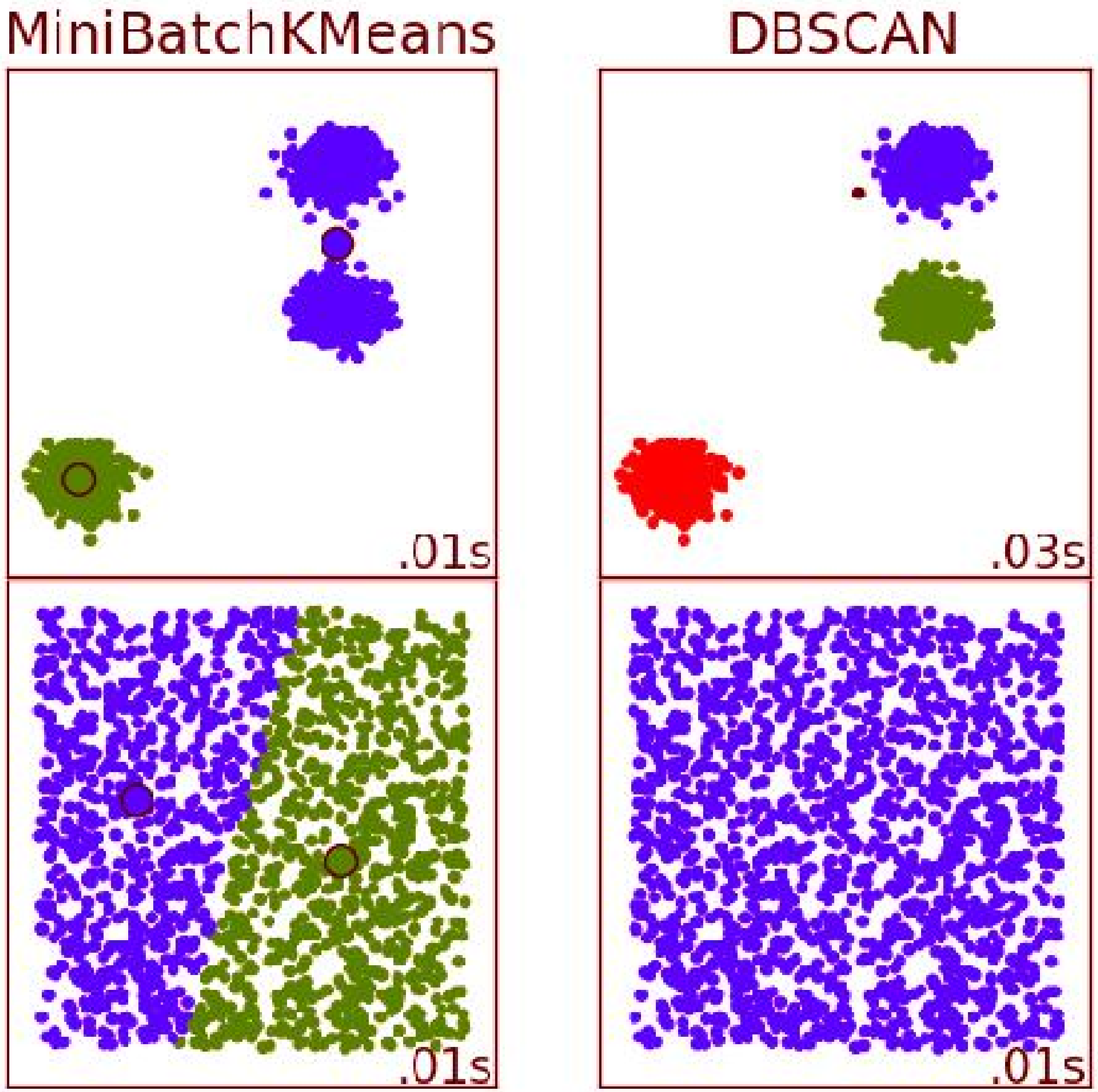
source: `http://scikit-learn.org`

Examples of Clustering with DBSCAN



source: http://scikit-learn.org

Examples of Clustering with DBSCAN



source: http://scikit-learn.org

Hypotheses and Limitations

Learning biases

- high-density regions have similar densities (ϵ and n_{min} are global)
- clusters do not overlap too much (separated by low-density regions)

Drawbacks

- results depends on ϵ and n_{min}
- ϵ may be difficult to estimate
- n_{min} depends on dataset size
- not entirely deterministic (non-core points)
- issues if large differences in density between clusters
- overlapping clusters are likely to be merged
- no representative point $\Rightarrow$ no interpretation

Hypotheses and Limitations

Learning biases

- high-density regions have similar densities (ϵ and n_{min} are global)
- clusters do not overlap too much (separated by low-density regions)

Drawbacks

- results depends on ϵ and n_{min}
- ϵ may be difficult to estimate
- n_{min} depends on dataset size
- not entirely deterministic (non-core points)
- issues if large differences in density between clusters
- overlapping clusters are likely to be merged
- no representative point $\Rightarrow$ no interpretation

Application: Clustering of Geographical Curves

Context of the Application

Common work with geographers

Clustering patterns of urban built-up areas with curves of fractal scaling behaviour. Thomas, I., Frankhauser, P., Frénay, B., Verleysen, M. Environment and planning B 37 (5), 942, 2010

