Data analysis and stochastic modeling

Romaric GAUDEL[1]

[1] Univ. Rennes, IRISA & Inria

Fall 2027

Part III

Clustering

Based on slides from

- Guillaume Gravier [Data analysis and probabilistic modeling]
- Jure Leskovec [Mining Massive Datasets]

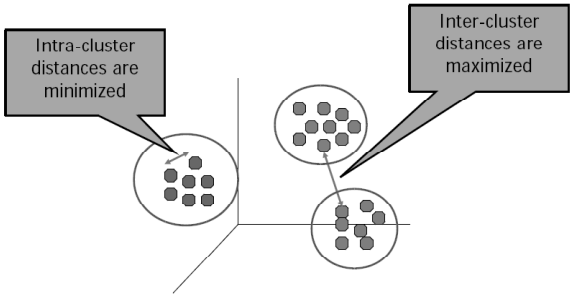
What are we gonna talk about today?

- What's cluster analysis?
- Partitioning clustering
  → k-means and the likes
- Hierarchical clustering
  → bottom-up clustering, linkage methods
- A quick survey of other methods
  → density methods, spectral clustering, etc.
- Case study

In short: an overview of the art of grouping data according to their similarity.

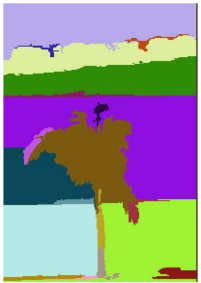
Objective

Clustering consists in grouping data together in “classes” where objects in a class are similar and objects from different classes are different.



Usage

- **(exploratory) data analysis and understanding** (data mining)
  - Biology: taxonomy of living things
  - Information retrieval: grouping similar documents
  - Land use: identifying areas with similar properties
  - Marketing: discover distinct groups of customers
  - City planning, earth quake analysis, etc.

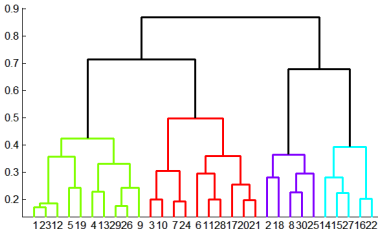


- **pre-processing tool**
  - quantization, coding and compression
  - classification, segmentation, etc.
  - k nearest neighbor search

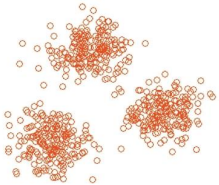
Two main strategies

Two group of methods:

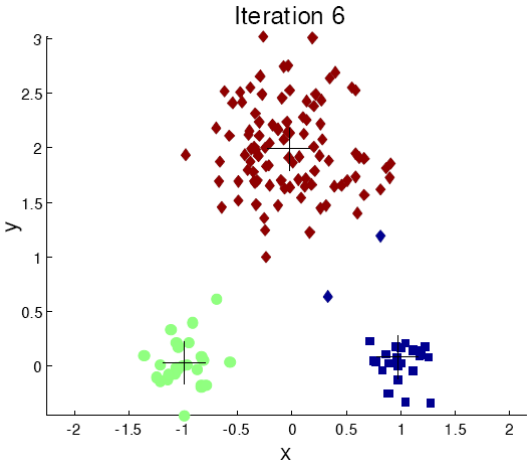
- **Hierarchical:**
  - **Agglomerative** (bottom up):
    - Initially, each point is a cluster
    - Repeatedly combine the two "nearest" clusters into one
  - **Divisive** (top down):
    - Start with one cluster and recursively split it



- **Point assignment:**
  - Maintain a set of clusters
  - Points belong to the "nearest" cluster



First example: *k*-means



Outline

1. k-means
2. Discussion on Clustering
3. Hierarchical clustering
4. Density-Based Clustering
5. Spectral clustering
6. Conclusion

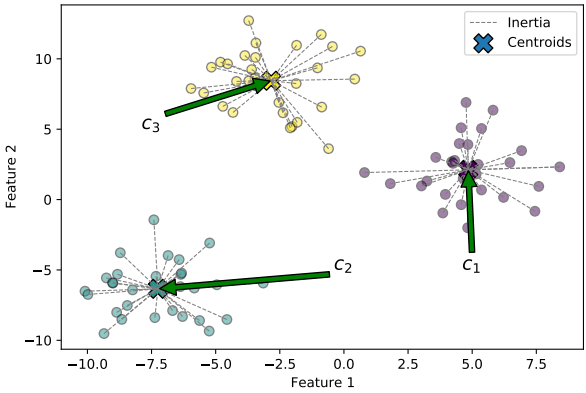
Model

Data

Examples $\{x_1, \ldots, x_N\}$, with $x_i \in^d$

Clusters

Centers $\{c_1, \ldots, c_K\}$, with $c_i \in^d$
Assignation function
$f : [1, N] \to [1, K]$, with $f(i)$ the index of the cluster / center assigned to example i
Cluster $S_k = \{x_i : f(i) = k\}$
Inertia $\sum_k \sum_{x \in S_k} \|x - c_k\|^2$



k-means objective

- It is a problem formulation, not an algorithm.
- **Problem**: Given Euclidean space/distance and ***k* =** number of clusters, find cluster centers that minimizes sum of squared distances from each point to its cluster center
- Finding an exact solution is NP-hard.
- The approximate solution is Lloyd's algorithm or the k-means algorithm.

k-means algorithm

Idea: Divide some data x_i into K clusters represented by the mean value of their members c_k (centroids), so as to minimize the overall quantization error

