Data Mining and Machine Learning
Clustering

Dmitry I. Ignatov

Computer Science Faculty
Dept. of Data Analysis and Artificial Intelligence

2019

Outline

Outline

Problem Formulation

Cluster (*Russian syn.*: гроздь, сгусток, пучок)
The main problem is to find a partition of an input set of objects into different groups consisting of similar elements.

Cluster structures

- Partition
- Hierarchy
- Fuzzy partition
- Biclusters
- Mixture of distributions

Clustering applications

- Biology and medicine
  - Gene expression analysis
  - Tomography clustering
- Social sciences
  - Sociology and anthropology
  - Psychology
- Technical systems
  - Telemetry
  - Image Segmentation
- Marketing
  - Market segmentation
  - Analysis of group behaviour
- Text Mining
- Social networks
  - Community detection

Classification of clustering techniques

- Partitioning methods
- Hierarchical methods
- Density-based and non-parametric methods
- Grid-based methods
- Multimodal clustering

Partitioning methods

- Pairwise disjoint clusters of spherical shape
- Based on distance between objects
- Each cluster is described by its centroid — center of mass (k-means) or one of its objects (k-medoids)

Hierarchical methods

- The output is a hierarchy of clusters
- Subsequent merging of one-element clusters (agglomerative or bottom-up) or partitioning of trivial cluster (all considered objects) into several smaller ones (divisive or top-down)

Outline

Examples of «dissimilarity measures» between objects

Objects $x \in \mathbb{R}^m$ are represented by «object-attribute» matrix

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \Longleftrightarrow \begin{bmatrix} x_1^1 & x_1^2 & \dots & x_1^m \\ x_2^1 & x_2^2 & \dots & x_2^m \\ \dots & \dots & \dots & \dots \\ x_n^1 & x_n^m & \dots & x_n^m \end{bmatrix}$$

Minkowski Metric

$$d(x,y) = \left[\sum_{i=1}^{m} |x^i - y^i|^p\right]^{\frac{1}{p}}$$

Cosine Distance

$$d(x,y) = 1 - \frac{\langle x, y\rangle}{\sqrt{\langle x, x\rangle}\sqrt{\langle y, y\rangle}}$$

Hamming distance

$$d(x,y) = \frac{1}{m}\sum_{i=1}^{m}[x^i \neq y^i]$$

k-means

k-means clustering technique is an iterative algorithm for partitioning objects into k subsets, $\mathcal{C}$.
A mass center (intra-cluster similarity by each attribute) C_j is called a **centroid** and computed as

$$c_j = \frac{1}{|C_j|} \sum_{i \in C_j} x_i$$

The cost function of the algorithm is the sum of squared distances between objects and their centroids

$$J(\mathcal{C}) = \sum_{j=1}^{k} \sum_{i \in C_j} d(x_i, c_j)^2$$

k-means

Main steps

Input: Data, k is a parameter

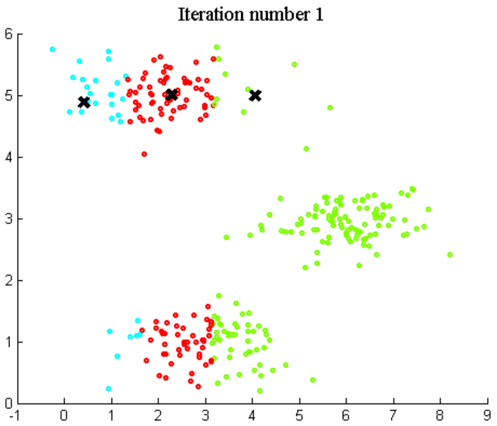
Output: Partition into k clusters

* * *

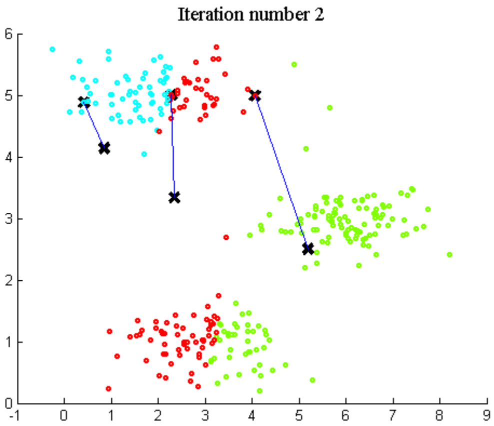
1. Initialisation: (Random) choice of k point as initial centroids.
2. Clusters' update: Having k centroids, assign each object to the nearest centroid. The object assigned to the same centroid c_j $(j = 1 \dots k)$ forms the cluster C_j.
3. Centroid's update: For each cluster C_j we need to compute its mass center, which should be the new centroid at the next iteration.

