

Clustering

Victor Kitov

v.v.kitov@yandex.ru

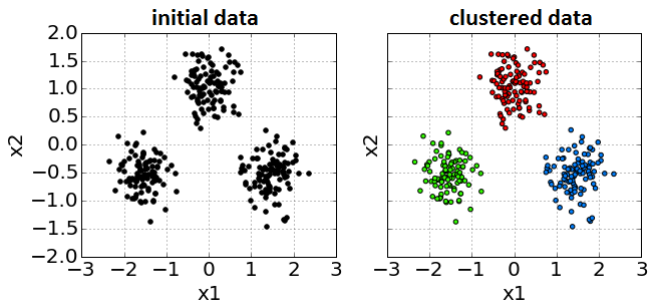
Table of Contents

- 1 Clustering introduction
- 2 Representative-based clustering
- 3 Hierarchical clustering
- 4 Probabilistic clustering
- 5 Density based approaches
- 6 Spectral clustering

Aim of clustering

- Clustering is partitioning of objects into groups so that:
 - inside groups objects are very similar
 - objects from different groups are dissimilar
- Unsupervised learning
- No definition of “similar”
 - different algorithms use different formalizations of similarity

Clustering demo



Applications of clustering

- data summarization
 - feature vector is replaced by cluster number
- feature extraction
 - cluster number, distance to native cluster center / other clusters
- customer segmentation
 - e.g. for recommender service
- community detection in networks
 - nodes - people, similarity - number of connections
- outlier detection
 - outliers do not belong any cluster

Clustering algorithms comparison

We can compare clustering algorithms in terms of:

- computational complexity
- do they build flat or hierarchical clustering?
- can the shape of clustering be arbitrary?
 - if not is it symmetrical, can clusters be of different size?
- can clusters vary in density of contained objects?
- robustness to outliers

Table of Contents

- 1 Clustering introduction
- 2 Representative-based clustering
 - K-means
 - Kernel K-means
 - Mahalanobis distance
 - K-medoids
- 3 Hierarchical clustering
- 4 Probabilistic clustering
- 5 Density based approaches

Representative-based clustering

- Clustering is flat (not hierarchical)
- Number of clusters K is specified in advance
- Each object x_n is associated cluster z_n
- Each cluster C_k is defined by its representative μ_k , $k = 1, 2, \dots, K$.¹
- Criterion to find representatives μ_1, \dots, μ_K :

$$Q(z_1, \dots, z_K) = \sum_{n=1}^N \min_k \rho(x_n, \mu_k) \rightarrow \min_{\mu_1, \dots, \mu_K} \quad (1)$$

¹Propose clustering algorithm that can extract a set of representatives for each cluster.

Generic algorithm

```
initialize  $\mu_1, \dots, \mu_K$  from random training objects

while not converged:
  for  $n = 1, 2, \dots, N$ :
     $z_n = \arg \min_k \rho(x_n, \mu_k)$ 

  for  $k = 1, 2, \dots, K$ :
     $\mu_k = \arg \min_{\mu} \sum_{n: z_n=k} \rho(x_n, \mu)$ 

return  $z_1, \dots, z_N$ 
```

- Comments:
 - different distance functions lead to different algorithms:
 - $\rho(x, x') = \|x - x'\|_2^2 \Rightarrow$ K-means
 - $\rho(x, x') = \|x - x'\|_1 \Rightarrow$ K-medians
 - μ_k may be arbitrary/constrained to be existing objects
 - converges in few iterations, complexity $O(NKD)$

Comments

- K - unknown parameter
 - if chosen small \Rightarrow distinct clusters will get merged
 - better to take K larger and then merge similar clusters.
- Shape of clusters is defined by $\rho(\cdot, \cdot)$
- Close clusters will have similar size

2 Representative-based clustering

- K-means
- Kernel K-means
- Mahalanobis distance
- K-medoids

K-means algorithm

- Suppose we want to cluster our data into K clusters.
- Cluster i has a center μ_i , $i=1,2,\dots,K$.
- Consider the task of minimizing

$$\sum_{n=1}^N \|x_n - \mu_{z_n}\|_2^2 \rightarrow \min_{z_1, \dots, z_N, \mu_1, \dots, \mu_K} \quad (2)$$

where $z_i \in \{1, 2, \dots, K\}$ is cluster assignment for x_i and μ_1, \dots, μ_K are cluster centers.

- Direct optimization requires full search and is impractical.
- K-means is a suboptimal algorithm for optimizing (2).

K-means algorithm

Initialize $\mu_j, j = 1, 2, \dots, K$.

repeat while stop condition not satisfied:

 for $i = 1, 2, \dots, N$:

 find cluster number of x_i :

$$z_i = \arg \min_{j \in \{1, 2, \dots, K\}} \|x_i - \mu_j\|_2^2$$

 for $j = 1, 2, \dots, K$:

$$\mu_j = \frac{1}{\sum_{n=1}^N \mathbb{I}[z_n = j]} \sum_{n=1}^N \mathbb{I}[z_n = j] x_n$$

Dynamic K-means algorithm

Initialize $\mu_j, j = 1, 2, \dots, K, z_i = 0, i = 1, 2, \dots, N$

repeat while stop condition not satisfied:

 for $i = 1, 2, \dots, N$:

 find cluster number of x_i :

$$z'_i = \arg \min_{j \in \{1, 2, \dots, K\}} \|x_i - \mu_j\|_2^2$$

 if $z'_i \neq z_i$:

 recalculate cluster means μ_{z_i} and $\mu_{z'_i}$:

$$\mu_{z_i} = \frac{1}{\sum_{n=1}^N \mathbb{I}[z'_n = z_i]} \sum_{n=1}^N \mathbb{I}[z'_n = z_i] x_n$$

$$\mu_{z'_i} = \frac{1}{\sum_{n=1}^N \mathbb{I}[z'_n = z'_i]} \sum_{n=1}^N \mathbb{I}[z'_n = z'_i] x_n$$

$$z_i = z'_i$$

Converges in less iterations, situation when no objects correspond to some cluster is impossible.