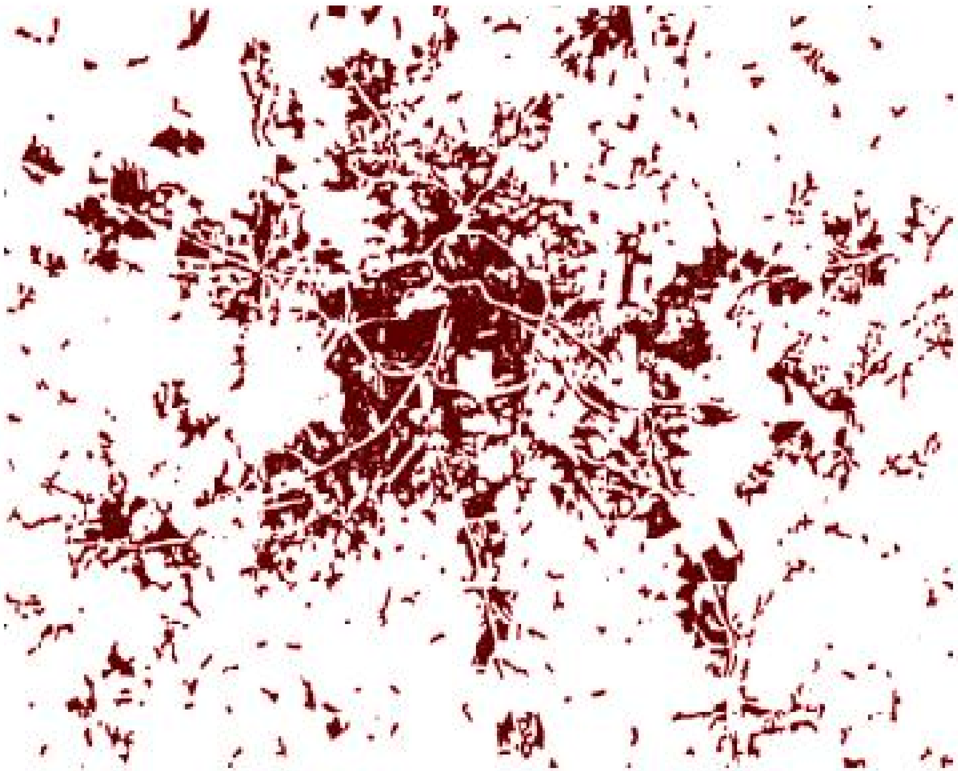
MASHS 2012 (Modèles et Apprentissage en Sciences Humaines et Sociales)

Problem statement

- geographers wanted to get knowledge about cities
- stated in machine learning terms
  - each city is represented as a curve
  - the goal is to find groups of cities

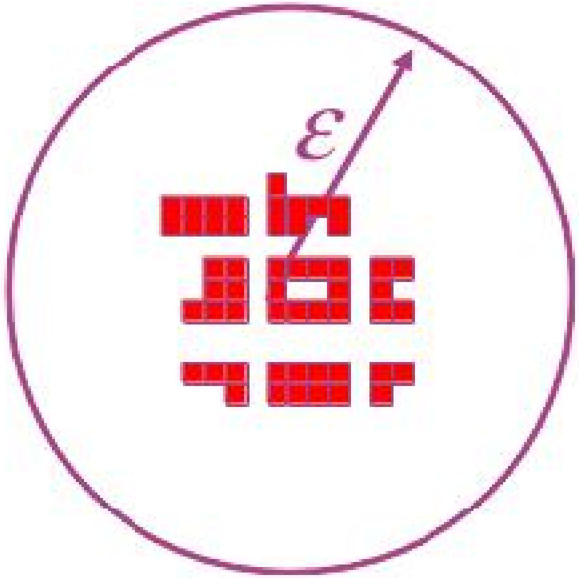
About the Data: Built-Up in Urban Areas

Question: how can we **characterise** the **built-up** within **urban areas** ?



The built-up area of Berlin

About the Data: Fractal Curves (1)

