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 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 versus iterative improvement
  - 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

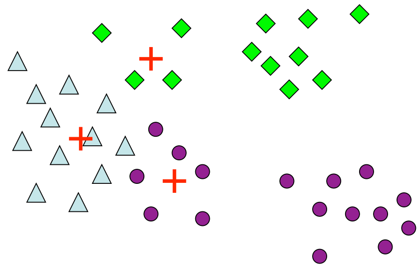
K-means overview

- 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
  - All assignment give initial clustering
- Until "happy" do:
  - Recompute centroids of clusters
    - New centroids may not be points of original set
  - Reassign all points to closest centroid
    - Updates clusters

12

An Example

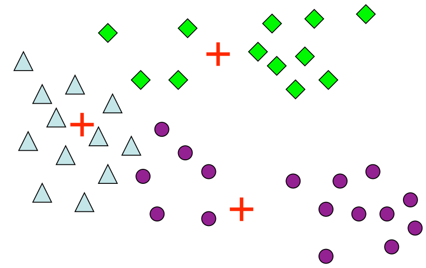
start: choose centroids and cluster



13

An Example

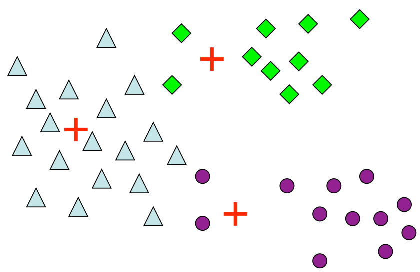
recompute centroids



14

An Example

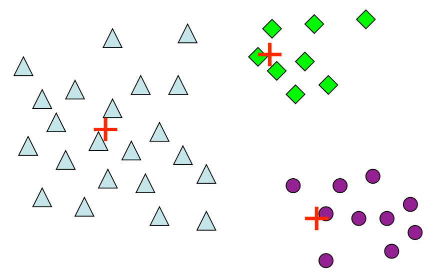
re-cluster around new centroids



15

An Example

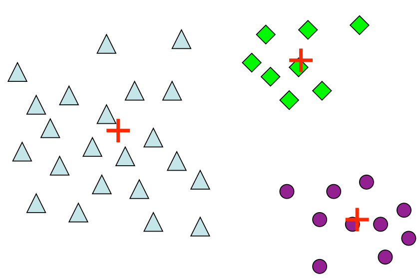
2^{nd} recompute centroids and re-cluster



16

An Example

3rd (final) recompute and re-cluster



17

Illustrations thanks to 2006 student Martin Makowiecki

18

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

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

19

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

20

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

21

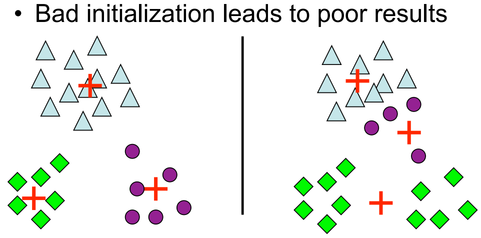
Space Complexity of K-means

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

22

Choosing Initial Centroids

- Bad initialization leads to poor results



Optimal **Not Optimal**

23

Choosing Initial Centroids

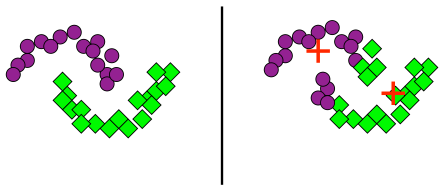
Many people spent much time examining how to choose seeds

- 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

24

K-means weakness

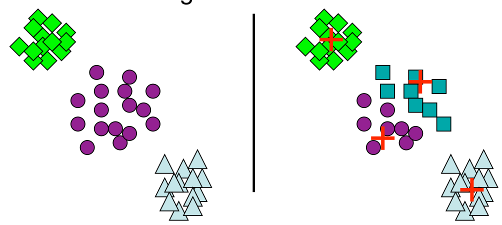
Non-globular clusters



25

K-means weakness

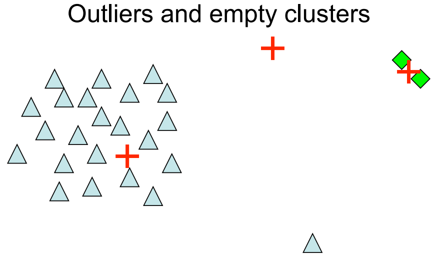
Wrong number of clusters



26

K-means weakness

Outliers and empty clusters



27

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

28

Clustering Algorithms
for general similarity measures

29

Types of general clustering methods

- agglomerative versus divisive algorithms
  - 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 ((((. .) .) .) .)

