Course. Introduction to Machine Learning

Work 1. Clustering Exercise

Session 2

Course 2023-2024

Dr. Maria Salamó Llorente

Dept. Mathematics and Informatics,

Faculty of Mathematics and Informatics,
University of Barcelona

Contents

1. Introduction (session 1)
2. Preprocess the data (session 1)
3. DBSCAN with sklearn (session 2)
4. BIRCH with sklearn (session 2)
5. K-Means + K-Modes (your own code) (session 2)
6. K-Medoids or K-Prototypes (your own code) (session 2)
7. Fuzzy clustering (your own code) (session 3)
8. Validation techniques (using sklearn validation metrics) (session 3)

Taxonomy of Clustering Algorithms

- *Clustering Techniques*
  - *Partitioning methods*
    - ***k-Means algorithm*** *[1957, 1967]*
    - ***k-Medoids algorithm***
      - *PAM [1990]*
      - *CLARA [1990]*
      - *CLARANS [1994]*
    - ***k-Modes*** *[1998]*
    - *Fuzzy c-means algorithm [1999]*
  - *Hierarchical methods*
    - *Divisive*
      - *DIANA [1990]*
    - *Agglomerative methods*
      - *AGNES [1990]*
      - ***BIRCH [1996]***
      - *CURE [1998]*
      - *ROCK [1999]*
      - *Chamelon [1999]*
  - *Density-based methods*
    - *STING [1997]*
    - ***DBSCAN [1996]***
    - *CLIQUE [1998]*
    - *DENCLUE [1998]*
    - *OPTICS [1999]*
    - *Wave Cluster [1998]*
  - *Graph based methods*
    - *MST Clustering [1999]*
    - *OPOSSUM [2000]*
    - *SNN Similarity Clustering [2001, 2003]*
  - *Model based clustering*
    - *EM Algorithm [1977]*
    - *Auto class [1996]*
    - *COBWEB [1987]*
    - *ANN Clustering [1982, 1989]*

DBSCAN
Density-Based Clustering

Using sklearn

Some Links

- http://www2.cs.uh.edu/~ceick/7363/Papers/dbscan.pdf
- https://youtu.be/sKRUfsc8zp4
- https://youtu.be/6jl9KkmgDIw
- https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html#sklearn.cluster.DBSCAN

Density-Based Clustering Methods

- Clustering based on density (local cluster criterion), such as **density-connected points** or based on an explicitly constructed density function
- Major features:
  - Discover clusters of arbitrary shape
  - Handle noise
  - One scan
  - Need density parameters
- Several interesting studies:
  - DBSCAN: Ester, et al. (KDD'96)
  - DENCLUE: Hinneburg & D. Keim (KDD'98/2006)
  - OPTICS: Ankerst, et al (SIGMOD'99).
  - CLIQUE: Agrawal, et al. (SIGMOD'98)

DBSCAN: Density-Based Clustering

- DBSCAN is a Density-Based Clustering algorithm
- Reminder: In density based clustering we partition points into dense regions separated by not-so-dense regions.
- Important Questions:
  - How do we measure density?
  - What is a dense region?
- DBSCAN:
  - Density at point p: number of points within a circle of radius Eps
  - Dense Region: A circle of radius Eps that contains at least MinPts points