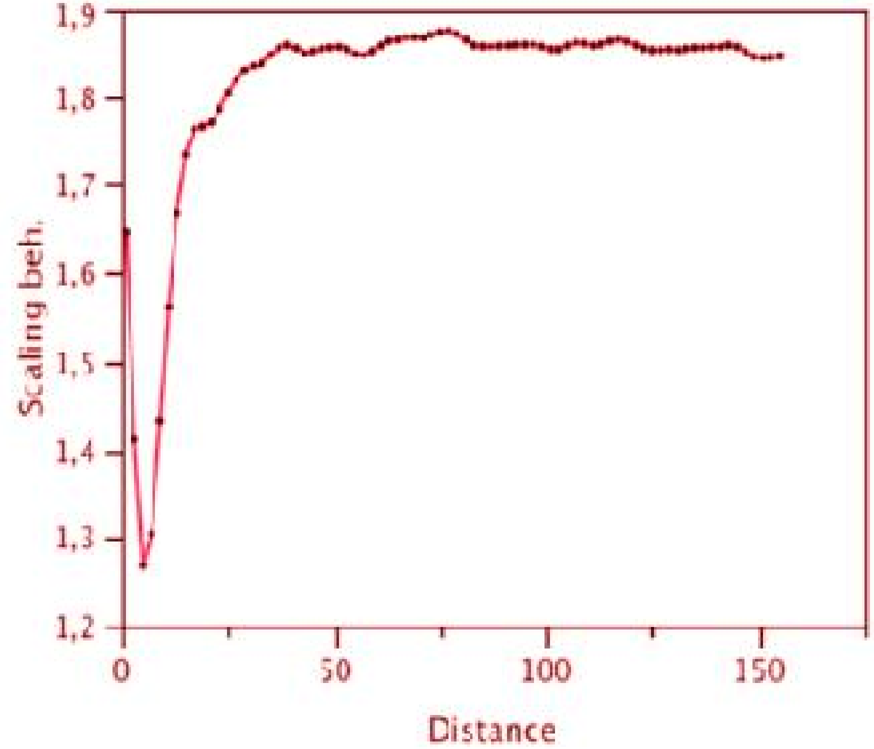


Curve of fractal behaviour

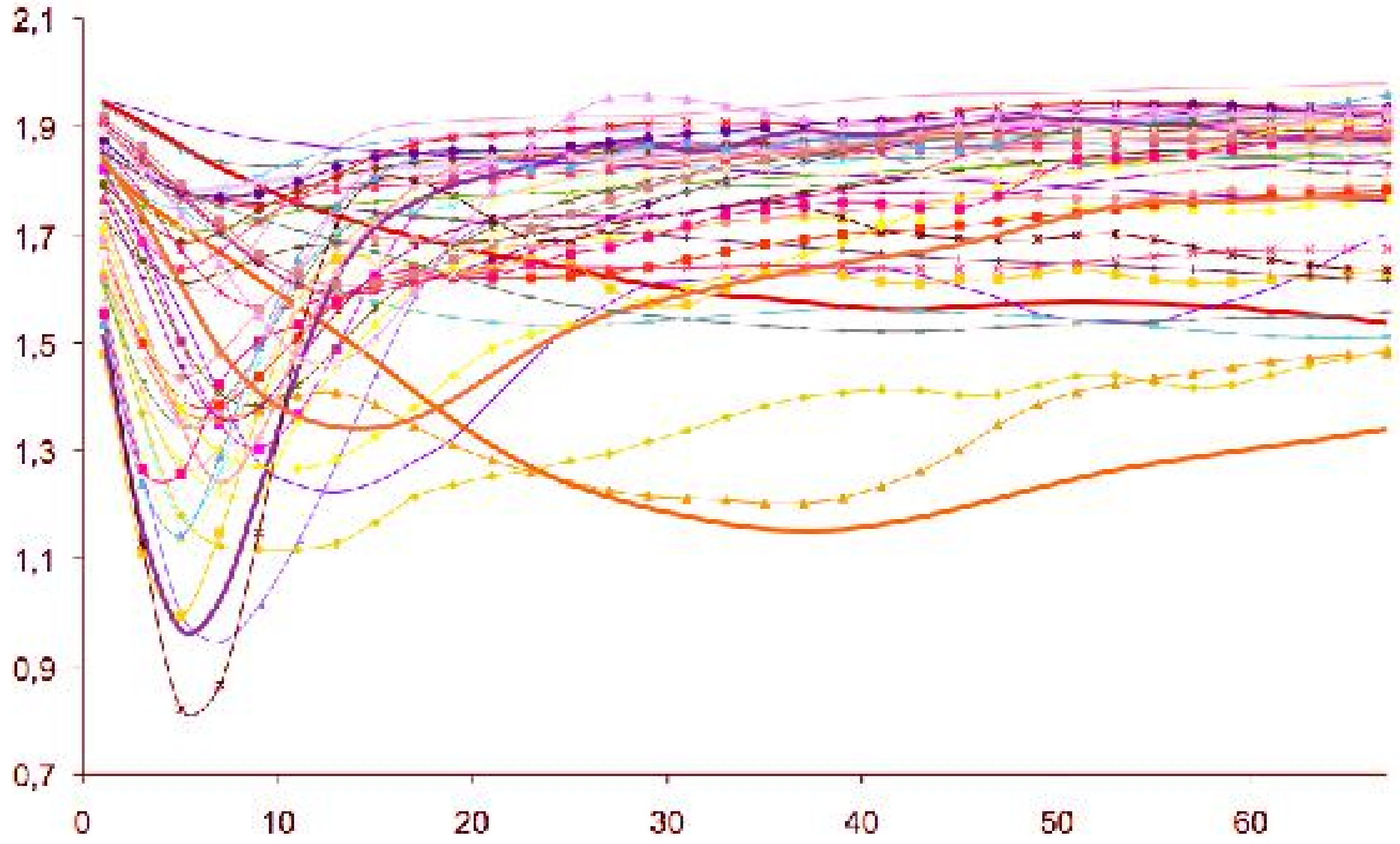
$\alpha(\epsilon)$ = built-up concentration at scale ϵ (as defined by geographers)

- $\alpha(\epsilon) = 2$: homogeneous mass distribution
- $1 < \alpha(\epsilon) < 2$: connected clusters
- $0 < \alpha(\epsilon) < 1$: detached clusters
- $\alpha(\epsilon) = 0$: isolated point

About the Data: Fractal Curves (2)



