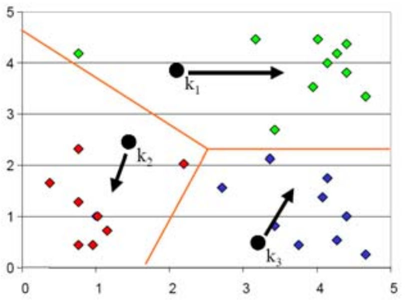
Machine Learning

10-701, Fall 2016

Clustering and Distance Metrics

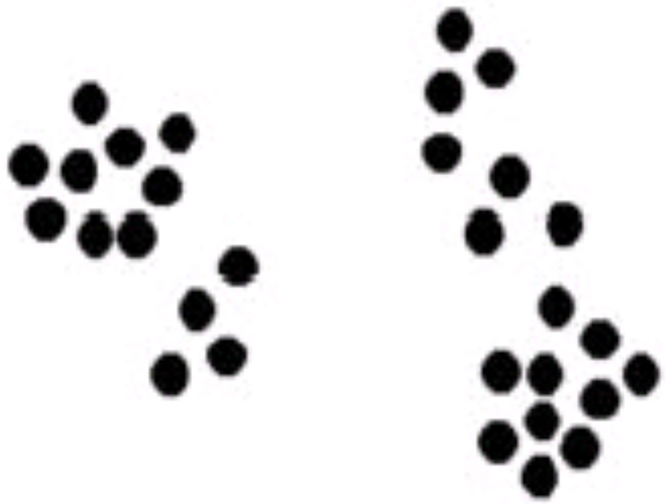
Eric Xing

Lecture 15, October 26, 2016



Reading: Chap. 9, C.B book

What is clustering?



- Are there any “grouping” them ?
- What is each group ?
- How many ?
- How to identify them?

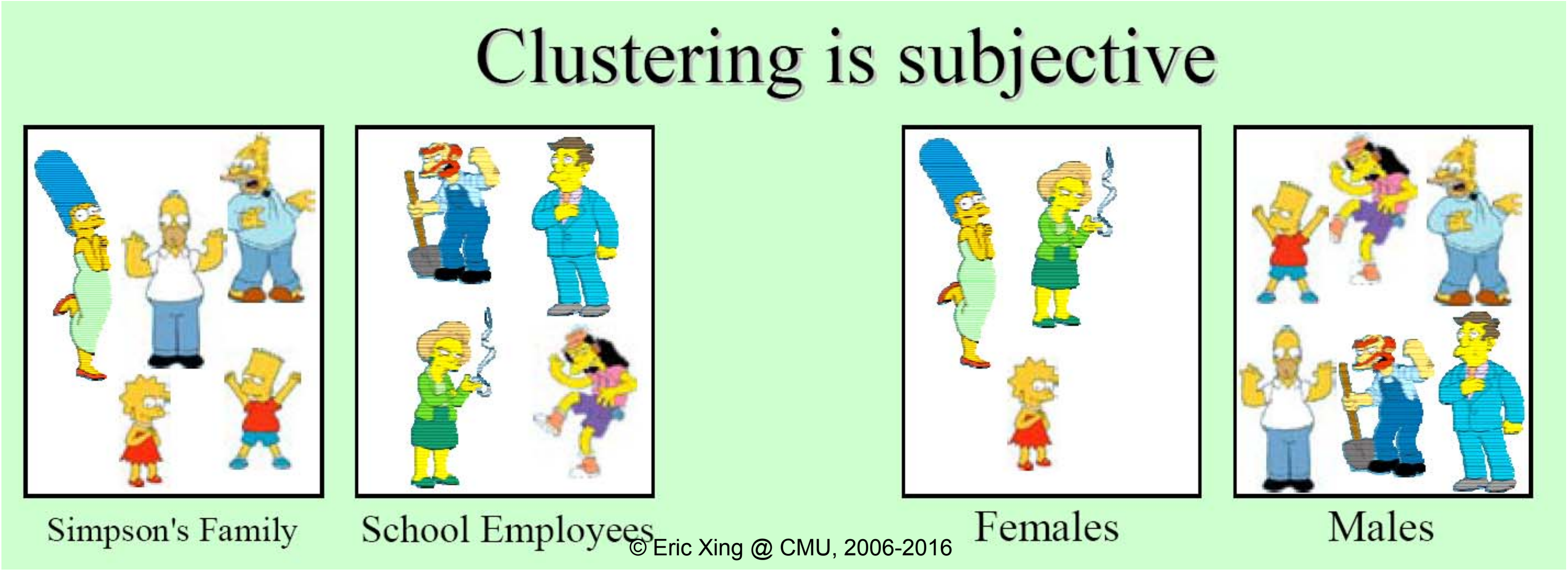
What is clustering?

- Clustering: the process of grouping a set of objects into classes of similar objects
  - high intra-class similarity
  - low inter-class similarity
  - It is the commonest form of unsupervised learning
- Unsupervised learning = learning from raw (unlabeled, unannotated, etc) data, as opposed to supervised data where a classification of examples is given
- A common and important task that finds many applications in Science, Engineering, information Science, and other places
  - Group genes that perform the same function
  - Group individuals that has similar political view
  - Categorize documents of similar topics
  - Ideality similar objects from pictures

Issues for clustering

- What is a natural grouping among these objects?
  - Definition of "groupness"
- What makes objects “related”?
  - Definition of "similarity/distance"
- Representation for objects
  - Vector space? Normalization?
- How many clusters?
  - Fixed a priori?
  - Completely data driven?
    - Avoid “trivial” clusters - too large or small
- Clustering Algorithms
  - Partitional algorithms
  - Hierarchical algorithms
- Formal foundation and convergence

What is a natural grouping among these objects?



Clustering is subjective

What is Similarity?

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
- Depends on representation and algorithm. For many rep./alg., easier to think in terms of a distance (rather than similarity) between vectors.

What properties should a distance measure have?

