http://www.youtube.com/watch?v=OR_-Y-eIlQo

Machine learning: Unsupervised learning

David Kauchak
cs160
Fall 2009

adapted from:
http://www.stanford.edu/class/cs276/handouts/lecture17-clustering.ppt

Machine learning code

- Weka
  - Java based
  - Tons of approaches
  - Good for playing with different approaches, but faster/better of individual approaches can be found
  - http://www.cs.waikato.ac.nz/ml/weka/
- SVMs
  - SVMLight (C-based… fast)
    - http://www.cs.cornell.edu/People/tj/svm_light/
  - LIBSVM (Java)
    - http://www.csie.ntu.edu.tw/~cjlin/libsvm/
- Others
  - many others out there… search for them
  - e.g. PyML (in Python, but I've never used it)

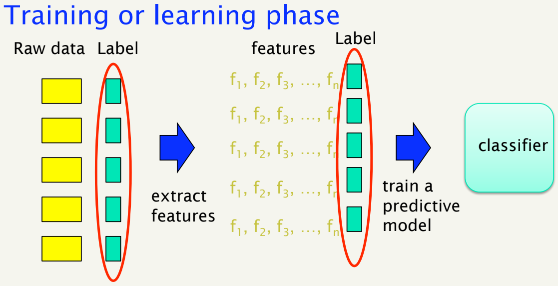
Administrative

- Keep up with the written homeworks
- Project proposals due this Friday
- Assignment 5 due next Wednesday!

- Will have assignment 4 back to you soon…
  - and literature reviews

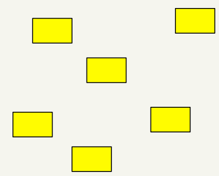
Supervised learning

Training or learning phase



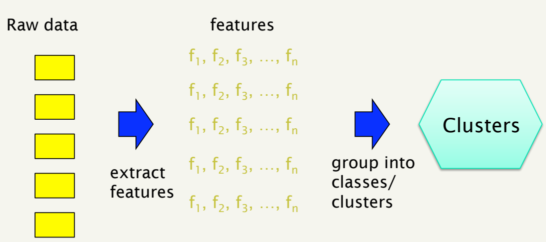
User "supervision", we're given the labels (classes)

Unsupervised learning



Given some example without labels, do something!

Unsupervised learning: clustering



No "supervision", we're only given data and want to find natural groupings

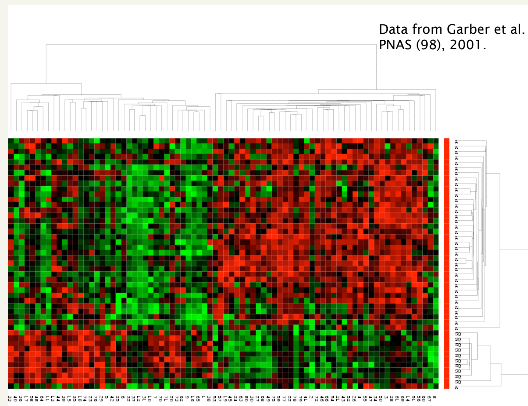
Unsupervised learning: modeling

- Most frequently, when people think of unsupervised learning they think clustering
- Another category: learning probabilities/parameters for models without supervision
  - Learn a translation dictionary
  - Learn an HMM from just sequences of data (i.e. not given the states)
  - Learn a grammar for a language
  - Learn the social graph

Clustering

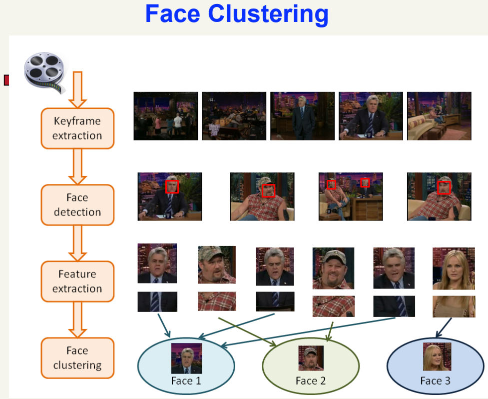
- Clustering: the process of grouping a set of objects into classes of similar objects
- Applications?

Gene expression data

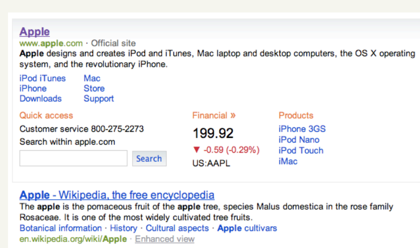


Data from Garber et al. PNAS (98), 2001.

