Clustering

1

Informal goal

- Given set of objects and measure of similarity between them, group similar objects together
- What mean by "similar"?
- What is good grouping?
- Computation time / quality tradeoff

2

General types of clustering

- "Soft" versus "hard" clustering
  - Hard: partition the objects
    - each object in exactly one partition
  - Soft: assign degree to which object in cluster
    - view as probability or score
- flat versus hierarchical clustering
  - hierarchical = clusters within clusters

3

Applications:

Many

- biology
- astronomy
- computer aided design of circuits
- information organization
- marketing
- …

4

Clustering in information search and analysis

- Group information objects
  - ⇒ discover topics
  - ? other groupings desirable
- Clustering versus classifying
  - classifying: have pre-determined classes with example members
  - clustering:
    - get groups of similar objects
    - added problem of labeling clusters by topic
      - e.g. common terms within cluster of docs.

5

Example applications in search

- Query evaluation: cluster pruning (§7.1.6)
  - cluster all documents
  - choose representative for each cluster
  - evaluate query w.r.t. cluster reps.
  - evaluate query for docs in cluster(s) having most similar cluster rep.(s)
- Results presentation: labeled clusters
  - cluster only query results
  - e.g. Yippy.com (metasearch)

hard / soft? flat / hier?

6

Issues

- What, if any, attributes represent items for clustering purposes?
- What is measure of similarity between items?
  - General objects and matrix of pairwise similarities
  - Objects with specific properties that allow other specifications of measure
    - Most common:
      Objects are d-dimensional vectors
      » Euclidean distance
      » cosine similarity
- What is measure of similarity between clusters?

7

Issues continued

- Cluster goals?
  - Number of clusters?
  - flat or hierarchical clustering?
  - cohesiveness of clusters?
- How evaluate cluster results?
  - relates to measure of closeness between clusters
- Efficiency of clustering algorithms
  - large data sets **=>** external storage
- Maintain clusters in dynamic setting?
- Clustering methods? - MANY!

8

Quality of clustering

- In applications, quality of clustering depends on how well solves problem at hand
- Algorithm uses measure of quality that can be optimized, but that may or may not do a good job of capturing application needs.
- Underlying graph-theoretic problems usually NP-complete
  - e.g. graph partitioning
- Usually algorithm not finding optimal clustering

9

General types of clustering methods

- constructive: decide in what cluster each object belongs and don't change
  - often faster
- iterative improvement: start with a clustering and move objects around to see if can improve clustering
  - often slower but better

10

Vector model: K- means algorithm

- Well known, well used
- Flat clustering
- Number of clusters picked ahead of time
- Iterative improvement
- Uses notion of centroid
- Typically uses Euclidean distance

11

K-means overview

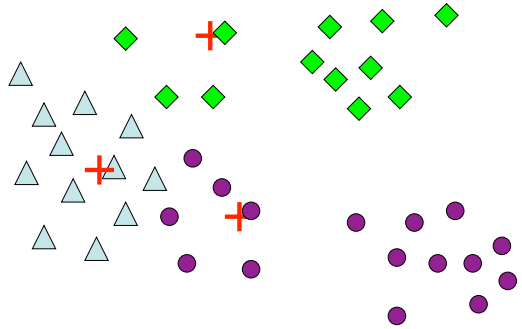
- Choose k points among set to be clustered
  - Call them k centroids
  - not required to be in set to be clustered
- For each point not selected, assign it to its closest centroid
  - Assignments give initial clustering
- Until "happy" do:
  - Recompute centroids of clusters:

    centroid of set of vectors $\{v_i \mid 1 \le i \le n\} = 1/n * \Sigma^n_{i=1} v_i$
    - New centroids may not be points of original set
  - Reassign all points to closest centroid
    - Updates clusters

