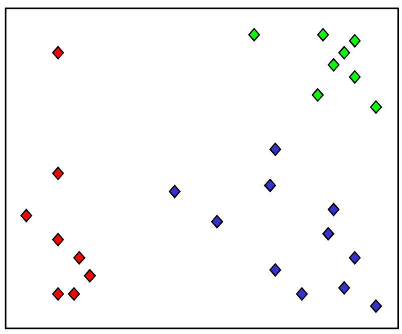
Clustering

15-381 Artificial Intelligence
Henry Lin

Modified from excellent slides of Eamonn Keogh, Ziv Bar-Joseph, and Andrew Moore

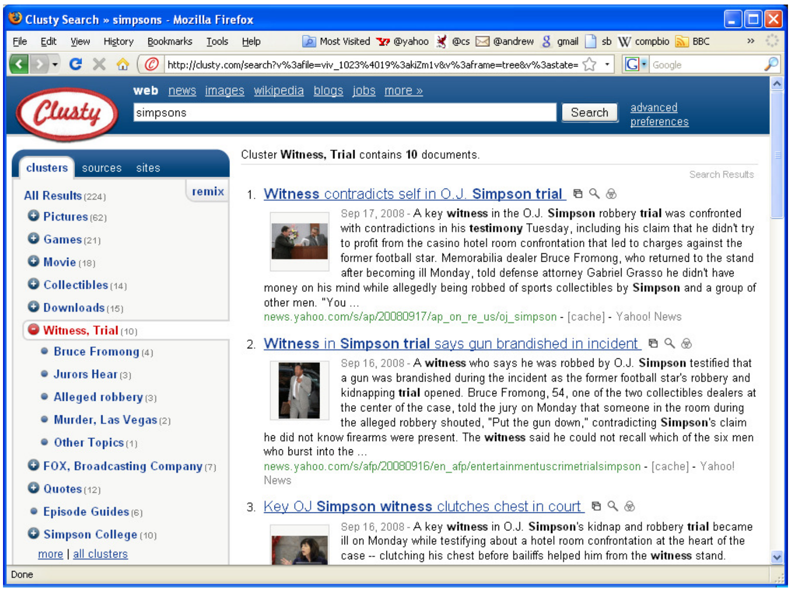
What is Clustering?

- Organizing data into *clusters* such that there is
  - high intra-cluster similarity
  - low inter-cluster similarity
- Informally, finding natural groupings among objects.



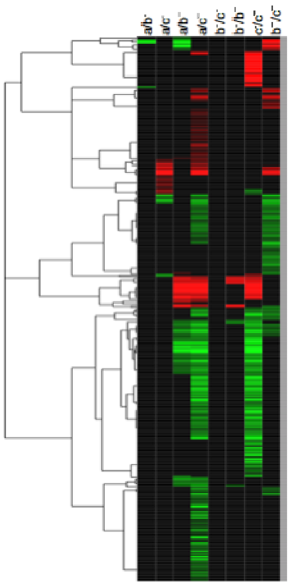
- Why do we want to do that?
- Any REAL application?

Example: clusty



Example: clustering genes

- Microarrays measures the activities of all genes in different conditions
- Clustering genes provide a lot of information about the genes
- An early "killer application" in this area
  - The most cited (7,812) paper in PNAS!



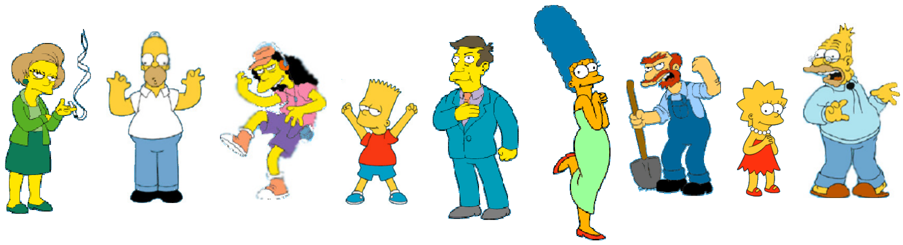
Why clustering?

- Organizing data into clusters shows internal structure of the data
  - Ex. Clusty and clustering genes above
- Sometimes the partitioning is the goal
  - Ex. Market segmentation
- Prepare for other AI techniques
  - Ex. Summarize news (cluster and then find centroid)
- Techniques for clustering is useful in knowledge discovery in data
  - Ex. Underlying rules, reoccurring patterns, topics, etc.

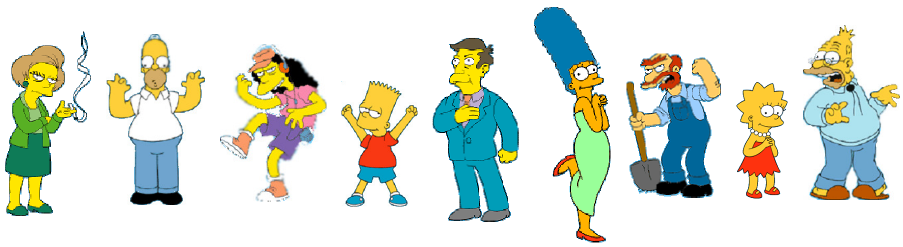
Outline

- Motivation
- Distance measure
- Hierarchical clustering
- Partitional clustering
  - K-means
  - Gaussian Mixture Models
  - Number of clusters

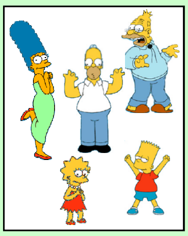
What is a natural grouping among these objects?



What is a natural grouping among these objects?

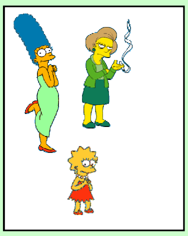


Clustering is subjective



Simpson's Family

School Employees



Females

Males

What is Similarity?

The quality or state of being similar; likeness; resemblance; as, a similarity of features.