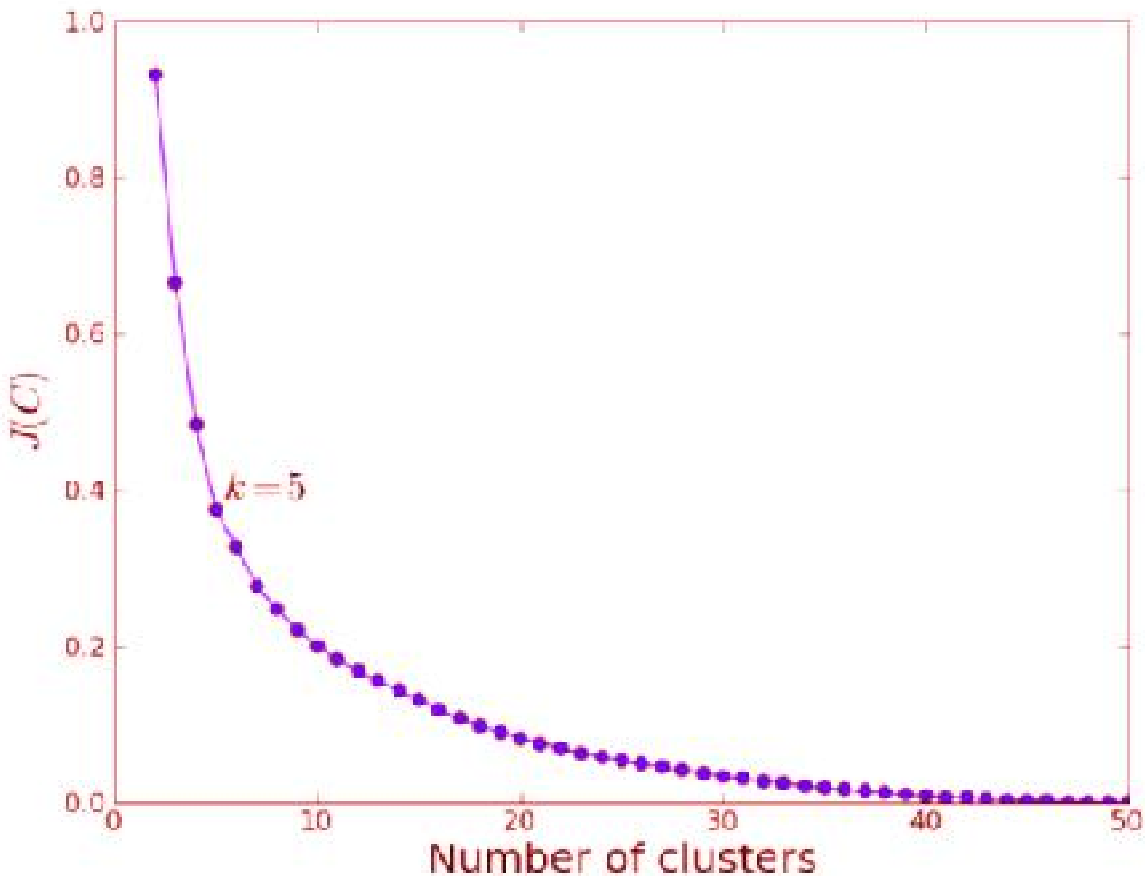
About the Data: Fractal Curves (3)