Face Clustering

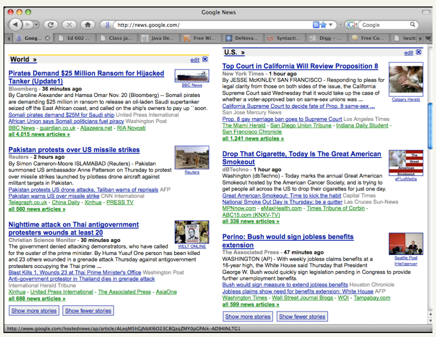


Applications of clustering in IR

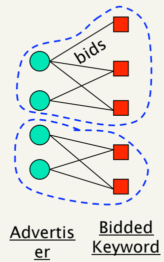


vary search results over clusters/ topics to improve recall

Google News: automatic clustering gives an effective news presentation metaphor



Clustering in search advertising

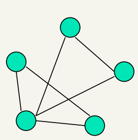


~10M nodes

- Find clusters of advertisers and keywords
  - Keyword suggestion
  - Performance estimation

14

Clustering applications



Who-messages-whom
IM graph

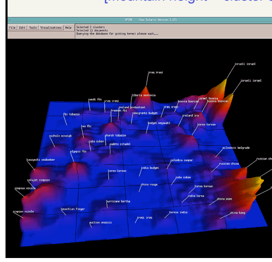
~100M
nodes

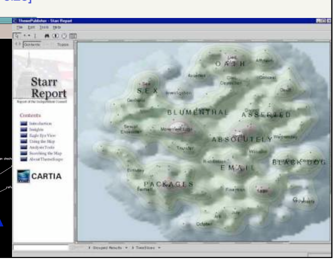
- Find clusters of users
  - Targeted advertising
  - Exploratory analysis
- Clusters of the Web Graph
  - Distributed pagerank computation

15

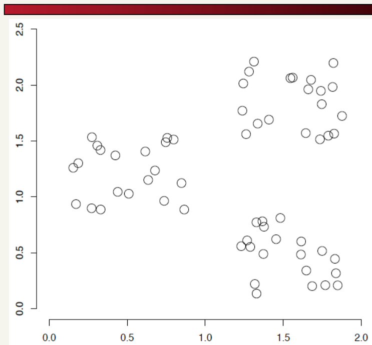
Data visualization

- Wise et al, "Visualizing the non-visual" PNNL
- ThemeScapes, Cartia
  - [Mountain height = cluster size]





A data set with clear cluster structure



What are some of the issues for clustering?

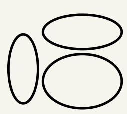
What clustering algorithms have you seen/used?

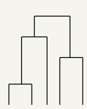
Issues for clustering

- Representation for clustering
  - How do we represent an example
    - features, etc.
  - Similarity/distance between examples
- Flat clustering or hierarchical
- Number of clusters
  - Fixed a priori
  - Data driven?

Clustering Algorithms

- Flat algorithms
  - Usually start with a random (partial) partitioning
  - Refine it iteratively
    - *K* means clustering
    - Model based clustering
  - Spectral clustering
- Hierarchical algorithms
  - Bottom-up, agglomerative
  - Top-down, divisive





Hard vs. soft clustering

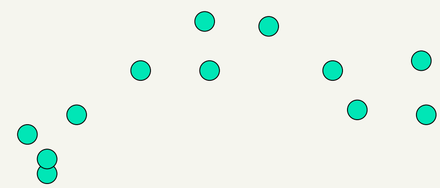
- Hard clustering: Each example belongs to exactly one cluster
- Soft clustering: An example can belong to more than one cluster (probabilistic)
  - Makes more sense for applications like creating browsable hierarchies
  - You may want to put a pair of sneakers in two clusters: (i) sports apparel and (ii) shoes

K-Means

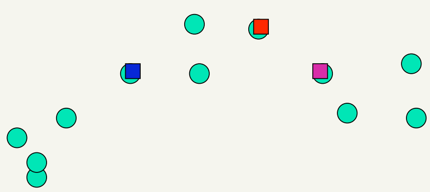
- Most well-known and popular clustering algorithm
- Start with some initial cluster centers
- Iterate:
  - Assign/cluster each example to closest center
  - Recalculate centers as the mean of the points in a cluster, *c*:

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

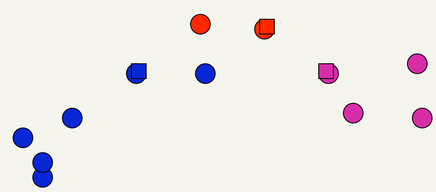
K-means: an example



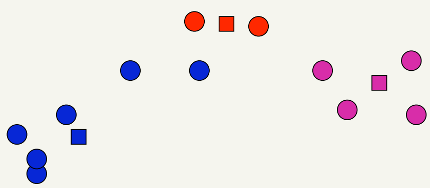
K-means: Initialize centers randomly



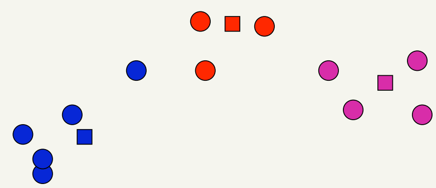
K-means: assign points to nearest center



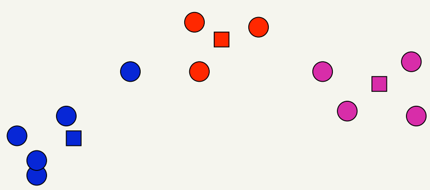
K-means: readjust centers



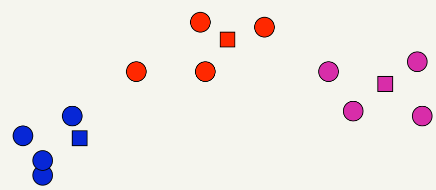
K-means: assign points to nearest center



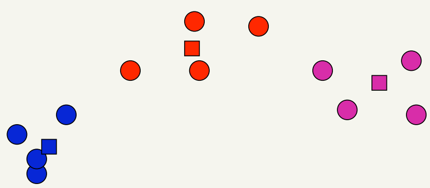
K-means: readjust centers



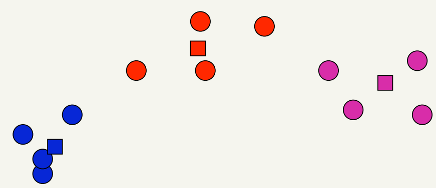
K-means: assign points to nearest center



K-means: readjust centers



K-means: assign points to nearest center

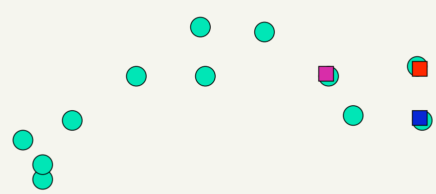


No changes: Done

K-means variations/parameters

- Initial (seed) cluster centers
- Convergence
  - A fixed number of iterations
  - partitions unchanged
  - Cluster centers don't change
- K

K-means: Initialize centers randomly



What would happen here?

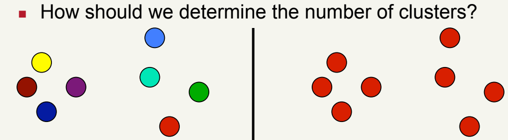
Seed selection ideas?

Seed Choice

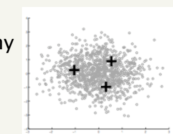
- Results can vary drastically based on random seed selection
- Some seeds can result in poor convergence rate, or convergence to sub-optimal clusterings
- Common heuristics
  - Random centers in the space
  - Randomly pick examples
  - Points least similar to any existing center
  - Try out multiple starting points
  - Initialize with the results of another clustering method

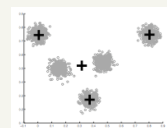
How Many Clusters?

- Number of clusters K must be provided
- Somewhat application dependent
- How should we determine the number of clusters?



too many



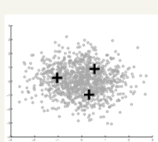


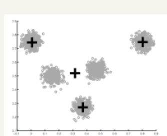
too few

One approach

- Assume data is Gaussian (i.e. spherical)

- Test for this
  - Testing in high dimensions doesn't work well
  - Testing in lower dimensions does work well

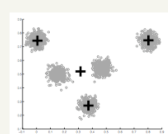




ideas?

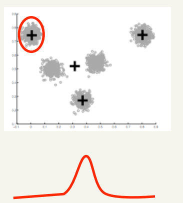
Project to one dimension and check

- For each cluster, project down to one dimension
  - Use a statistical test to see if the data is Gaussian



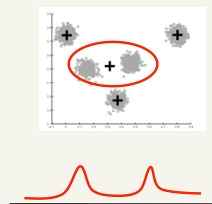
Project to one dimension and check

- For each cluster, project down to one dimension
  - Use a statistical test to see if the data is Gaussian



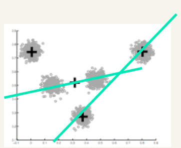
Project to one dimension and check

- For each cluster, project down to one dimension
  - Use a statistical test to see if the data is Gaussian