Iteration through steps 2-3 continue until clusters changes.

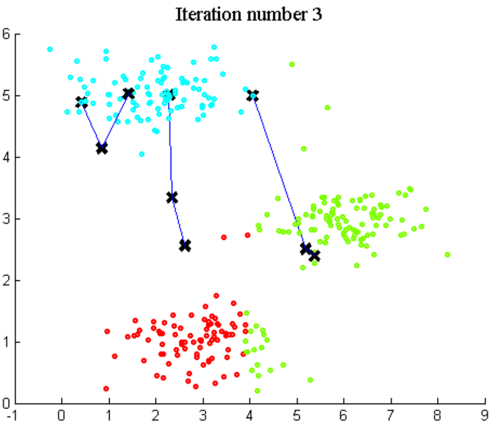
k-means example



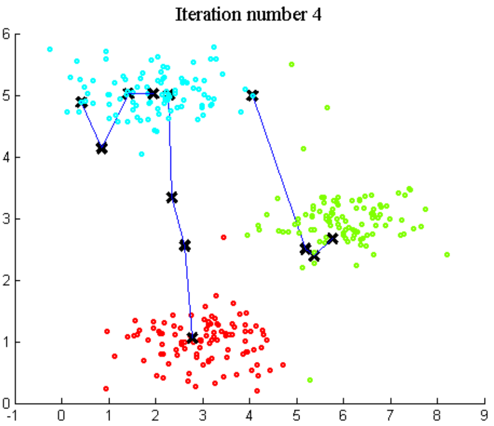
k-means example



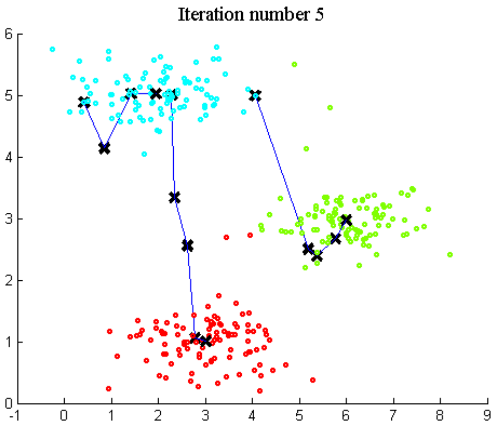
k-means example



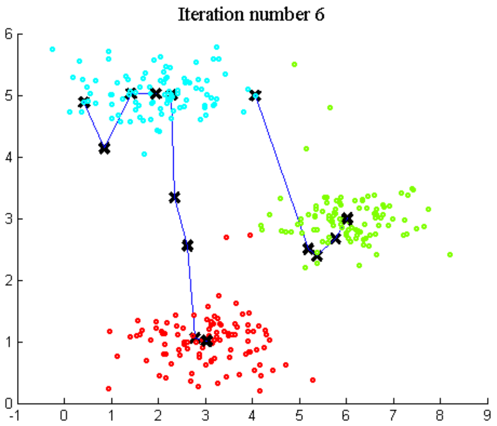
k-means example



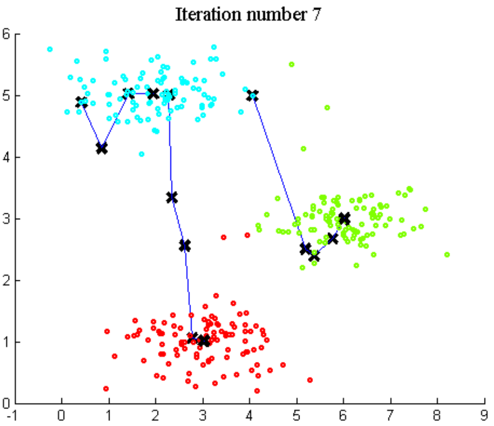
k-means example



k-means example



k-means example



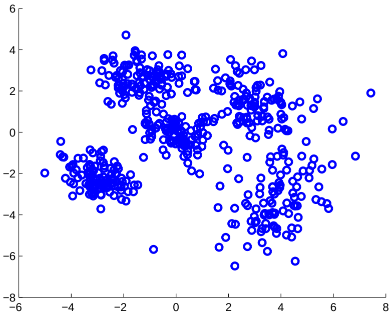
Quality measures and number of clusters

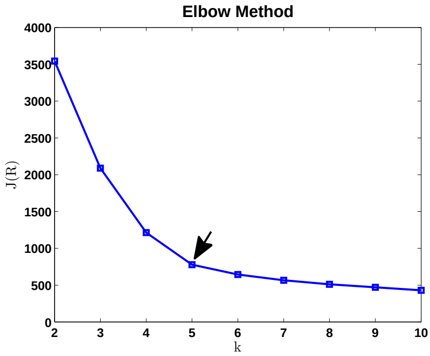
Elbow method

For each k one needs to calculate $J(\mathcal{C}_k)$.
Decision on the choice of the number of clusters is done at k' such that $J(\mathcal{C}_{k'+1})$ drops «not so fast», I.e. the following fraction is not high:

$$D(k) = \frac{|J(\mathcal{C}_k) - J(\mathcal{C}_{k+1})|}{|J(\mathcal{C}_{k-1}) - J(\mathcal{C}_k)|}$$

Quality measures and number of clusters
Elbow method





Quality measures and number of clusters
Silhouette

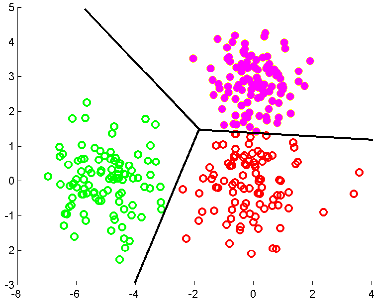
The **Silhouette** of cluster C_h is the following function:

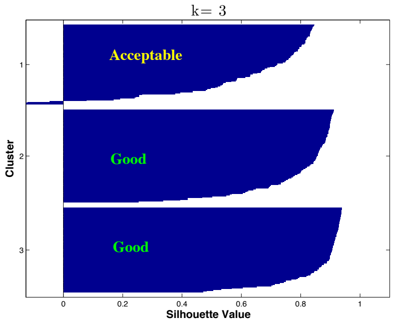
$$s_h(i) = \frac{\min_m \{b_m(i)\} - a(i)}{\max\{a(i), min(b_m(i))\}} \qquad (m = 1, \ldots, k, m \neq h),$$