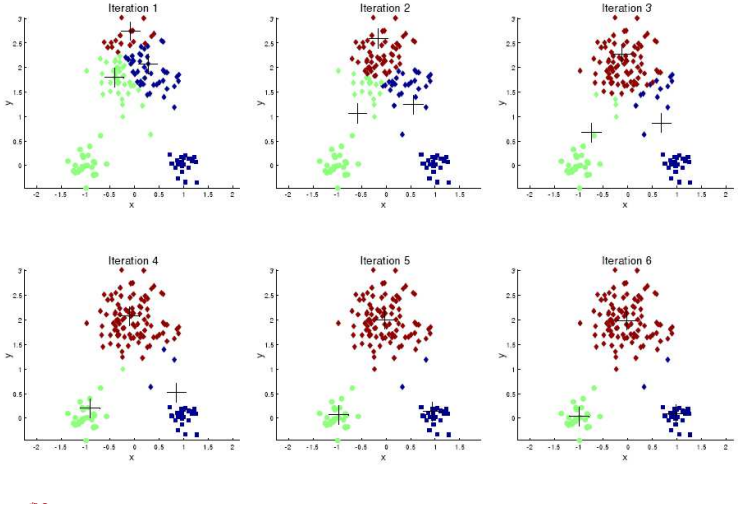
$$e = \sum_i d(x_i, c_{f(i)})$$

Algorithm:

initialize K centroids c_k
while not converged **do**
  for $i = 1 \rightarrow N$ **do**
    assign x_i to the closest centroid ($f(i) \leftarrow \arg\min_k d(x_i, c_k)$)
  end for
  for $i = 1 \rightarrow K$ **do**
    update centroid c_k from all assigned points
  end for
end while

[J. B. McQueen. Some methods for classification and analysis of multivariate observations. Proc. Symposium on Math., Statistics, and Probability, pp. 281-297, 1967]

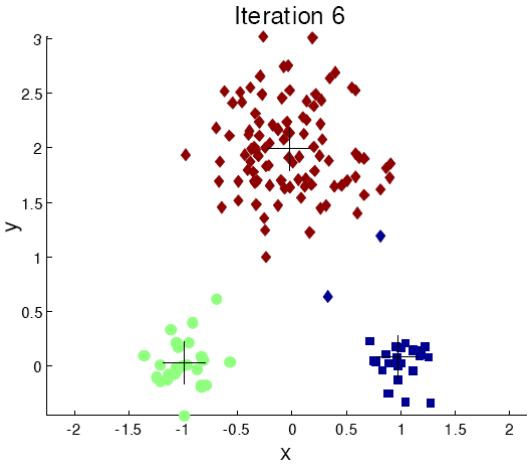
An execution



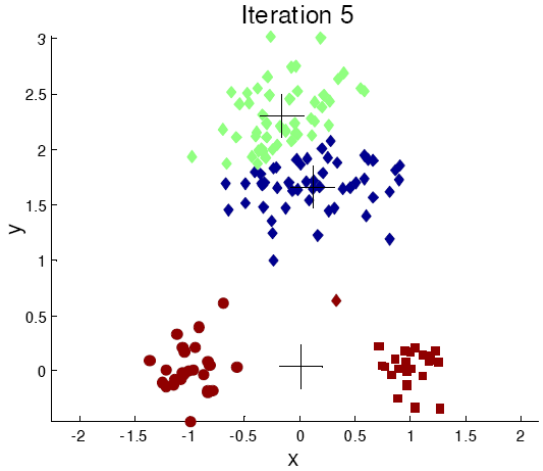
Remarks

- the distance is the key
  - mean has to have a meaning ...
  - ... but we can use the median instead (the *k-medians* or *k*-medioids algorithm)
- convergence = nothing moves anymore!
  - convergence is guaranteed
  - often in 10 to 20 iterations
- complexity = O(iKNd)
  - i = #iterations, d=dimension(x_i)
- initialization is tricky!
  - random choice
  - multiple runs
  - hierarchical k-means (LBG)

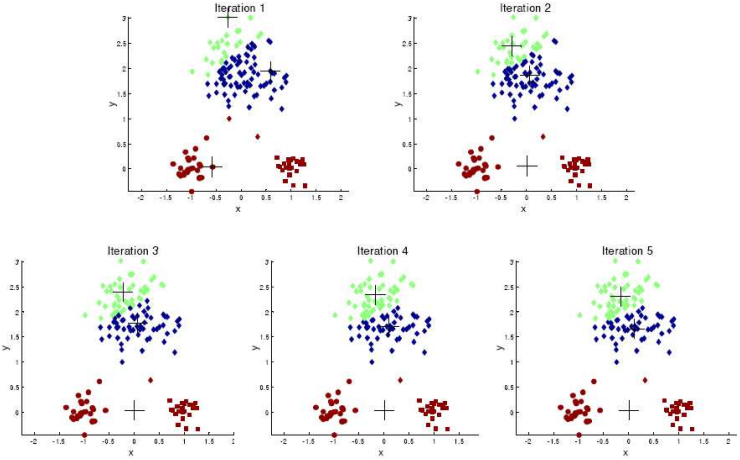
With good/lucky initialization



With bad/unlucky initialization



Execution with bad/unlucky initialization



Final remarks on initialization

- **multiple runs**
  - but that's costly!
  - have to deal with "*dead*" clusters (e.g., replace them)
- **hierarchical k-means**
  - the original Linde-Buzo-Gray algorithm (aka bisecting k-means)

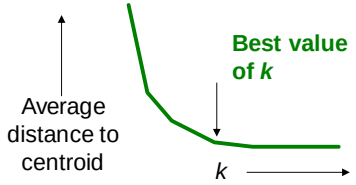
initialize centroids $c_1(1)$ to gravity center
$i \leftarrow 1$
while not enough clusters ($i < p$) **do**
  split each centroids $c_i(j)$ (along maximum variance line)
  run k-means
end while

  - and its many variants
    - split only the biggest cluster $\rightarrow$ arbitrary number of clusters instead of 2^p
    - points stay within their parent cluster $\rightarrow$ much faster

In any case, local optima!