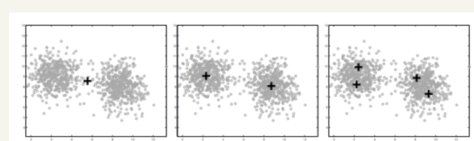
Project to one dimension and check

- For each cluster, project down to one dimension
  - Use a statistical test to see if the data is Gaussian



The dimension of the projection is based on the data

On synthetic data



Split too far

Compared to other approaches

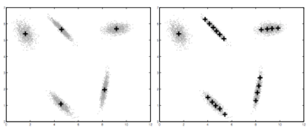


Figure 4: 2-*d* synthetic dataset with 5 true clusters. On the left, G-means correctly chooses 5 centers and deals well with non-spherical data. On the right, the BIC causes *X*-means to overfit the data, choosing 20 unevenly distributed clusters.

http://cs.baylor.edu/~hamerly/papers/nips_03.pdf

K-Means time complexity

- Variables: *K* clusters, *n* data points, *m* features/dimensions, *I* iterations
- What is the runtime complexity?
  - Computing distance between two points
  - Reassigning clusters
  - Computing new centers
  - Iterate…

K-Means time complexity

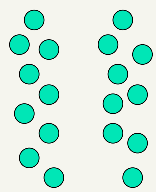
- Variables: *K* clusters, *n* data points, *m* features/dimensions, *I* iterations
- What is the runtime complexity?
  - Computing distance between two points is O*(m)* where *m* is the dimensionality of the vectors.
  - Reassigning clusters: O*(Kn)* distance computations, or O*(Knm)*
  - Computing centroids: Each points gets added once to some centroid: O*(nm)*
  - Assume these two steps are each done once for *I* iterations: O*(Iknm)*

In practice, K-means converges quickly and is fairly fast

Problems with K-means

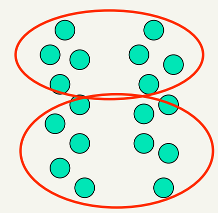
- Determining K is challenging
- Spherical assumption about the data (distance to cluster center)
- Hard clustering isn't always right
- Greedy approach

Problems with K-means



What would K-means give us here?

Assumes spherical clusters

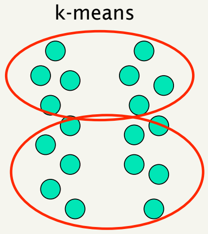


k-means

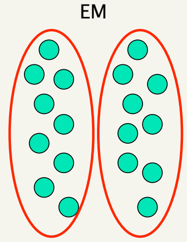
EM clustering: mixtures of Gaussians

Assume data came from a mixture of Gaussians (**elliptical data**), assign data to cluster with a certain *probability*

k-means



EM



EM is a general framework

- Create an initial model, θ'
  - Arbitrarily, randomly, or with a small set of training examples
- Use the model θ' to obtain another model θ such that

  $\sum_i \log P_{\theta}(data_i) > \sum_i \log P_{\theta'}(data_i)$ i.e. better models data (increased log likelihood)
- Let $\theta' = \theta$ and repeat the above step until reaching a local maximum
  - Guaranteed to find a better model after each iteration

Where else have you seen EM?

EM shows up all over the place

- Training HMMs (Baum-Welch algorithm)
- Learning probabilities for Bayesian networks
- EM-clustering
- Learning word alignments for language translation
- Learning Twitter friend network
- Genetics
- Finance
- Anytime you have a model and unlabeled data!

E and M steps: creating a better model

Expectation: Given the current model, figure out the expected probabilities of the data points to each cluster

$p(x|\theta_c)$ What is the probability of each point belonging to each cluster?

Maximization: Given the probabilistic assignment of all the points, estimate a new model, θ_c

Just like NB maximum likelihood estimation, except we use fractional counts instead of whole counts

Similar to *K*-Means

- Iterate:
  - Assign/cluster each point to closest center

    Expectation: Given the current model, figure out the expected probabilities of the points to each cluster $p(x|\theta_c)$

  - Recalculate centers as the mean of the points in a cluster

    Maximization: Given the probabilistic assignment of all the points, estimate a new model, θ_c

E and M steps

Expectation: Given the current model, figure out the expected probabilities of the data points to each cluster

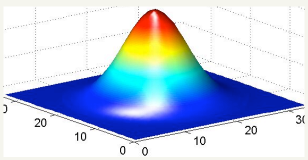
Maximization: Given the probabilistic assignment of all the points, estimate a new model, θ_c

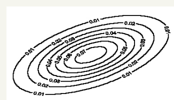
Iterate:

each iterations increases the likelihood of the data and guaranteed to converge (though to a local optimum)!