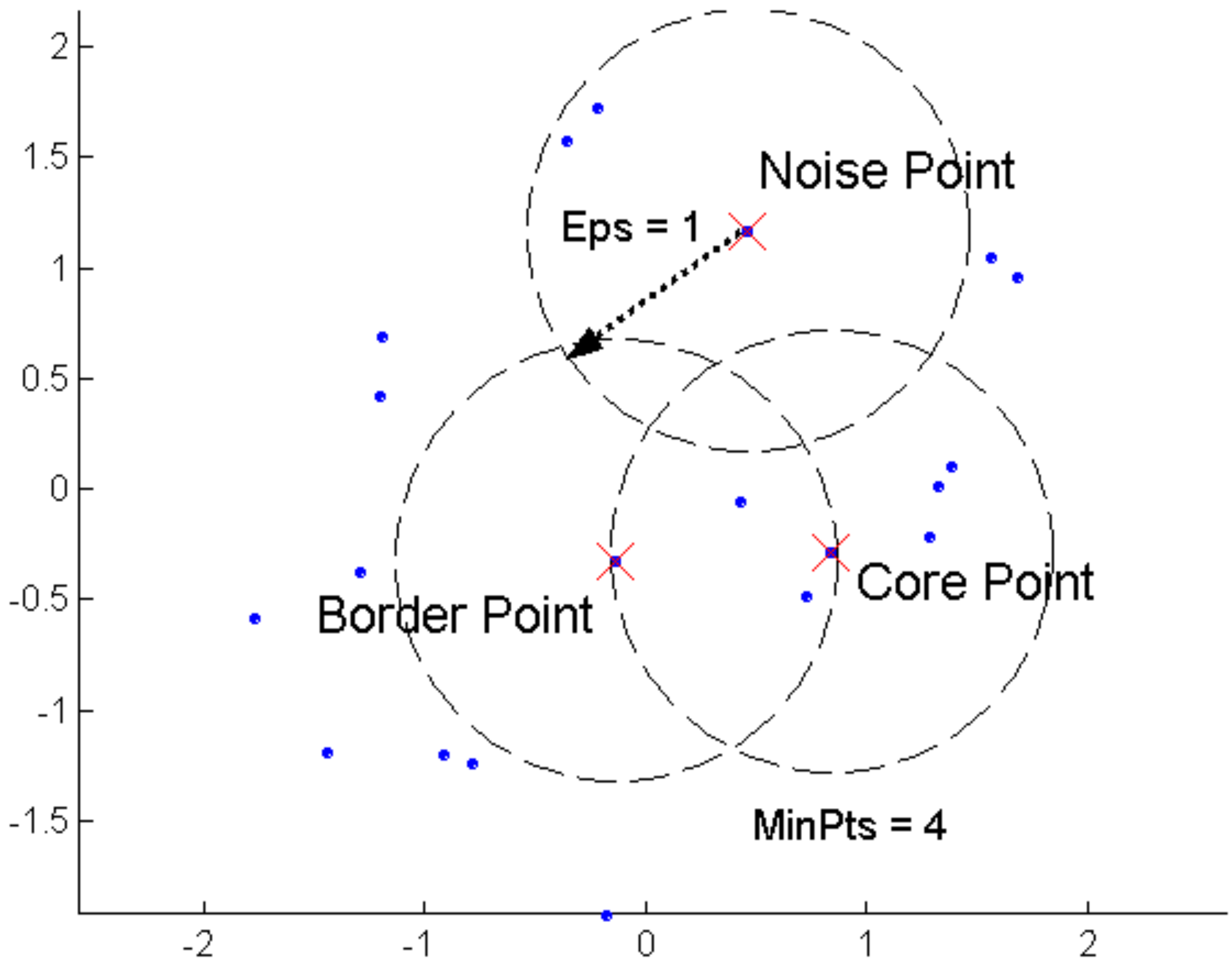
DBSCAN: Preliminary concepts

Characterization of points

- Density = number of points within a specified radius (Eps)
- A point is a **core point** if it has more than a specified number of points (*MinPts*) within Eps
  - These points belong in a dense region and are at the interior of a cluster
- A **border point** has fewer than *MinPts* within Eps, but is in the neighborhood of a core point
- A **noise point** is any point that is not a core point or a border point

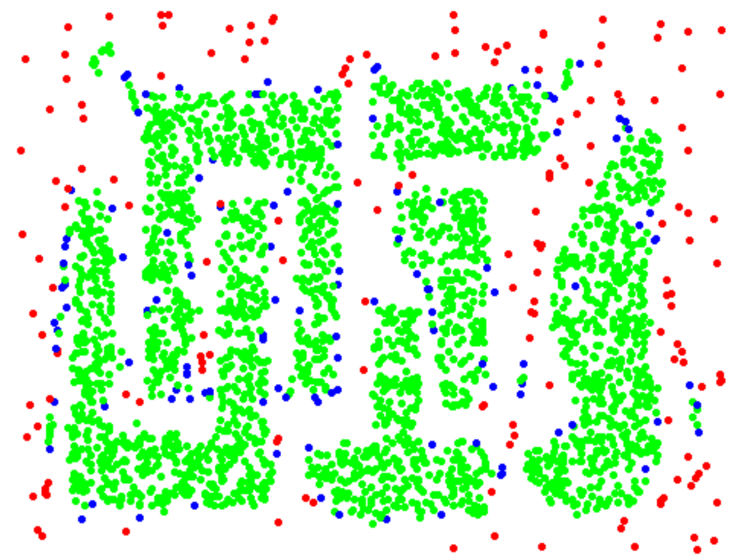
UNIVERSITAT DE BARCELONA

DBSCAN: Core, Border, and Noise Points



DBSCAN: Core, Border and Noise Points

Original Points



Point types: core, border and noise

Eps = 10, MinPts = 4

Parameter Estimation

- Parameters must be specified by the user
  - Ɛ = physical distance (radius),
  - *minPts* = desired minimum cluster size

