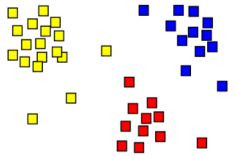
15-853:Algorithms in the Real World

Clustering: Lectures 1 and 2

Given a set of objects and a similarity (or distance) measure among the objects, cluster into groups of similar (close) objects

Also called:

- Unsupervised learning
- Classification
- Typology
- Numerical taxonomy



Applications

Biology: Multiple allignments, evolutionary trees

Business: Market research, risk analysis

Liberal arts: Classifying painters, writers, musicians

Sociology: personality types, classifying criminals, classifying survey results.

Computer Science: compression, information retrieval, text mining, image segmentation, recommender systems, anomaly detection

Types of clustering

Hard vs. soft

Hierarchical vs flat

Distance vs similarity based

Distance metrics:

- Euclidean, Minkowski, Hamming, Edit, …

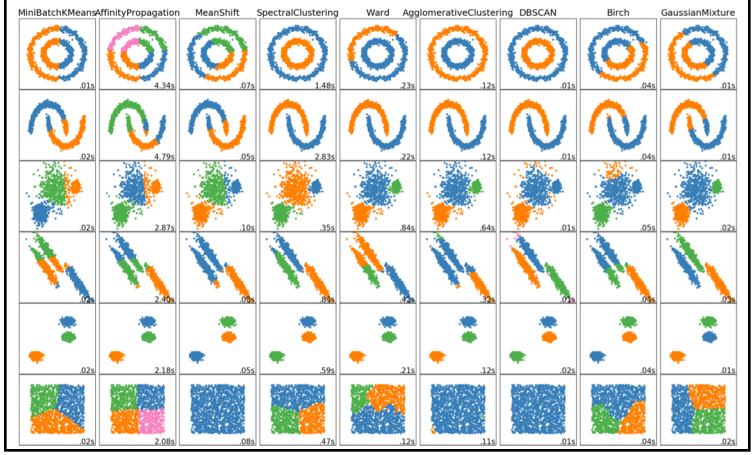
Similarity measures:

- Cosine, Kernel functions, or $S_{ij} = (1 + d_{ij})$

Main Approaches

Centroid based: K-means
Distribution/Model-based (EM)
Mixture of gaussians
Spectral
Agglomerate
Density based
Neural nets

Examples



K-means

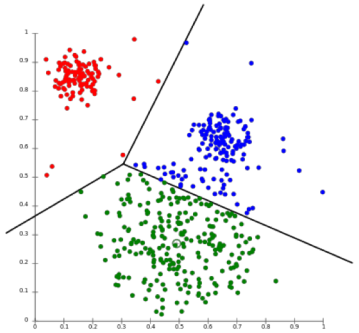
K-clustering: given a metric space (X,d), and a point set $S \subset X$ partition S into k sets $C_1, C_2, \cdots, C_k$

Cost: $\phi(C) = \sum_1^k min_{c \epsilon X} \sum_{x \epsilon C_i} d(c,x)^2$

Goal: minimize the cost
Problem is NP-hard. Can find approximations.
Typically looking for an approximation.
Related to expectation-maximization, mixture of gaussians, and k-median.

K-means

Dedines Voronoi Cells:



Lloyds algorithm for K-means

A greedy local search algorithm:

Start with a set of centers: c1, c2, …, ck in X
Repeat until "convergence":

- Assign each x in S to nearest center
- Update location of each center to minimize sum of distances to points assigned to it

Will converge but perhaps slowly and perhaps to a local minimum
Often tried with many starting sets
Often dimensionality reduction is applied first

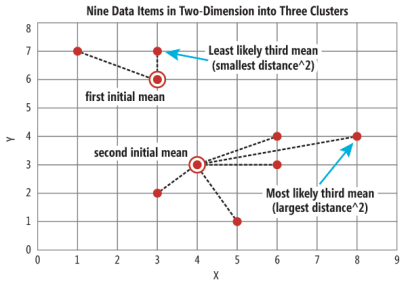
K-means++

Picks the starting set more intelligently:

Pick a center uniformly at random from X and add to centers Y (initially empty)
For k-1 steps:

- For each x in X calculate min distance d(x) from points in Y
- Pick an x in X with probability proportional to $d(x)^2$ and add to Y

K-means++



Gives an 8(ln k + 2) approximation even without Lloyds
Using Lloyds will only improve the result.