Model: mixture of Gaussians

$$N[x;\mu,\Sigma] = \frac{1}{(2\pi)^{d/2}\sqrt{\det(\Sigma)}}\exp[-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)]$$





Covariance determines the shape of these contours

- Fit these Gaussian densities to the data, one per cluster

EM example

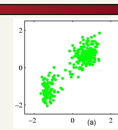


Figure from Chris Bishop

EM example

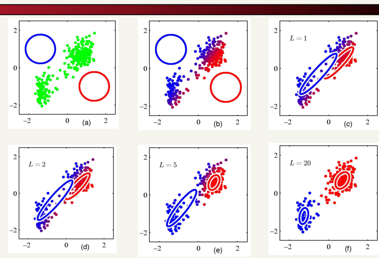


Figure from Chris Bishop

EM

- EM is a general purpose approach for training a model when you don't have labels
- Not just for clustering!
  - K-means is just for clustering
- One of the most general purpose unsupervised approaches
  - can be hard to get right!

Finding Word Alignments

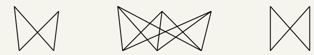
… la maison … la maison bleue … la fleur …

… the house … the blue house … the flower …

- In machine translation, we train from pairs of translated sentences
- Often useful to know how the words align in the sentences
- Use EM!
    - learn a model of P(french-word | english-word)

Finding Word Alignments

… la maison … la maison bleue … la fleur …



… the house … the blue house … the flower …

All word alignments equally likely

All P(french-word | english-word) equally likely

Finding Word Alignments

… la maison … la maison bleue … la fleur …

… the house … the blue house … the flower …

“la” and “the” observed to co-occur frequently, so P(la | the) is increased.

Finding Word Alignments

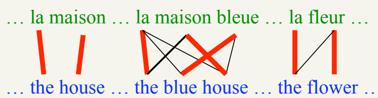
… la maison … la maison bleue … la fleur …

… the house … the blue house … the flower …

“house” co-occurs with both “la” and “maison”, but P(maison | house) can be raised without limit, to 1.0, while P(la | house) is limited because of “the”

(pigeonhole principle)

Finding Word Alignments



settling down after another iteration

Finding Word Alignments

Inherent hidden structure revealed by EM training!

For details, see

- "A Statistical MT Tutorial Workbook" (Knight, 1999).
  - 37 easy sections, final section promises a free beer.
- "The Mathematics of Statistical Machine Translation" (Brown et al, 1993)
- Software: GIZA++

Statistical Machine Translation

P(maison | house) = 0.411
P(maison | building) = 0.027
P(maison | manson) = 0.020
…

Estimating the model from training data

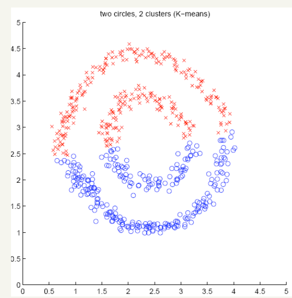
Discussion

- How does an infant learning to understand and speak language fit into our model?
- How about learning to play tennis?
- How are these different/similar?

Other clustering algorithms

- K-means and EM-clustering are by far the most popular for clustering
- However, they can't handle all clustering tasks
- What types of clustering problems can't they handle?

Non-gaussian data



What is the problem?

Similar to classification: global decision vs. local decision

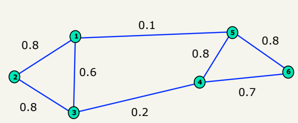
Spectral clustering

Similarity Graph

- Represent dataset as a weighted graph $G(V,E)$
- Data points: $\{x_1, x_2, \ldots, x_6\}$

$V=\{x_i\}$ Set of n vertices representing points

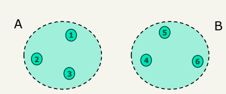
$E=\{w_{ij}\}$ Set of weighted edges indicating pair-wise similarity between points



What does clustering represent?

Graph Partitioning

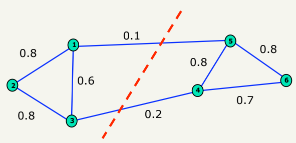
- Clustering can be viewed as partitioning a similarity graph
- *Bi-partitioning* task:
  - Divide vertices into two disjoint groups (A,B)



What would define a good partition?

Clustering Objectives

- Traditional definition of a "good" clustering:
  1. within cluster should be highly similar.
  2. between different clusters should be highly dissimilar.
- Apply these objectives to our graph representation

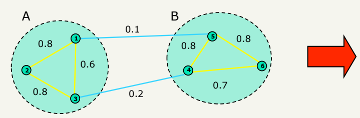


1. Maximise weight of within-group connections
2. Minimise weight of between-group connections

Graph Cuts

- Express partitioning objectives as a function of the "edge cut" of the partition.
- *Cut:* Set of edges with only one vertex in a group.