where $a(i)$ is the average distance from i-th element of C_h to each of its remaining elements, and $b_m(i)$ is the average distance to elements of the «remaining» clusters.

Silhouette

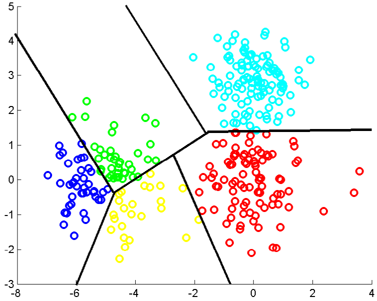
Appropriate number of clusters

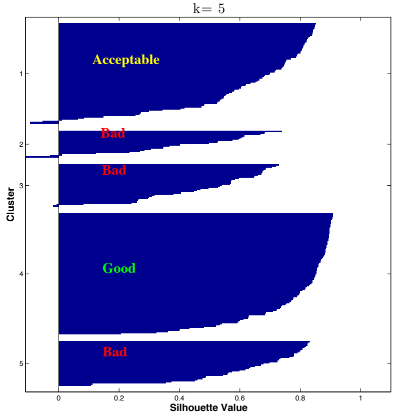




Silhouette

Inappropriate number of clusters





k-medoids

The idea of the method is similar to k-means, however, the centroids (here, medoids) are always the objects of an input sample.

- Robustness to noise
- Slow computations

Algorithm stages

1. Initialisation: Choose k objects as initial medoids M

2. Clusters' update: With k medoids, each object is assigned to the nearest medoid. The objects assigned to a certain medoid c_i $(i = 1 \dots k)$ form the cluster C_i

3. Medoids' replacement: For each pair (c_i, x_h), $c_i \in M$ and $x_h \in \overline{M}$ compute $Cost(c_i, x_h) = J(\mathcal{C}') - J(\mathcal{C})$, where $\mathcal{C}'$ is a partition with the medoid x_h instead c_i.
If $Cost < 0$, then c_i is replaced by x_h.

Iterative process 2–3 is continued until clusters do not change.