minPts

- derived from the number of dimensions *D* in the data set, as *minPts* ≥ *D* + *1*
- *minPts* = *1* does not make sense, as then every point on its own will already be a cluster
- *minPts* must be chosen at least 3. Larger is better.
- larger the dataset, the larger the value of *minPts* should be chosen

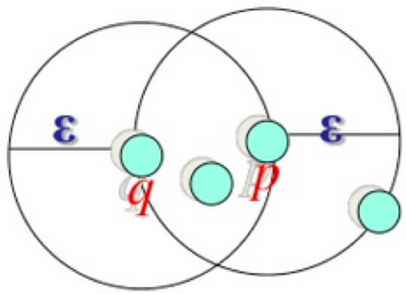
Ɛ

- value can be chosen by using a k-distance graph
- If Ɛ is chosen much too small, a large part of the data will not be clustered
- If too high value, majority of objects will be in the same cluster
- In general, small values of Ɛ are preferable

Concepts: Ɛ-Neighborhood

- Ɛ-Neighborhood: Objects within a radius of Ɛ from an object (epsilon-neighborhood)

- Core objects: Ɛ-Neighborhood of an object contains at least *MinPts* of objects



ε-Neighborhood of *p*
ε-Neighborhood of *q*
p is a core object (MinPts = 4)
q is not a core object

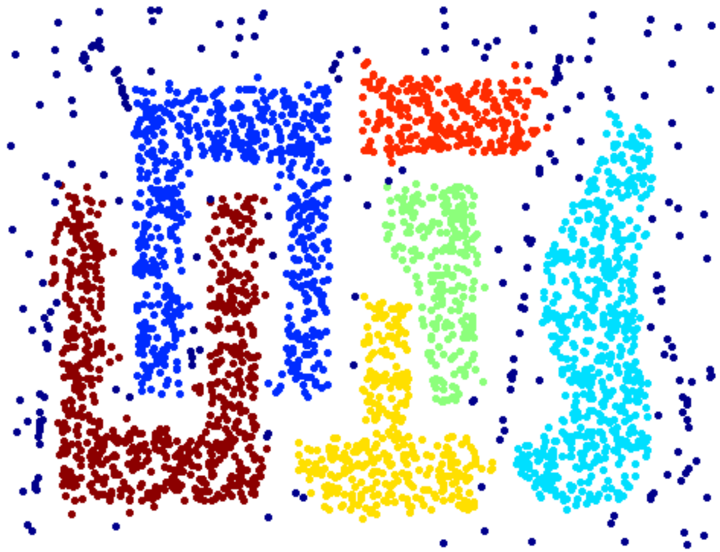
DBSCAN Algorithm (simplified view for teaching)

1. Create a graph whose nodes are the points to be clustered
2. For each core-point c create an edge from c to every point p in the **ε-neighborhood** of c
3. Set N to the nodes of the graph;
4. If N does not contain any core points terminate
5. Pick a core point c in N
6. Let X be the set of nodes that can be reached from c by going forward;
    1. create a cluster containing $X \cup \{c\}$
    2. $N = N / (X \cup \{c\})$
7. Continue with step 4

Remark: points that are not assigned to any cluster are outliers;