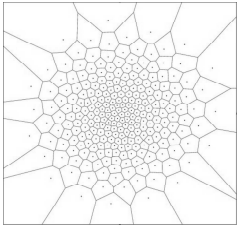
How to select *k*?

- Try different **k**, looking at the change in the average distance to centroid as **k** increases
- Average falls rapidly until right **k**, then changes little

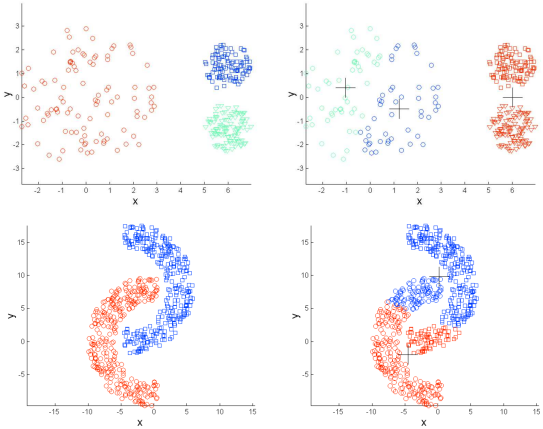


Pros & Cons

- **Pros**
  - simple, clear and popular
  - decently efficient
  - guaranteed convergence
  - can accomodate any shape (with enough clusters)
- **Cons**
  - initialization and local optima
  - need to define the number of clusters
  - convex clusters of roughly the same size and density
  - tends to create unbalanced cells
  - highly sensitive to noise and outliers



Pros & Cons illustrated



Outline

1. k-means
2. Discussion on Clustering
3. Hierarchical clustering
4. Density-Based Clustering
5. Spectral clustering
6. Conclusion

Main objective for clusters

Good clustering

= low within-class similarity = high across-class similarity (enventually)

An ambiguous notion



How many clusters?

An ambiguous notion



How many clusters?



Six Clusters



Two Clusters

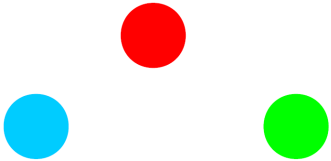


Four Clusters

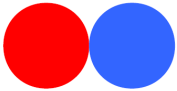
Other cluster objectives

- **well separated clusters**
  - every point in the cluster is closer to every point in the cluster than to any point outside the cluster
- **center clusters**
  - every point in the cluster is closer to the center of the cluster than to the center of any other cluster
- **contiguous cluster**
  - every point in the cluster is closer to at least one point in the cluster than to any point in another cluster
- **density-based cluster**
  - dense region of points in a cluster separated from other clusters by low-density regions

Other cluster objectives (illustrated)



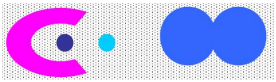
well separated



center



contiguous

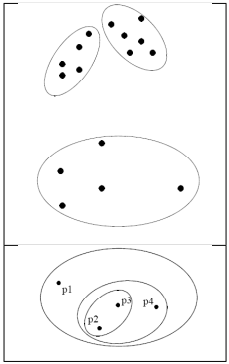


dense

Multiple clustering types

Clustering $\Rightarrow$ a set of clusters

- partitioning data: divide the entire data set
  - non-overlapping clusters
  - each object belongs to one a only one cluster
- aggregating data: group similar data
  - nested clusters
  - hierarchical structure



But other philosophies do exist: density-based, grid-based, model-based, constraints-based, etc.

Multiple clustering types (cont'd)

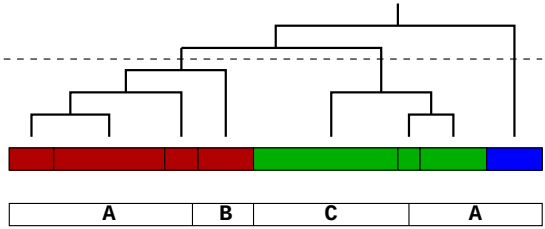
- **exclusive or not?**
  - points might belong to several clusters (with or without weights)
- **fuzzy or not?**
  - in fuzzy algorithms, a point belong to all clusters with a weight $\in [0, 1]$
- **partial or not?**
  - only part of the data is clustered
- **homogeneous or not?**
  - clusters of very different shape

Even more requirements

- Dynamic behavior
- Noise and outliers
- Insensitive to the order of input
- ...
- Scalability
- ...

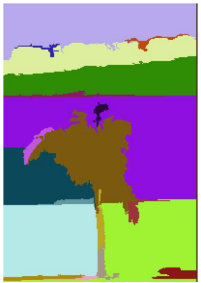
Quality criterion

- subjective evaluation by inspecting clusters
- within cluster distortion (if a meaningful metric exists)
- any oritrary objective quality criterion (but none out of the shelf)
- special case with temporal segmentations



Quality criterion given the application

- **(exploratory) data analysis and understanding** (data mining)
  - Biology: taxonomy of living things
  - Information retrieval: grouping similar documents
  - Land use: identifying areas with similar properties
  - Marketing: discover distinct groups of customers
  - City planning, earth quake analysis, etc.



- **pre-processing tool**
  - quantization, coding and compression
  - classification, segmentation, etc.
  - k nearest neighbor search

A key ingredient: pairwise distances

similar objects $\Rightarrow$ notion of distance

- Any measure of the similarity $d(x_i, x_j)$ between two objects can be used to solve the problem
- The adequate measure highly depends on the nature of the data and on the nature of the problem
  - distance measures
    - Euclidian, Manhattan, Mahalanobis, χ^2, etc.
  - similarity (no triangular inequality)
    - cosine, template matching, edit distance, generalized likelihood, etc.
  - conceptual measures
    - whatever one can think of...