Expectation Maximization (EM)

K-means is a special case.

Start with an arbitrary set of clusters defined by parameters: p1, p2, …, pk
Repeat until "convergence":

- **Expectation**: Assign each x in S to cluster that best matches parameters.
  This can be a probabilistic (soft) assignment
- **Maximization**: Update parameters to best fit the assignment.

Coverges to a local maximal likelihood estimator.

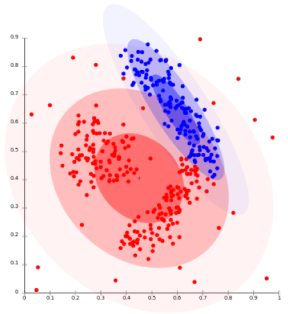
EM: mixture of gaussians

Here the parameters are (anisotropic) Gaussians. The parameters form a matrix

Can deal with elongated structures.

More generally can be any parameterized model of the data

Useful if you know what form the clusters will have.



Spectral Clustering

Input is a similarity graph (often sparse)

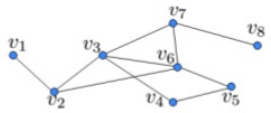
- Cosine measure (x dot y)/(||x|| ||y||)
- K-nearest neighbor graph

Uses Eigenvectors of the Graph Laplacian

- If W is the weight matrix
- and D is a diagonal matrix summing each row
- L = D – W (often normalized)

GFT Example

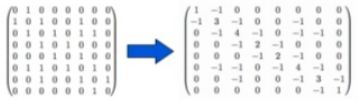
❑Graph Laplacian



$G = \{V, E\}$ with unitary weights

$D = diag(degree(v_1) \quad \ldots \quad degree(v_n))$

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



$\mathbf{W}$ $\mathcal{L}$

- Symmetric
- Off-diagonal entries non-positive
- Rows sum up to zero
- Has a complete set of orthonormal eigenvectors: $L = \chi \Lambda \chi^T$

$0 = \lambda_0 < \lambda_1 \leq \ldots \leq \lambda_{n-1}$

Spectral Clustering

Used in two possible ways:

- Divisive hierarchical clustering
  Use second eigenvector to split in two
  Recurse on the parts
- K-clustering
  Use first l eigenvectors as a reduced dimensional space
  Use k-means on the result

The Eigenvectors

First is trivial (all 1s) with eigenvalue 0 (if normalized)

The second gives information about how well the graph can be separated.

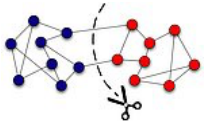
Cheeger constant :

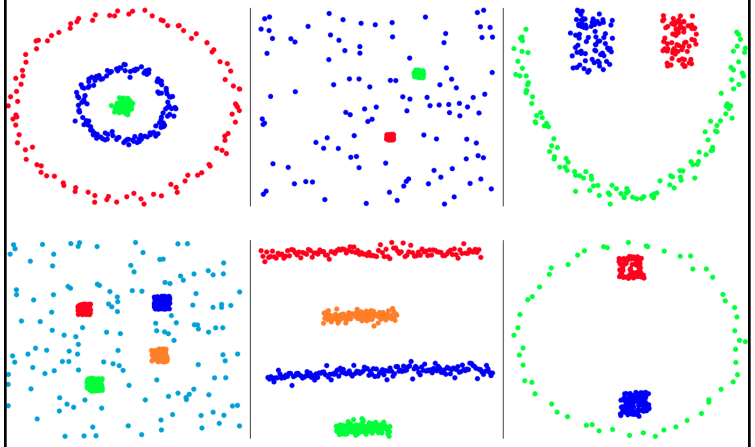
$E(A,V\setminus A)/|A|$ for any A, $|A| < |V|/2$

A measure of how well graph separates

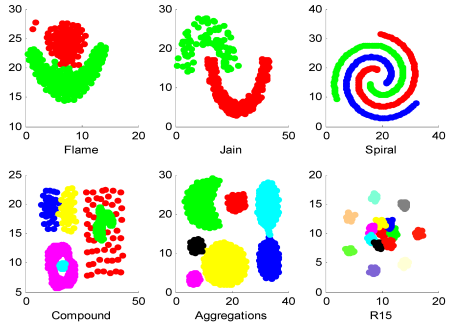
Related to expander graphs (do not separate)

Related to second eigenvalue.