K-means properties

Possible stop conditions:

- cluster assignments z_1, \dots, z_N stop to change (typical)
- maximum number of iterations reached
- cluster means $\{\mu_i\}_{i=1}^K$ stop changing significantly

Initialization:

- typically $\{\mu_i\}_{i=1}^K$ are initialized to randomly chosen training objects

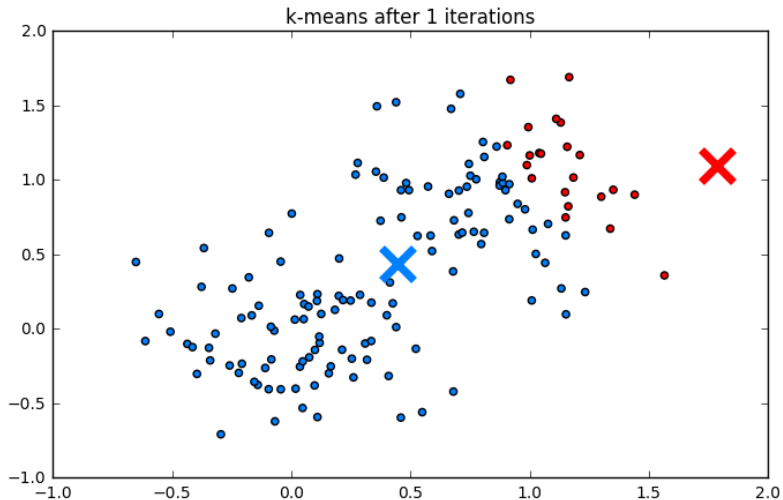
Optimality:

- criteria is non-convex
- solution depends on starting conditions
- we may restart several times from diff. random starting points and select solution giving minimal value of (2).

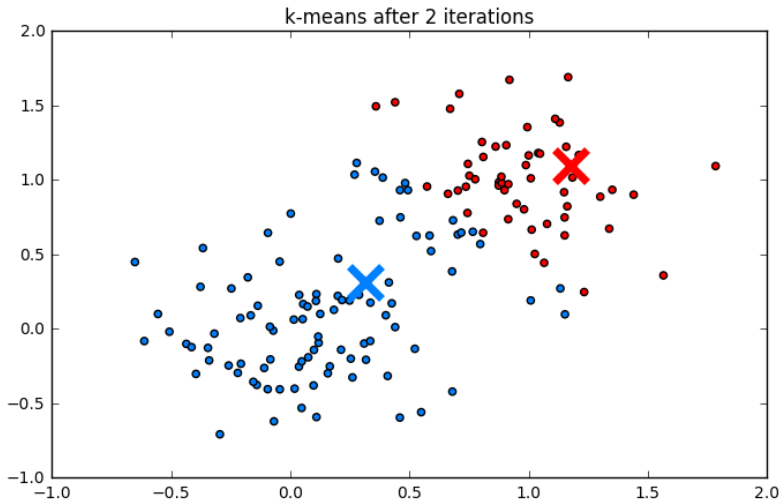
Complexity: $O(NDKI)$, where K is the number of clusters and I is the number of iterations.

- Usually algorithm converges in small number of iterations I .