30

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

     . . . .
     ^   ^

31

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

32

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.

33

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^2logn)*(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^2logn)*
    (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?

34

Single pass agglomerative-like

Given arbitrary order of objects to cluster: v_1, … 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?

35

Issues

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

36

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

37

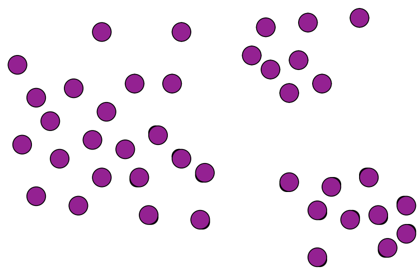
Hierarchical Divisive: Template

1. Put all objects in one cluster
2. Repeat until all clusters are singletons
   a) choose a cluster to split
      - what criterion?
   b) replace the chosen cluster with the sub-clusters
      - split into how many?
      - how split?
      - "reversing" agglomerative => split in two

- cutting operation: cut-based measures seem to be a natural choice.
  - focus on similarity across cut - lost similarity
- not necessary to use a cut-based measure

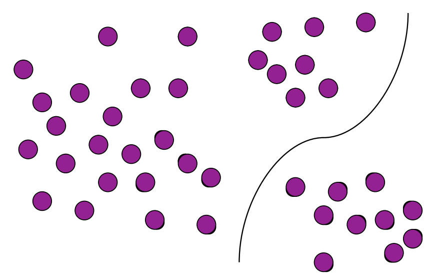
38

An Example



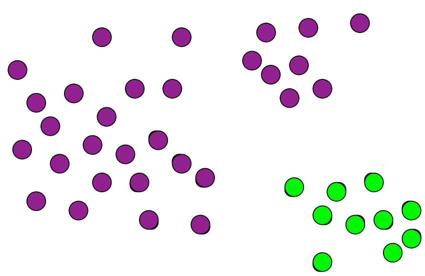
39

An Example: 1st cut



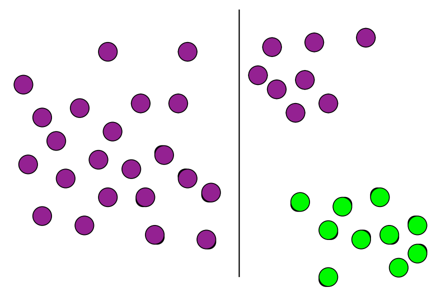
40

An Example: result of 1st cut



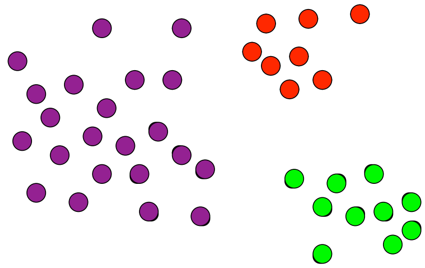
41

An Example: 2nd cut



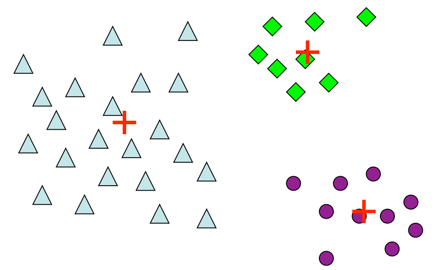
42

An Example: stop at 3 clusters



43

Compare k-means result



44

Cut-based optimization

- weaken the connection between objects in different clusters *rather than* strengthening connection between objects within a cluster

- Are many cut-based measures
- We will look at two

45

Inter / Intra cluster costs

Given:
- $V = \{v_1, \ldots, v_n\}$, the set of all objects
- A partitioning clustering $C_1, C_2, \ldots C_k$ of the objects:

$$V = \cup_{i=1,\ldots,k} C_i .$$

Define:
- $\text{cutcost}(C_p) = \sum_{\substack{v_i \text{ in } C_p \\ v_j \text{ in } V\text{-}C_p}} \text{sim}(v_i, v_j).$