When DBSCAN Works Well

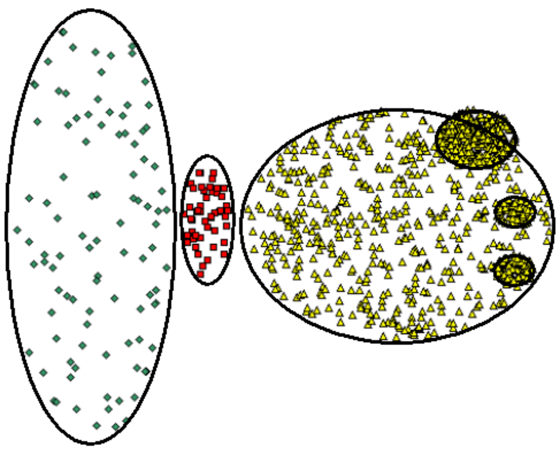
Original Points



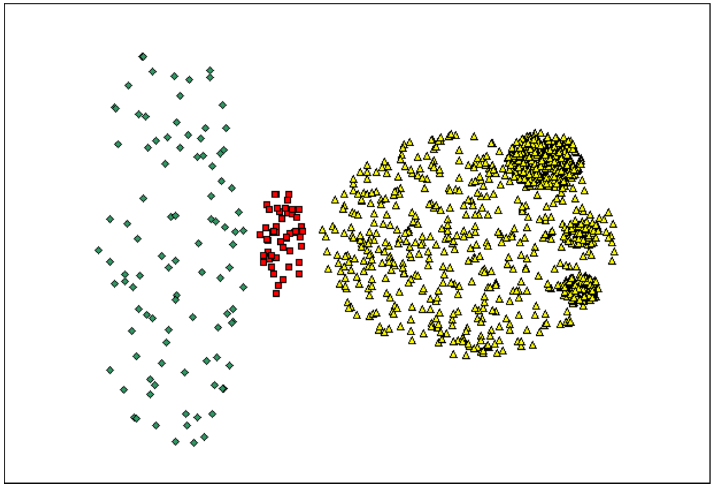
Clusters

- Resistant to Noise
- Can handle clusters of different shapes and sizes

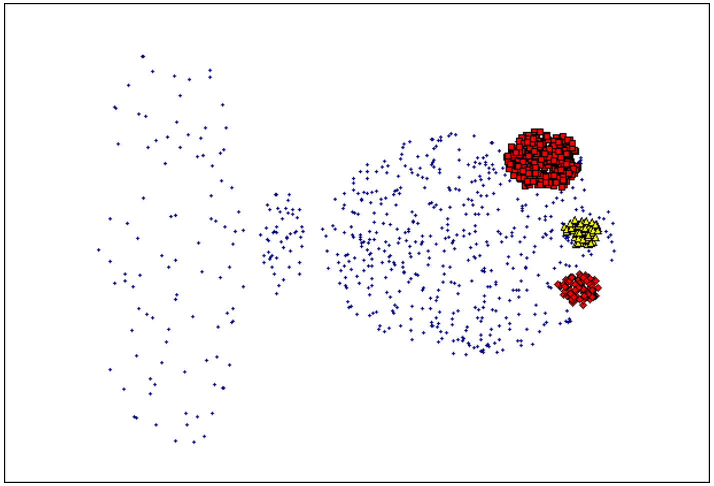
When DBSCAN Does NOT Work Well



Original Points



(MinPts=4, Eps=9.75).



(MinPts=4, Eps=9.92)

- Varying densities
- High-dimensional data

Complexity DBSCAN

- Time Complexity: `O(n²)`
  - For each point it has to be determined if it is a core point
  - Can be reduced to `O(n*log(n))` in lower dimensional spaces by using efficient data structures (where n is the number of objects to be clustered);

- Space Complexity: `O(n)`

Birch

Balanced Iterative Reducing and Clustering Using Hierarchies

Using sklearn

Some Links

- https://youtu.be/xw3RwYs7fUM
- https://scikit-learn.org/stable/modules/generated/sklearn.cluster.Birch.html

Extensions to Hierarchical Clustering

- Major weakness of agglomerative clustering methods
  - Can never undo what was done previously
  - Do not scale well: time complexity of at least $O(n^2)$, where n is the number of total objects
- Integration of hierarchical & distance-based clustering
  - BIRCH (1996): uses CF-tree and incrementally adjusts the quality of sub-clusters
  - CHAMELEON (1999): hierarchical clustering using dynamic modeling

BIRCH

- Zhang, Ramakrishnan & Livny, SIGMOD'96
- Incrementally construct a CF (Clustering Feature) tree, a hierarchical data structure for multiphase clustering
  - **Phase 1**: scan DB to build an initial in-memory CF tree (a multi-level compression of the data that tries to preserve the inherent clustering structure of the data)
  - **Phase 2**: use an arbitrary clustering algorithm to cluster the leaf nodes of the CF-tree
- *Scales linearly*: finds a good clustering with a single scan and improves the quality with a few additional scans
- *Weakness:* handles only numeric data, and sensitive to the order of the data record