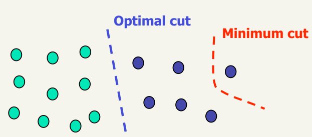
$$cut(A,B) = \sum_{i \in A, j \in B} w_{ij}$$



$cut(A,B) = 0.3$

Can we use the minimum cut?

Graph Cut Criteria



- **Problem:**
  - **Only considers external cluster connections**
  - **Does not consider internal cluster density**

Graph Cut Criteria

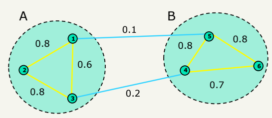
- **Criterion: Normalised-cut** (Shi & Malik,'97)
  - Consider the connectivity between groups relative to the density of each group:

$$\min Ncut(A,B) = \frac{cut(A,B)}{vol(A)} + \frac{cut(A,B)}{vol(B)}$$

  - Normalize the association between groups by *volume*
    - $Vol(A)$: The total weight of the edges originating from group A

Why does this work?

Normalized cut



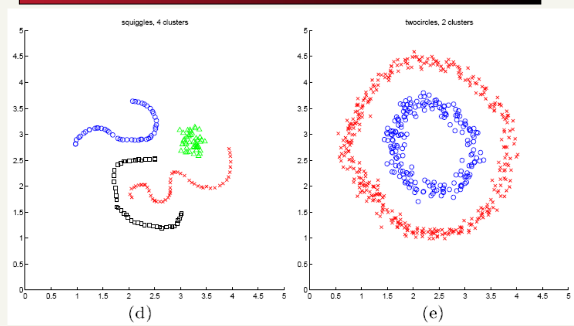
- *Vol(A)*: The total weight of the edges originating from group *A*

minimize between group connections

$$\min Ncut(A,B) = \frac{cut(A,B)}{vol(A)} + \frac{cut(A,B)}{vol(B)}$$

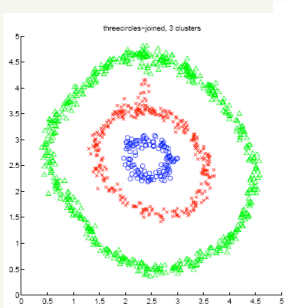
maximize within group connections

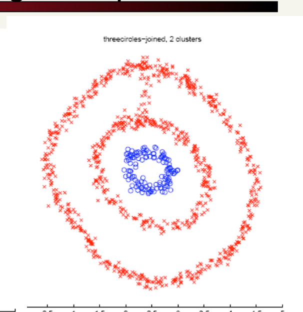
Spectral clustering examples



Ng et al On Spectral clustering: analysis and algorithm

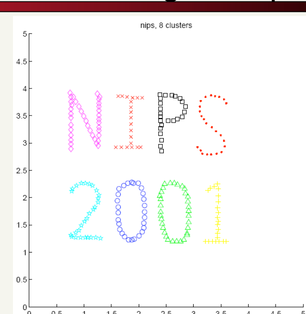
Spectral clustering examples





Ng et al On Spectral clustering: analysis and algorithm

Spectral clustering examples



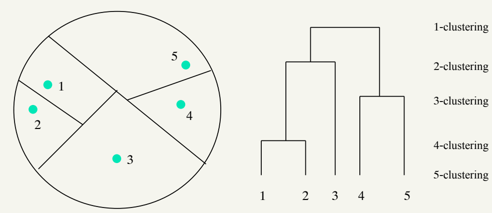
Ng et al On Spectral clustering: analysis and algorithm

Hierarchical clustering

- We didn't cover this, but if you're interested, take a look through these slides for the common approaches

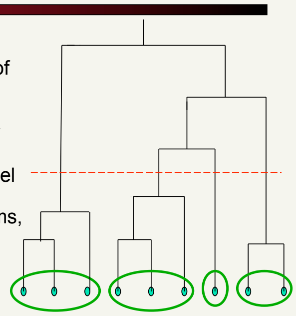
Hierarchical Clustering

Recursive partitioning/merging of a data set



Dendogram

- Represents all partitionings of the data
- We can get a K clustering by looking at the **connected** components at any given level
- Frequently binary dendograms, but n-ary dendograms are generally easy to obtain with minor changes to the algorithms



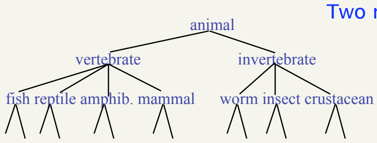
Advantages of hierarchical clustering

- Don't need to specify the number of clusters
- Good for data visualization
  - See how the data points interact at many levels
  - Can view the data at multiple levels of granularity
  - Understand how all points interact
- Specifies all of the K clusterings/partitions