Fuzzy c-means
[Dunn, 1973; Bezdek, 1981]

- Let w_{ij} be the membership degree of an object x_i to cluster C_j, $i = 1, \ldots, n$, $j = 1, \ldots, k$.
- $\sum\limits_j w_{ij} = 1$.
- The objective function

$$J(C) = \sum_{j=1}^{k} \sum_{i \in C_j} w_{ij}^p d(x_i, c_j)^2$$

- p is the level of cluster fuzziness

Fuzzy c-means

Algorithm stages

Input: Data, parameter k

Output: Membership matrix $W^{n \times k}$

* * *

1. Initialisation: Assignment of k points as initial centroids

2. Membership degrees' update: $w_{ij} = \dfrac{(1/d(x_i, c_j)^2)^{\frac{1}{p-1}}}{\sum\limits_{q=1}^{k} (1/d(x_i, c_q)^2)^{\frac{1}{p-1}}}$

3. Centroids' update: $c_j = \dfrac{\sum\limits_{i=1}^{n} w_{ij}^p x_i}{\sum\limits_{i=1}^{n} w_{ij}^p}$

The iterative process 2-3 is continued until the obtained clusters do not change (or the objective function changes slightly).

Hierarchical approaches

From «object-attribute» matrices one need to go to matrices of pairwise distances

$$\begin{bmatrix} x_1^1 & x_1^2 & \cdots & x_1^m \\ x_2^1 & x_2^2 & \cdots & x_2^m \\ \cdots & \cdots & \cdots & \cdots \\ x_n^1 & x_n^m & \cdots & x_n^m \end{bmatrix} \Rightarrow \begin{pmatrix} d(x_1,x_1) & d(x_1,x_2) & \ldots & d(x_1,x_n) \\ d(x_2,x_1) & \ddots & \ddots & d(x_2,x_n) \\ \vdots & \ddots & \ddots & \vdots \\ d(x_n,x_1) & d(x_n,x_2) & \cdots & d(x_n,x_n) \end{pmatrix}$$

The matrix is symmetric with zeros on its main diagonal.

Hierarchical approaches

From «object-attribute» matrices one need to go to matrices of pairwise distances

$$\begin{bmatrix} x_1^1 & x_1^2 & \cdots & x_1^m \\ x_2^1 & x_2^2 & \cdots & x_2^m \\ \dots & \dots & \dots & \dots \\ x_n^1 & x_n^m & \cdots & x_n^m \end{bmatrix} \Rightarrow \begin{pmatrix} 0 & d(x_1,x_2) & d(x_1,x_3) & \cdots & d(x_1,x_n) \\ & 0 & d(x_2,x_3) & \cdots & d(x_2,x_n) \\ & & \ddots & \dots & \dots \\ & & & 0 & d(x_{n-1},x_n) \\ & & & & 0 \end{pmatrix}$$

The matrix is symmetric with zeros on its main diagonal.

Agglomerative clustering

Agglomerative approach performs consequent merging of similar clusters.

0. Start with 1-element clusters
1. Find the most similar clusters
2. Merge two the most similar clusters

Continue steps 1–2 until all objects are in in one cluster.

Assume that we have selected the distance metric and found a pair of the most similar objects. Then we merged that objects in a larger cluster.
Question: How to claculate the distance between the new and the remaining clusters?