- $D(A,B) = D(B,A)$ *Symmetry*
- $D(A,A) = 0$ *Constancy of Self-Similarity*
- $D(A,B) = 0$ IIf A= B *Positivity Separation*
- $D(A,B) \leq D(A,C) + D(B,C)$ *Triangular Inequality*

Intuitions behind desirable distance measure properties

- $D(A,B) = D(B,A)$ *Symmetry*
  - *Otherwise you could claim "Alex looks like Bob, but Bob looks nothing like Alex"*

- $D(A,A) = 0$ *Constancy of Self-Similarity*
  - *Otherwise you could claim "Alex looks more like Bob, than Bob does"*

- $D(A,B) = 0$ IIf A= B *Positivity Separation*
  - *Otherwise there are objects in your world that are different, but you cannot tell apart.*

- $D(A,B) \leq D(A,C) + D(B,C)$ *Triangular Inequality*
  - *Otherwise you could claim "Alex is very like Bob, and Alex is very like Carl, but Bob is very unlike Carl"*

Distance Measures: Minkowski Metric

- Suppose two object *x* and *y* both have *p* features

$$x = (x_1, x_2, \cdots, x_p)$$
$$y = (y_1, y_2, \cdots, y_p)$$

- The Minkowski metric is defined by

$$d(x, y) = \sqrt[r]{\sum_{i=1}^{p} | x_i - y_i |^r}$$

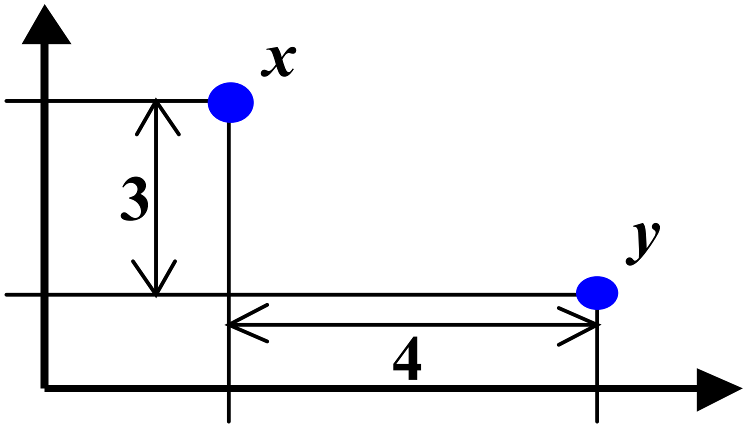
- Most Common Minkowski Metrics

$$1, r = 2 \text{ (Euclidean distance)} \qquad d(x, y) = \sqrt[2]{\sum_{i=1}^{p} | x_i - y_i |^2}$$

$$2, r = 1 \text{ (Manhattan distance)} \qquad d(x, y) = \sum_{i=1}^{p} | x_i - y_i |$$

$$3, r = +\infty \text{ ("sup" distance)} \qquad d(x, y) = \max_{1 \le i \le p} | x_i - y_i |$$

An Example



$$1:\ \text{Euclidean distance}:\quad \sqrt[2]{4^2+3^2}=5.$$

$$2:\ \text{Manhattan distance}:\quad 4+3=7.$$

$$3:\ \text{"sup" distance}:\quad \max\{4,3\}=4.$$

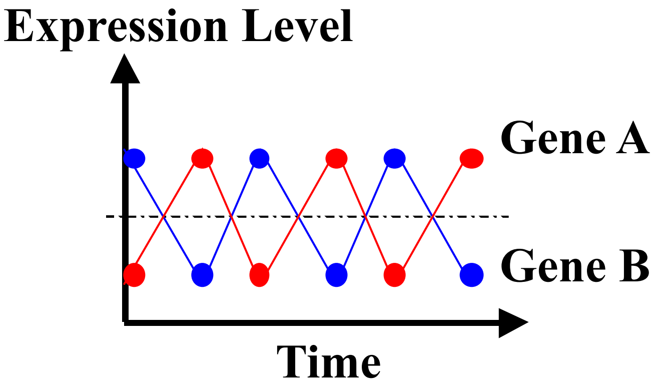
Hamming distance

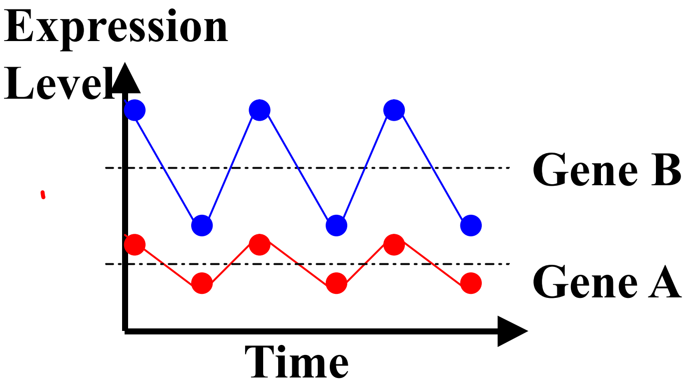
- Manhattan distance is called *Hamming distance* when all features are binary.
  - Gene Expression Levels Under 17 Conditions (1-High,0-Low)

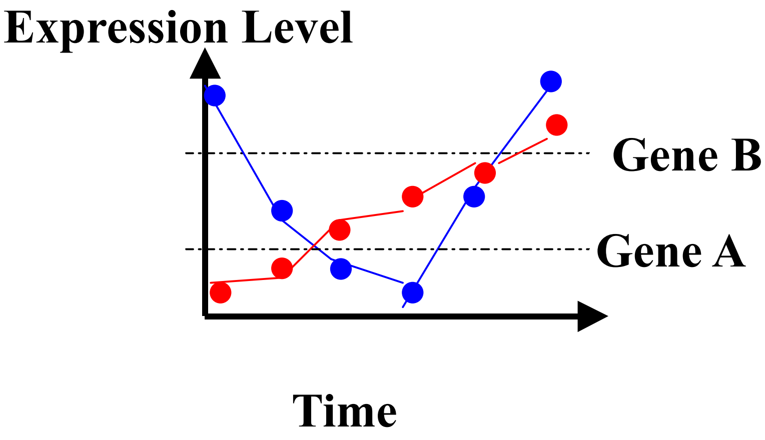
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
GeneA	0	1	1	0	0	1	0	0	1	0	0	1	1	1	0	0	1
GeneB	0	1	1	1	0	0	0	0	1	1	1	1	1	1	0	1	1

$$\text{Hamming Distance}: \#(01) + \#(10) = 4 + 1 = 5.$$

Similarity Measures: Correlation Coefficient







Similarity Measures: Correlation Coefficient

- Pearson correlation coefficient

$$s(x,y)=\frac{\sum_{i=1}^{p}(x_i-\overline{x})(y_i-\overline{y})}{\sqrt{\sum_{i=1}^{p}(x_i-\overline{x})^2\times\sum_{i=1}^{p}(y_i-\overline{y})^2}}$$

$$\text{where } \overline{x}=\tfrac{1}{p}\sum_{i=1}^{p}x_i \text{ and } \overline{y}=\tfrac{1}{p}\sum_{i=1}^{p}y_i.$$

$$|s(x,y)|\leq 1$$

- Special case: cosine distance

$$s(x,y)=\frac{\vec{x}\cdot\vec{y}}{|\vec{x}|\cdot|\vec{y}|}$$

Edit Distance:
A generic technique for measuring similarity

- To measure the similarity between two objects, transform one of the objects into the other, and measure how much effort it took. The measure of effort becomes the distance measure.

The distance between Patty and Selma.