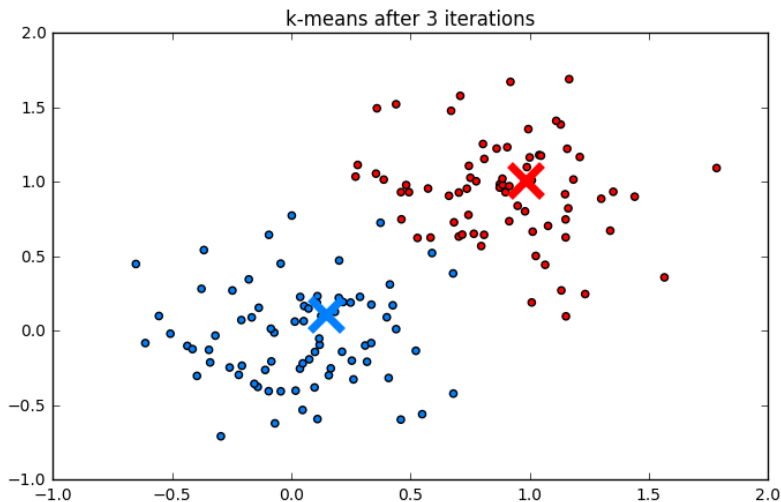
Example of K-means



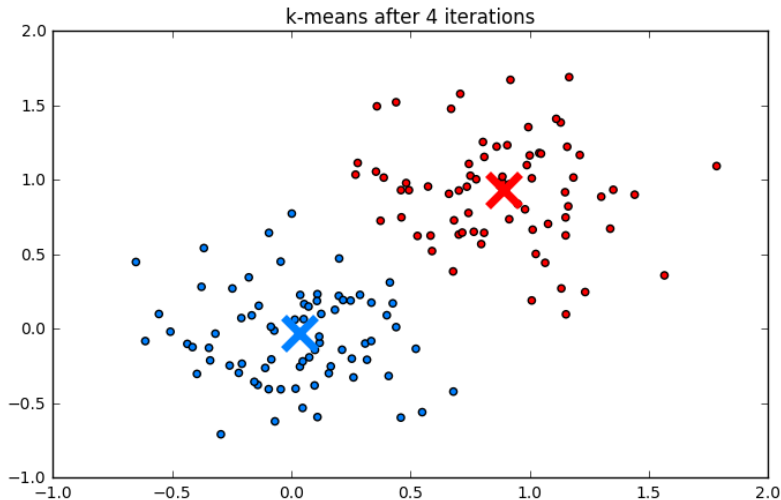
Example of K-means



Example of K-means



Example of K-means



Gotchas

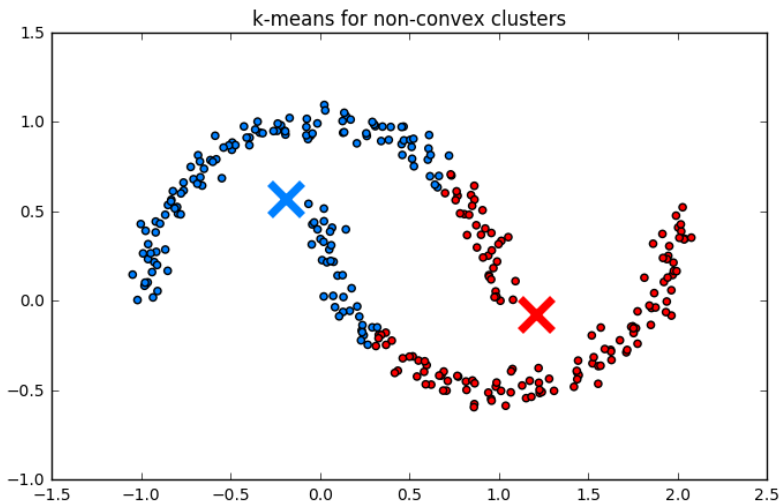
- K-means assumes that clusters are convex:

K-means clustering on the digits dataset (PCA-reduced data)
Centroids are marked with white cross

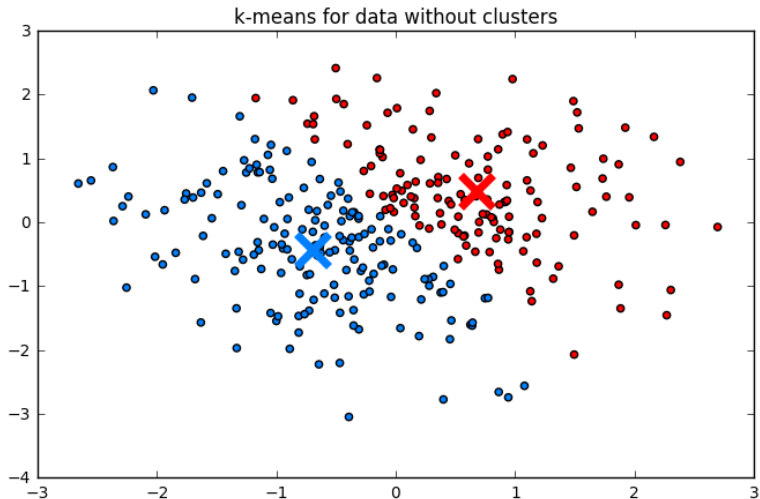


- It always finds clusters even if none actually exist
 - need to control cluster quality metrics

K-means for non-convex clusters



K-means for data without clusters



K-means and EM algorithm

```
Initialize  $\mu_j, j = 1, 2, \dots, K$ .
```

```
repeat while stop condition not satisfied:
```

```
  for  $i = 1, 2, \dots, N$ :
```

```
    find cluster number of  $x_i$ :
```

```
     $z_i = \arg \min_{j \in \{1, 2, \dots, g\}} \|x_i - \mu_j\|$ 
```

```
  for  $j = 1, 2, \dots, K$ :
```

```
    
$$\mu_j = \frac{1}{\sum_{n=1}^N \mathbb{I}[z_n = j]} \sum_{n=1}^N \mathbb{I}[z_n = j] x_n$$

```

- K-means is EM-algorithm when:

K-means and EM algorithm

```
Initialize  $\mu_j, j = 1, 2, \dots, K.$ 
```

```
repeat while stop condition not satisfied:
```

```
  for  $i = 1, 2, \dots, N$ :
```

```
    find cluster number of  $x_i$ :
```

```
     $z_i = \arg \min_{j \in \{1, 2, \dots, g\}} \|x_i - \mu_j\|$ 
```

```
  for  $j = 1, 2, \dots, K$ :
```

$$\mu_j = \frac{1}{\sum_{n=1}^N \mathbb{I}[z_n = j]} \sum_{n=1}^N \mathbb{I}[z_n = j] x_n$$

- K-means is EM-algorithm when:
 - applied to Gaussians
 - with equal priors
 - with unity covariance matrices
 - with hard clustering

K-means

- Not robust to outliers
 - K-medians is robust
- K-representatives may create singleton clusters in outliers if centroids get initialized with outlier
 - better to init centroids with mean of m randomly chosen objects
- Constructs spherical clusters of similar radii
 - Allows kernel version which can find non-convex clusters in original space

2 Representative-based clustering

- K-means
- **Kernel K-means**
- Mahalanobis distance
- K-medoids

Kernel K-means

- Let $C_k := \{n : z_n = k\}$ - indices of objects in cluster k .
- Squared distance to centroid:

$$\begin{aligned} \rho(x, \mu_k)^2 &= \|x - \mu_k\|^2 = \langle \varphi(x) - \frac{1}{|C_k|} \sum_{i \in C_k} \varphi(x_i), \varphi(x) - \frac{1}{|C_k|} \sum_{i \in C_k} \varphi(x_i) \rangle \\ &= \langle \varphi(x), \varphi(x) \rangle - 2 \langle \varphi(x), \frac{1}{|C_k|} \sum_{i \in C_k} \varphi(x_i) \rangle + \frac{1}{|C_k|^2} \sum_{i, j \in C_k} \langle \varphi(x_i), \varphi(x_j) \rangle \\ &= K(x, x) - 2 \frac{1}{|C_k|} \sum_{i \in C_k} K(x, x_i) + \frac{1}{|C_k|^2} \sum_{i, j \in C_k} K(x_i, x_j) \end{aligned}$$

```
initialize  $C_1, \dots, C_K$ 
```

```
while not converged:
```

```
  for  $n = 1, 2, \dots, N$ :
```

```
     $z_n = \arg \min_k \rho(x_n, \mu_k)^2$ 
```

```
return  $z_1, \dots, z_N$ 
```