Clustering Feature Vector in BIRCH

Clustering Feature (CF): CF = (N, LS, SS)

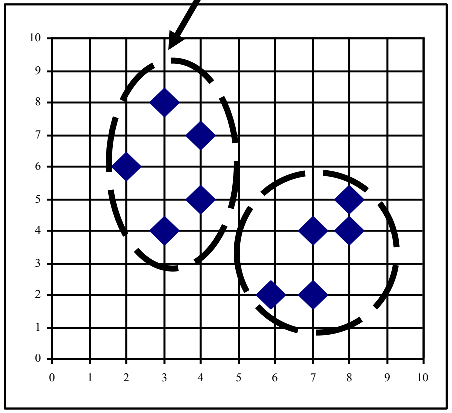
***N**: Number of data points*

***LS**: linear sum of N points:* $\sum_{i=1}^{N} X_i$

***SS**: square sum of N points*

$$\sum_{i=1}^{N} X_i^{\,2}$$

CF = (5, (16,30),(54,190))

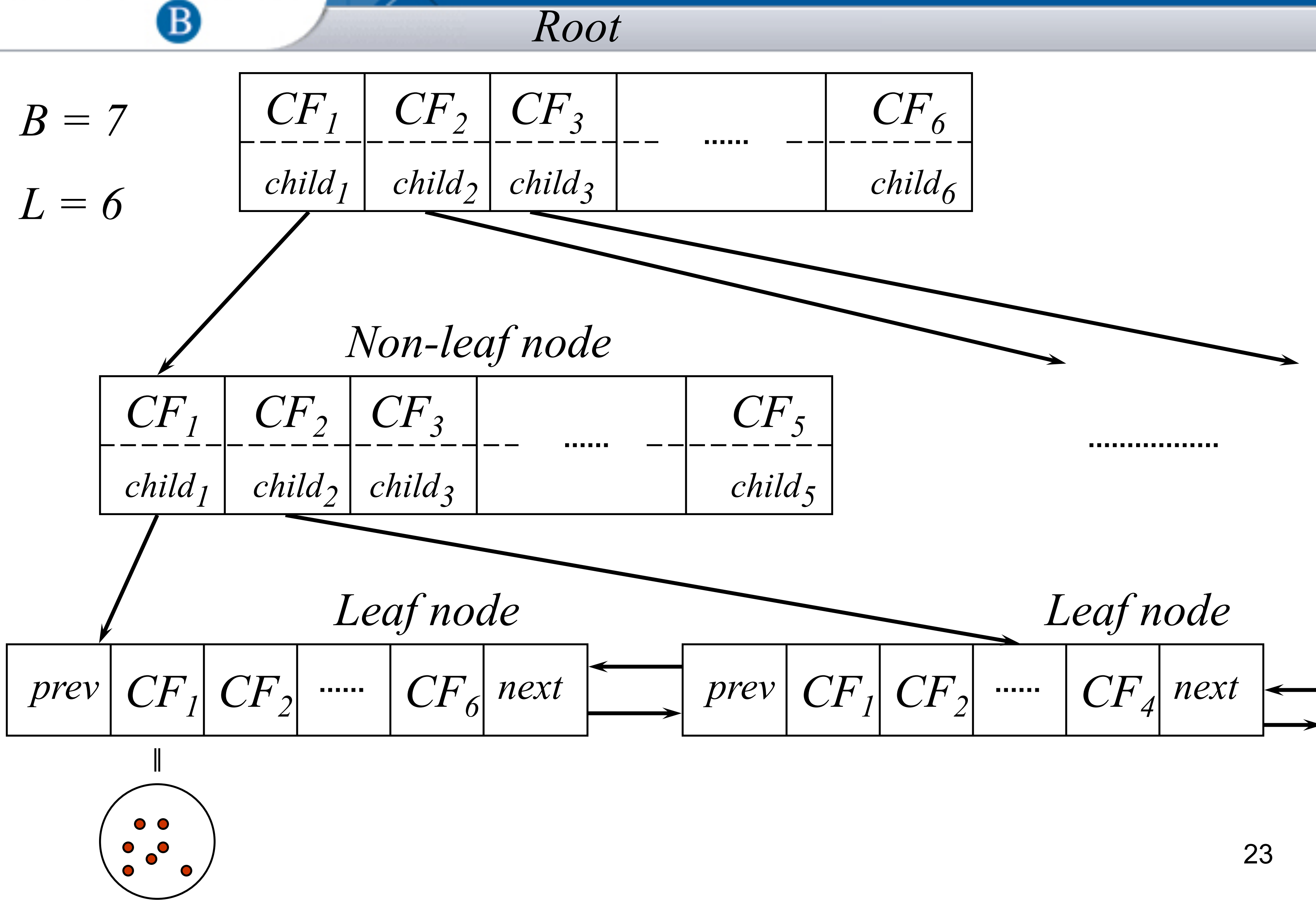


(3,4)
(2,6)
(4,5)
(4,7)
(3,8)

CF-Tree in BIRCH

- Clustering feature:
  - Summary of the statistics for a given subcluster: the 0-th, 1st, and 2nd moments of the subcluster from the statistical point of view
  - Registers crucial measurements for computing cluster and utilizes storage efficiently
- A CF tree is a **height-balanced** tree that stores the clustering features for a hierarchical clustering
  - A nonleaf node in a tree has descendants or "children"
  - The nonleaf nodes store sums of the CFs of their children
- A CF tree has two **parameters**
  - **Branching factor**: max # of children
  - **Threshold**: max diameter of sub-clusters stored at the leaf nodes

The CF Tree Structure



The Birch Algorithm

- Cluster Diameter $$\sqrt{\frac{1}{n(n-1)}\sum(x_i - x_j)^2}$$
- For each point in the input
  - Find closest leaf entry
  - Add point to leaf entry and update CF
  - If **entry diameter > max_diameter**, then split leaf, and possibly parents
- Algorithm is O(n)
- **Concerns**
  - Sensitive to insertion order of data points
  - Since we fix the size of leaf nodes, so clusters may not be so natural
  - Clusters tend to be spherical given the radius and diameter measures

K-Means

Implement your own code

UNIVERSITAT DE BARCELONA

K-Means basis

- It is a partitional algorithm that …
  - Assumes instances are **real-valued vectors**