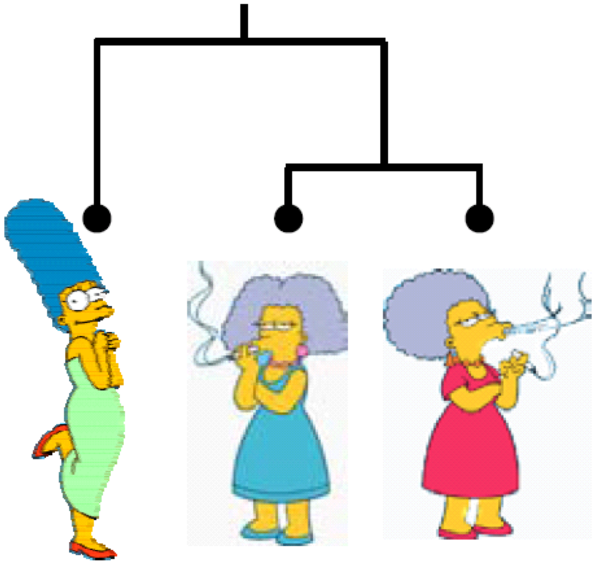
Change dress color, 1 point
Change earring shape, 1 point
Change hair part, 1 point

D(Patty,Selma) = 3

The distance between Marge and Selma.

Change dress color, 1 point
Add earrings, 1 point
Decrease height, 1 point
Take up smoking, 1 point
Lose weight, 1 point

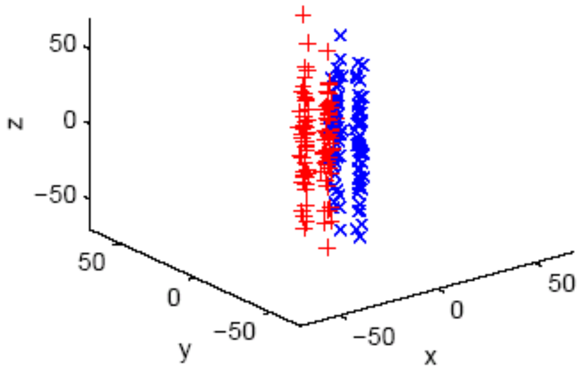
DPMarge,Selma) = 5



This is called the Edit distance or the Transformation distance

Learning Distance Metric

Original data

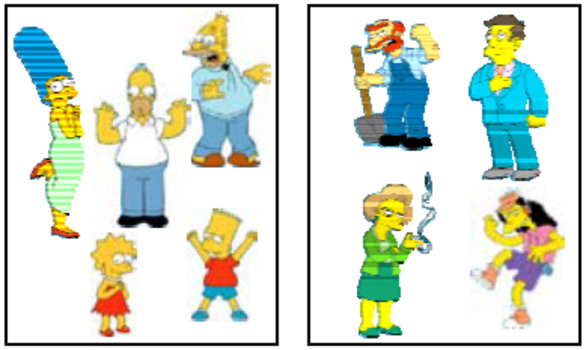


(a)

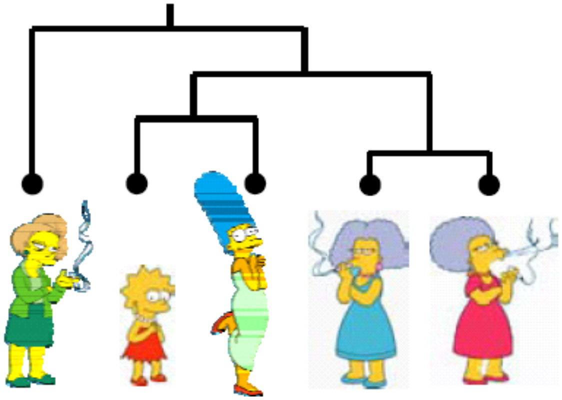
More later …

Clustering Algorithms

- Partitional algorithms
  - Usually start with a random (partial) partitioning
  - Refine it iteratively
    - K means clustering
    - Mixture-Model based clustering

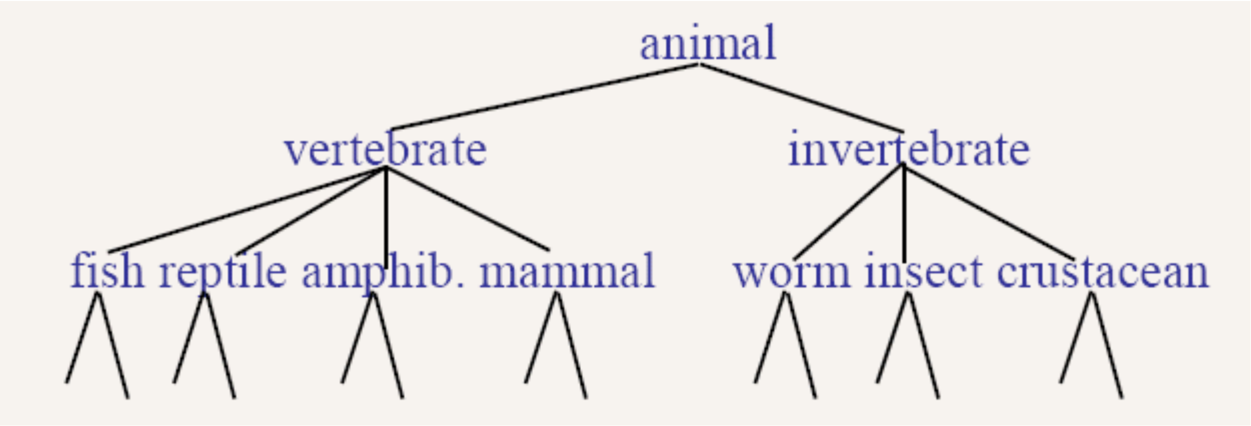


- Hierarchical algorithms
  - Bottom-up, agglomerative
  - Top-down, divisive



Hierarchical Clustering

- Build a tree-based hierarchical taxonomy (dendrogram) from a set of documents.