High dimension space

- many applications require high-dimension spaces: text documents, images, DNA data, etc.
- high dimensions raises new challenges
  - many irrelevant dimensions might mask clusters
  - distance measure becomes meaningless (curse of dimensionality)
  - cluster may exist only in some subspaces
- several workarounds
  - feature transformation
    - PCA/SVD if features are correlated/redundant
  - feature selection
    - select feature where nice clusters appear
  - subspace clustering
    - find clusters in all possible subspaces (CLIQUE, Proclus)

Scalability (Large Datasets)

- Many algorithms in $\mathcal{O}\left(n^2\right)$ or $\mathcal{O}\left(n^3\right)$ (too high computation and/or memory)

Scalability (Large Datasets)

- Many algorithms in $\mathcal{O}\left(n^2\right)$ or $\mathcal{O}\left(n^3\right)$ (too high computation and/or memory)
- $\mathcal{O}\left(n\right)$ may already be too high to fit in the memory

To recap

To select the most appropriate solution one must look at the following elements

- Type of proximity or density measure
- Sparseness
  - Dictates type of similarity; Adds to efficiency
- Attribute type
  - Dictates type of similarity
- Type of Data
- Dimensionality
- Noise and Outliers
- Type of Distribution

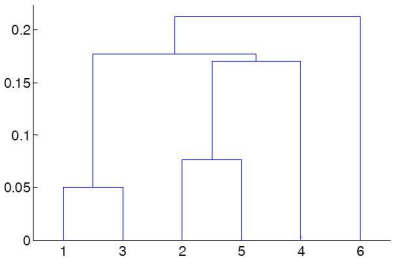
Outline

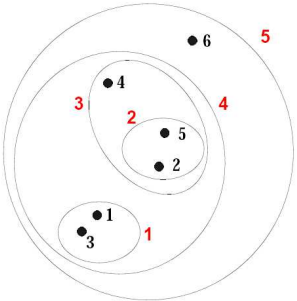
1. k-means
2. Discussion on Clustering
3. Hierarchical clustering
4. Density-Based Clustering
5. Spectral clustering
6. Conclusion

Hierarchical clustering

Idea: Progressively generate clusters by merging or divising data

- generate nested clusters
- can be visualized as a dendogram





Agglomerative vs. divisive

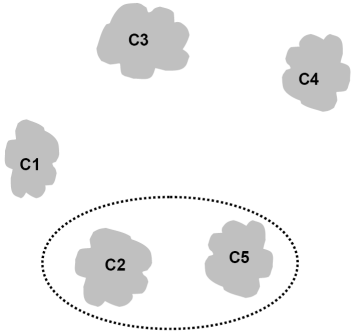
- **Agglomerative bottom-up clustering**
  - Bottom-up construction of the dendogram by progressively merging clusters

initialize N singleton clusters
nclusters $\leftarrow N$
while nclusters > 0 **do**
  merge the two closest clusters
  nclusters $\leftarrow$ nclusters $- 1$
end while

- **Divisive bottom-up clustering**
  - top-down construction of the dendogram
  - DIANA (Divisive ANAlysis)

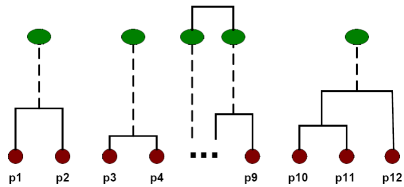
Proximity matrix and bottom-up clustering

update the proximity matrix.

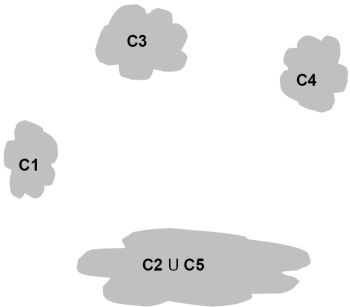


	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

Proximity Matrix

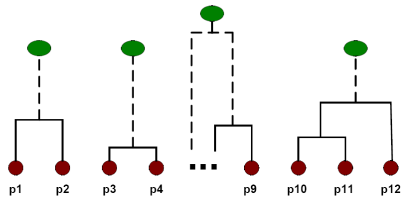


Proximity matrix and bottom-up clustering (cont'd)



	C1	C2 U C5	C3	C4
C1		?		
C2 U C5	?	?	?	?
C3		?		
C4		?		

Proximity Matrix

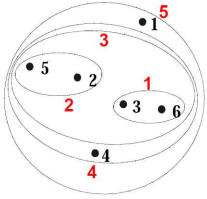


“Linkage” types

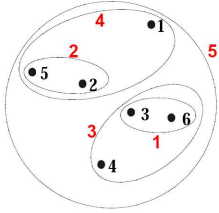
linkage = how to measure the distance between clusters

- single linkage: $D(A, B) = \min(d(x, y) \quad \forall (x, y) \in A \times B)$
  - $\rightarrow$ for well-separated classes only
- total linkage: $D(A, B) = \max(d(x, y) \quad \forall (x, y) \in A \times B)$
  - $\rightarrow$ favors large clusters
- average linkage = average distance between elements in A and B
  - $\rightarrow$ robust to noise and outliers but biased towards globular clusters
- Ward’s linkage = increase in variance for the cluster being merged
- and many others, including distance between mean/median or between (statistical) models of the data

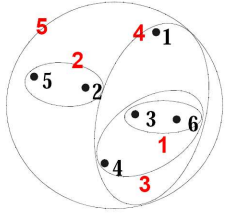
“Linkage” types (cont’d)