  - Clusters based on ***centroids, center of gravity***, or **mean of points** in a cluster, ***c***:

$$\vec{\mu}(\mathrm{c}) = \frac{1}{|c|} \sum_{\vec{x} \in c} \vec{x}$$

  - Reassignment of instances to clusters is **based on distance** to the current cluster centroids
    - Manhattan distance (L_1 norm), Euclidean distance (L_2 norm), Cosine similarity

K-Means basis

Algorithm	Basic K-means algorithm.

1: Select K points as initial centroids.
2: **repeat**
3:     Form K clusters by assigning each point to its closest centroid.
4:     Recompute the centroid of each cluster.
5: **until** Centroids do not change.

UNIVERSITAT DE BARCELONA

Discussion on the K-Means method

- K-Means clustering often **terminates at a local optimal**
  - Initialization can be important to find high-quality clusters
- **Need to specify K**, the number of clusters, in advance
  - There are ways to automatically determine the "*best*" K
  - In practice, one often runs a range of values and selected the "*best*" K value
- **Sensitive to noisy data and outliers**
  - Variations: Using K-medians, K-medoids, etc.
- K-Means is applicable only to objects in a **continuous n-dimensional space**
  - Using the K-Modes for **categorical data**
- Non suitable to discover clusters with **non-convex shapes**
  - Using density-based clustering, kernel k-means, etc.

Variations of K-Means

- There are many variants of the K-Means methods, varying different aspects
  - Choosing better initial centroid estimates
    - K-Means++, Intelligent K-Means, Genetic K-Means
  - Choosing different representatives for the clusters
    - K-Medoids, K-Medians, K-Modes
  - Applying feature transformation techniques *(explained at the supervised part of the course)*
    - Weighted K-Means, Kernel K-Means