Intuition

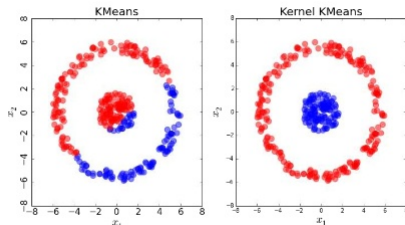
- Consider RBF kernel $K(x, \mu) = e^{-\gamma\|x-\mu\|^2}$.

$$\rho(x, \mu_k)^2 = 1 - 2 \frac{1}{|C_k|} \sum_{i \in C_k} e^{-\gamma\|x-x_i\|^2} + \frac{1}{|C_k|^2} \sum_{i,j \in C_k} e^{-\gamma\|x_i-x_j\|^2}$$

- $\frac{1}{|C_k|} \sum_{i \in C_k} e^{-\gamma\|x-x_i\|^2}$ - average similarity of x to points in cluster k
- $\frac{1}{|C_k|^2} \sum_{i,j \in C_k} e^{-\gamma\|x_i-x_j\|^2}$ - constant offset for cluster k , measuring its compactness.

Kernel K-means

Kernel K-means vs. K-means



Pyclus: Open Source Data Clustering Package

- Complexity: with respect to N each iteration $O(N^2)$, assuming small num of iterations total $O(N^2)$.
- Centroids are not calculated directly
- Allows non-convex clustering in original feature space.

2 Representative-based clustering

- K-means
- Kernel K-means
- Mahalanobis distance
- K-medoids

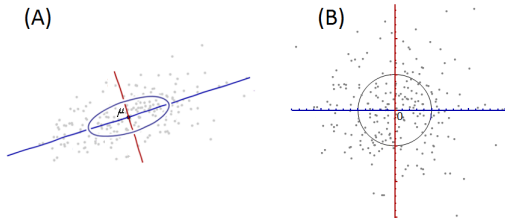
Mahalanobis distance

- Consider statistical distribution $F(\mu, \Sigma)$ with mean μ and covariance matrix Σ :
- Mahalanobis distance from x to $F(\mu, \Sigma)$:

$$\rho(x, F(\mu, \Sigma))^2 = (x - \mu)^T \Sigma^{-1} (x - \mu)$$

- Mahalanobis distance from x to another point x' , given $F(\mu, \Sigma)$:

$$\rho(x, x')^2 = (x - x')^T \Sigma^{-1} (x - x')$$

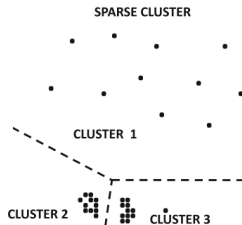


Mahalanobis distance clustering

- Mahalanobis distance in clustering:

$$\rho(x, \mu_k) = (x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k)$$

- is different for each k
- μ_k and Σ_k - sample mean and covariance matrix for objects from cluster k
- Mahalanobis distance allow modeling clusters
 - elliptically elongated
 - of different size and density



- 2 Representative-based clustering
 - K-means
 - Kernel K-means
 - Mahalanobis distance
 - K-medoids

K-medoids

- K-medoids - each cluster representative μ_k should be existing object from the training set.
- Motivation:
 - robust to outliers
 - more interpretable (representative is existing object)
 - the only option if we can calculate $\rho(x, x')$ but x, x' are incomparable elementwise
 - e.g. x_n - time series of varying length

K-medoids algorithm

```
initialize  $\mu_1, \dots, \mu_K$  from random training objects

while not converged:
    generate replacement candidates  $R = (\mu_k(i), x_n(i))_{i=1}^I$ 
    select replacement maximally improving  $\sum_{n=1}^N \min_k \rho(x_n, \mu_k)$ 

    if improvement was not achieved:
        fallback to previous state
        break

for  $n = 1, 2, \dots, N$ :
     $z_n = \arg \min_k \rho(x_n, \mu_k)$ 

return  $z_1, \dots, z_N$ 
```

As replacement candidates we may generate all variants or random subset.

General comments on K-representatives

- Init $\{\mu_k\}_{k=1}^K$ with
 - random objects from training set
 - centroids of m randomly selected objects from training set (more robust to outliers)
- K-representatives has non-convex optimization criteria
 - depends in initialization of $\{\mu_k\}_{k=1}^K$
 - so we can restart clustering from different starting conditions and select the one, maximizing (1)
- Outliers can create singleton clusters consisting of 1 point.
 - apply outlier filtering beforehand
 - alternatively during clustering for clusters with too few points replace cluster centroids with random objects.