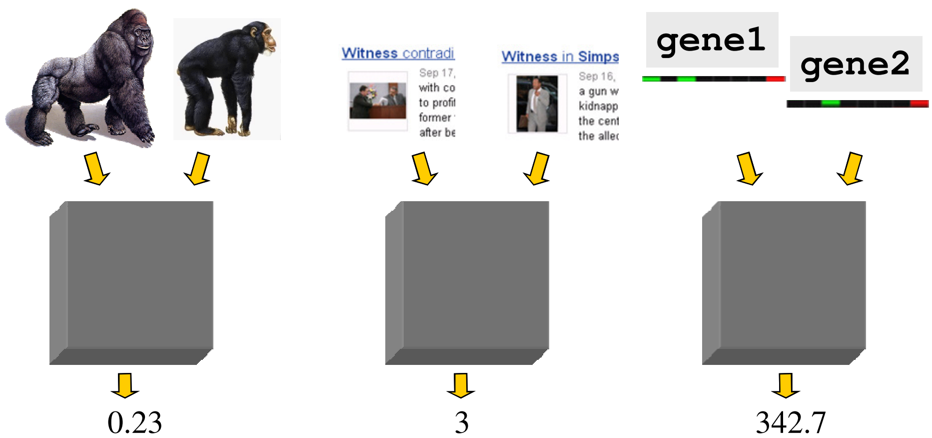
Webster's Dictionary

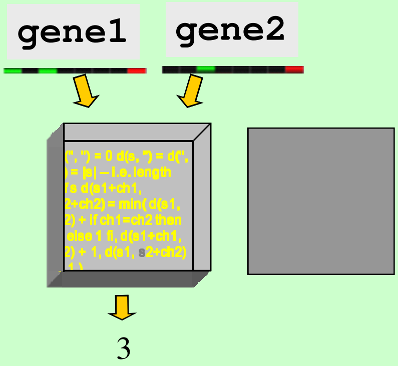
Similarity is hard to define, but… "*We know it when we see it*"

The real meaning of similarity is a philosophical question. We will take a more pragmatic approach.

Defining Distance Measures

Definition: Let O_1 and O_2 be two objects from the universe of possible objects. The distance (dissimilarity) between O_1 and O_2 is a real number denoted by $D(O_1,O_2)$





Inside these black boxes: some function on two variables (might be simple or very complex)

What properties should a distance measure have?

- $D(\mathrm{A,B}) = D(\mathrm{B,A})$ *Symmetry*
- $D(\mathrm{A,B}) = 0$ iif A = B *Constancy of Self-Similarity*
- $D(\mathrm{A,B}) \geq 0$ *Positivity*
- $D(\mathrm{A,B}) \leq D(\mathrm{A,C}) + D(\mathrm{B,C})$ *Triangular Inequality*

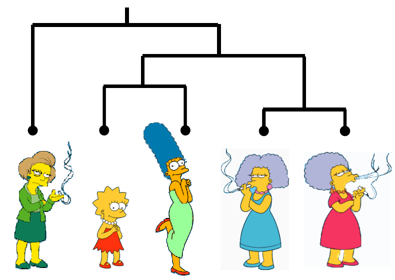
Outline

- Motivation
- Distance measure
- Hierarchical clustering
- Partitional clustering
  - K-means
  - Gaussian Mixture Models
  - Number of clusters

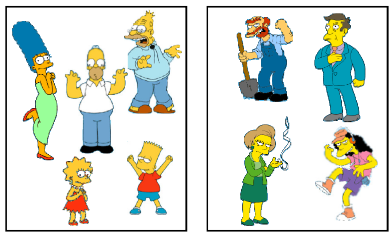
Two Types of Clustering

• **Partitional algorithms:** Construct various partitions and then evaluate them by some criterion (we will see an example called BIRCH)
• **Hierarchical algorithms:** Create a hierarchical decomposition of the set of objects using some criterion

Hierarchical



Partitional



Desirable Properties of a Clustering Algorithm

• Scalability (in terms of both time and space)

• Ability to deal with different data types

• Minimal requirements for domain knowledge to determine input parameters

• Able to deal with noise and outliers

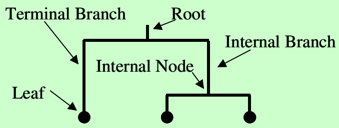
• Insensitive to order of input records

• Incorporation of user-specified constraints

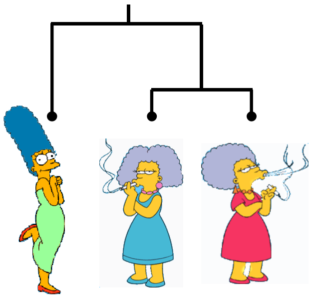
• Interpretability and usability

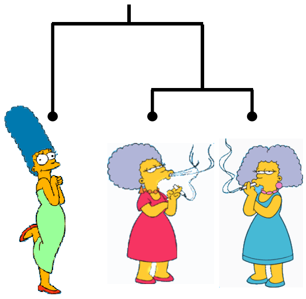
A Useful Tool for Summarizing Similarity Measurements

In order to better appreciate and evaluate the examples given in the early part of this talk, we will now introduce the *dendrogram*.



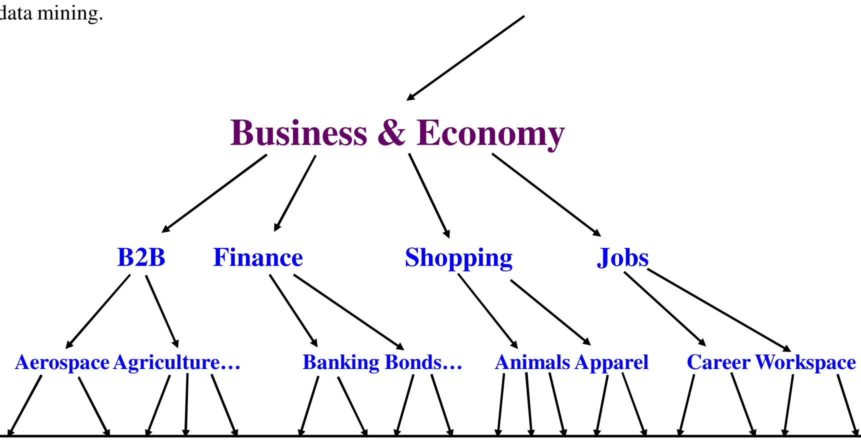
The similarity between two objects in a dendrogram is represented as the height of the lowest internal node they share.