Initialization of K-Means

- Different initializations may generate rather different clustering results
- Original proposal (MacQueen,1967): selects the k seed randomly
  - Need to run the algorithm multiple times using different seeds
- There are many methods proposed for better initialization of K seeds
  - **K-Means++** (Arthur and Vassilvitskii,2007):
    - The first centroid is selected randomly
    - The next centroid selected is the one that is farthest from the currently selected (selection is based on a weighted probability score).
    - The selection continues until K centroids are obtained

k-Means++

- K-Means algorithm is sensitive to the initialization of the centroids or the mean points
- K-Means++ ensures a smarter initialization of the centroids and improves the quality of the clustering
  – The initialization is different
  – The remaining of the algorithm is the same as standard k-Means

Arthur, D.; Vassilvitskii, S. (2007). **K-means++: the advantages of careful seeding**. Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. Society for Industrial and Applied Mathematics Philadelphia, PA, USA. pp. 1027–1035.

Some k-Means references

- MacQueen, J. B. (1967). **Some Methods for classification and Analysis of Multivariate Observations**. Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability. University of California Press. pp. 281–297.

- Celebi, M. E., Kingravi, H. A., and Vela, P. A. (2013). **A comparative study of efficient initialization methods for the k-means clustering algorithm**. Expert Systems with Applications. 40 (1): 200–210.
- Arthur, D.; Vassilvitskii, S. (2007). **K-means++: the advantages of careful seeding**. Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. Society for Industrial and Applied Mathematics Philadelphia, PA, USA. pp. 1027–1035.

Note **all the documents with this icon are in a zip file in campus virtual**

K-Modes

K-Modes for categorical data

- K-Means cannot handle non-numerical (categorical) data
  - Mapping categorical value to 1/0 cannot generate quality clusters for high-dimensional data
- K-Modes is a variation of the *K-Means* Method (Huang'98)
  - Replacing means of clusters with <u>modes</u>
  - Using new dissimilarity measures to deal with categorical objects
  - Using a <u>frequency</u>-based method to update modes of clusters

K-Modes basis

- K-Modes: an extension to K-Means by replacing means with **modes**

$$\Phi(x_j, z_j) = 1 - n_j^r/n_l \text{ when } x_j = z_j \text{ ; } 1 \text{ when } x_j \neq z_j$$

where z_j is the categorical value of attribute j in Z_l, n_l is the number of objects in cluster l, and n_j^r is the number of objects whose attribute value is r

- Dissimilarity measure between object X and the center of a cluster Z
- The dissimilarity measure (distance function) is **frequency-based**

$$d(X_i, X_l) \equiv \sum_{j=1}^{m} \delta(x_{i,j}, x_{l,j})$$

where

$$\delta(x_{i,j}, x_{l,j}) = \begin{cases} 0, & x_{i,j} = x_{l,j} \\ 1, & x_{i,j} \neq x_{l,j} \end{cases}$$

K-Modes algorithm

- K-Modes deals with categorical attributes

```
Insert the first K objects into K new clusters.
Calculate the initial K modes for K clusters.
Repeat {
    For (each object O) {
       Calculate the similarity between object O and the
       modes of all clusters.
       Insert object O into the cluster C whose mode is the
       least dissimilar to object O.
    }
       Recalculate the cluster modes so that the cluster
       similarity between mode and objects is maximized.
} until (num_iterations or few objects change clusters).
```

K-Modes

- Algorithm is still based on iterative object cluster assignment and centroid update

- A **fuzzy k-modes** method is proposed to calculate a **fuzzy cluster membership** value for each object to each cluster

- A mixture of categorical and numerical data: Using a **K-prototype** method

References of K-Modes

- Zhexue Huang and Michael K. Ng. 2003. **A Note on K-Modes Clustering**. J. Classif. 20, 2 (September 2003), 257-261. DOI=http://dx.doi.org/10.1007/s00357-003-0014-4
- Anil Chaturvedi, Paul E. Green, and J. Douglas Caroll. 2001. **K-Modes Clustering**. J. Classif. 18, 1 (January 2001), 35-55. DOI=http://dx.doi.org/10.1007/s00357-001-0004-3
- Zengyou He, **Approximation algorithms for K-Modes clustering**. https://arxiv.org/pdf/cs/0603120.pdf
- Fuyuan Cao, Jive Liang, Deyu Li, Liang Bai, Chuangyin Dang. **A dissimilarity measure for the K-Modes clustering algorithm**. Knowledge-based Systems, Volume 26, 2012, ISSN 0950-7051.DOI= https://doi.org/10.1016/j.knosys.2011.07.011.