12

An Example

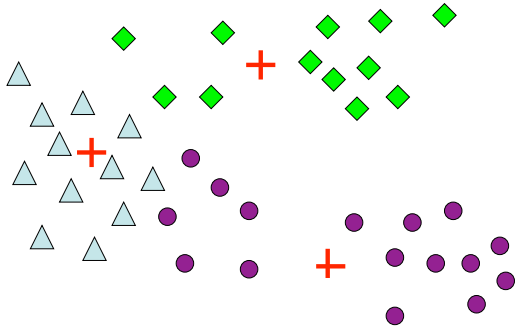
start: choose centroids and cluster



13

An Example

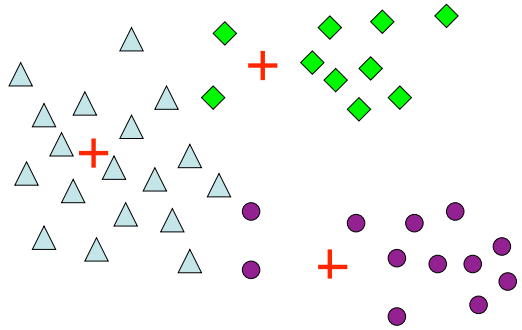
recompute centroids



14

An Example

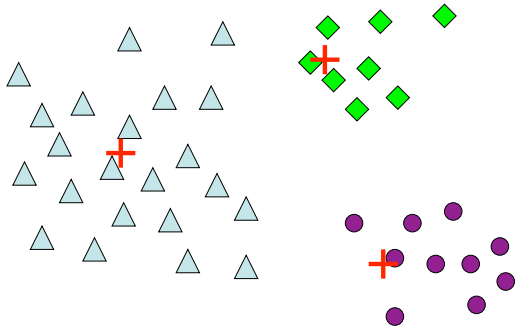
re-cluster around new centroids



15

An Example

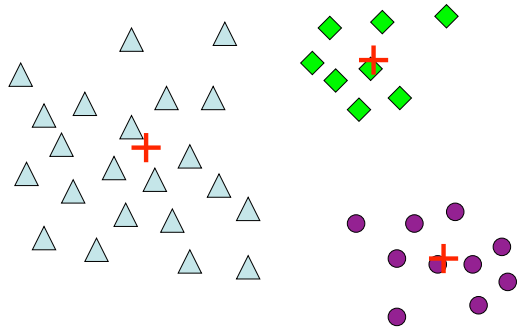
2nd recompute centroids and re-cluster



16

An Example

3rd (final) recompute and re-cluster



17

Illustrations thanks to 2006 student Martin Makowiecki

Details for K-means

- Need definition of centroid

  $c_i = 1/|C_i| \sum_{x \in C_i} x$ for i^{th} cluster C_i containing objects x

  notion of sum of objects ?
- Need definition of distance to (similarity to) centroid
- Typically vector model with Euclidean distance
- minimizing sum of squared distances of each point to its centroid = Residual Sum of Squares

$$RSS = \sum_{i=1}^{K} \sum_{x \in C_i} dist(c_i, x)^2$$

18

K-means performance

- Can prove RSS decreases with each iteration, so converge
- Can achieve local optimum
  - No change in centroids
- Running time depends on how demanding stopping criteria
- Works well in practice
  - speed
  - quality

19

Time Complexity of K-means

- Let t_{dist} be the time to calculate the distance between two objects
- Each iteration time complexity:

  $O(K*n*t_{dist})$

  n = number of objects
- Bound number of iterations I giving

  $O(I*K*n*t_{dist})$
- for m-dimensional vectors:

  $O(I*K*n*m)$

  m large and centroids not sparse

20

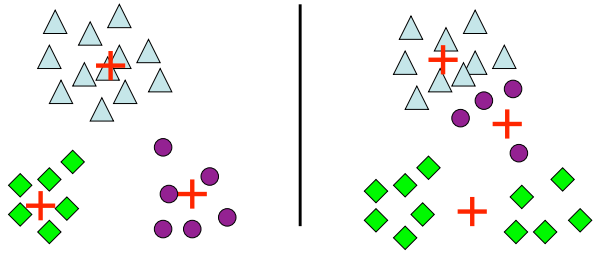
Space Complexity of K-means

- Store points and centroids
  - vector model: O((n + K)m)
- External algorithm versus internal?
  - store k centroids in memory
  - run through points each iteration

21

Choosing Initial Centroids

- Bad initialization leads to poor results



Optimal

Not Optimal

22

Choosing Initial Centroids

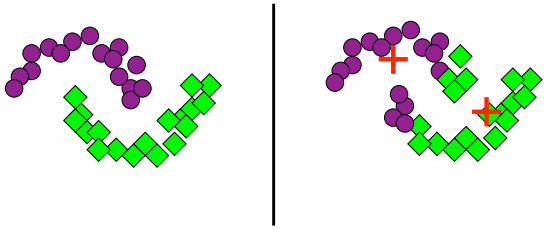
Many people spent much time examining how to choose seeds

- Random
  - Fast and easy, but often poor results
- Run random multiple times, take best
  - Slower, and still no guarantee of results
- Pre-conditioning
  - remove outliers
- Choose seeds algorithmically
  - run hierarchical clustering on sample points and use resulting centroids
  - Works well on small samples and for few initial centroids