Agglomerative clustering

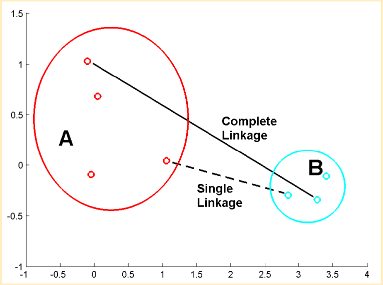
Linkage types

1. Single Linkage

$$d(A,B) = \min_{x \in A, y \in B} d(x,y)$$

2. Complete Linkage

$$d(A,B) = \max_{x \in A, y \in B} d(x,y)$$



Agglomerative clustering

Linkage types

3. Average Linkage

$$d(A,B) = \frac{1}{|A||B|} \sum_{i \in A} \sum_{j \in B} d(x_i, y_j)$$

4. Weighted (or Group) Average Linkage
   Let cluster $A$is obtained after union of q and p. Then

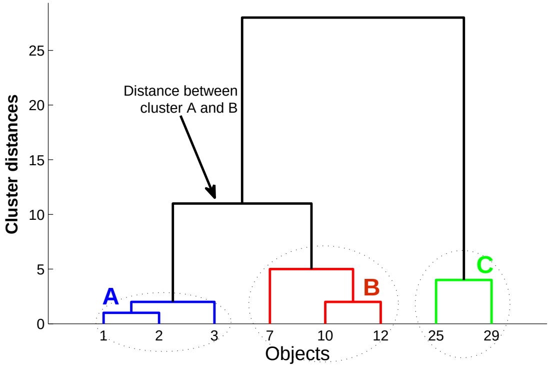
$$d(A,B) = \frac{d(p,B) + d(q,B)}{2}$$

5. Centroid Linkage

$$d(A,B) = \|c_A - c_B\|_2$$

Agglomerative clustering

The process of subsequent merging can be represented by a tree-like structure, i.e. its **dendrogram**.
Assume we have one-dimensional sample $\{\ 1,\ \ 2,\ \ 3,\ \ 7,\ \ 10,\ \ 12,\ \ 25,\ \ 29\ \}$

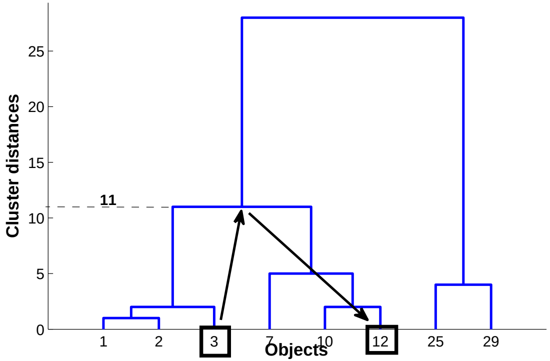


Quality Assessment

Cophenetic correlation

Dendrogrammatic distance

The dendrogrammatic distance between objects x_i and x_j is called the height of dendrogramm tree such that those objects are united in one cluster.



Quality Assessment

Cophenetic correlation

Cophenetic correlation

Cophenetic correlation is correlation coefficient between pairwise distances and the corresponding dendrogrammatic distances of the same set of objects.
With a «niclely» built tree those distance series should correlate well.

$$cophCorr = \frac{\sum\limits_{i<j}(d(x_i, x_j) - \overline{d})(coph(x_i, x_j) - \overline{coph})}{\sqrt{\sum\limits_{i<j}(d(x_i, x_j) - \overline{d})^2 \cdot \sum\limits_{i<j}(coph(x_i, x_j) - \overline{coph})^2}}$$

Divisive clustering

Divisive clustering is performed in the reverse order, i.e, by splitting larger clusters into smaller ones.

1. Find x_{i*} with the greatest average distance to the remaining objects. Add to the set of dismounted objects S.
2. For each object $x_i \notin S$ calculate the difference between average distances to objects from S and those which are not from S:

$$D_i = \frac{1}{|S|}\sum_{j\in S} d(x_i, x_j) - \frac{1}{|\overline{S}|}\sum_{j\notin S} d(x_i, x_j)$$

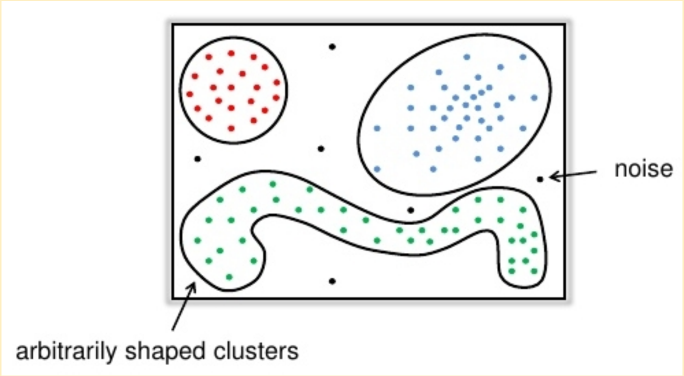
3. Add the object x_h with the lowest D_h to S
4. Repeat steps 2–3 until all D_i are positive
5. Repeat steps 1–4 for cluster with the greatest diameter (the greatest pairwise distance between a pair of objects)

Stop the process when the are only single element clusters.

Density-based methods

DBSCAN algorithm

DBSCAN (Density Based Spatial Clustering of Applications with Noise) is a density-based algorithm for clustering spatial data with noise which is able to recognise cluster of different shape.



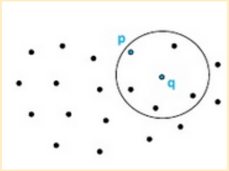
DBSCAN algorithm

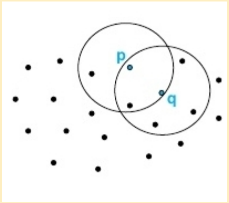
Main idea

For each point of a cluster its neighbourhood of a fixed radius should contain no less than M points, i.e. $N_{eps}(p) \geq M$, where $N_{eps}(p)$ is a set of points located within distance ε from p.
There is a problem with boundary points.