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.652.5571&rep=rep1&type=pdf

K-Medoids

K-Medoids basis

- The **k-Means algorithm is sensitive to outliers**!!
  - since an object with an extremely large value may substantially distort the distribution of the data
- **K-Medoids**:
  - Instead of taking the **mean** value of the object in a cluster as a reference point, **medoids** can be used, which is the **most centrally located object** in a cluster

K-Medoids

- The K-Medoids clustering algorithm:
  - Select *K* points as the initial representative objects (i.e., as initial k-Medoids)
  - Repeat
    - Assigning each point to the cluster with the closest medoid
    - Randomly select a non-representative object o_i
    - Compute the total cost S of swapping the medoid m with o_i
    - If S<0, then swap m with o_i to form the new set of medoids

Discussion on K-Medoids Clustering

- **K-Medoids Clustering**: find representative objects (**medoids**) in clusters
- **PAM (Partitioning Around Medoids)**
  - Starts from an initial set of medoids, and
  - Iteratively replaces one of the medoids by one of the non-medoids if it improves the total sum of the squared errors (SSE) of the resulting clustering
  - PAM works effectively for small data sets but does not scale well for large data sets (due to the computational complexity)
  - Computational Complexity: PAM $O(K(n-K)^2)$ (quite expensive!)
- Efficiency improvements on PAM
  - **CLARA** (Kaufmann & Rousseeuw, 1987)
    - PAM on samples; $O(Ks^2 + K(n-k))$, s is the sample size
  - **CLARANS** (ng & Han, 1994): Randomized re-sampling, ensuring efficiency + quality

References K-Medoids

- R. T. Ng and Jiawei Han (2002), "**CLARANS: a method for clustering objects for spatial data mining**" in *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, no. 5, pp. 1003-1016, Sep/Oct 2002. doi: 10.1109/TKDE.2002.1033770

- Kaufman, L. and Rousseeuw, P.J. (1987), **Clustering by means of Medoids**, in Statistical Data Analysis Based on the L_1 –Norm and Related Methods, edited by Y. Dodge, North-Holland, 405–416

- H.S. Park , C.H. Jun, **A simple and fast algorithm for K-medoids clustering**, Expert Systems with Applications, 36, (2) (2009), 3336–3341

- J. Xie and S. Jiang, "**A Simple and Fast Algorithm for Global K-means Clustering**", 2010 Second International Workshop on Education Technology and Computer Science, Wuhan, 2010, pp. 36-40. doi: 10.1109/ETCS.2010.347

UNIVERSITAT DE BARCELONA

K-Prototypes

K-prototypes Algorithm

- To integrate the k-means and k-modes algorithms into the k-prototypes algorithm that is used to cluster the mixed-type objects

- The dissimilarity between two mixed-type objects X and Y, which are described by attributes $A_1^r, A_2^r, \ldots, A_p^r, A_{p+1}^c, \ldots, A_m^c$ (*m* is the attribute numbers the first *p* means numeric data, the rest means categorical data), can be measured by::

$$d_2(X,Y) = \sum_{j=1}^{p}(x_j - y_j)^2 + \gamma \sum_{j=p+1}^{m} \delta(x_j, y_j)$$

K-prototypes Algorithm(cont.)

$$d_2(X,Y) = \sum_{j=1}^{p}(x_j - y_j)^2 + \gamma \sum_{j=p+1}^{m}\delta(x_j, y_j)$$

- The first term is the Euclidean distance measure on the numeric attributes and the second term is the simple matching dissimilarity measure on the categorical attributes
- The weight γ is used to avoid favoring either type of attribute

References of K-prototypes

- Zhexue Huang, **Clustering large datasets with mixed numerical and categorical values**. https://pdfs.semanticscholar.org/d42b/b5ad2d03be6d8fefa63d25d02c0711d19728.pdf
- Byoungwook Kim. **A Fast K-prototypes Algorithm Using Partial Distance Computation**. https://www.researchgate.net/publication/316348009_A_Fast_K-prototypes_Algorithm_Using_Partial_Distance_Computation

Course. Introduction to Machine Learning

Work 1. Clustering Exercise

Session 2

Course 2023-2024

Dr. Maria Salamó Llorente

Dept. Mathematics and Informatics,

Faculty of Mathematics and Informatics,
University of Barcelona