Table of Contents

- 1 Clustering introduction
- 2 Representative-based clustering
- 3 Hierarchical clustering
 - Bottom-up hierarchical clustering
 - Top-down hierarchical clustering
- 4 Probabilistic clustering
- 5 Density based approaches
- 6 Spectral clustering

Motivation

- Number of clusters K not known a priori.
- Clustering is usually not flat, but hierarchical with different levels of granularity:
 - sites in the Internet
 - books in library
 - animals in nature

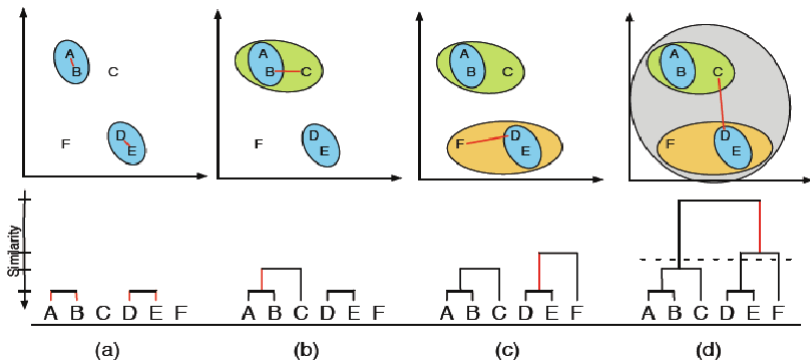
Hierarchical clustering

Hierarchical clustering may be:

- top-down
 - hierarchical K-means
- bottom-up
 - agglomerative clustering

- 3 Hierarchical clustering
 - Bottom-up hierarchical clustering
 - Top-down hierarchical clustering

Bottom-up clustering demo



Algorithm

```
initialize  $N \times N$  distance matrix  $M$  between  
singleton clusters  $\{x_1\}, \dots, \{x_N\}$ 
```

REPEAT:

- 1) pick closest pair of clusters i and j
- 2) merge clusters i and j
- 3) delete rows/columns i, j from M and add
new row/column for merged cluster

UNTIL 1 cluster is left

RETURN hierarchical clustering of objects

- Early stopping is possible when:
 - K clusters are left
 - distance between most close clusters \geq threshold

Agglomerative clustering - distances

- Consider clusters $A = \{x_{i_1}, x_{i_2}, \dots\}$ and $B = \{x_{j_1}, x_{j_2}, \dots\}$.
- We can define the following natural distances

- nearest neighbour (or single link)

$$\rho(A, B) = \min_{a \in A, b \in B} \rho(a, b)$$

- furthest neighbour (or complete-link)

$$\rho(A, B) = \max_{a \in A, b \in B} \rho(a, b)$$

- group average link

$$\rho(A, B) = \text{mean}_{a \in A, b \in B} \rho(a, b)$$

- closest centroid

$$\rho(A, B) = \rho(\mu_A, \mu_B)$$

where $\mu_U = \frac{1}{|U|} \sum_{x \in U} x$ or $m_U = \text{median}_{x \in U} \{x\}$

Intercluster distance properties³

- Nearest neighbour
 - extracts clusters of arbitrary shape
 - may merge distinct clusters connected by mistake by outliers
 - $M_{(i \cup j)k} = \min\{M_{ik}, M_{jk}\}$
- Furthest neighbour
 - creates very compact clusters
 - diameter of clusters grows
 - $M_{(i \cup j)k} = \max\{M_{ik}, M_{jk}\}$
- Group average link² and closest centroid distance give the compromise between nearest and furthest neighbour.

²How $M_{(i \cup j)k}$ will be recalculated for average link?

³Suppose we modify distance $\rho(x, x')$ with monotone transformation F : $\rho'(x, x') = F(\rho(x, x'))$. Which of the cluster distances will not be affected by this change?

Intercluster distance properties

Group average link is preferred to closest centroid distance, because

- centroid distance may lead to non-monotonous joining distance sequences in agglomerative algorithm.
- in contrast nearest neighbour, furthest neighbour and group average link always lead to monotonous joining distance sequences
- representation of cluster by mean/median ignores cluster shape
- centroid and median distance tend to prefer larger clusters, for which means are generally closer.

Variance based clustering

- For each cluster i keeps statistics:

$$m_i = |C_i|, F_i^d = \sum_{k \in C_i} x_k^d, S_i^d = \sum_{k \in C_i} (x_k^d)^2$$

- Using statistics we can calculate in-cluster variance

$$V_i = \sum_{d=1}^D \left[\frac{S_i^d}{m_i} - \left(\frac{F_i^d}{m_i} \right)^2 \right]$$

- Distance:

$$\rho(A, B) = V_{A \cup B} - V_A - V_B$$

Complexity

- Memory requirements: $O(N^2)$ - keep all pairwise distances.
- Computational requirements:
 - $O(D)$ - distance calculation
 - $O(N^2 D)$ - calculate all pairwise distances
 - Binary min-heap of size m : $O(\ln m)$ -insert element, $O(\ln m)$ -delete element, $O(1)$ -find min
 - Create heap of N^2 pairwise distances: $O(N^2 \ln N)$
 - merging of clusters:
 - find minimum $O(1)$, delete $O(\ln N)$, calculate $O(N)$, insert $O(\ln N)$
 - do it N times: $O(N^2)$
 - total complexity: $(N^2 D + N^2 \ln N)$
- When N is large we can:
 - use only random subsample of objects
 - merge points with K —representatives to K clusters to which apply agglomerative clustering.

K-representatives+agglomerative clustering

- Efficient combination:
 - 1 apply K-representatives with $M > K$ clusters
 - 2 use agglomerative clustering to merge excessive clusters to K
 - K-means has complexity $O(N)$
 - agglomerative clustering complexity $O(M^2 \ln M)$
 - but agglomerative clustering allows non-convex clusters!