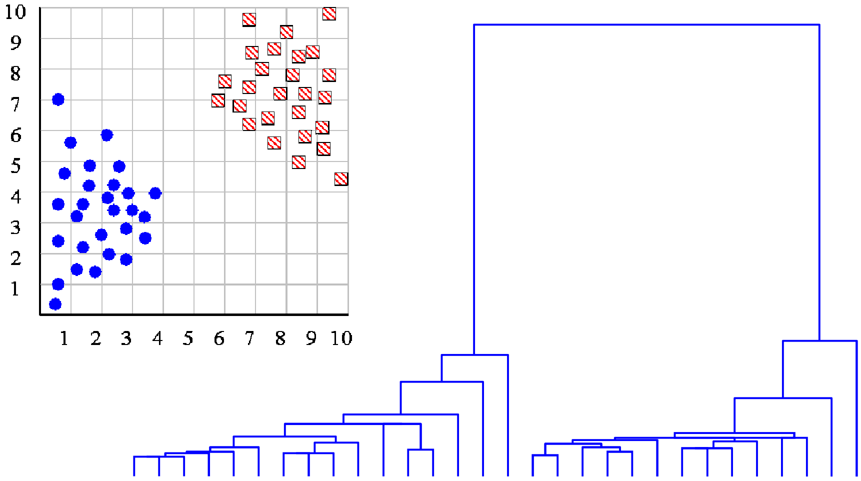


Note that hierarchies are commonly used to organize information, for example in a web portal.

Yahoo's hierarchy is manually created, we will focus on automatic creation of hierarchies in data mining.

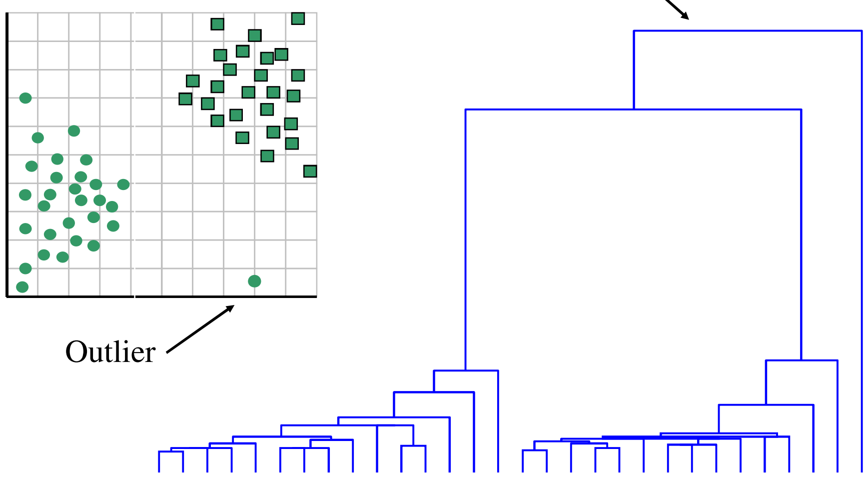


We can look at the dendrogram to determine the “correct” number of clusters. In this case, the two highly separated subtrees are highly suggestive of two clusters. (Things are rarely this clear cut, unfortunately)



One potential use of a dendrogram is to detect outliers

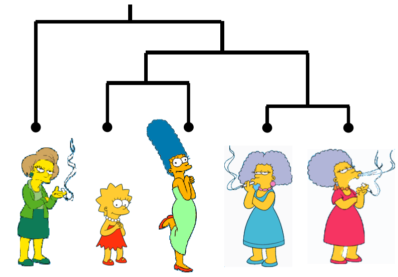
The single isolated branch is suggestive of a data point that is very different to all others



(How-to) Hierarchical Clustering

The number of dendrograms with n leafs $= (2n-3)!/[(2^{(n-2)})\,(n-2)!]$

Number of Leafs	Number of Possible Dendrograms
2	1
3	3
4	15
5	105
...	…
10	34,459,425

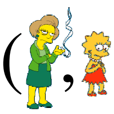


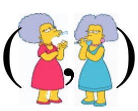
Since we cannot test all possible trees we will have to heuristic search of all possible trees. We could do this..

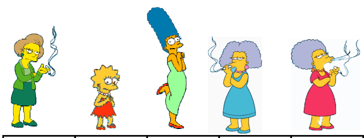
Bottom-Up (agglomerative): Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

Top-Down (divisive): Starting with all the data in a single cluster, consider every possible way to divide the cluster into two. Choose the best division and recursively operate on both sides.

We begin with a distance matrix which contains the distances between every pair of objects in our database.

D() = 8

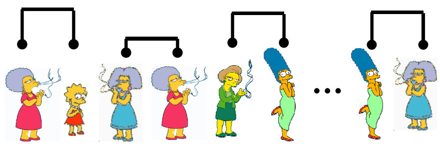
D() = 1



0	8	8	7	7
	0	2	4	4
		0	3	3
			0	1
				0

Bottom-Up (agglomerative):
Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

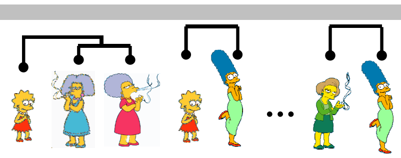
Consider all possible merges…



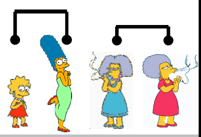
Choose the best

Bottom-Up (agglomerative):
Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

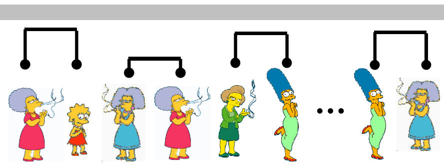
Consider all possible merges…



Choose the best



Consider all possible merges…

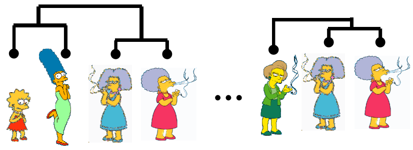


Choose the best

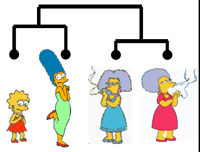
Bottom-Up (agglomerative):

Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

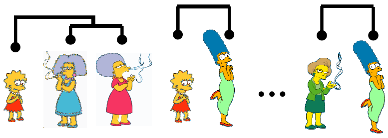
Consider all possible merges…



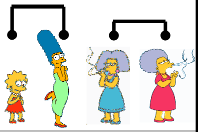
Choose the best



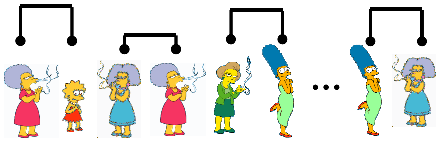
Consider all possible merges…



Choose the best



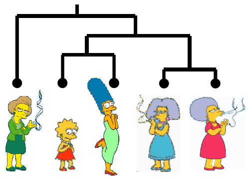
Consider all possible merges…



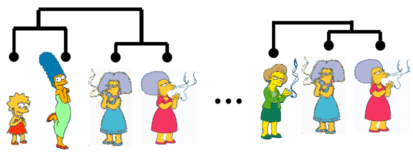
Choose the best

Bottom-Up (agglomerative):

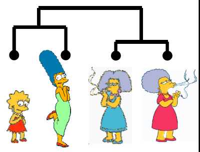
Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.



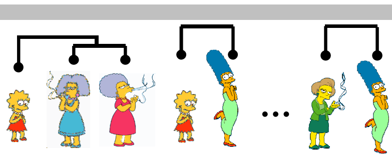
Consider all possible merges…



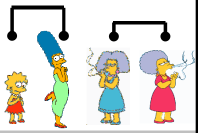
Choose the best



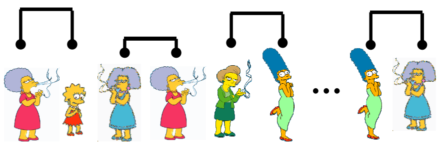
Consider all possible merges…



Choose the best



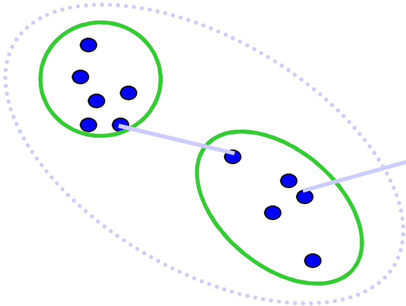
Consider all possible merges…



Choose the best

Similarity criteria: Single Link

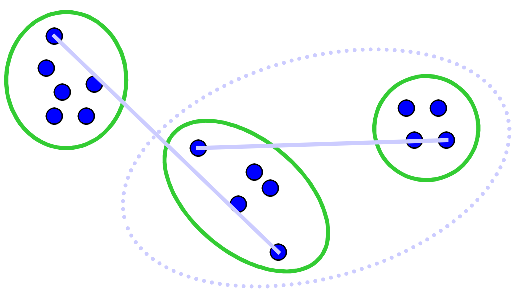
- cluster similarity = similarity of two most similar members



- Potentially long and skinny clusters

Hierarchical: Complete Link

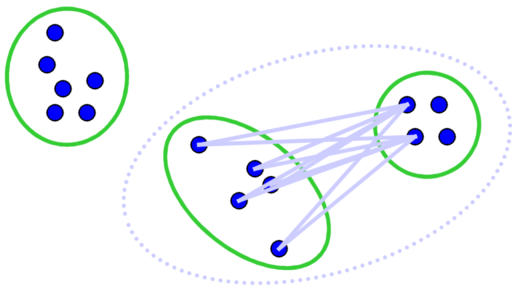
- cluster similarity = similarity of two least similar members



+ tight clusters

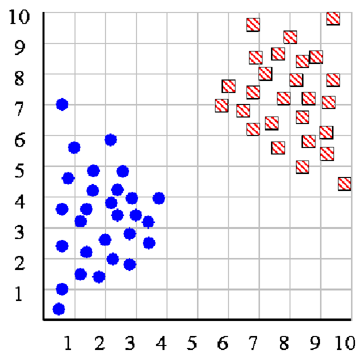
Hierarchical: Average Link

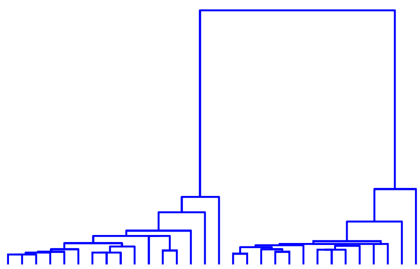
- cluster similarity = average similarity of all pairs



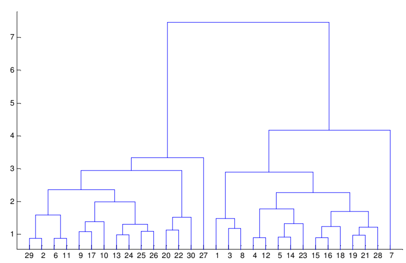
the most widely used similarity measure

Robust against noise





Single linkage



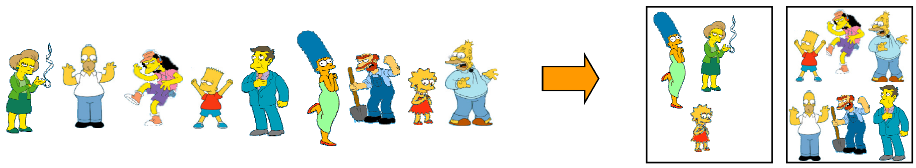
Average linkage

Summary of Hierarchal Clustering Methods

• No need to specify the number of clusters in advance.
• Hierarchical structure maps nicely onto human intuition for some domains
• They do not scale well: time complexity of at least $O(n^2)$, where n is the number of total objects.
• Like any heuristic search algorithms, local optima are a problem.
• Interpretation of results is (very) subjective.

