15-826: Multimedia Databases and Data Mining

Data Mining - clustering (revisited)
C. Faloutsos

Outline
Goal: ‘Find similar / interesting things’
• Intro to DB
• Indexing - similarity search
  • Data Mining

Data Mining - Detailed outline
• Statistics
• AI - decision trees
• DB
  – data warehouses; data cubes; OLAP
  – classifiers
  – association rules
  – misc. topics:
    • clustering (revisited)
    • reconstruction of info

Clustering - outline
• Partitioning methods (eg., k-means)
  • Hierarchical methods (eg., agglomerative)
  • Density-based methods
  • Grid-based methods

Partitioning methods
K-means (k is given). Goal:
• find k-points and assign the nearest data points to them
• to minimize the sum of squared distances
Algorithm: Iterative improvement
Can be expensive (needs many iterations)
Sensitive to outliers

Variations:
• k-medoids / CLARANS (but: O(n**2) )
• k harmonic means [Zhang, Hsu, Dayal+ HPLabs, 99]
Clustering - outline

- Partitioning methods (e.g., k-means)
- Hierarchical methods (e.g., agglomerative)
- Density-based methods
- Grid-based methods

Hierarchical methods

- Agglomerative / Divisive Hierarchical Clustering
- BIRCH
- CURE
- CHAMELEON

Hierarchical methods

BIRCH:
- uses the ‘CF’-tree (count, center, sum of squares, for the points in each node
- needs 1 or 2 passes
- finds spherical clusters
- needs two numbers: branching factor; radius threshold

Hierarchical methods

BIRCH Algorithm - phase 1:
- start inserting points in the CF-tree (~ a sphere tree)
- if the radius, is exceeded, split the node
phase 2:
  - cluster the leaf nodes of the tree

CURE

Main ideas:
- use sampling
- represent a cluster by many centroids (thus clusters can have arbitrary shapes)

CURE

Algorithm:
- get a sample
- partition it into a set of partitions
- partially cluster each partition
- cluster the partial clusters
CURE

- O(n) complexity
- good quality clusters
- relatively small sensitivity to user parameters

CHAMELEON

- for each object, link it to its k-nn
- use graph partitioning, to create many small clusters (= connected components)
- merge clusters that are ‘close enough’

CHAMELEON

- similarity of a pair of clusters Ci, Cj: use
  - ‘relative inter-connectivity’ (= avg # of cross-links, normalized)
  - ‘relative closeness’ (= avg pairwise distance, normalized)
- Empirically: better quality clusters, but O(n^2)

Clustering - outline

- Partitioning methods (eg., k-means)
- Hierarchical methods (eg., agglomerative)
  - Density-based methods
  - Grid-based methods

DBSCAN

- High level idea: group high-density areas, that are within a threshold
- in detail: decide on
  - ε: a distance threshold and
  - minPts: minimum number of points in a neighborhood
- ‘core object’: iff it has >= minPts within ε
DBSCAN

- Pictorially (say, minPts=3):

  ![DBSCAN diagram](image)

- 'directly density reachable'
- 'density-connected'

DBSCAN

- can give elongated clusters
- needs $O(n^2)$ time at worst; less, with a spatial index (but: dim. curse...)
- Also, sensitive to $\epsilon$ (which is global \textit{not adaptive})
- hence: OPTICS

OPTICS

- Given 'minPts'
- impose a sequential ordering on the points
- estimate the 'reachability distance' and
- look for plateaus

(Details are tricky - intuitively, somehow similar to traversing an MST, and plotting the edge length for each node)

OPTICS

- reachable distance
- rank of object

OPTICS

- reachable distance
- rank of object
OPTICS

- speed: like DBSCAN (O(n*n), or less, with a spatial index)

Clustering - outline

- Partitioning methods (eg., k-means)
- Hierarchical methods (eg., agglomerative)
- Density-based methods
  - Grid-based methods

Grid-based methods

- Main idea: impose a grid; group together ‘similar’ nearby cells
- STING: do a quad-tree decomposition. For each cell:
  - keep statistics,
  - test (chi-square) whether the distribution is known (uniform, Gauss, etc)
  - merge similar cells together and
  - repeat recursively

Grid-based methods

- ‘WaveCluster’ (2- or 3-d address space)
  - do wavelet transform first
  - create a hierarchy of clusters, one for each resolution, by grouping the connected components

  hi-res  med-res  low-res

Grid-based methods

- ‘CLIQUE’: for high dimensions
  - ‘Dense’ cell: has >= k points
  - Goal: find ‘dense’ cells, in m, or lower, dimensions

‘WaveCluster’: Fast (O(n)), but only for low-dimensionality
Grid-based methods

Idea:
• project in lower dimensionalities;
• report dense cells, that are connected
• look for higher-dimensionality dense cells

Uses an ‘a-priori’-like argument:
if a cell is ‘dense’, so are all its projections