- 3 Hierarchical clustering
 - Bottom-up hierarchical clustering
 - Top-down hierarchical clustering

Algorithm

INPUT:

data D , flat clustering algorithm A
leaf selection criterion, termination criterion

Initialize tree T to root, containing all data

REPEAT

based on selection criterion, select leaf L
using algorithm A split L into children L_1, \dots, L_K
add L_1, \dots, L_K as child nodes to tree T

UNTIL termination criterion

Comments

- Leaf selection criterion:
 - split leaf most close to the root
 - balanced tree by height
 - split leaf with maximum elements
 - balanced tree by cluster weight
- Building hierarchy top-down is more natural for a human

Table of Contents

- 1 Clustering introduction
- 2 Representative-based clustering
- 3 Hierarchical clustering
- 4 Probabilistic clustering**
- 5 Density based approaches
- 6 Spectral clustering

EM-algorithm for normal mixtures

Initialize ϕ_j, μ_j and Σ_j , $j = 1, 2, \dots, g$.

repeat while stopping condition not satisfied:

E-step. Calculate correspondences of x_n
to component z :

for $n = 1, 2, \dots, N$:

for $z = 1, 2, \dots, Z$:

$$w_{nz} = \frac{\phi_z N(x_n; \mu_z, \Sigma_z)}{\sum_k \phi_k N(x_n; \mu_k, \Sigma_k)} \quad \# = p(z|x(n))$$

M-step. Update component parameters:

for $z = 1, 2, \dots, Z$:

$$\hat{\phi}_z = \frac{1}{N} \sum_{n=1}^N w_{nz}$$

$$\hat{\mu}_z = \frac{\sum_{n=1}^N w_{nz} x_n}{\sum_{n=1}^N w_{nz}}$$

$$\hat{\Sigma}_z = \frac{1}{\sum_{n=1}^N w_{nz}} \sum_{n=1}^N w_{nz} (x_n - \hat{\mu}_z)(x_n - \hat{\mu}_z)^T$$

K-means versus EM clustering

- For each x_n EM algorithm gives $w_{nz} = p(z|x_n)$.
- This is soft or probabilistic clustering into Z clusters, having priors ϕ_1, \dots, ϕ_Z and probability distributions $p(x; \theta_1), \dots, p(x; \theta_Z)$.
- We can make it hard clustering using $z_n = \arg \max_z w_{nz}$.

- EM clustering becomes K-means clustering when:
 - applied to Gaussians
 - with equal priors
 - with unity covariance matrices
 - with hard clustering

Table of Contents

- 1 Clustering introduction
- 2 Representative-based clustering
- 3 Hierarchical clustering
- 4 Probabilistic clustering
- 5 Density based approaches**
 - Grid-based clustering
 - DBScan
 - Clustering by relevant density peaks
- 6 Spectral clustering

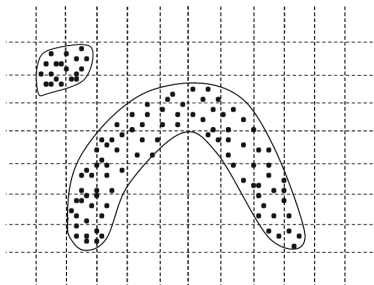
- 5 Density based approaches
 - Grid-based clustering
 - DBScan
 - Clustering by relevant density peaks

Grid-based clustering

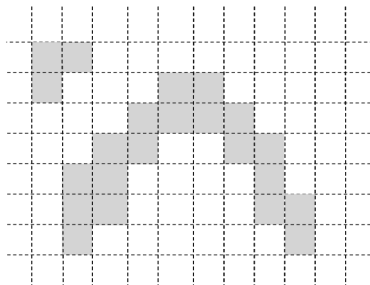
- Divide each dimension into p equal intervals
- Obtain p^D hypercubes
- Consider hypercube filled when it contains $\geq k$ points.
 - need not consider all possible hypercubes - look at data distribution along each axis.
- Consider hypercubes locally connected if they share $r < D$ common dimensions
 - $r=0$: corner, $r=1$: border, $r>1$: side
- Create graph:
 - node - filled hypercube
 - edges - between locally connected hypercubes
- Clusters: connected components in the graph⁴

⁴Propose an algorithm to index all objects with connected components they belong to.

Illustration



(a) Data points and grid



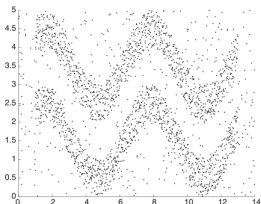
(b) Agglomerating adjacent grids

Discussion

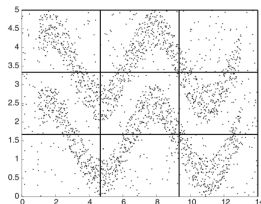
- Number of clusters is determined automatically
- Clusters may have arbitrary shape
- Need to specify: p, k, r .⁵
- Method will fail when cluster has varying density.
 - K-representatives - not, but it will fail for clusters of different size
 - mixture of Gaussians - not, but it will fail for non-elliptic clusters

⁵Under what selection of p, k the algorithm will have tendency to:

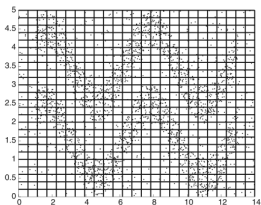
- join distinct clusters?
- separate true cluster due to local variations in density?