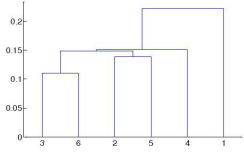
Sing Link (MIN)

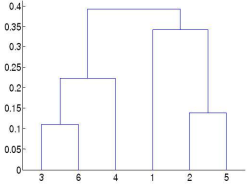


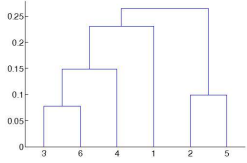
Complete Link (MAX)



Group Average





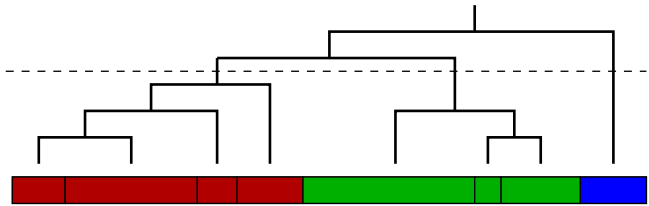


Pros and Cons

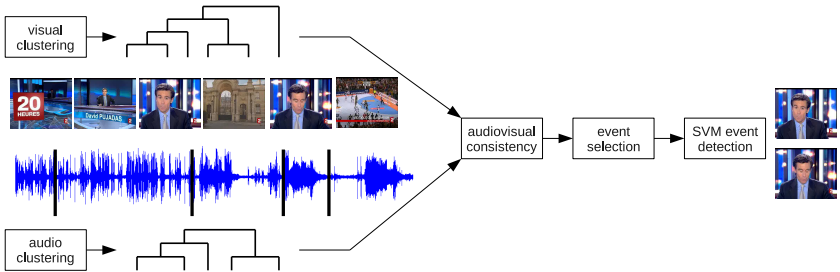
- **Pros**
  - better than k-means with non-metric distances
  - can combine metrics and cluster balancing criteria
  - possibility to define a posteriori the number of clusters (though not so easy in practice)
- **Cons**
  - quite slow and computationally demanding (O(N^3) or O($N^2 \log(N)$))
  - local optima may not be globally good
    - cannot undo what was done previously
    - require relocation methods
- cutting the dendogram is not as easy as it seems to be

Bottom-up clustering for temporal partitioning

- detect boundaries of segments $\longrightarrow$ see hypothesis testing
- group together segments with similar characteristics
  - model based representation of clusters (Gaussian densities and mixtures)
  - Kullback-Leibler divergence, generalized likelihood ratio
- find out where to cut the dendogram
  - model selection approaches: Bayesian information criterion



Mining TV sequences with bottom-up clustering

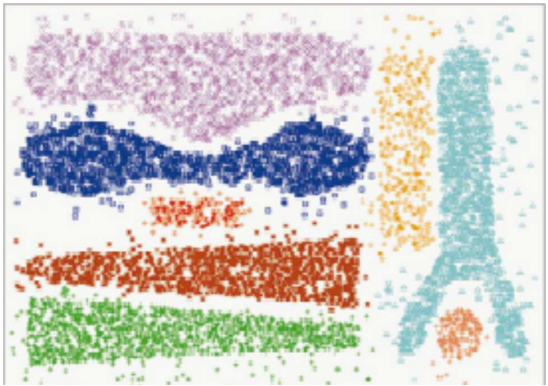


Outline

1. k-means
2. Discussion on Clustering
3. Hierarchical clustering
4. Density-Based Clustering
5. Spectral clustering
6. Conclusion

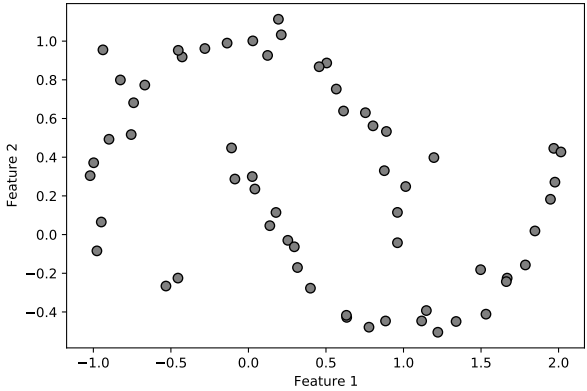
Density-based clustering in a nutshell

- Track density-connected points = neighborhood analysis
- arbitrary shaped clusters, robustness to noise, one pass over the data
- Typical algorithms: DBSCAN, OPTICS, DENCLUE



DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

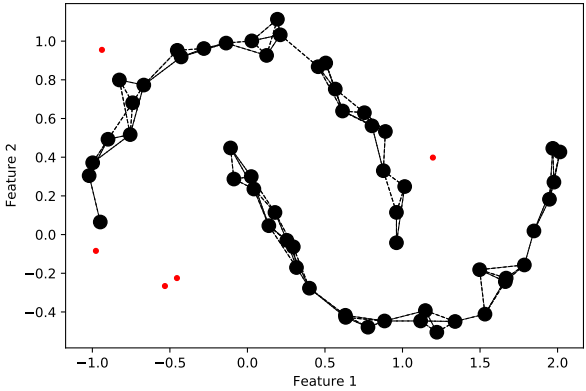
Algorithm (first step)



DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

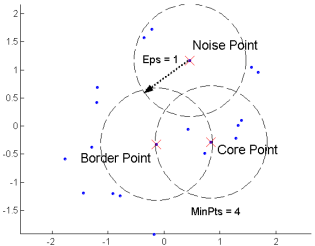
Algorithm (first step)