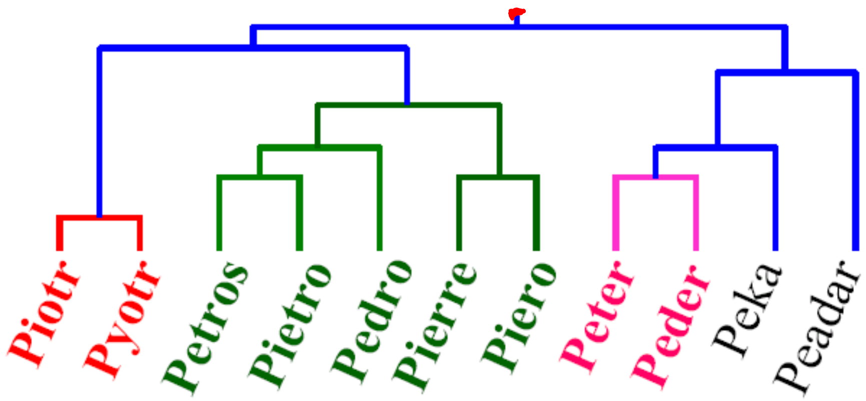


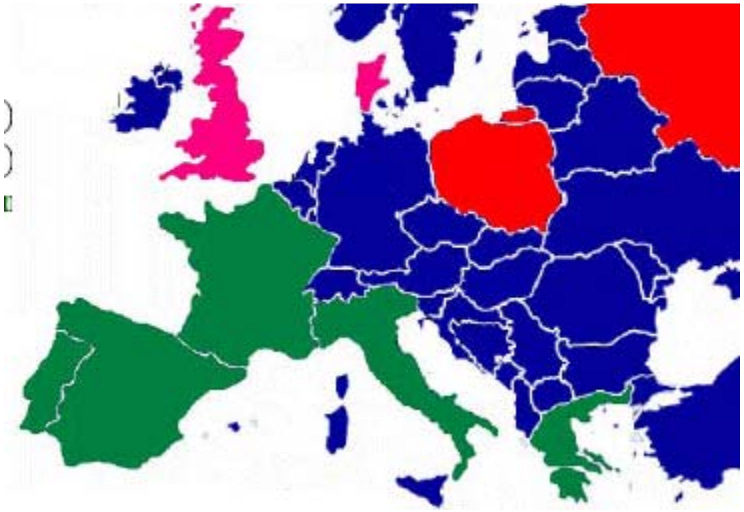
- Note that hierarchies are commonly used to organize information, for example in a web portal.
  - Yahoo! is hierarchy is manually created, we will focus on automatic creation of hierarchies in data mining.

Dendogram

- A Useful Tool for Summarizing Similarity Measurement
  - The similarity between two objects in a dendrogram is represented as the height of the lowest internal node they share.

- Clustering obtained by cutting the dendrogram at a desired level: each connected component forms a cluster.





Hierarchical Clustering

- Bottom-Up Agglomerative Clustering
  - Starts with each obj in a separate cluster
  - then repeatedly joins the closest pair of clusters,
  - until there is only one cluster.

  The history of merging forms a binary tree or hierarchy.

- 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.

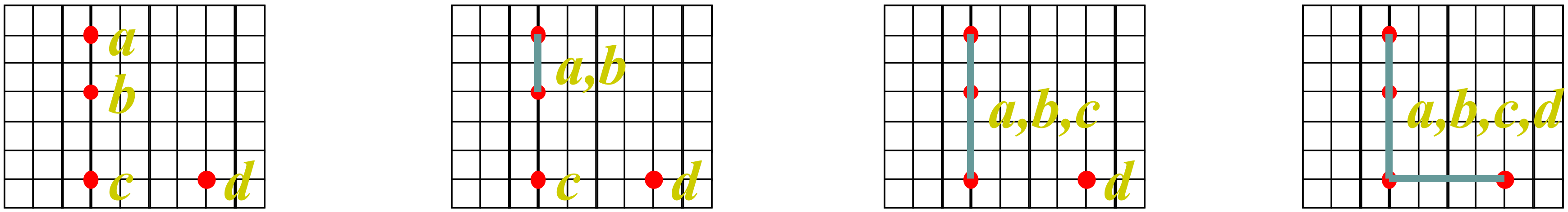
Closest pair of clusters

The distance between two clusters is defined as the distance between

- Single-Link
  - Nearest Neighbor: their closest members.
- Complete-Link
  - Furthest Neighbor: their furthest members.
- Centroid:
  - Clusters whose centroids (centers of gravity) are the most cosine-similar
- Average:
  - average of all cross-cluster pairs.

Single-Link Method

Euclidean Distance



(1)

(2)

(3)

	b	c	d
a	2	5	6
b		3	5
c			4

	b	c	d
a	2	5	6
b		3	5
c			4

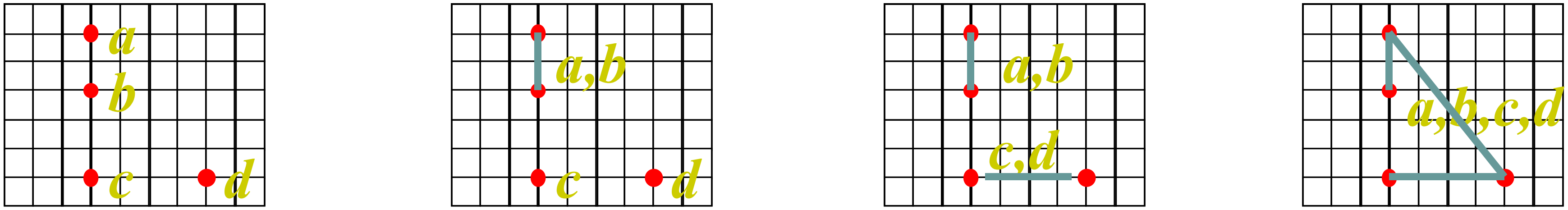
	c	d
a,b	3	5
c		4

	d
a,b,c	4

Distance Matrix

Complete-Link Method

Euclidean Distance



(1)

(2)

(3)