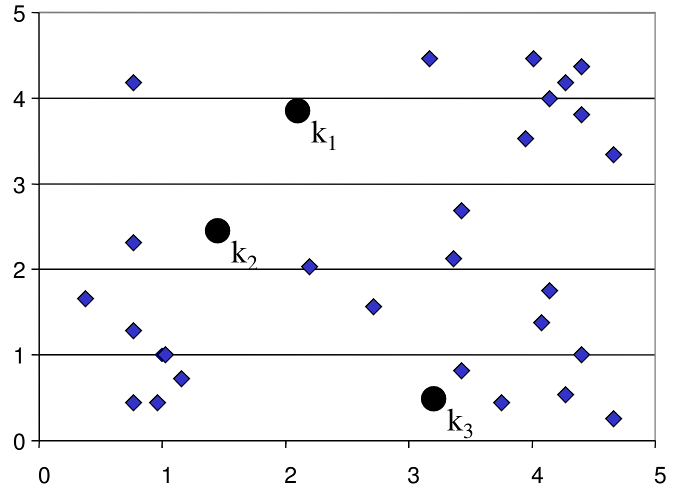
Partitional Clustering

- Nonhierarchical, each instance is placed in exactly one of K non-overlapping clusters.
- Since only one set of clusters is output, the user normally has to input the desired number of clusters K.



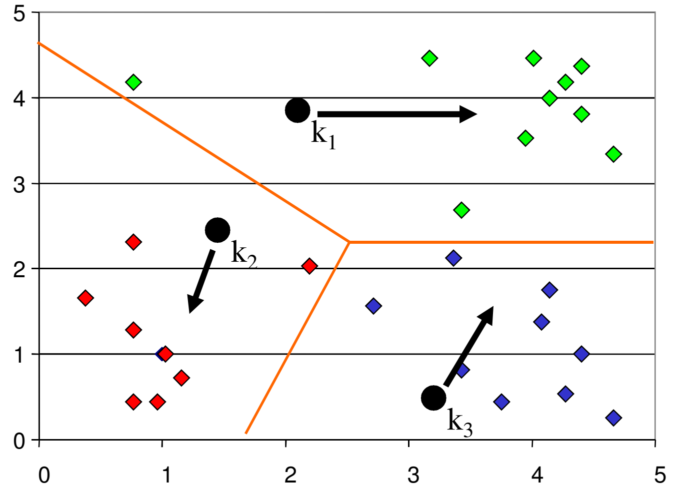
K-means Clustering: Initialization

Decide K, and initialize K centers (randomly)



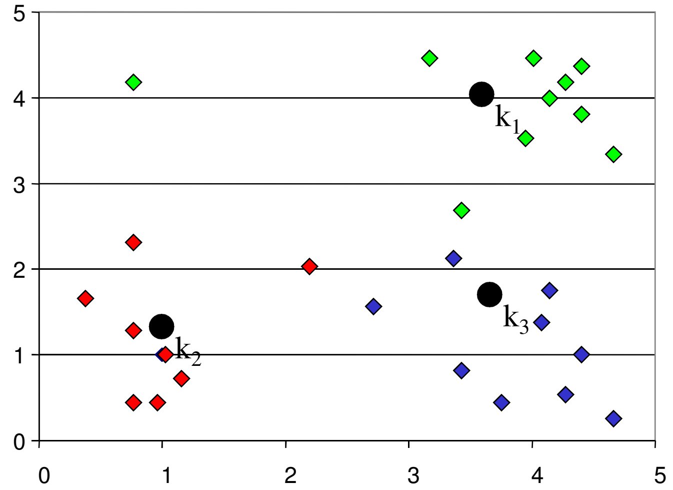
K-means Clustering: Iteration 1

Assign all objects to the nearest center.
Move a center to the mean of its members.



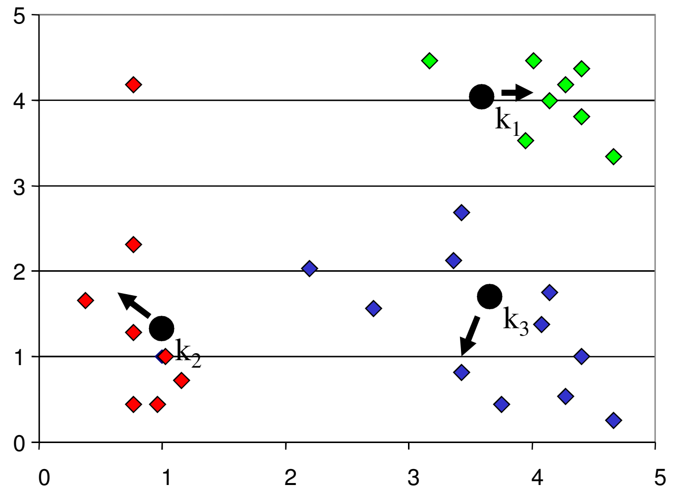
K-means Clustering: Iteration 2

After moving centers, re-assign the objects…



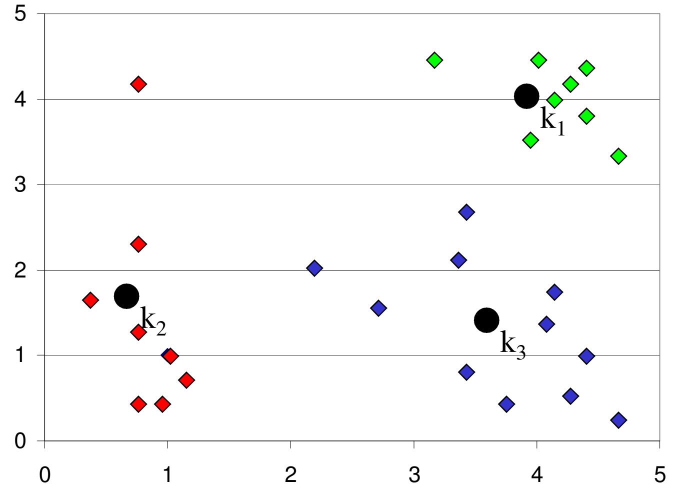
K-means Clustering: Iteration 2

After moving centers, re-assign the objects to nearest centers.
Move a center to the mean of its new members.