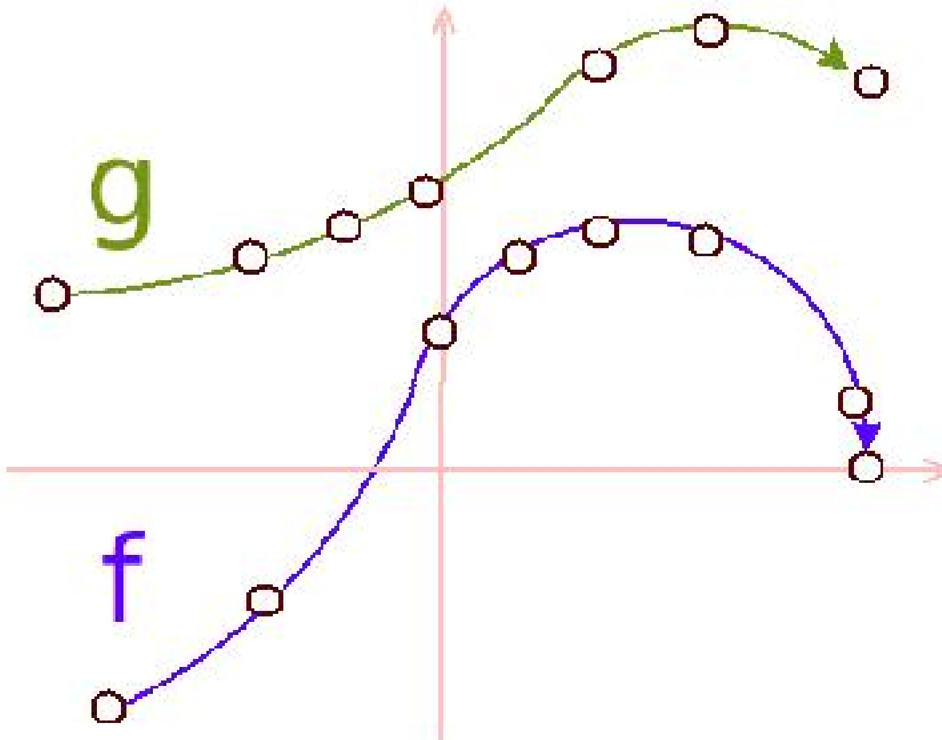
Clustering of Curves with k-medoids

k-medoids: find m representative curves c_j / clusters C_j minimising

$$J(c_1, \ldots, c_m) = \sum_{j=1}^{m} \sum_{i \in C_j} d(\alpha_i, c_j)^2$$



Computing Distance Between Unaligned Curves

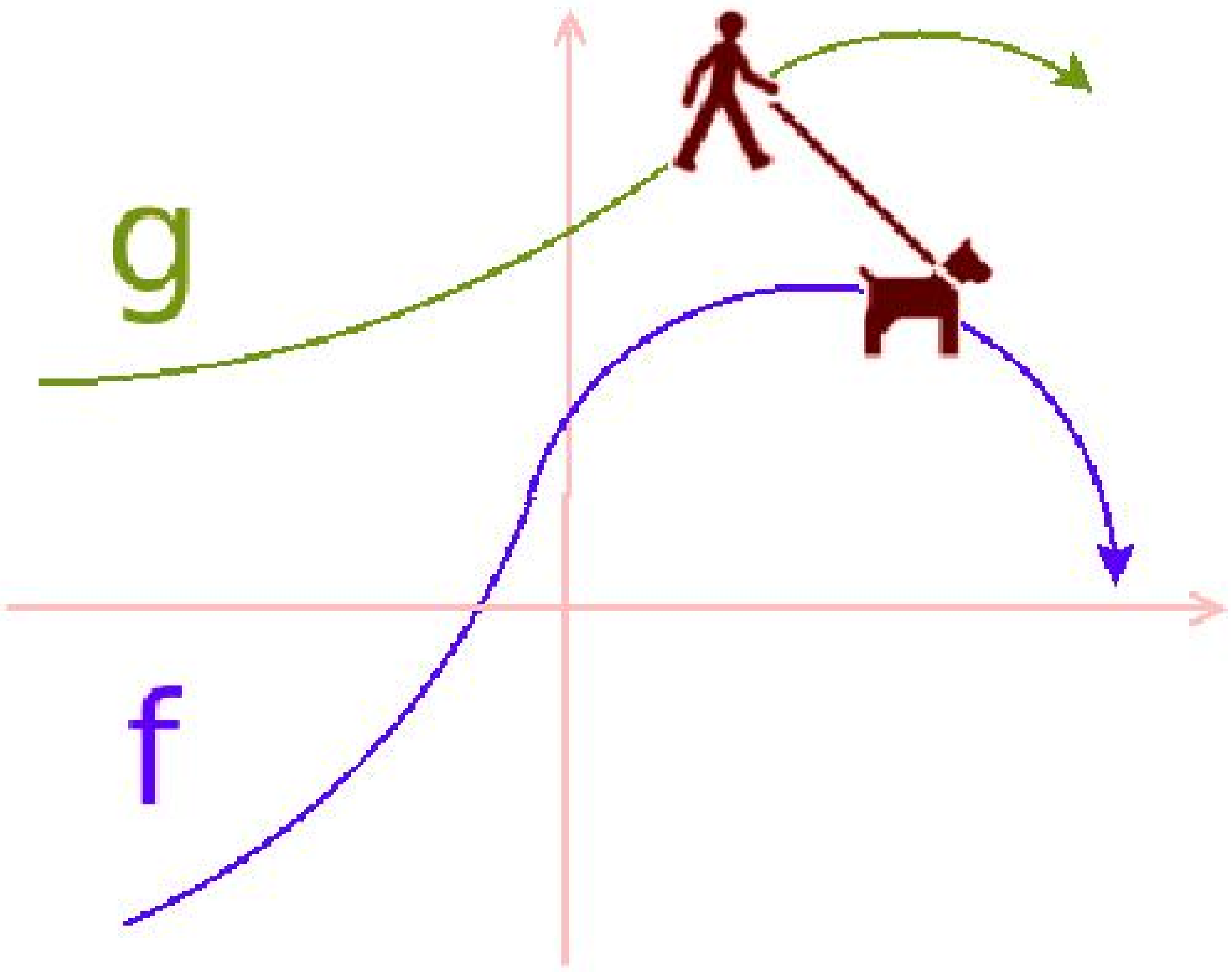


Problem statement

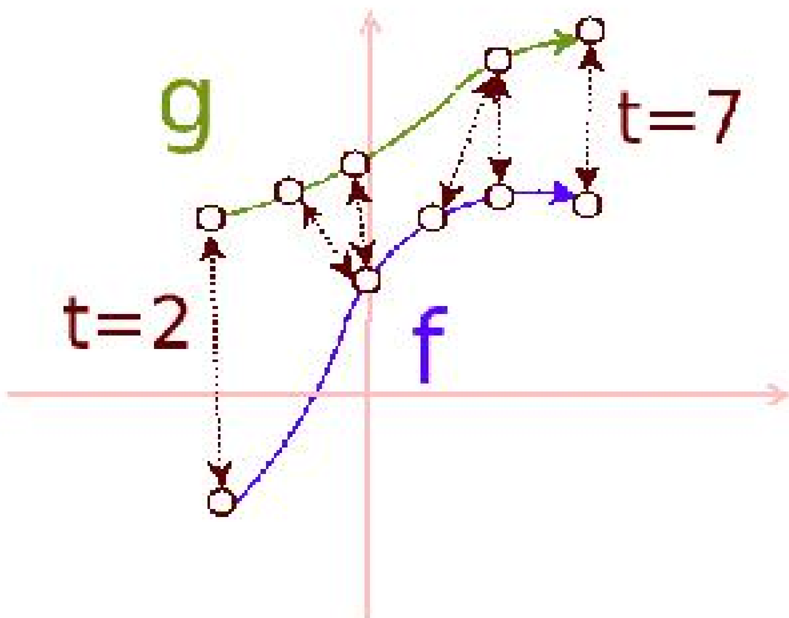
The **curves** are described by **ordered 2D points** and