	b	c	d
a	2	5	6
b		3	5
c			4

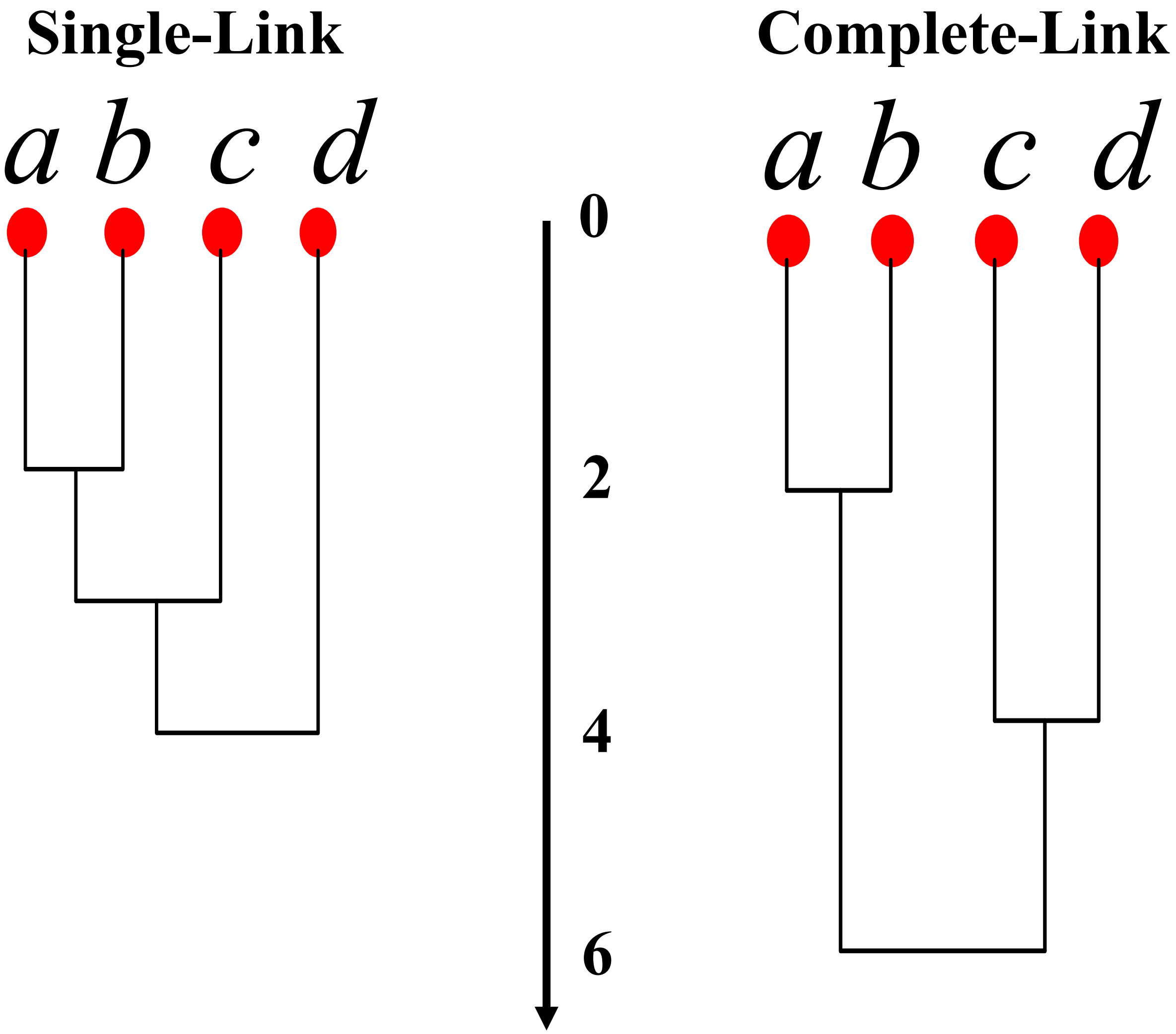
	b	c	d
a	2	5	6
b		3	5
c			4

	c	d
a,b	5	6
c		4

	c,d
a,b	6

Distance Matrix

Dendrograms


Single-Link

Complete-Link

Computational Complexity

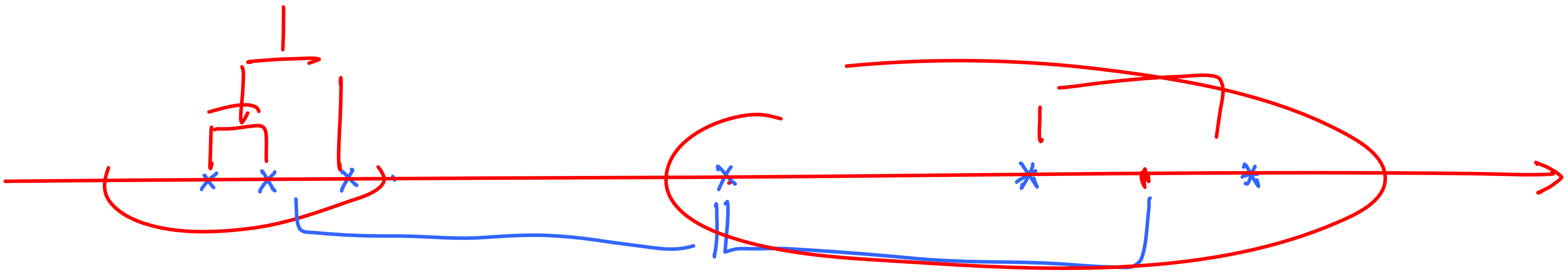
- In the first iteration, all HAC methods need to compute similarity of all pairs of *n* individual instances which is $O(n^2)$.

- In each of the subsequent *n−2* merging iterations, compute the distance between the most recently created cluster and all other existing clusters.

- In order to maintain an overall $O(n^2)$ performance, computing similarity to each other cluster must be done in constant time.

- Else $O(n^2 \log n)$ or $O(n^3)$ if done naively

Local-optimality of HAC



Partitioning Algorithms

- Partitioning method: Construct a partition of *n* objects into a set of *K* clusters

- Given: a set of objects and the number *K*

- Find: a partition of *K* clusters that optimizes the chosen partitioning criterion
  - Globally optimal: exhaustively enumerate all partitions
  - Effective heuristic methods: K-means and K-medoids algorithms