Definition

A point p is density directly reachable from another point q (with ε и M) if $p \in N_{\varepsilon}(q)$ and $|N_{\varepsilon}(q)| \geq M$.

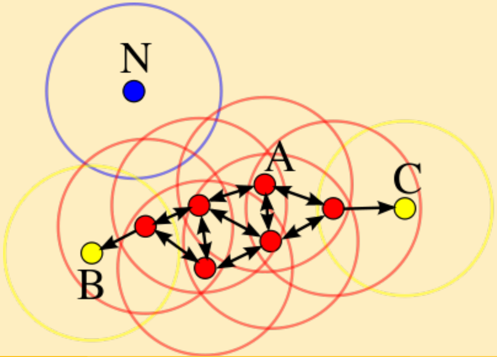




DBSCAN algorithm

Definiton

A point p is density reachable from another point q (with ε and M) if there exists sequence of points between them such that every its point is directly density reachable from the preceding one.
A point B is density connected (with ε and M) with a point C if there exists a point A such that B and C are density reachable from A (with the same ε and M).



DBSCAN

Cluster definition

A cluster C_j (with ε and M) is a non-empty set of objects:
1) $\forall p, q : p \in C_j, q$ is density reachable (with ε and M) from $p \Rightarrow q \in C_j$
2) $\forall p, q \in C_j : p$ is density connected (with ε and M) with q.

Pseudocode

```
Input: Data $D$, $\varepsilon$, $M$
Output: Clusters $\mathcal{C} = \{C_j\}$.
  1: All elements of $D$ has the flag value «not visited», $j \leftarrow 0$, $Noise \leftarrow \emptyset$
  2: for all $d_i \in D : flag(d_i) =$ «not visited» do
  3:     $flag(d_i) \leftarrow$ «visited», $N_i \leftarrow N_\varepsilon(d_i)$
  4:     if $|N_i| < M$ then
  5:         $Noise \leftarrow Noise \cup \{d_i\}$ (noise elimination)
  6:     else
  7:         $j \leftarrow j + 1$ (the index of the next cluster)
  8:         $Expand(d_i, N_i, C_j, \varepsilon, M)$ (cluster expansion)
  9: return $\mathcal{C} = \{C_j\}$
```

DBSCAN

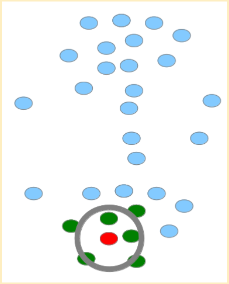
Pseudocode. Cluster expansion

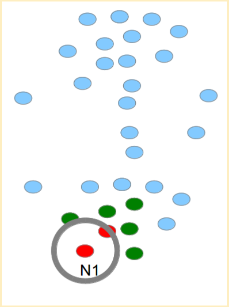
Input: Current object d_i and its N_i neighbourhood, current cluster C_j, ε, and M
Output: Cluster C_j.

```
 1: $C_j = C_j \cup \{d_i\}$
 2: for all $d_k \in N_i : flag(d_i) =$ «not visited» do
 3:     if $flag(d_k) =$ «not visited» then
 4:         $flag(d_k) \leftarrow$ «visited»
 5:         $N_{ik} \leftarrow N_\varepsilon(d_k)$
 6:         if $|N_{ik}| \geq M$ then
 7:             $N_i \leftarrow N_i \cup N_{ik}$
 8:     if $\nexists p : d_k \in C_p$ ($d_k$ is not assigned to any cluster yet) then
 9:         $C_j \leftarrow C_j \cup \{d_k\}$
10: return $\{C_j\}$
```

DBSCAN. An example

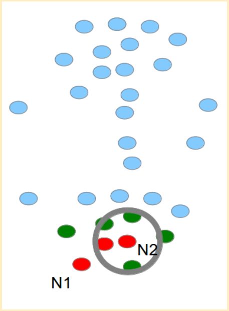
Initial parameters: $M = 4, \varepsilon > 0$.
We have taken randomly the first object. It has 6 neighbors from N_ε (left Fig.)
Then we create the first cluster (in red) and start its expansion. Since the first neighbour $N1$ is on the border, we add it to the cluster (right Fig.).