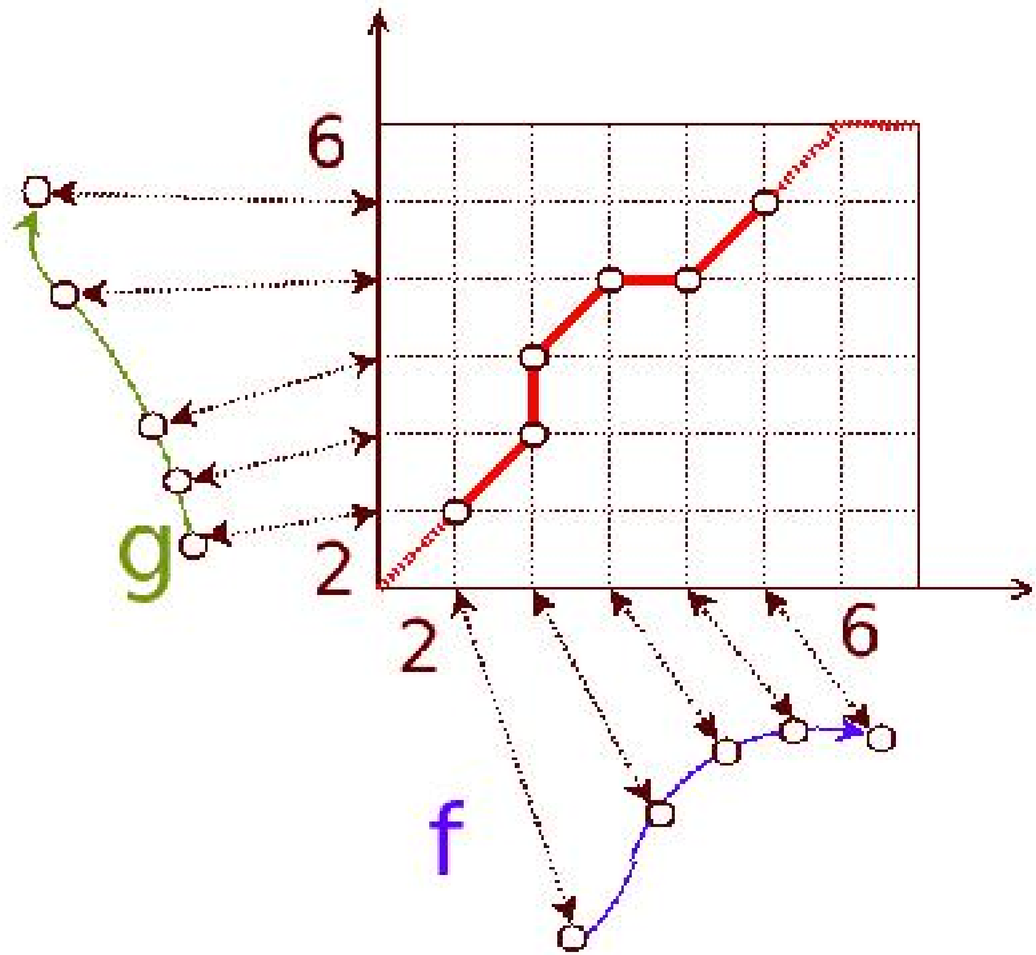
- each curve can be described by a **different number** of points
- the **x-components** are not necessarily the **same**