1. label points & build the graph



Two parameters

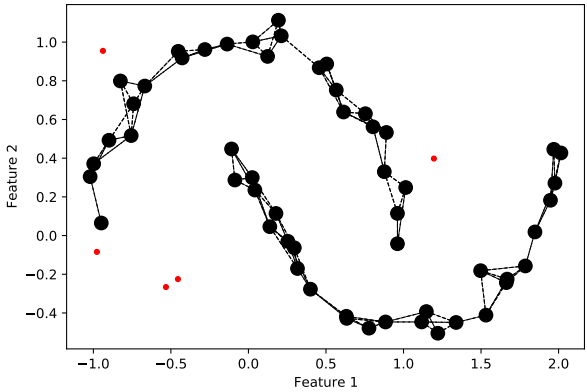
1. label points
   - core points: enough points around
   - border points: not enough points around but close to a core point
   - noise points: none of the above
2. eliminate noise points
3. create clusters from core points
4. assign border points to core clusters



DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Algorithm

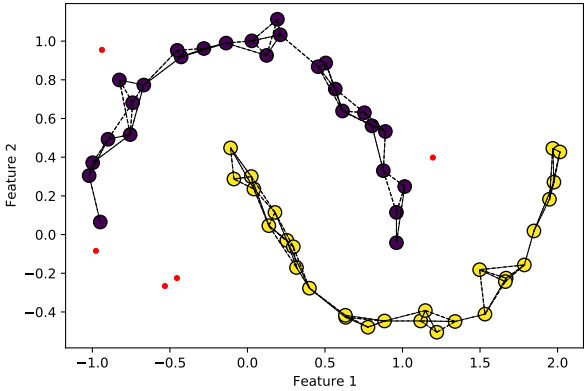
1. label points & build the graph



DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Algorithm

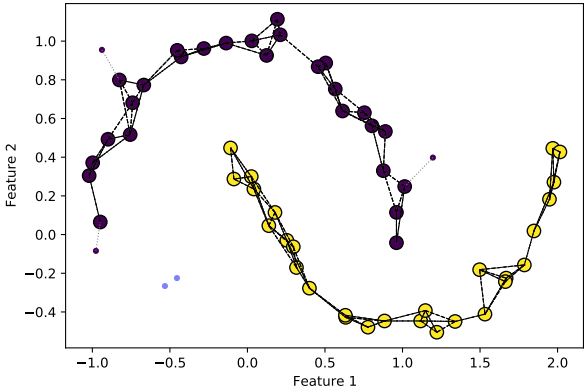
1. label points & build the graph
2. search for connected components



DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Algorithm

1. label points & build the graph
2. search for connected components
3. label border points



DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

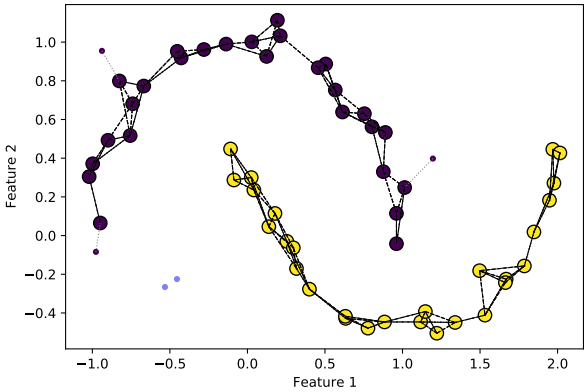
Algorithm

1. label points & build the graph
2. search for connected components
3. label border points

Search for connected components

While it remains unlabeled points

1. pick an unlabeled point & choose a new label
2. identify its component thanks to DFS or BFS



Pros & Cons

- Pros
  - **No assumption of convex clusters:** DBSCAN is capable of discovering clusters of various shapes and sizes.
  - **Handles noise:** It identifies outliers as noise points, making it robust to noise in the data.
  - **No need to specify the number of clusters:** DBSCAN automatically determines the number of clusters based on the data.
- Cons
  - **Sensitive to parameter selection:** The performance of DBSCAN is influenced by the choice of parameters like epsilon (ε) and minimum points.
    - ⋆ **Challenges with varying cluster densities:** Choosing appropriate parameters becomes challenging when clusters have significantly different densities.
  - **Requires a sensible distance function.**

Outline

1. k-means
2. Discussion on Clustering
3. Hierarchical clustering
4. Density-Based Clustering
5. Spectral clustering
6. Conclusion

DBSCAN facing noisy graphs

DBSCAN fails due to **greedy**/**strict** connected components search

Replacement: Graph Laplacian

The unnormalized graph Laplacian is the $n \times n$ matrix defined as

$$\mathbf{L} = \mathbf{D} - \mathbf{W}$$

with $\mathbf{D}$ a diagonal matrix with elements $d_i = \sum_j w_{ij}$.

Important properties of $\mathbf{L}$:

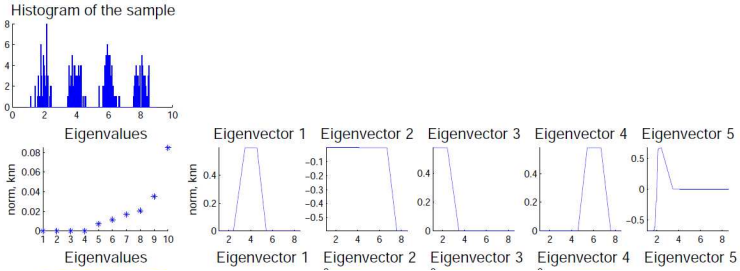
- $\mathbf{L}$ as n non-negative, real valued eigenvalues
- the smallest eigenvalue is 0 and corresponds to the unit eigen vector $\mathbb{1}$
- the multiplicity of the eigenvalue 0 is the number of connected components

$\Rightarrow$ **exploit the eigenvalues of the Laplacian of the similarity graph to perform clustering**



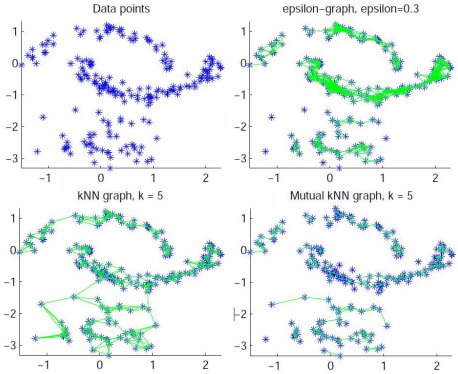
Normalized versions of the Laplacian are often used in practice.