- $\text{intracost}(C_p) = \sum_{(v_i, v_j) \text{ in } C_p} \text{sim}(v_i, v_j).$

46

Cost of a clustering

total relative cut cost $(C_1, \ldots, C_k)$ =

$$\sum_{p=1}^{k} \frac{\text{cutcost}(C_p)}{\text{intracost}(C_p)}$$

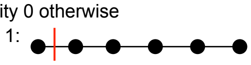
- contribution each cluster:
  ratio external similarity to internal similarity

Optimization

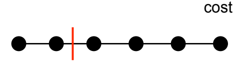
Find clustering $C_1, \ldots, C_k$ that minimizes
total relative cut cost$(C_1, \ldots, C_k)$

47

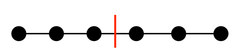
Simple example

- six objects
- similarity 1 if edge shown
- similarity 0 otherwise
- choice 1: 

cost UNDEFINED + 1/4

- choice 2: 

cost 1/1 + 1/3 = 4/3

- choice 3: 

cost 1/2 + 1/2 = 1 *prefer balance

48

Second cut-based measure: Conductance

- define:

  s_degree(C_p) = cutcost(C_p)+2*intracost(C_p)

  - model as graph, similarity = edge weights
  - s_degree is sum over all vertices in C_p of weights of edges touching vertex

- conductance (C_p) =

$$\frac{\text{cutcost}(C_p)}{\min\{\text{s\_degree}(C_p), \text{s\_degree}(V-C_p)\}}$$

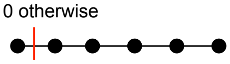
Optimization using conductance

- Choices:
  - minimize $\sum^k_{p=1}$ conductance (C_p)
  - minimize $\text{MAX}^k_{p=1}$ conductance (C_p)

- Observations
  - conductance (C_p) = conductance ($V-C_p$)
  - Finding a cut (C, V-C) with minimum conductance is NP-hard

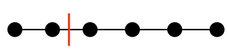
Simple example

- six objects
- similarity 1 if edge shown
- similarity 0 otherwise
- choice 1:



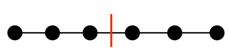
conductance 1/min(1,9) = 1

- choice 2:



conductance 1/min(3, 7) = 1/3

- choice 3:



conductance1/min(5, 5) = 1/5 *prefer balance

Hierarchical divisive revisited

- can use one of cut-based algorithms to split a cluster
- how choose cluster to split next?
  - if building entire tree, doesn't matter
  - if stopping a certain point, choose next cluster based on measure optimizing
    - e.g. for total relative cut cost, choose C_i with largest cutcost(C_i) / intracost(C_i)

Divisive Algorithm: Iterative Improvement; no hierarchy

1. Choose initial partition $C_1, \ldots, C_k$
2. repeat {

    unlock all vertices

    repeat {

        choose some C_i at random

        choose an unlocked vertex v_j in C_i

        move v_j to that cluster, if any, such that move gives maximum decrease in cost

        lock vertex v_j

    } until all vertices locked

}until converge

Observations on algorithm

- heuristic
- uses randomness
- convergence usually improvement < some chosen threshold between outer loop iterations
- vertex "locking" insures that all vertices are examined before examining any vertex twice
- there are many variations of algorithm
- can use at each division of hierarchical divisive algorithm with k=2
  - more computation than an agglomerative merge

Compare to k-means

- Similarities:
  - number of clusters, k, is chosen in advance
  - an initial clustering is chosen (possibly at random)
  - iterative improvement is used to improve clustering
- Important difference:
  - divisive algorithm can minimize a cut-based cost
    - total relative cut cost, conductance use external and internal measures
  - k-means maximizes only similarity within a cluster
    - ignores cost of cuts

Eigenvalues and clustering

General class of techniques for clustering a graph using eigenvectors of adjacency matrix (or similar matrix) called

Spectral clustering

First described in 1973

spectrum of a graph is list of eigenvalues, with multiplicity, of its adjacency matrix

Spectral clustering: *brief* overview

Given: k: number of clusters
nxn object-object sim. matrix S of non-neg. val.s

Compute:

1. Derive matrix L from S (straightforward computation)
   - e.g. Laplacian L=I-E, are variations in def.
2. find eigenvectors corresp. to k smallest eigenval.s of L
3. use eigenvectors to define clusters
   - variety of ways to do this
   - all involve another, simpler, clustering
     - e.g. points on a line

Spectral clustering optimizes a cut measure
similar to total relative cut cost

HITS and clustering

Recall HITS matrix formulation:

$\boldsymbol{a} = E^T\boldsymbol{h}$ $\quad$ $\boldsymbol{a} = E^TE\boldsymbol{a}$

$\boldsymbol{h} = E\boldsymbol{a}$ $\quad$ $\boldsymbol{h} = EE^T\boldsymbol{h}$

for adjacency matrix E, authority vector **a**, hub vector **h**

- a is the eigenvector corresponding to the eigenvalue 1 for E^TE

HITS and clustering

- Non-principal eigenvectors of EE^T and E^TE have positive and negative component values
  - Denote $a_{e2}, a_{e3}, \ldots$
    matching $h_{e2}, h_{e3}, \ldots$
  - E is adjacency matrix
- For a matched pair of eigenvectors $\boldsymbol{a}_{ej}$ and $\boldsymbol{h}_{ej}$
  - Denote k^{th} component of j^{th} pair: $\boldsymbol{a}_{ej}(k)$ and $\boldsymbol{h}_{ej}(k)$
  - Make a "community" of size *c* (chosen constant):
    - Choose c pages with most positive $\boldsymbol{h}_{ej}(k)$ - hubs
    - Choose c pages with most positive $\boldsymbol{a}_{ej}(k)$ - authorities
  - Make another "community" of size *c*:
    - Choose c pages with most negative $\boldsymbol{h}_{ej}(k)$ - hubs
    - Choose c pages with most negative $\boldsymbol{a}_{ej}(k)$ - authorities

Comparing clusterings

- Define external measure to
  - comparing two clusterings as to similarity
  - if one clustering "correct", one clustering by an algorithm, measures how well algorithm doing
    - refer to "correct" clusters as classes
      - "gold standard"
    - refer to computed clusters as clusters
- External measure independent of cost function optimized by algorithm

One measure: motivated by F-score in IR

- Given:
  - a set of classes $S_1, \ldots S_k$ of the objects use to define relevance
  - a computed clustering $C_1, \ldots C_k$ of the objects use to define retrieval
- Consider pairs of objects
  - pair in same class, call ***similar pair*** ≡ relevant
  - pair in different classes ≡ irrelevant
  - pair in same clusters ≡ retrieved
  - pair in different clusters ≡ not retrieved
- Use to define precision and recall

61

Clustering f-score

precision of the clustering w.r.t the gold standard =

$$\frac{\text{\# similar pairs in the same cluster}}{\text{\# pairs in the same cluster}}$$

recall of the clustering w.r.t the gold standard =

$$\frac{\text{\# similar pairs in the same cluster}}{\text{\# similar pairs}}$$

f-score of the clustering w.r.t the gold standard =

$$\frac{2\text{*precision*recall}}{\text{precision + recall}}$$

62

Properties of cluster F-score

- always ≤ 1
- Perfect match computed clusters to classes gives F-score = 1
- Symmetric
  - Two clusterings $\{C_i\}$ and $\{K_j\}$, neither "gold standard"
  - treat $\{C_i\}$ as if are classes and compute F-score of $\{K_j\}$ w.r.t. $\{C_i\}$ = F-score$_{\{Ci\}}(\{K_j\})$
  - treat $\{K_j\}$ as if are classes and compute F-score of $\{C_i\}$ w.r.t. $\{K_j\}$ = F-score$_{\{Kj\}}(\{C_i\})$

$\Rightarrow$ F-score$_{\{Ci\}}(\{K_j\})$ = F-score$_{\{Kj\}}(\{C_i\})$

63

another related external measure Rand index

$$\frac{(\text{\# similar pairs in the same cluster} + \text{\# dissimilar pairs in the different clusters})}{N\,(N-1)/2}$$

percentage pairs that are correct

64

Clustering: wrap-up

- many applications
  - application determines similarity between objects
- menu of
  - cost functions to optimizes
  - similarity measures between clusters
  - types of algorithms
    - flat/hierarchical
    - constructive/iterative
  - algorithms within a type

65