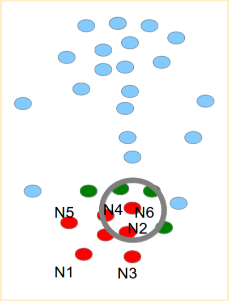




DBSCAN. An example

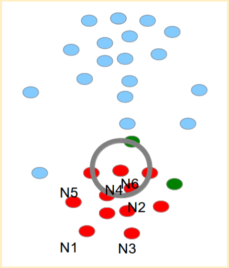
Then we go to the next neighbour $N2$. It has 5 own neighbours from N_{ik} (left Fig.). Then we add new neighbours to the existing ones (the new «green» neighbours). And so on. After visiting all the initial neighbours through $N1$ to $N6$ (right Fig.), we continue with the new «green» ones.

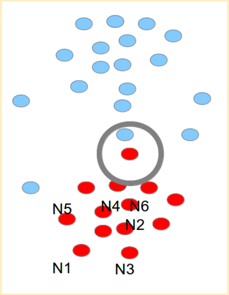




DBSCAN. An example

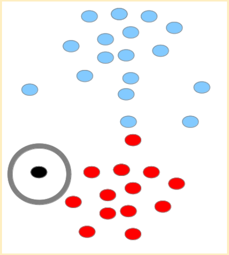
After processing the neighbours of points $N1$—$N6$ there are only two «green» points (left Fig.); after their processing the first cluster is formed (right Fig.) and then we randomly take a remaining point from the input set.

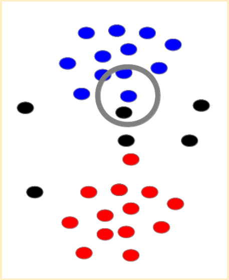




DBSCAN. An example

If the «lonely» point is taken, with the number of neighbours less than $M = 4$ (left Fig.), it is added to the set of noisy objects, *Noise*. Then again the algorithm randomly takes a new non-visited point. Finally, we have 2 clusters and 6 noisy objects (right Fig.). Note that there two point located between the clusters among the noisy objects («bottle neck»).





Drawbacks and advantages of DBSCAN

Advantages

+ Is able to find arbitrary shaped clusters
+ Easy to implement
+ Is able to eliminate noise
+ Time-efficient, $O(n \log n)$, with a proper choice of data structure (otherwise $O(n^2)$)

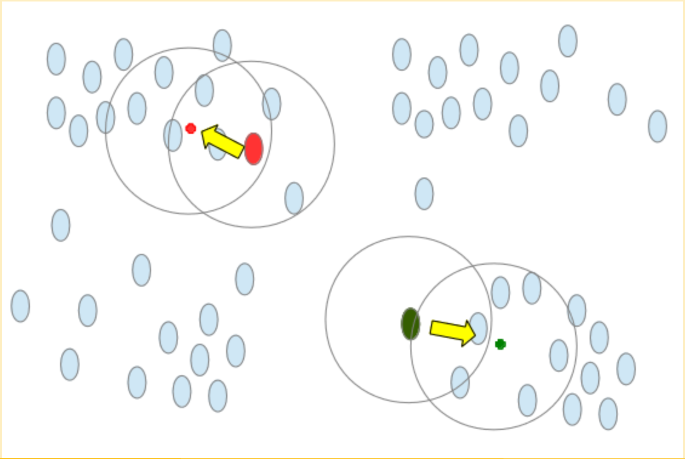
Drawbacks

- Is parametric. It works inappropriately with large density difference due to the fixed M (the minimal neighbours number). There are its modifications to cope with the problem.
- Depends of the choice of distance metric.

Mean-Shift

Idea

Find the mass center of input objects where where the density is maximal and use it as a centroid.

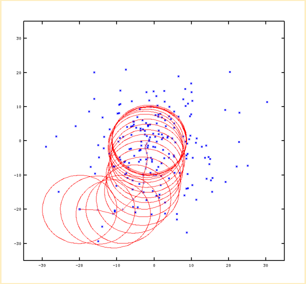


Mean-Shift

Idea

- Define a neighbourhood around each sample point
- Calculate its centroid
- Move the center of the neighbourhood into the centroid

After each iteration, the centroids are moved to the «more dense regions» until convergence to the density modes.