Toy Example


Histogram of the sample
Eigenvalues
norm, knn
Eigenvalues
Eigenvector 1
Eigenvector 2
Eigenvector 3
Eigenvector 4
Eigenvector 5
norm, knn
Eigenvector 1
Eigenvector 2
Eigenvector 3
Eigenvector 4
Eigenvector 5

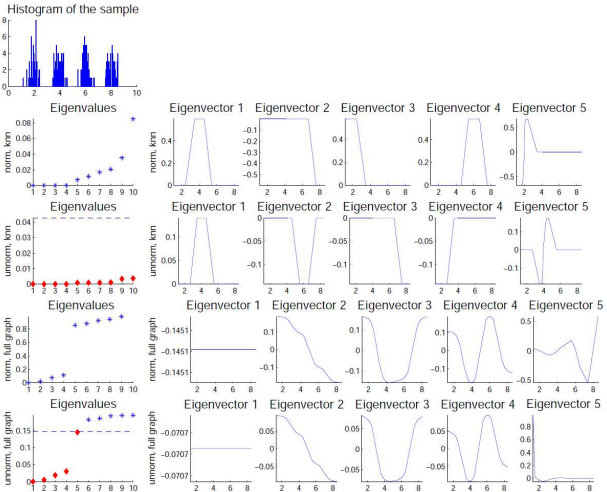
Similarity Graphs

- data point = node in a graph
- edges encode the similarity between two nodes with weights w_{ij}
  - ϵ-neighbor graphs
  - k-nearest neighbor graphs
  - fully connected graphs



[Ulrike von Luxburg. A tutorial on spectral clustering. Statistics and Computing, 17(4):395-416, 2007]

Toy example (bis)



Spectral clustering algorithm

Input: similarity matrix $\mathbf{S} \in \mathbb{R}^{n \times n}$, number of clusters k

Algorithm:

construct a similarity graph from $\mathbf{S}$ with adjacency matrix $\mathbf{W}$
compute the Laplacian $\mathbf{L}$
compute the k eigenvectors $\mathbf{u}_1, \ldots, \mathbf{u}_k$ of $\mathbf{L}$ associated with the k lowest eigenvalues
define $\mathbf{U} \in \mathbb{R}^{n \times k}$ with $\mathbf{u}_1, \ldots, \mathbf{u}_k$ as columns
define $\mathbf{y}_i \in \mathbb{R}^k$ the i-th row of $\mathbf{U}$ for $i \in [1, n]$
run k-means clustering (or other) on the vector $\mathbf{y}_i$

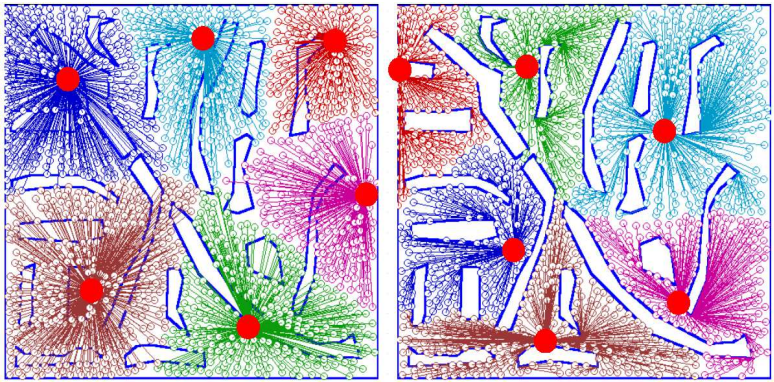
Strong links with

- graph cut algorithms (NCut, RatioCut, MinMaxCut)
- random walks theory (Markov clustering)

Pros & Cons (w.r.t DBSCAN)

- Advantages over DBSCAN
  - **Allows affinity matrix customization:** Users can customize affinity matrices based on domain knowledge.
    - ⋆ **Less sensible to locality parameters:** Keep more connections and let k-means discard less important ones.
  - **Handles varying cluster densities more robustly**
- Disadvantages compared to DBSCAN
  - **Sensitivity to parameter tuning.** (including affinity matrix choice)
  - **Higher computational complexity.**
  - **Does not handle noise and outliers.**
  - **Requires to specify the number of clusters.**

To go beyond: clustering with obstacles



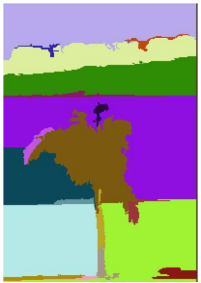
Outline

1. k-means
2. Discussion on Clustering
3. Hierarchical clustering
4. Density-Based Clustering
5. Spectral clustering
6. Conclusion

Many usages, with corresponding objectives

- **(exploratory) data analysis and understanding** (data mining)
  - Biology: taxonomy of living things
  - Information retrieval: grouping similar documents
  - Land use: identifying areas with similar properties
  - Marketing: discover distinct groups of customers
  - City planning, earth quake analysis, etc.