K-Means

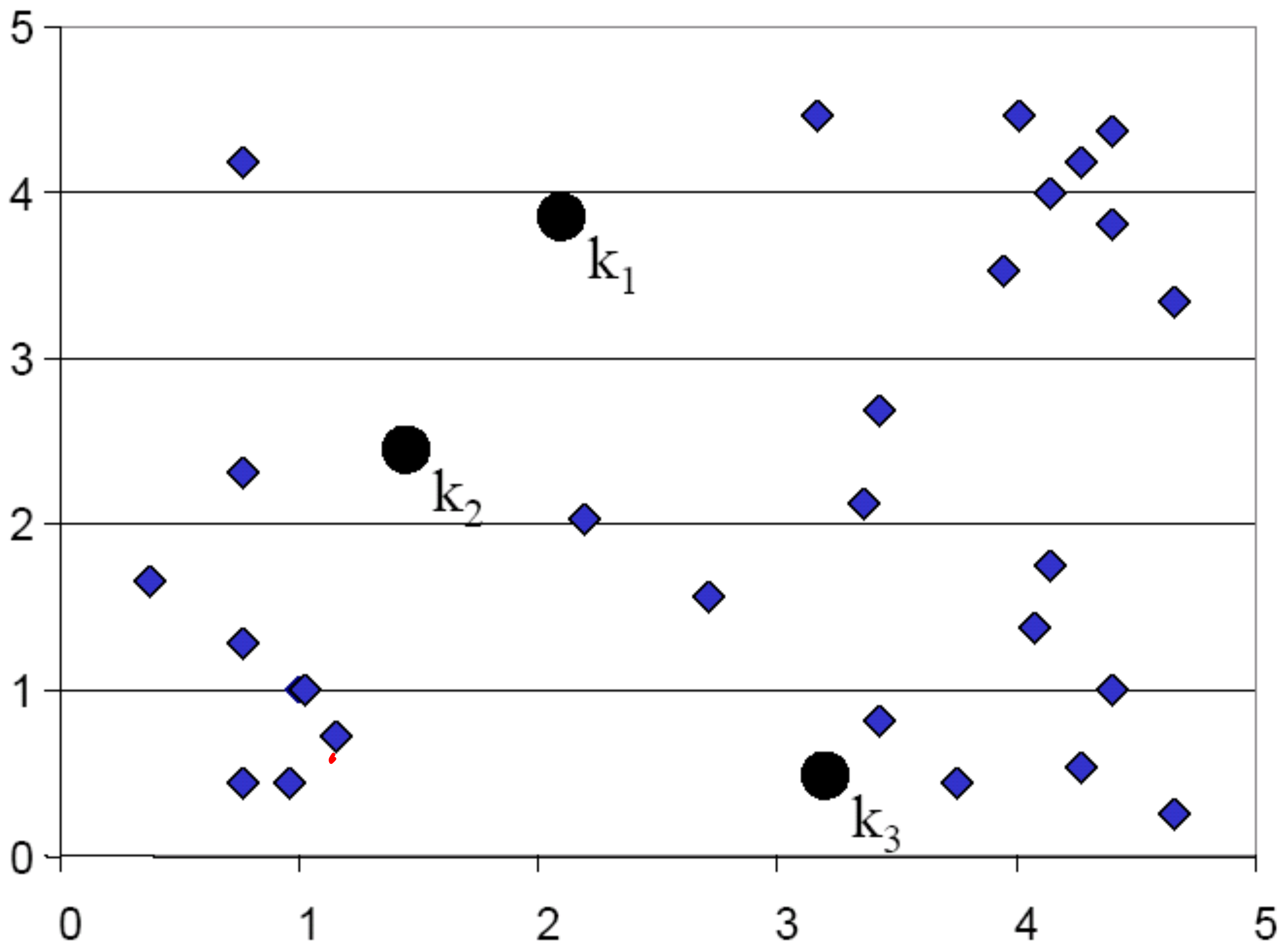
Algorithm

1. Decide on a value for *k*.
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 centroids (aka the center of gravity or mean)

$$\vec{\mu}_k = \frac{1}{\mathcal{C}_k} \sum_{i \in \mathcal{C}_k} \vec{x}_i$$

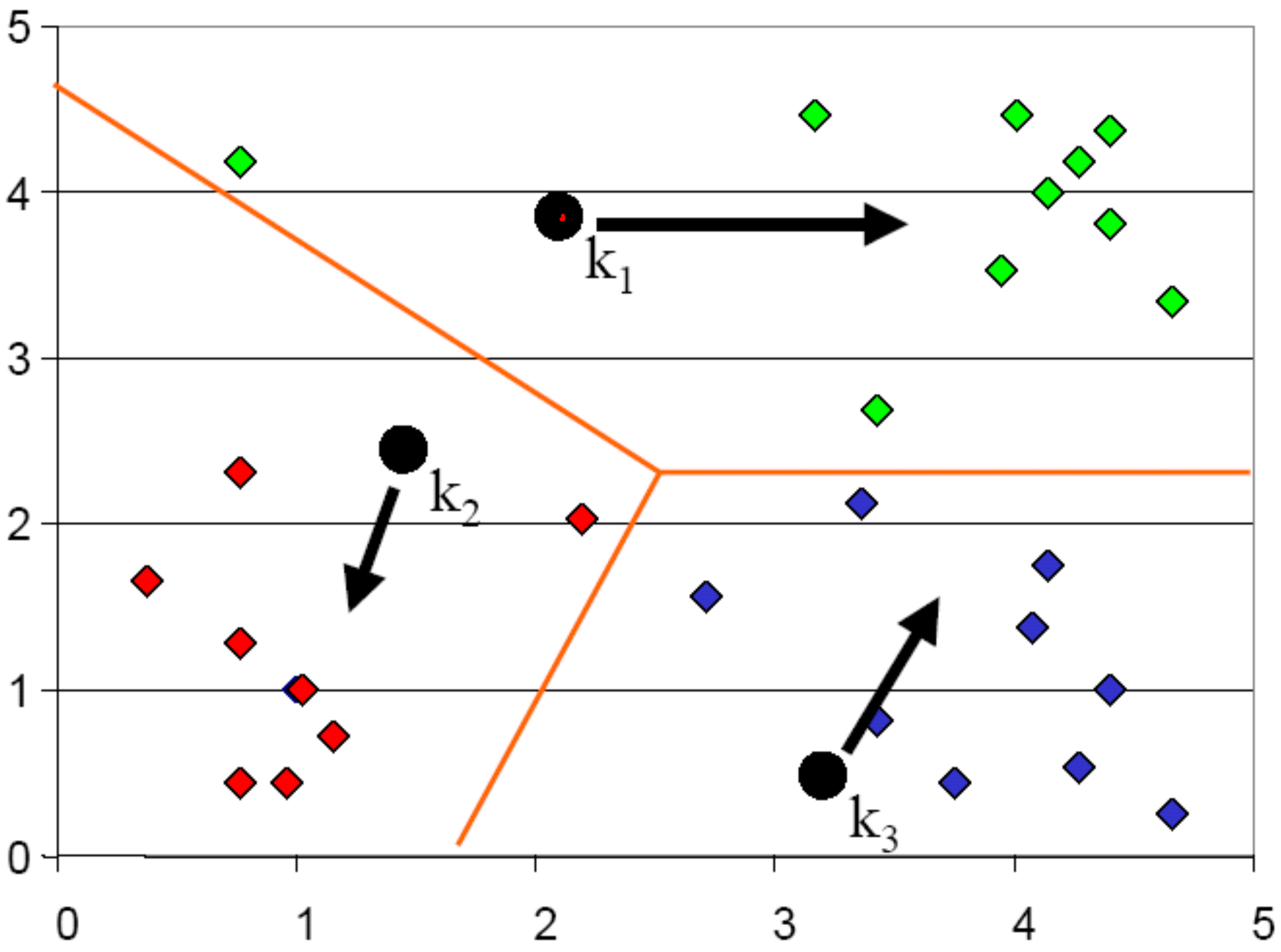
4. Re-estimate the *k* cluster centers, by assuming the memberships found above are correct.
5. If none of the *N* objects changed membership in the last iteration, exit. Otherwise go to 3.