This uses number of shared nearest neighbors as the weight. Note this is very similar to SVD since the number of shared neighbors is AA^T



Agglomerate Clustering

Hierarchical: bottom up

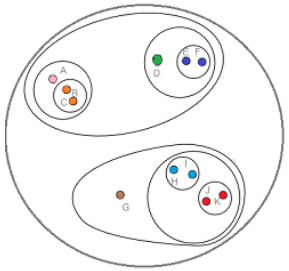
Assuming a distance measure

Initially one group per object: $G = P$

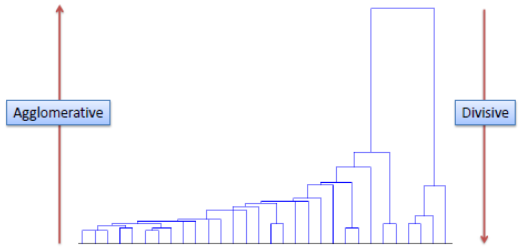
While $|G| > 1$

find "closest" pair in G and join pair into group

Algorithms vary depending on distance between groups.

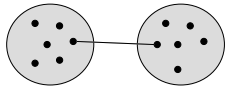


Hierarchical Clustering



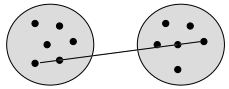
Agglomerate: Distances

Choice 1: min distance (single linkage)

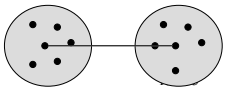


Min Spanning Tree

Choice 2: max distance (complete linkage)

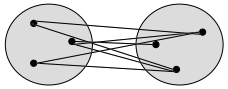


Choice 3: centroid distance

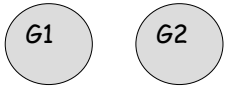


Agglomerate: Distances

Choice 4: average distance



Choice 5: Min sum of squares (Ward's method)



$$SS(G) = \frac{1}{2|G|}\sum_{x\epsilon G, y\epsilon G} d(x,y)^2$$

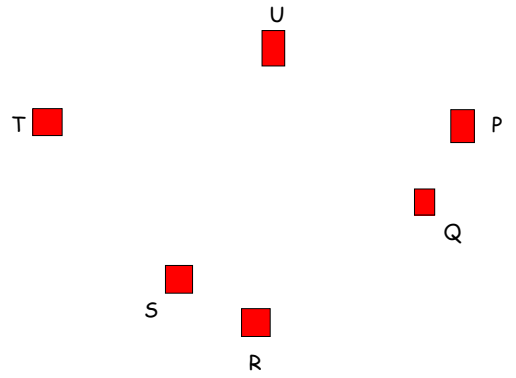
$$SS(G_{12}) - SS(G_1) - SS(G_2)$$

Heap-based Algorithm

Initialize KD-Tree with elements
Initialize heap with best match for each element
Repeat {
    Remove best pair <A,B> from heap
    If A and B are active clusters {
        Create new cluster C = A+B
        Update KD-Tree, removing A and B and inserting C
        Use KD-Tree to find best match for C and insert into heap
    } else if A is active cluster {
        Use KD-Tree to find best match for A and insert into heap}
} until only one active cluster left