K-means Clustering: Finished!

Re-assign and move centers, until …
no objects changed membership.



Algorithm *k-means*

1. Decide on a value for *K*, the number of clusters.

2. Initialize the *K* cluster centers (randomly, if necessary).

3. Decide the class memberships of the *N* objects by assigning them to the nearest cluster center.

4. Re-estimate the *K* cluster centers, by assuming the memberships found above are correct.

5. Repeat 3 and 4 until none of the *N* objects changed membership in the last iteration.

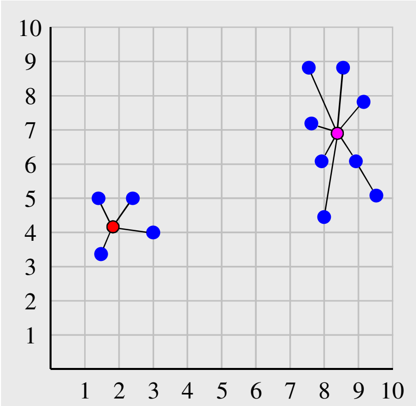
Why K-means Works

- What is a good partition?
- High intra-cluster similarity
- K-means optimizes
  - the average distance to members of the same cluster

$$\sum_{k=1}^{K} \frac{1}{n_k} \sum_{i=1}^{n_k} \sum_{j=1}^{n_k} \left\| x_{ki} - x_{kj} \right\|^2$$

  - which is twice the total distance to centers, also called squared error

$$se = \sum_{k=1}^{K} \sum_{i=1}^{n_k} \left\| x_{ki} - \mu_k \right\|^2$$



Comments on *K-Means*

- Strength
  - Simple, easy to implement and debug
  - Intuitive objective function: optimizes intra-cluster similarity
  - *Relatively efficient*: $O(tkn)$, where n is # objects, k is # clusters, and t is # iterations. Normally, $k, t \ll n$.
- Weakness
  - Applicable only when *mean* is defined, then what about categorical data?
  - Often terminates at a *local optimum*. Initialization is important.
  - Need to specify *K*, the *number* of clusters, in advance
  - Unable to handle noisy data and *outliers*
  - Not suitable to discover clusters with *non-convex shapes*
- Summary
  - Assign members based on current centers
  - Re-estimate centers based on current assignment

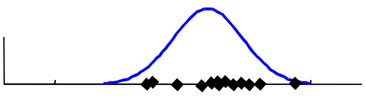
Outline

- Motivation
- Distance measure
- Hierarchical clustering
- Partitional clustering
  - K-means
  - Gaussian Mixture Models
  - Number of clusters

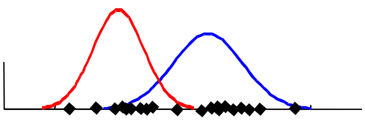
Gaussian Mixture Models

- Gaussian

$$P(x) = \varphi(x;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}}\exp(-\frac{(x-\mu)^2}{2\sigma^2})$$



  - ex. height of one population



- Gaussian Mixture

$$P(C=i) = \omega_i, \quad P(x \mid C=i) = \varphi(x;\mu_i,\sigma_i)$$

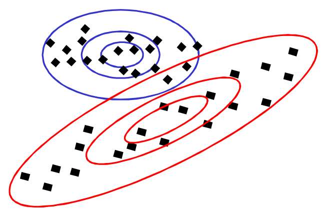
$$P(x) = \sum_{i=1}^{K} P(x, C=i) = \sum_{i=1}^{K} P(C=i)P(x \mid C=i) = \omega_i \varphi(x;\mu_i,\sigma_i)$$

  - ex. height of two population

Gaussian Mixture Models

- Mixture of Multivariate Gaussian

$$P(C=k)=\omega_i,\quad P(x\,|\,C=i)=\varphi(x;\mu_i,\Sigma_i)$$

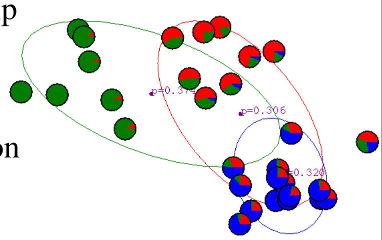


  - ex. y-axis is blood pressure and x-axis is age

GMM+EM = “Soft K-means”

- Decide the number of clusters, K
- Initialize parameters (randomly)
- E-step: assign *probabilistic* membership

$$p_{ij}=P(C=i|\mathbf{x}_j)=\alpha P(\mathbf{x}_j|C=i)P(C=i)$$
$$p_i=\textstyle\sum_j p_{ij}.$$