The Dog-Man Analogy



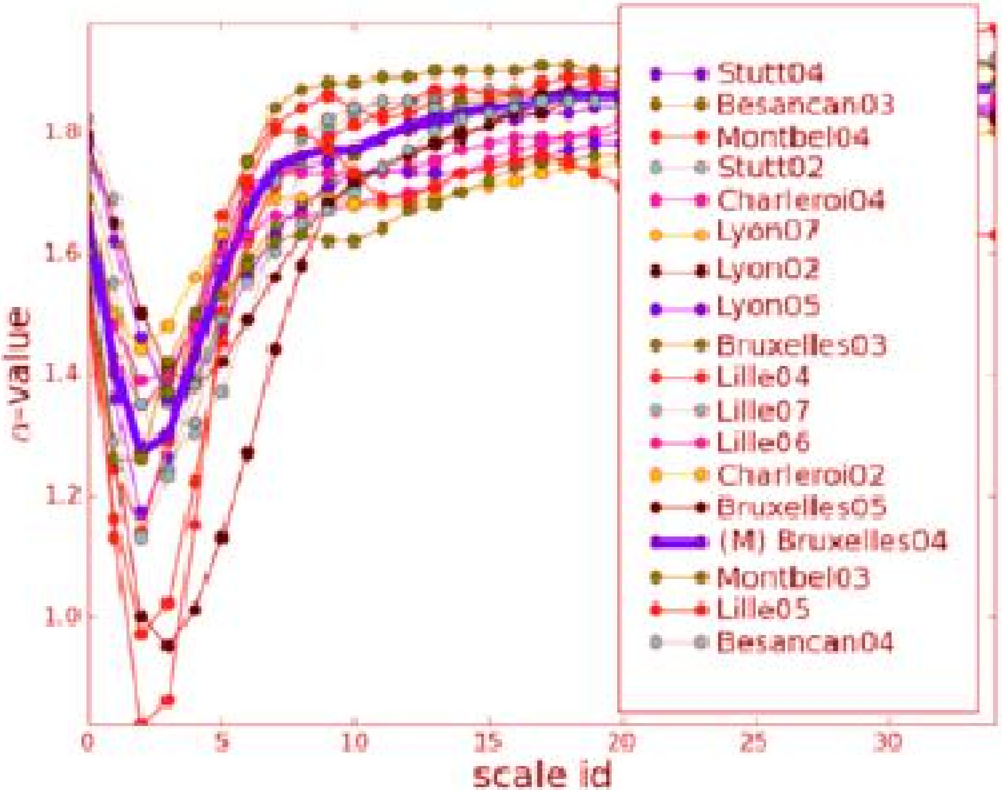
Example of Discrete Time Warping

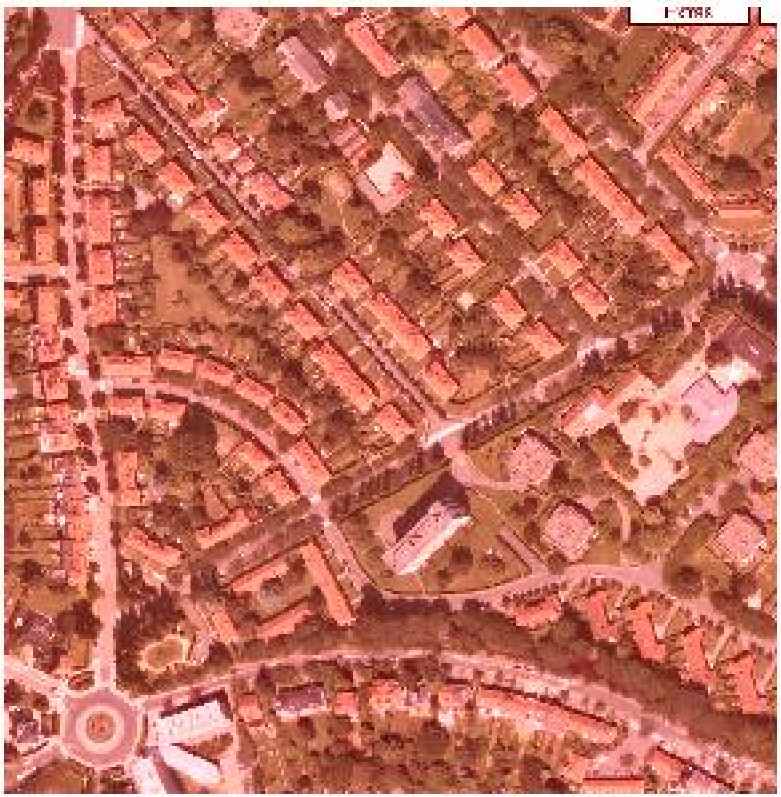




Clustering Result with $k = 5$: Cluster 1

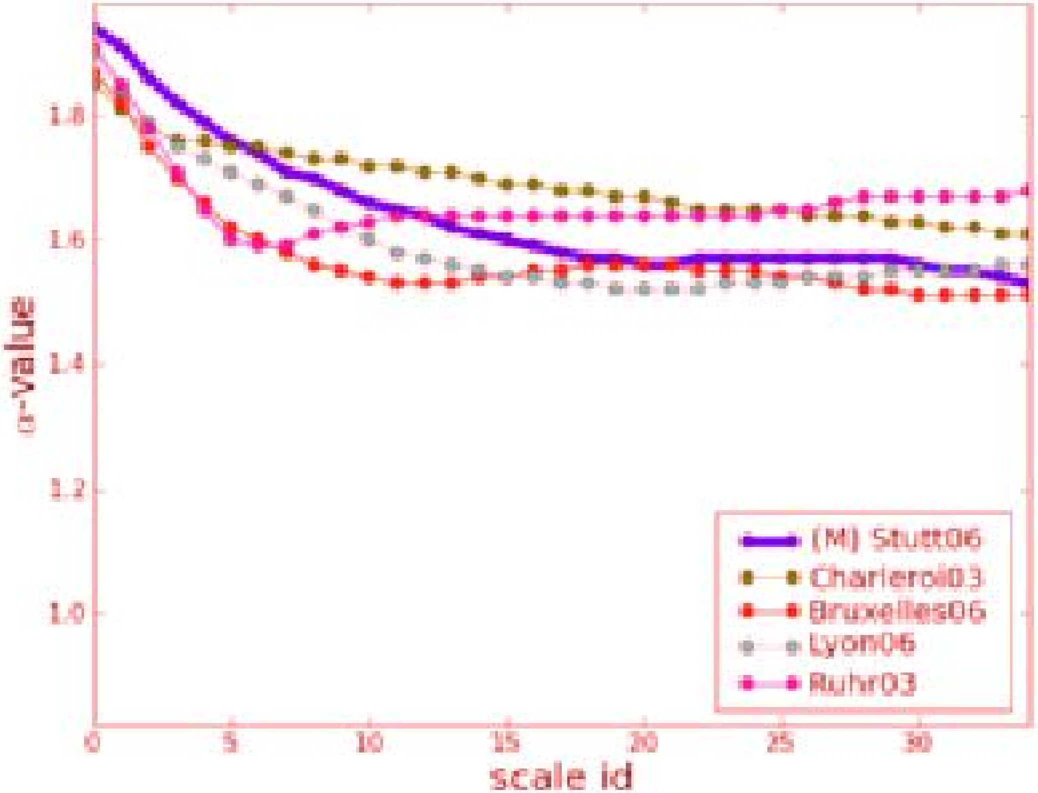
Classic dense urban areas: city centres with root-like built-up patterns and detached houses aligned along roads with small distances between buildings.





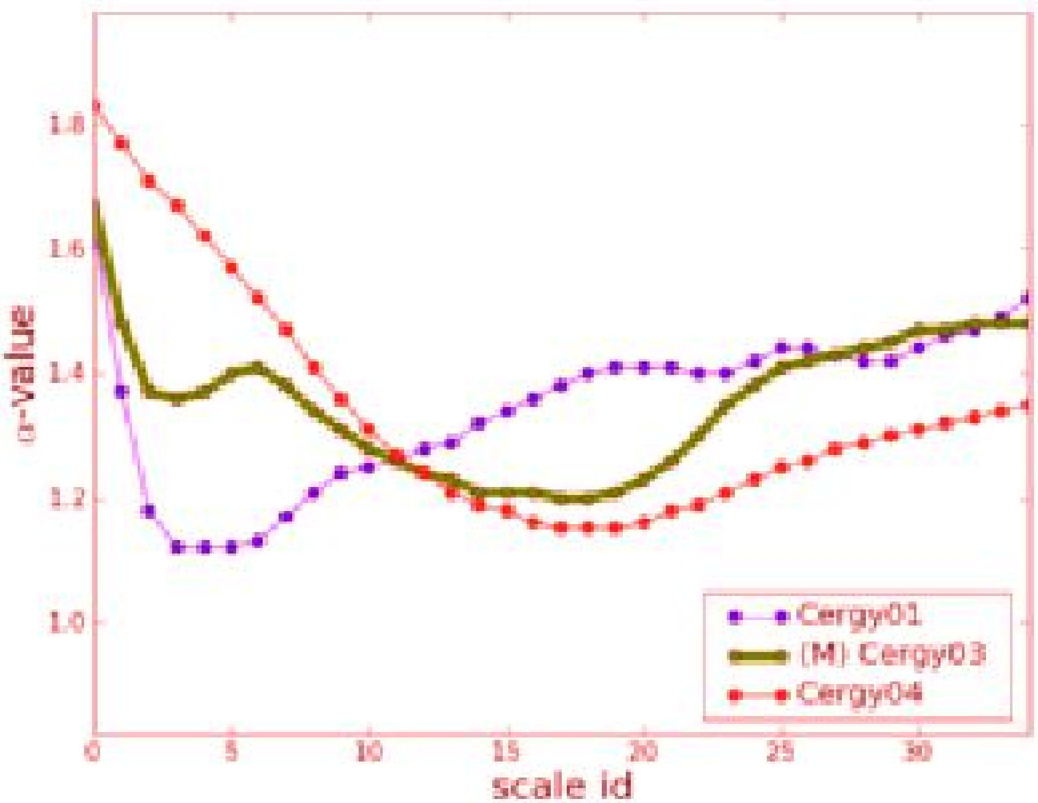
Clustering Result with $k = 5$: Cluster 2