Selection of p 

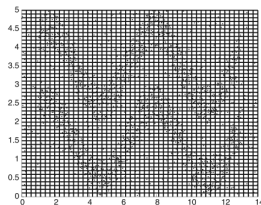
(a) Arbitrarily-shaped clusters



(b) Rough-grained grid

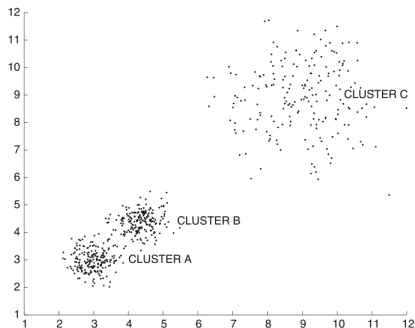


(c) Moderate-grained grid



(d) Fine-grained grid

Failure for varying density

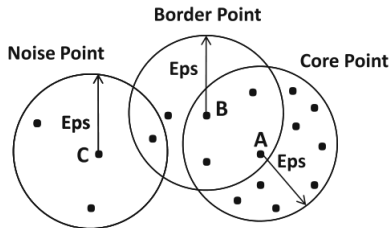


- Large k : cluster C is missed
- Small k : clusters A and B get merged

- 5 Density based approaches
 - Grid-based clustering
 - DBScan
 - Clustering by relevant density peaks

DBScan

- Core point: point having $\geq k$ points in its ε neighbourhood
- Border point: not core point, having at least 1 core point in its ε neighbourhood
- Noise point: neither a core point nor a border point



- k, ε - parameters of the method.

Algorithm

INPUT: training set, parameters ϵ, k .

- 1) Determine core, border and noise points with ϵ, k .
- 2) Create graph in which core points are connected if they are within ϵ of one another
- 3) Determine connected components in the graph
- 4) Assign each border point to connected component with which it is best connected

RETURN points in each connected component as a cluster

Comments

- Connecting core points - agglomerative clustering with single linkage, stopping at distance ϵ .
- Resistant to outliers by ignoring noise points.
- Similar to grid-based clustering:
 - automatically determines the number of clusters
 - works badly for density varying clusters
- Complexity $O(N^2 D)$
 - can be reduced to $O(N \ln N)$ for small D with spatial indexing.
 - grid-based methods find objects in the same region in $O(D)$.

- 5 Density based approaches
 - Grid-based clustering
 - DBScan
 - Clustering by relevant density peaks

Gradient ascent clustering

INPUT: training set x_1, \dots, x_N , step size η ,
kernel $K(\cdot)$, bandwidth h .

Define kernel density of objects: $\rho(x) = \frac{1}{Nh^D} \sum_{n=1}^N K\left(\frac{\rho(x, x_n)}{h}\right)$

FOR $n = 1, \dots, N$:

$z_0 = x_n, i = 0$

REPEAT while not converged:

$z_{i+1} = z_i + \eta \nabla \rho(z_i)$

$i = i + 1$

associate x_n to peak z_i

Merge almost identical peak positions z_1, \dots, z_N

RETURN clusters of data points, converging to the same peak.

Mean shift clustering

INPUT: training set x_1, \dots, x_N , step size η ,
kernel $K(\cdot)$, bandwidth h .

FOR $n = 1, \dots, N$:

$z_0 = x_n, i = 0$

REPEAT while not converged:

$$z_{i+1} = \frac{\sum_{k=1}^N K(\rho(z, x_k)/h) x_k}{\sum_{k=1}^N K(\rho(z, x_k)/h)}$$

$i = i + 1$

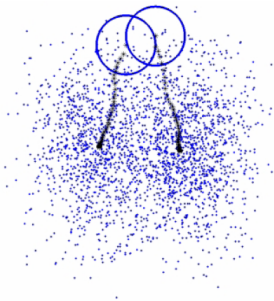
associate x_n to peak z_i

Merge almost identical peak positions z_1, \dots, z_N

RETURN clusters of data points, converging to the same peak.

Comments

Mean shift convergence process



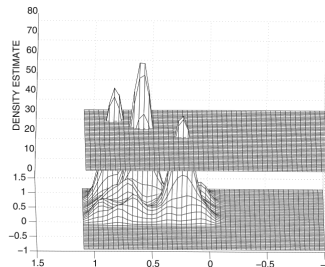
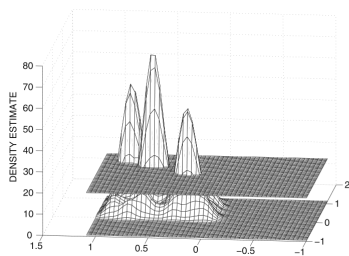
- Mean shift clustering is equivalent to steepest gradient clustering.
- Usually RBF kernel $K(\rho(x, x')/h) = e^{-\rho(x, x')^2/h^2}$ is used
- Efficient to discard objects that are outside some ε -neighbourhood of z_i in z_i recalculation.

DENCLUE clustering

- INPUT: training set x_1, \dots, x_N , threshold τ , step size η , kernel $K(\cdot)$, bandwidth h .
- Cluster points using gradient ascent or mean shift algorithm.
- Discard clusters, corresponding to peaks with $p(x) < \tau$
- Merge clusters, connected by path of data points $\{p(x_{i(k)})\}_{k=1}^K$, having $p(x_{i(k)}) \geq \tau$ $k = 1, 2, \dots, K$.
- OUTPUT: cluster indices of x_1, \dots, x_N .

DENCLUE Comments

- Depending on threshold τ may obtain different number of clusters:



- Automatically determines number of clusters, given τ .
- Clusters can be of arbitrary shape
- By varying τ , we can build hierarchical clustering.