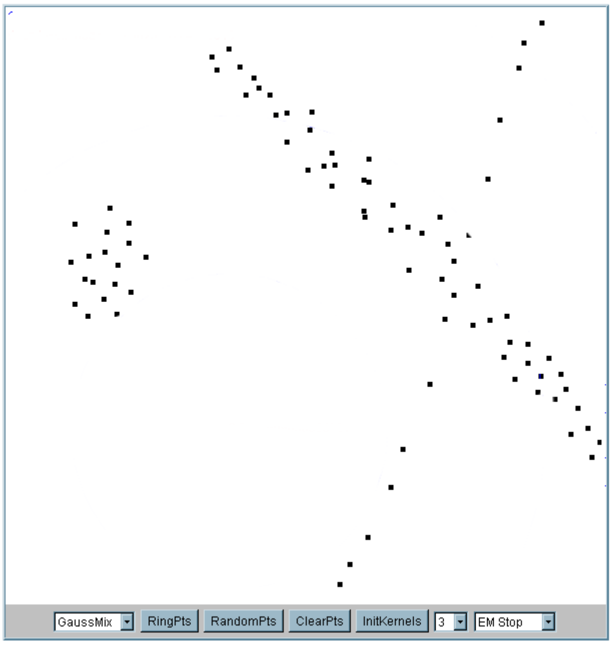


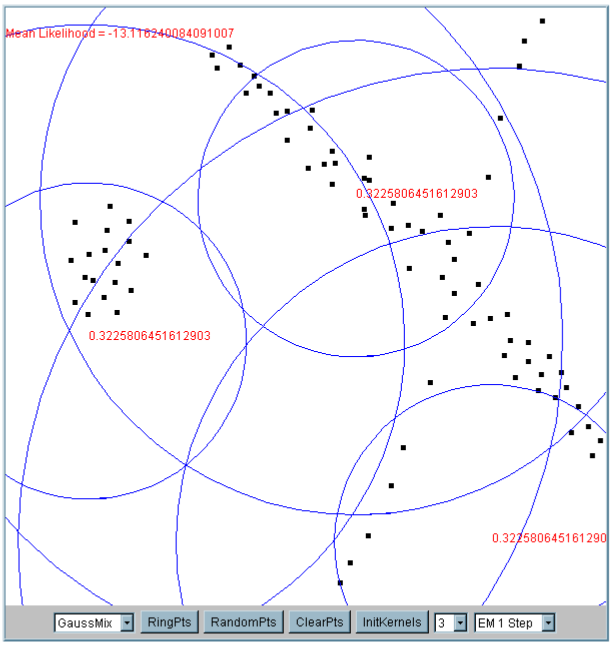
- M-step: re-estimate parameters based on *probabilistic* membership

$$\mu_i \leftarrow \sum_j p_{ij}\mathbf{x}_j/p_i$$
$$\Sigma_i \leftarrow \sum_j p_{ij}\mathbf{x}_j\mathbf{x}_j^\top/p_i$$
$$w_i \leftarrow p_i\,.$$

- Repeat until change in parameters are smaller than a threshold

See R&N for details

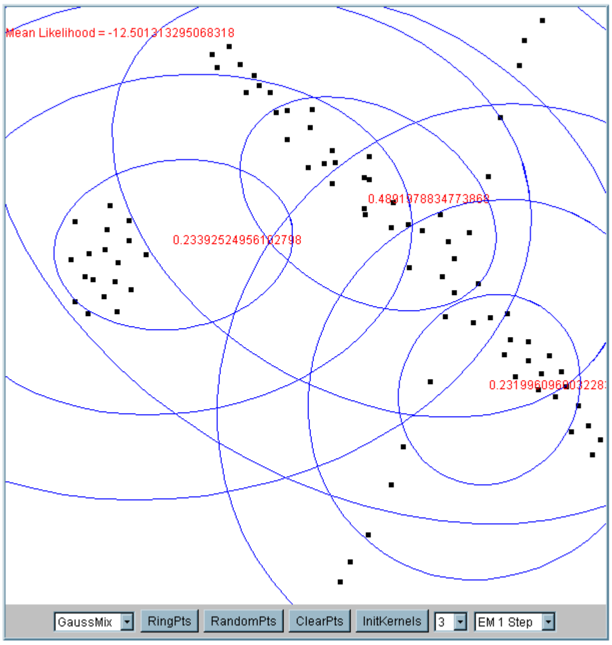




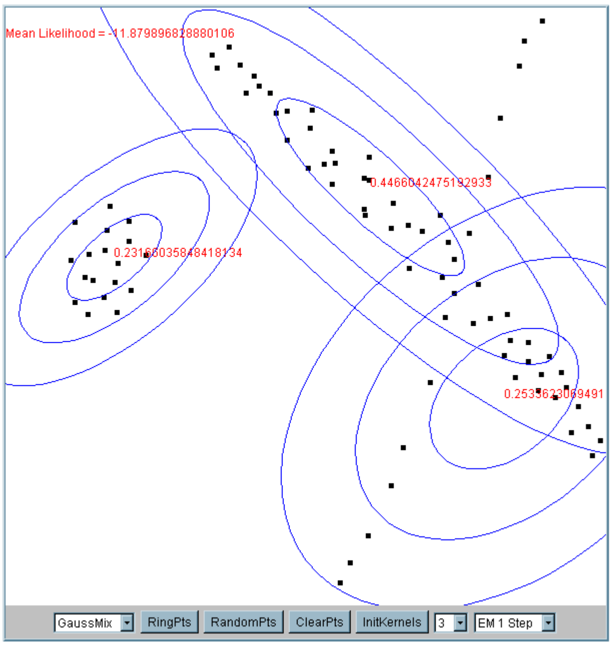
Iteration 1

The cluster means are randomly assigned

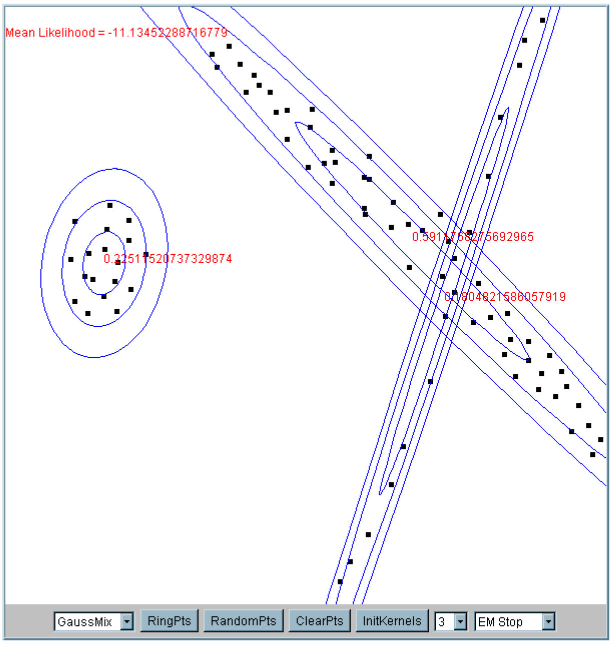
Iteration 2



Iteration 5



Iteration 25



Strength of Gaussian Mixture Models

- *Interpretability*: learns a generative model of each cluster
  - you can generate new data based on the learned model
- *Relatively efficient*: $O(tkn)$, where n is # objects, k is # clusters, and t is # iterations. Normally, k, t << n.
- Intuitive (?) objective function: optimizes data likelihood
- Extensible to other mixture models for other data types
  - e.g. mixture of multinomial for categorical data
  - maximization instead of mean
  - sensitivity to noise and outliers depend on the distribution