Areas with buildings covering large irregular areas: free-standing industrial or office buildings, where intrabuilding distances are considerable.



Clustering Result with $k = 5$: Cluster 3

Atypical scaling curves: new town of Cergy-Pontoise in France, which was created in 1969 to manage the development of the Paris Region.



Outcome of the Data Analysis Task

Advantages of the machine learning approach

- machine learning allowed analysing a large number of curves
- typically difficult to do manually (without introducing bias)
- another advantage is that you can easily update the result

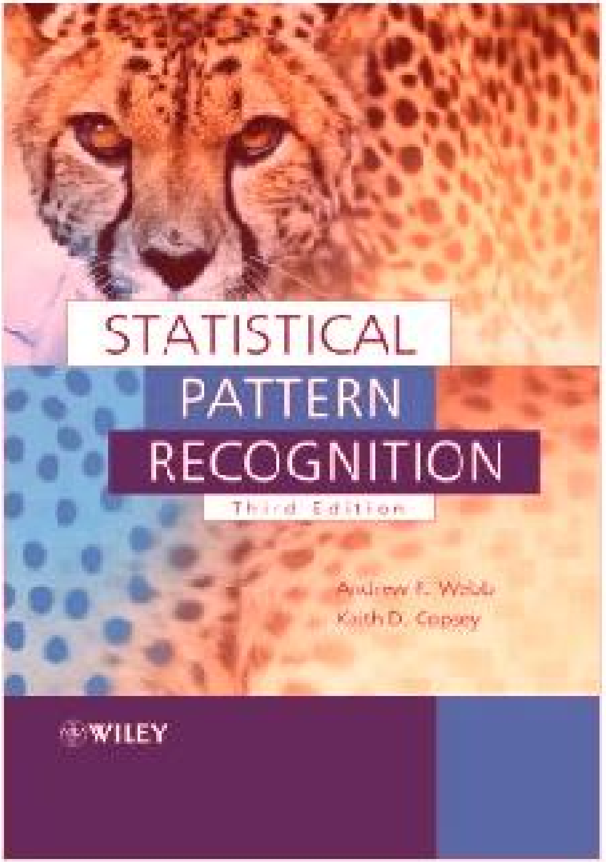
Interaction with users

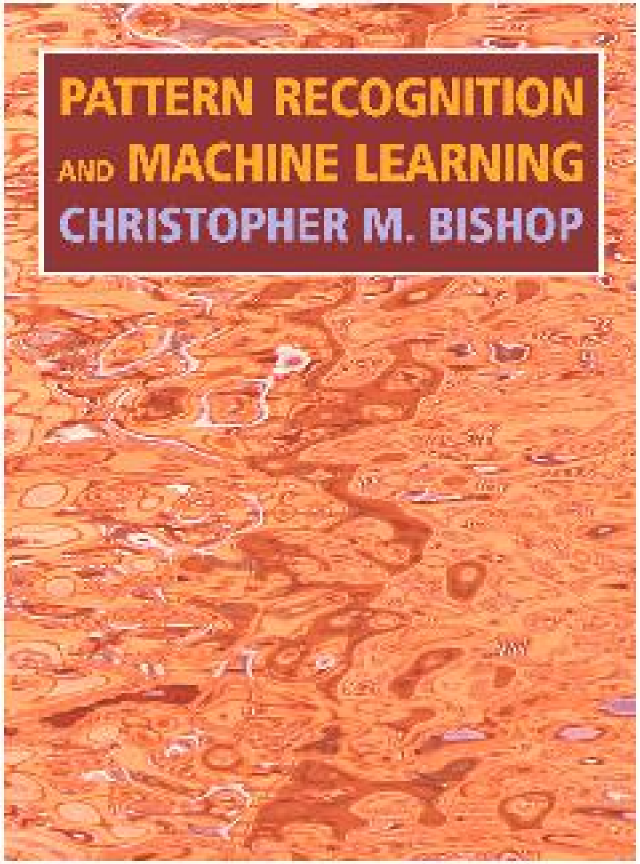
- no objective criterion to choose the number of clusters
- geographers chose 5 clusters and were very happy with the results
- this analysis confirmed the interest of the curves of fractal behaviour

Outline of this Lesson

- clustering
  - problem statement
  - the k-means algorithm
  - choosing the number of clusters
  - the DBSCAN algorithm
- application: clustering of geographical curves

References


STATISTICAL
PATTERN
RECOGNITION
Third Edition
WILEY


PATTERN RECOGNITION
AND MACHINE LEARNING
CHRISTOPHER M. BISHOP