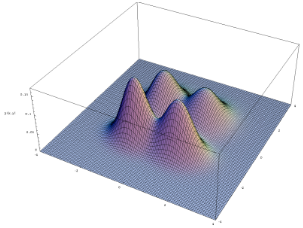
Mean-Shift

Density modes are defined by kernel density estimation:

$$\hat{f}(\mathbf{x}) = \frac{1}{nh^d}\sum_{i=1}^{n} K(\frac{\mathbf{x}-\mathbf{x_i}}{h}),$$

where d is the dimension of the feature space, h is a bandwidth.
$K(\frac{\mathbf{x}-\mathbf{x_i}}{h})$ is a kernel function that depends on the distance and calculates the contribution of neighbours x_i within the bandwidth. Often Gaussian kernel is used:

$$K(\frac{\mathbf{x}-\mathbf{x_i}}{h}) = \frac{1}{2\pi^{d/2}} e^{-\frac{||\mathbf{x}-\mathbf{x_i}||^2}{2h^2}}$$



Mean-Shift

Mean-Shift follows gradient ascent.

$$\nabla \hat{f}(\mathbf{x}) = \frac{1}{nh^d} \sum_{i=1}^{n} \frac{\partial}{\partial \mathbf{x}} K(\frac{\mathbf{x} - \mathbf{x_i}}{h})$$

$$\nabla \hat{f}(\mathbf{x}) = 0$$

For Gaussian kernel:

$$\frac{\partial}{\partial \mathbf{x}} K(\frac{\mathbf{x} - \mathbf{x_i}}{h}) = K(\frac{\mathbf{x} - \mathbf{x_i}}{h}) \frac{\mathbf{x} - \mathbf{x_i}}{h} \frac{1}{h}$$

$$\Longrightarrow \sum_{i=1}^{n} K(\frac{\mathbf{x} - \mathbf{x_i}}{h})\mathbf{x} = \sum_{i=1}^{n} K(\frac{\mathbf{x} - \mathbf{x_i}}{h})\mathbf{x_i}$$

Mean-Shift

The direction of the largest density increase is defined by the vector below:

$$\mathbf{m}(\mathbf{x}) = \frac{\sum\limits_{i=1}^{n} K(\frac{\mathbf{x}-\mathbf{x_i}}{h})\mathbf{x_i}}{\sum\limits_{i=1}^{n} K(\frac{\mathbf{x}-\mathbf{x_i}}{h})}$$

By the way, the mean shift is defined as follows:

$$\mathbf{m}(\mathbf{x}) - \mathbf{x} = \frac{\sum\limits_{i=1}^{n} K(\frac{\mathbf{x}-\mathbf{x_i}}{h})\mathbf{x_i}}{\sum\limits_{i=1}^{n} K(\frac{\mathbf{x}-\mathbf{x_i}}{h})} - \mathbf{x}$$

Mean-Shift

Steps

Input: Data D

Output: Clusters C_j

1. Compute mean-shift: For each point of the initial dataset $x_i \in D$ compute mean-short vector $\mathbf{m}(\mathbf{x_i})$

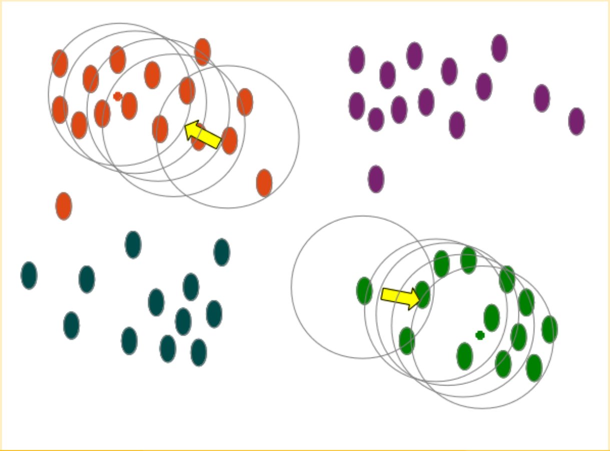
2. Expand cluster: The argument of the kernel density function is shifted by $\mathbf{m}(\mathbf{x}) : \hat{f}(\mathbf{x}) \to \hat{f}(\mathbf{m}(\mathbf{x}) - \mathbf{x})$.

Steps 1 and 2 are repeated until convergence to modes of the kernel density function.

Mean-Shift

Convergence to a local maxima is guaranteed

Yizong Cheng, Mean Shift, Mode Seeking, and Clustering. IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE,(8) 1995



Questions and contacts

www.hse.ru/staff/dima

Thank you!

dmitrii.ignatov[at]gmail.com