Hierarchical Clustering

- Common in many domains
  - Biologists and social scientists
  - Gene expression data
  - Document/web page organization
    - DMOZ
    - Yahoo directories

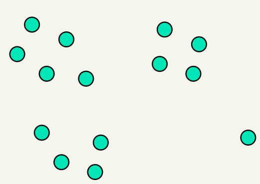
Two main approaches…



Divisive hierarchical clustering

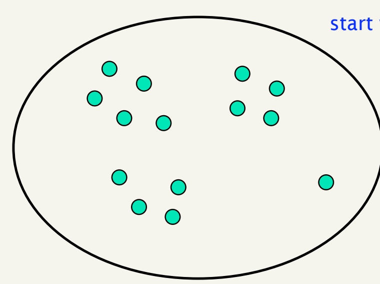
- Finding the best partitioning of the data is generally exponential in time
- Common approach:
  - Let **C** be a set of clusters
  - Initialize **C** to be the one-clustering of the data
  - While there exists a cluster *c* in **C**
    - remove *c* from **C**
    - partition *c* into 2 clusters using a flat clustering algorithm, c_1 and c_2
    - Add to c_1 and c_2 **C**
- Bisecting k-means

Divisive clustering



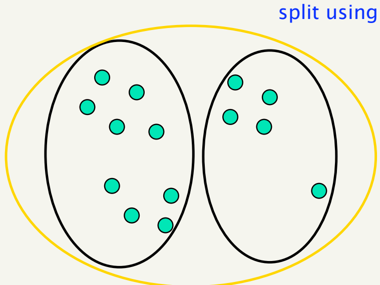
Divisive clustering

start with one cluster

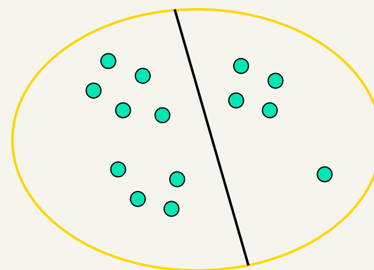


Divisive clustering

split using flat clustering

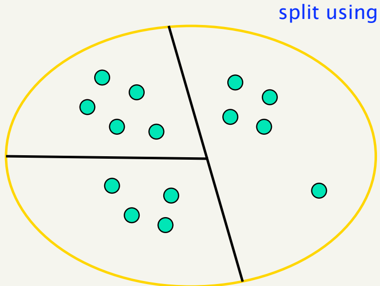


Divisive clustering



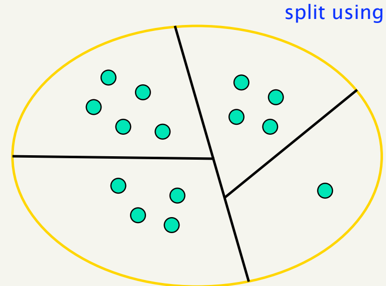
Divisive clustering

split using flat clustering

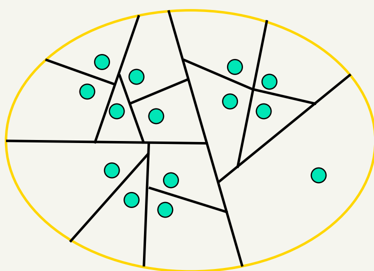


Divisive clustering

split using flat clustering



Divisive clustering



Note, there is a “temporal” component not seen here

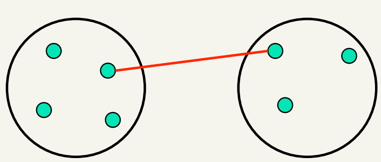
Hierarchical Agglomerative Clustering (HAC)

- Let **C** be a set of clusters
- Initialize **C** to be all points/docs as separate clusters
- While **C** contains more than one cluster
  - find c_1 and c_2 in **C** that are closest together
  - remove c_1 and c_2 from **C**
  - merge c_1 and c_2 and add resulting cluster to **C**
- The history of merging forms a binary tree or hierarchy

- How do we measure the distance between clusters?