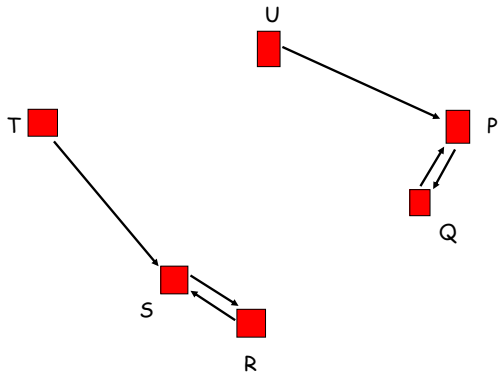
Walter, Bala, Kulkarni, Pingali

Heap-based Algorithm Example



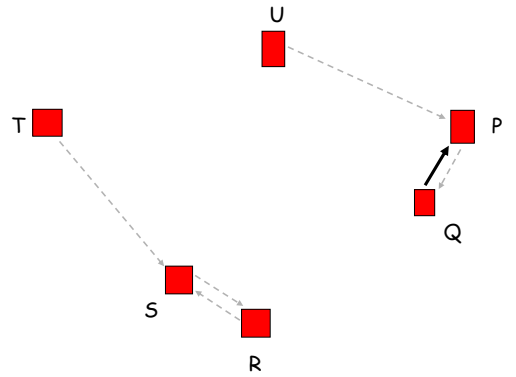
Walter, Bala, Kulkarni, Pingali

Heap-based Algorithm Example



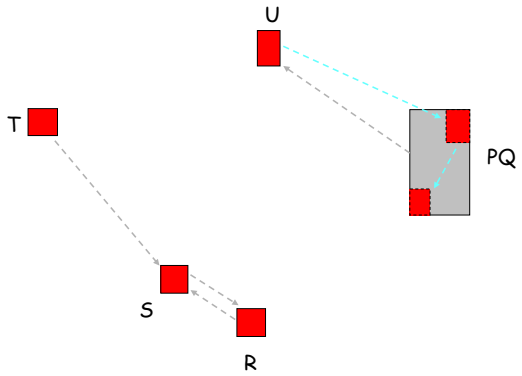
Walter, Bala, Kulkarni, Pingali

Heap-based Algorithm Example



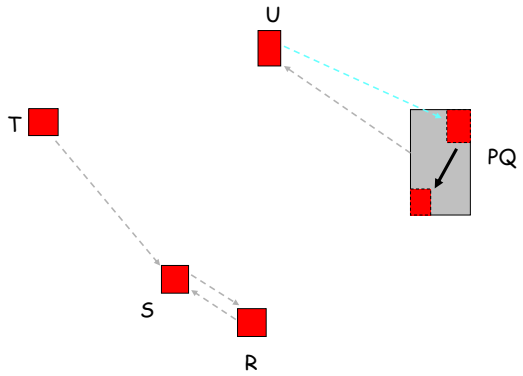
Walter, Bala, Kulkarni, Pingali

Heap-based Algorithm Example



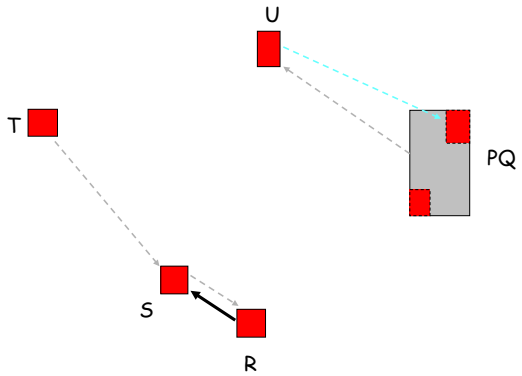
Walter, Bala, Kulkarni, Pingali

Heap-based Algorithm Example



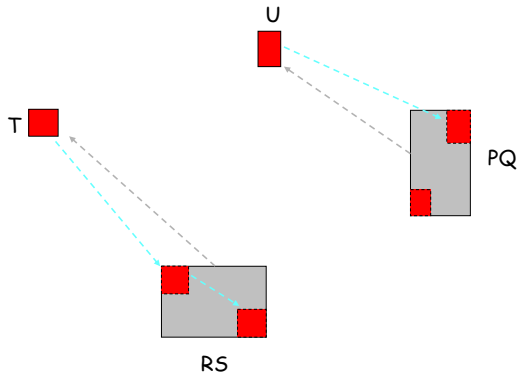
Walter, Bala, Kulkarni, Pingali

Heap-based Algorithm Example



Walter, Bala, Kulkarni, Pingali

Heap-based Algorithm Example



Walter, Bala, Kulkarni, Pingali

Main Approaches

Centroid based: K-means
Distribution/Model-based (EM)
Mixture of gaussians
Spectral
Agglomerate
Density based
Neural nets

Examples