- **pre-processing tool**
  - quantization, coding and compression
  - classification, segmentation, etc.
  - k nearest neighbor search



Multiple algorithms for various behaviors

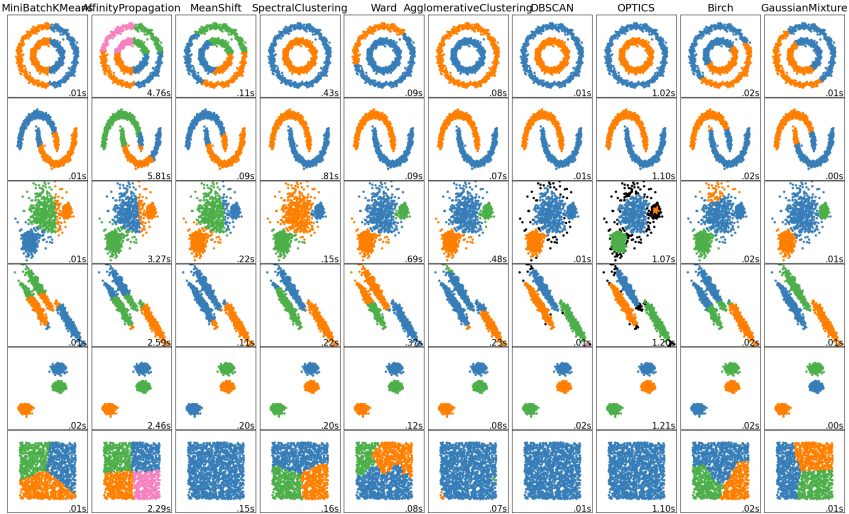
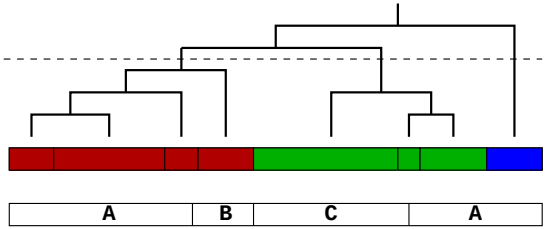


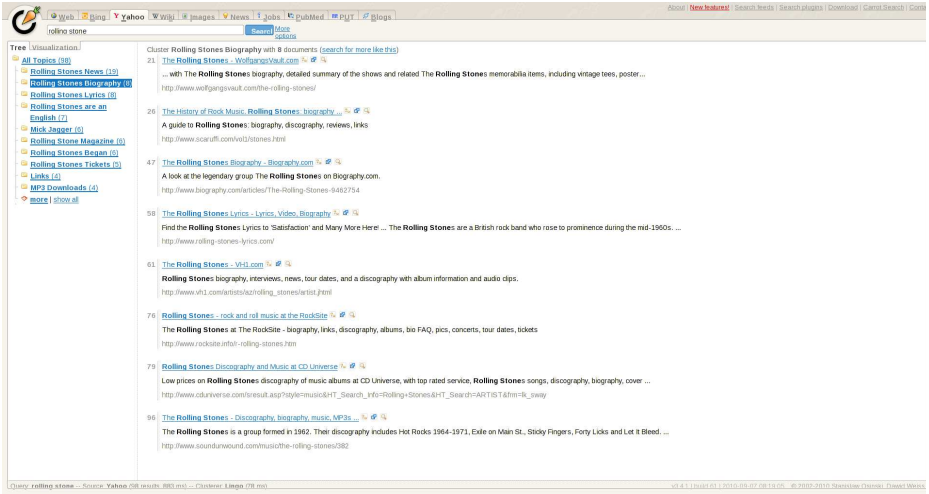
Illustration: sklearn documentation

Hard to evaluate the quality of a clustering

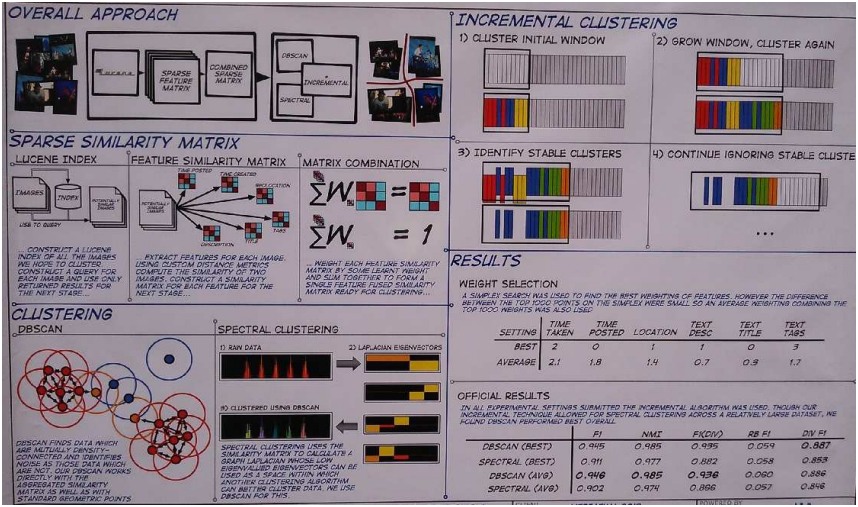
- subjective evaluation by inspecting clusters
- within cluster distortion (if a meaningful metric exists)
- any oritrary objective quality criterion (but none out of the shelf)
- special case with temporal segmentations



A fun (and practical) use of text clustering



Another example mixing all you've seen



MediaEval 2013 Social Event Detection Task

Further readings

- So many textbooks (look for clustering and data mining)!
- So many resources on the Web (not in Wikipedia this time)
  - free code available everywhere
  - lot's of tutorials/lessons
    - Dr. HOI Chu's course:

      https://svn.mosuma.net/r4000/doc/course/ci6227/public/lectures/lecture07cluster.pdf
    - Prof. Jiawei Han courses: http://www.cs.uiuc.edu/ hanj