23

K-means weakness

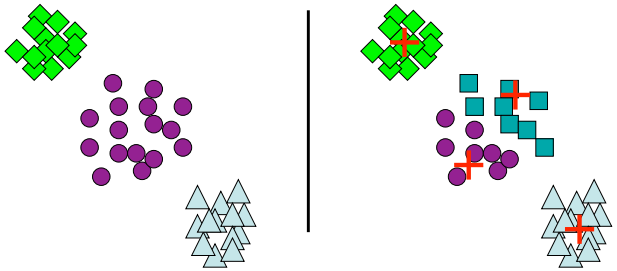
Non-globular clusters



24

K-means weakness

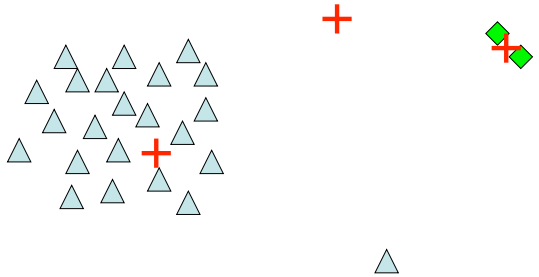
Wrong number of clusters



25

K-means weakness

Outliers and empty clusters



26

Real cases tend to be harder

- Different attributes of the feature vector have vastly different sizes
  - size of star versus color
- Can weight different features
  - how weight greatly affects outcome
- Difficulties can be overcome

27

Clustering Algorithms
for general similarity measures

general similarity measure:
specified by object X object similarity matrix

28

Types of general clustering methods

- constructive algorithms
- agglomerative versus divisive construction
  - agglomerative = bottom-up
    - build up clusters from single objects
  - divisive = top-down
    - break up cluster containing all objects into smaller clusters
  - both agglom'tive and divisive give hierarchies
  - hierarchy can be trivial:

1 (. .) . . . 2 ((. .) .) . .
3 (((. .) .) .) . 4 ((((. .) .) .) .)

29

Similarity between clusters

Possible definitions:

I. similarity between most similar pair of objects with one in each cluster
   - called single link

. . . .
^ ^

II. similarity between least similar pair objects, one from each cluster
   - called complete linkage

. . . .
^ ^

30

Similarity between clusters, cont.

Possible definitions:

III. average of pairwise similarity between all pairs of objects, one from each cluster
   - "centroid" similarity

IV. average of pairwise similarity between all pairs of distinct objects, including w/in same cluster
   - "group average" similarity

- Generally no representative point for a cluster;
  - compare K-means
- If using Euclidean distance as metric
  - centroid
  - bounding box

31

General Agglomerative

- Uses any computable cluster similarity measure sim(C_i, C_j)
- For n objects v_1, …, v_n, assign each to a singleton cluster $C_i = \{v_i\}$.
- repeat {
  - identify two most similar clusters C_j and C_k (could be ties – chose one pair)
  - delete C_j and C_k and add (C_j U C_k) to the set of clusters

  } until only one cluster
- Dendrograms diagram the sequence of cluster merges.

32

Agglomerative: remarks

- *Intro. to IR* discusses in great detail for cluster similarity:
  - single-link, complete-link, group average, centroid
- Uses priority queues to get time complexity
  $O((n^2 \log n)*$(time to compute cluster similarity))
  - one priority queue for each cluster: contains similarities to all other clusters plus bookkeeping info
  - time complexity more precisely:
    O((n^2) *(time to compute object-object similarity) + $(n^2 \log n)$*
    (time to compute sim($cluster_z$, $cluster_j$ U $cluster_k$) if know sim($cluster_z$, $cluster_j$) and sim($cluster_z$, $cluster_k$)) **)**
- Problem with priority queue?

33

Single pass agglomerative-like

Given arbitrary order of objects to cluster: $v_1, \ldots v_n$ and threshold τ

Put v_1 in cluster C_1 by itself
For i = 2 to n {
  for all existing clusters C_j
    calculate sim(v_i, C_j);
  record most similar cluster to v_i as $C_{max(i)}$
  if sim(v_i, $C_{max(i)}$) > τ add v_i to $C_{max(i)}$
  else create new cluster {v_i}
}

ISSUES?

34

Issues

- put v_i in cluster after seeing only $v_1, \ldots v_{i-1}$
- not hierarchical
- tends to produce large clusters
  - depends on τ
- depends on order of v_i

35

Alternate perspective for single-link algorithm

- Build a minimum spanning tree (MST)
  - graph algorithm
    - edge weights are pair-wise similarities
    - since in terms of similarities, not distances, really want maximum spanning tree
- For some threshold τ, remove all edges of similarity < τ
- Tree falls into pieces => clusters
- Not hierarchical, but get hierarchy for sequence of τ

36