Weakness of Gaussian Mixture Models

- Often terminates at a *local optimum*. Initialization is important.
- Need to specify *K,* the *number* of clusters, in advance
- Not suitable to discover clusters with *non-convex shapes*

- Summary
  - To learn Gaussian mixture, assign probabilistic membership based on current parameters, and re-estimate parameters based on current membership

Clustering methods: Comparison

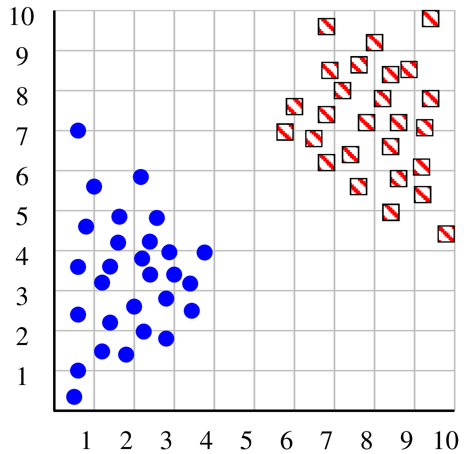
	Hierarchical	**K-means**	**GMM**
Running time	naively, O(N^3)	fastest (each iteration is linear)	fast (each iteration is linear)
Assumptions	requires a similarity / distance measure	strong assumptions	strongest assumptions
Input parameters	none	*K* (number of clusters)	*K* (number of clusters)
Clusters	subjective (only a tree is returned)	exactly *K* clusters	exactly *K* clusters

Outline

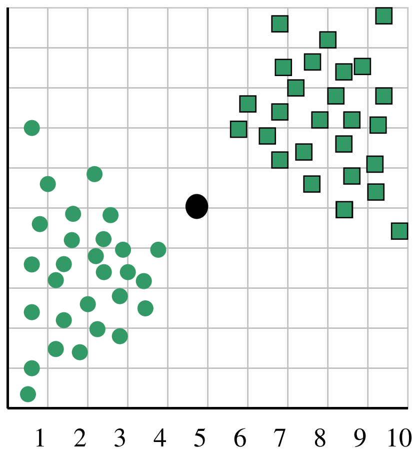
- Motivation
- Distance measure
- Hierarchical clustering
- Partitional clustering
  - K-means
  - Gaussian Mixture Models
  - Number of clusters

How can we tell the *right* number of clusters?

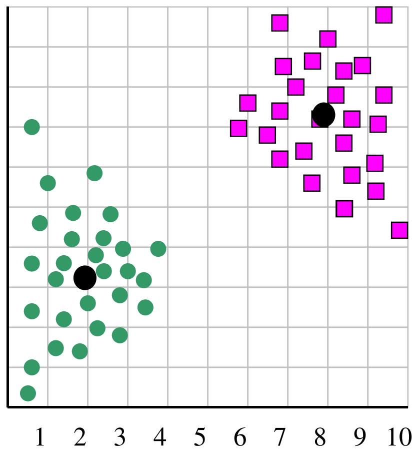
In general, this is a unsolved problem. However there are many approximate methods. In the next few slides we will see an example.



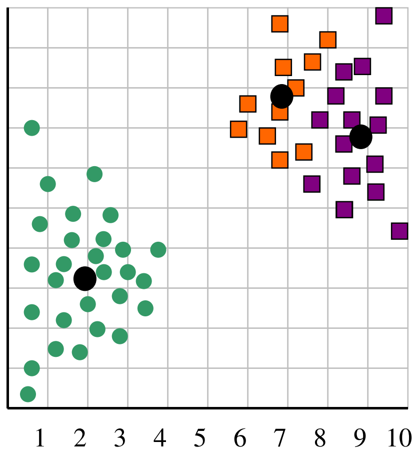
When k = 1, the objective function is 873.0



When k = 2, the objective function is 173.1

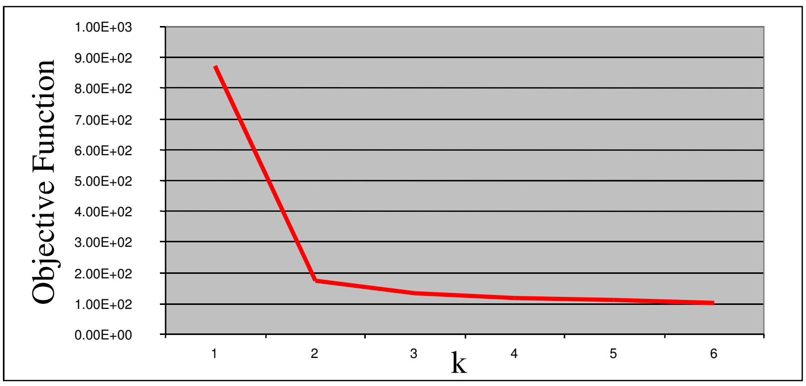


When k = 3, the objective function is 133.6



We can plot the objective function values for k equals 1 to 6…

The abrupt change at k = 2, is highly suggestive of two clusters in the data. This technique for determining the number of clusters is known as “knee finding” or “elbow finding”.



Note that the results are not always as clear cut as in this toy example

What you should know

- Why is clustering useful
- What are the different types of clustering algorithms
- What are the assumptions we are making for each, and what can we get from them
- Unsolved issues: number of clusters, initialization, etc.

Acknowledgement: modified from excellent slides of Eamonn Keogh, Ziv Bar-Joseph, Andrew Moore, and others.