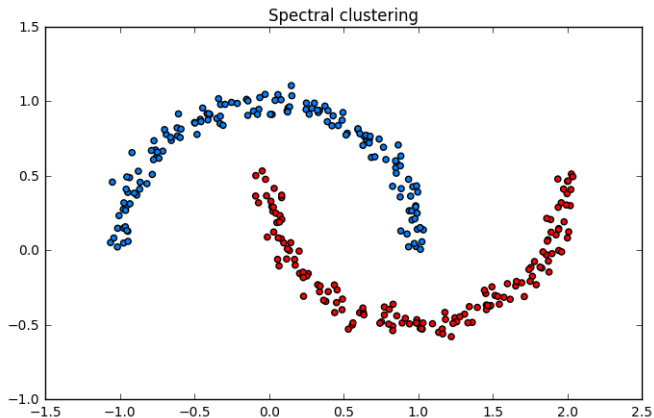
DENCLUE Comments

- DENCLUE becomes DBSCAN for
 - $K(\rho(x, x')) = \mathbb{I}[\rho(x, x') \leq \varepsilon]$
 - $\tau = k/V_D(\varepsilon)$, where $V_D(\varepsilon)$ -volume of sphere with radius ε in D -dimensional space.
- Complexity $O(N^2I)$, I -number of iterations in gradient ascent.
 - for N points I times need to calculate $\rho(x)$
 - $\rho(x)$ can be calculated faster by looking only at neighborhood points, found with spatial index
 - using: ball trees, KD-trees, mapping: bin on the axis->objects in that bin.
- In contrast to DBSCAN, density peak clustering can find clusters with different object density but is more computationally expensive.

Table of Contents

- 1 Clustering introduction
- 2 Representative-based clustering
- 3 Hierarchical clustering
- 4 Probabilistic clustering
- 5 Density based approaches
- 6 Spectral clustering**

Spectral clustering - example



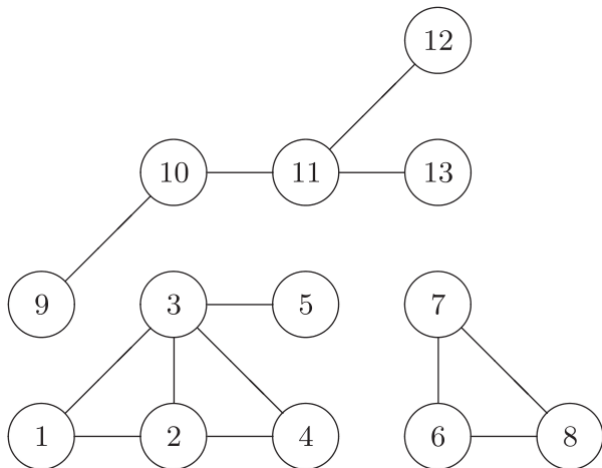
Description

- Spectral clustering relies upon similarity matrix W between objects.
- Similarity matrix \leftrightarrow weighted connection graph
- Examples:
 - nodes represent people, edge weights - how much they communicate
 - nodes represent web-pages, edge weights - scalar products of $TF - IDF$

Similarity matrix calculation

- $\|x_i - x_j\| < \textit{threshold}$
- RBF
- based on nearest neighbourhood property

Graph with disjoint components



Graph Laplacian

- $W = W^T$, $w_{ij} \geq 0$ - the similarity between object i and object j .
- Define $D = \text{diag}\{d_1, \dots, d_N\}$, where $d_i = \sum_{j=1}^N w_{ij}$ -weighted degree of node i .
- Define graph Laplacian

$$L = D - W$$

- Properties of graph Laplacian:
 - it is symmetric
 - *It has eigenvector $\mathbf{1} \in \mathbb{R}^N$ consisting of ones with eigenvalue 0. Why?*
 - it is positive semi-definite: $\forall f \in \mathbb{R}^N : f^T L f \geq 0$.
 - L has eigenvalues $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_N = 0$

Positive semi-definiteness of Laplacian

Consider arbitrary $f \in \mathbb{R}^N$:

$$\begin{aligned} f^T Lf &= f^T Df - f^T Wf = \sum_i d_i f_i^2 - \sum_{i,j} f_i f_j w_{ij} = \\ &= \frac{1}{2} \left(\sum_i d_i f_i^2 - 2 \sum_{i,j} w_{ij} f_i f_j + \sum_j d_j f_j^2 \right) = \\ &= \frac{1}{2} \left(\sum_{i,j} w_{ij} f_i^2 - 2 \sum_{i,j} w_{ij} f_i f_j + \sum_{j,i} w_{ji} f_j^2 \right) = \quad (3) \\ &= \frac{1}{2} \left(\sum_{i,j} w_{ij} f_i^2 - 2 \sum_{i,j} w_{ij} f_i f_j + \sum_{i,j} w_{ij} f_j^2 \right) = \\ &= \frac{1}{2} \sum_{i,j} w_{ij} (f_i - f_j)^2 \geq 0 \end{aligned}$$

Eigenvectors of Laplacian

- Consider eigenvector f corresponding to eigenvalue $\lambda = 0$.
 - $f^T Lf = \lambda f^T f = 0$
- Using (3) we have that

$$0 = f^T Lf = \frac{1}{2} \sum_{i,j} w_{i,j} (f_i - f_j)^2 \quad (4)$$

- If objects i and j are connected on the graph, there exists a path with $w_{uv} > 0$ along the path and from (4) it should be that $f_i = f_j$.
- So the set of eigenvectors of L is spanned by indicator vectors $I_{A_1}, I_{A_2}, \dots, I_{A_K}$ where A_i is i -th isolated region on the graph.
- Order of $\lambda = 0$ gives the number of isolated components.

Spectral clustering algorithm

- 1 Find order K of singular value $\lambda = 0$ for L
- 2 Find set of eigenvectors v_1, \dots, v_K corresponding to $\lambda = 0$
- 3 Cluster *rows* of $V = [v_1, \dots, v_K] \in \mathbb{R}^{N \times K}$ with K -means.

RETURN clustering for rows as clustering for initial objects x_1, \dots, x_N .

Practical application

- $L' = D^{-1}L$ is considered instead of L (“normalized” Laplacian)
 - to account for different connectivity levels of different nodes
- Most often singular values of L' are not exactly zero, but close to zero. So we select K almost-zero eigenvalues and corresponding K eigenvectors.

Example

Histogram of the sample