Distance between clusters

- **Single-link**
  - Similarity of the *most* similar (single-link)

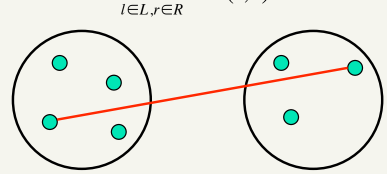


$$\max_{l \in L, r \in R} sim(l,r)$$

Distance between clusters

- **Complete-link**
  - Similarity of the “furthest” points, the *least* similar

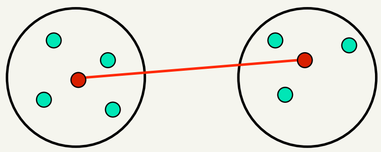
$$\min_{l \in L, r \in R} sim(l,r)$$



Why are these “local” methods used? efficiency

Distance between clusters

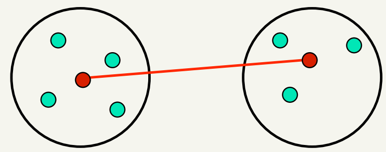
- **Centroid**
  - Clusters whose centroids (centers of gravity) are the most similar



$$\|\mu(L)-\mu(R)\|^2$$

Distance between clusters

- **Centroid**
  - Clusters whose centroids (centers of gravity) are the most similar

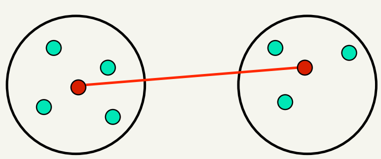


$$\frac{|L|\cdot|R|}{|L|+|R|}\|\mu(L)-\mu(R)\|^2$$ Ward's method

What does this do?

Distance between clusters

- **Centroid**
  - Clusters whose centroids (centers of gravity) are the most similar

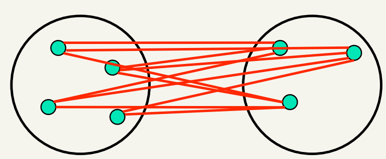


$$\frac{|L|\cdot|R|}{|L|+|R|}\|\mu(L)-\mu(R)\|^2$$ Ward's method

Encourages similar sized clusters

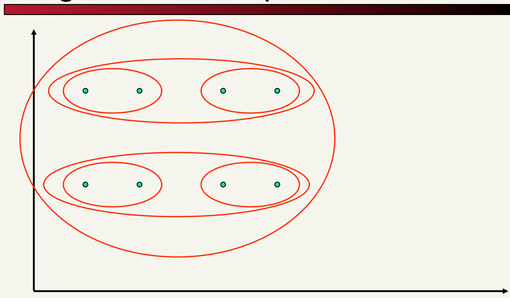
Distance between clusters

- **Average-link**
  - Average similarity between all pairs of elements

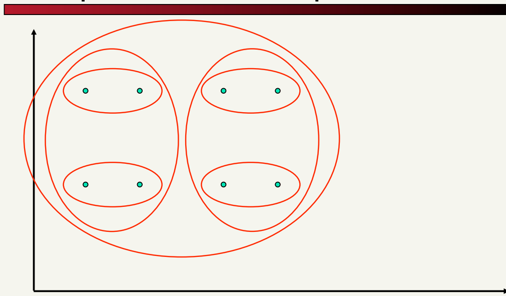


$$\frac{1}{|L|\cdot|R|}\sum_{x\in L, y\in R}\|x-y\|^2$$

Single Link Example

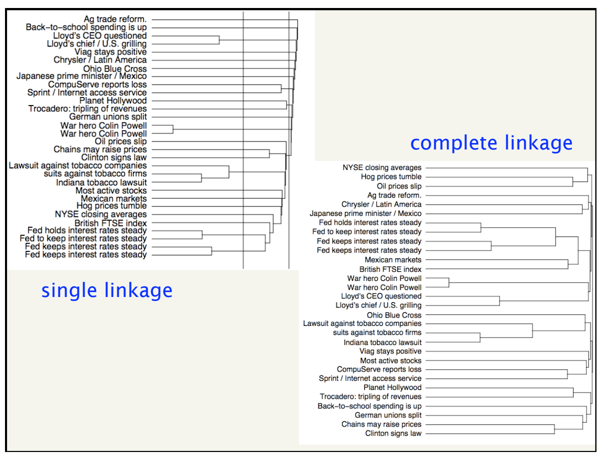


Complete Link Example



Computational Complexity

- For
  - *m* dimensions
  - *n* documents/points
- How many iterations?
  - $n-1$ iterations
- First iteration
  - Need to compute similarity of all pairs of *n* points/documents: $O(n^2m)$
- Remaining n-2 iterations
  - compute the distance between the most recently created cluster and all other existing clusters: $O(nm)$
  - Does depend on the cluster similarity approach
- Overall run-time: $O(n^2m)$ – generally slower than flat clustering!



Problems with hierarchical clustering

Problems with hierarchical clustering

- Locally greedy: once a merge decision has been made it cannot be changed

Single-linkage: chaining effect