K-means Clustering: Step 1

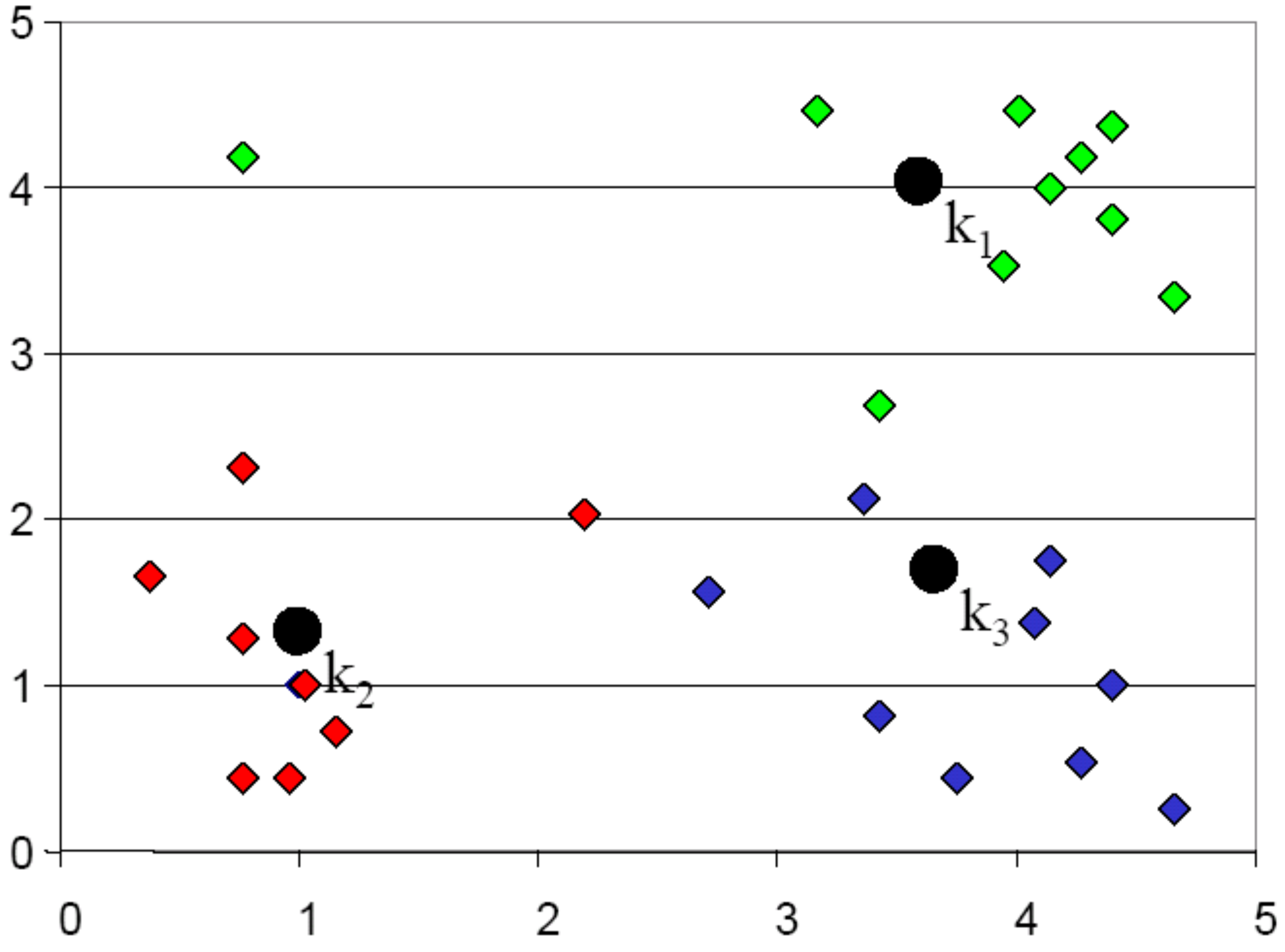


© Eric Xing @ CMU, 2006-2016

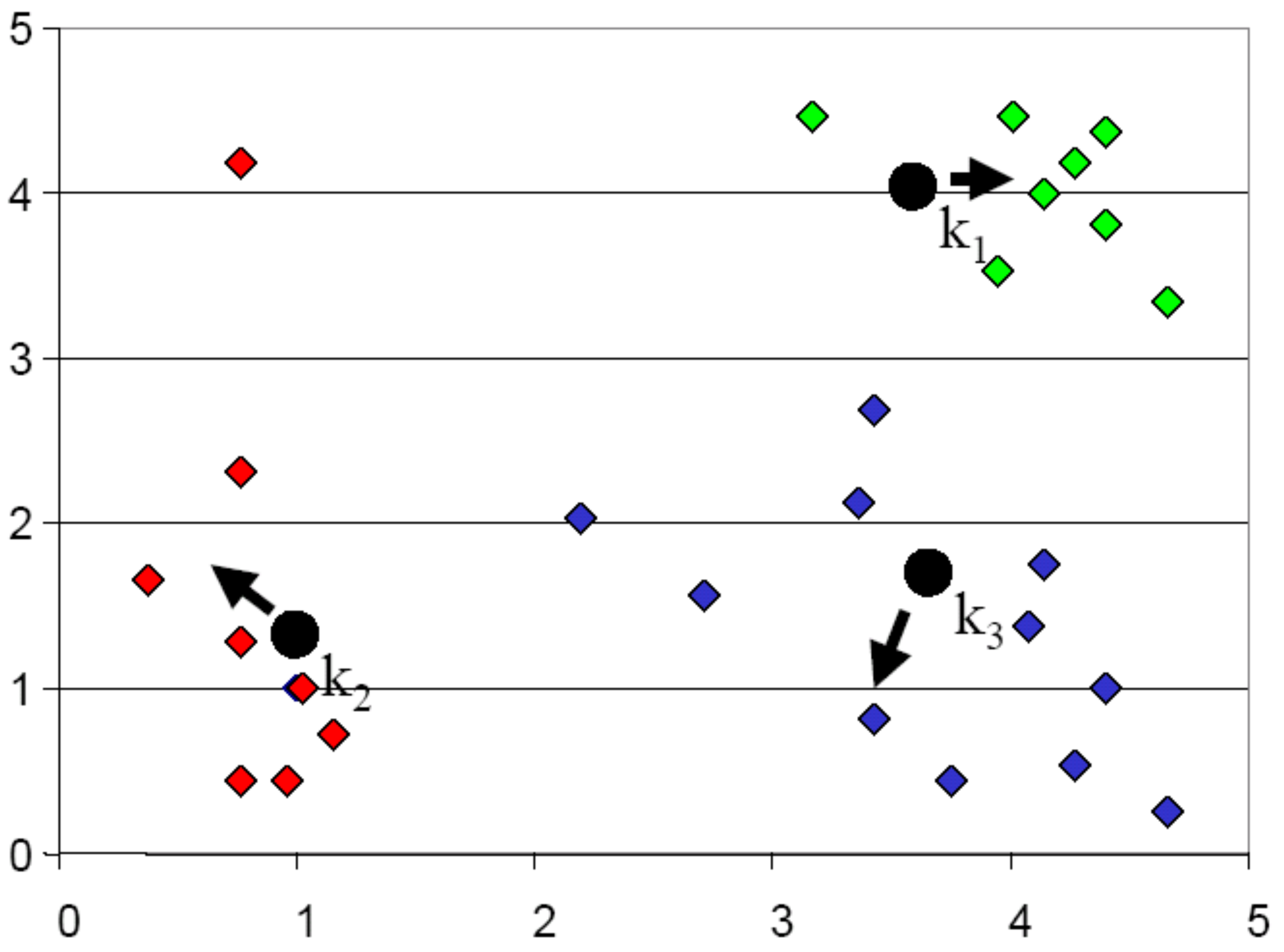
K-means Clustering: Step 2



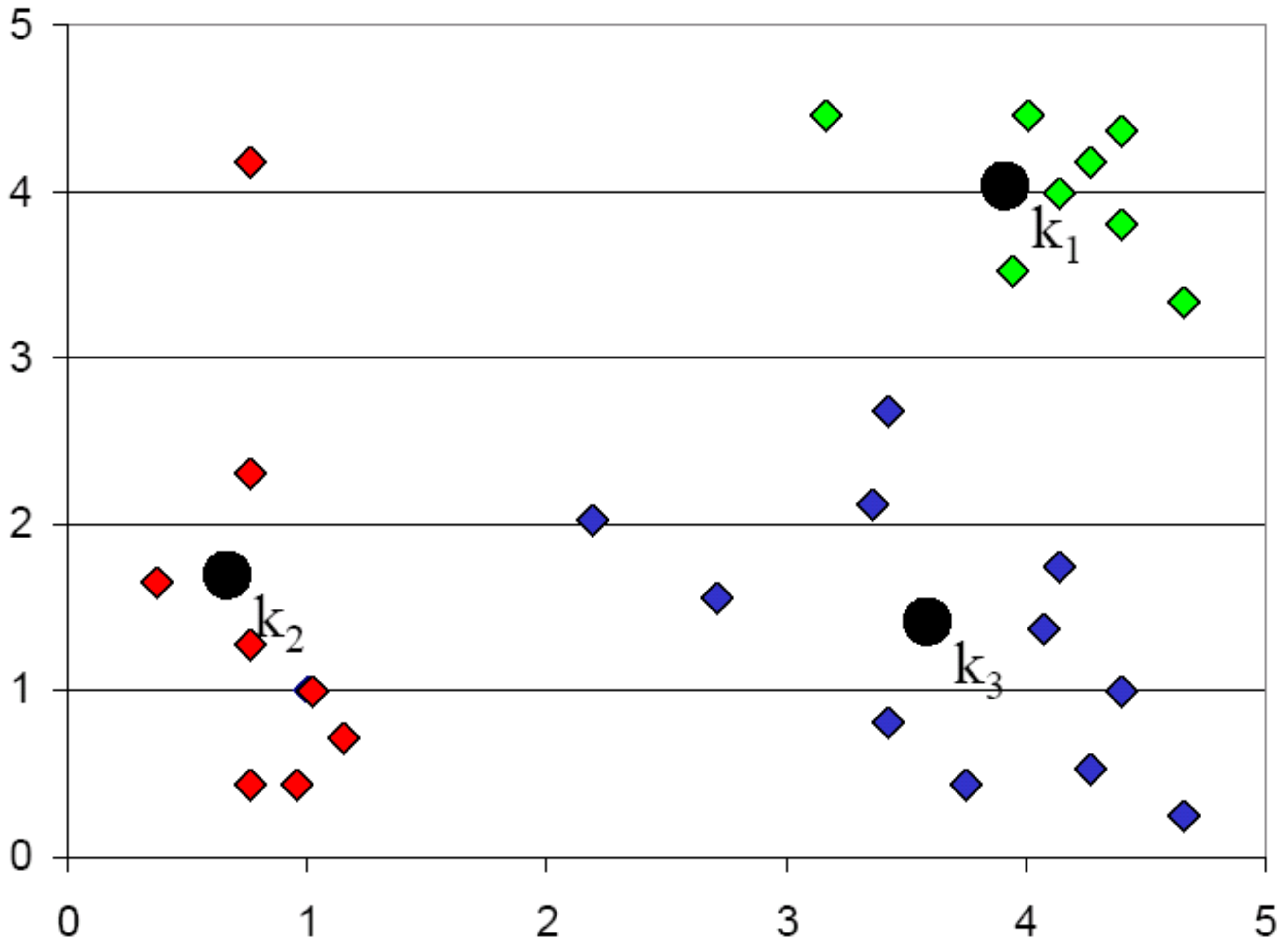
K-means Clustering: Step 3



K-means Clustering: Step 4



K-means Clustering: Step 5



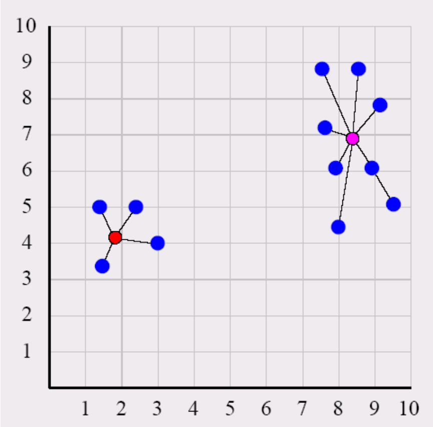
Convergence

- Why should the K-means algorithm ever reach a fixed point?
  - -- A state in which clusters don't change.

- K-means is a special case of a general procedure known as the Expectation Maximization (EM) algorithm.
  - EM is known to converge.
  - Number of iterations could be large.

- Goodness measure
  - sum of squared distances from cluster centroid:

$$SD_{K_i} = \sum_{j=1}^{m_k} ||x_{ij} - \mu_i||^2 \qquad SD_K = \sum_{i=1}^{k} SD_{K_i}$$



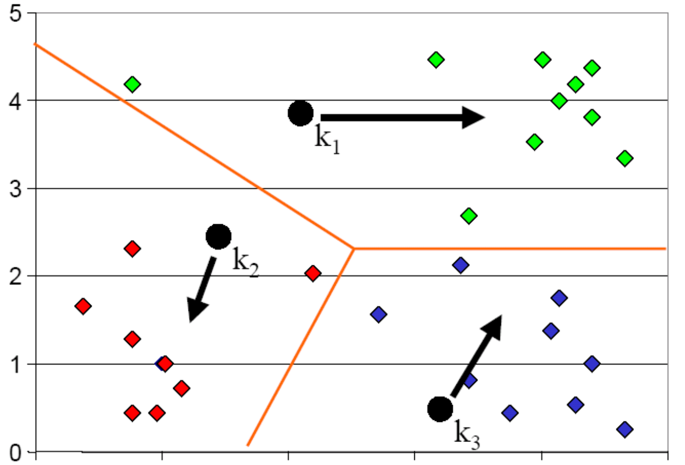
- Reassignment monotonically decreases SD since each vector is assigned to the closest centroid.

Time Complexity

- Computing distance between two objs is O(m) where m is the dimensionality of the vectors.
- Reassigning clusters: O(Kn) distance computations, or O(Knm).
- Computing centroids: Each doc gets added once to some centroid: O(nm).
- Assume these two steps are each done once for l iterations: O($lKnm$).

Seed Choice

- Results can vary based on random seed selection.



- Some seeds can result in poor convergence rate, or convergence to sub-optimal clusterings.
  - Select good seeds using a heuristic (e.g., doc least similar to any existing mean)
  - Try out multiple starting points (very important!!!)
  - Initialize with the results of another method.

How Many Clusters?

- Number of clusters K is given
  - Partition n docs into predetermined number of clusters
- Finding the “right” number of clusters is part of the problem
  - Given objs, partition into an “appropriate” number of subsets.
  - E.g., for query results - ideal value of K not known up front - though UI may impose limits.
- Solve an optimization problem: penalize having lots of clusters
  - application dependent, e.g., compressed summary of search results list.
  - Information theoretic approaches: model-based approach
- Tradeoff between having more clusters (better focus within each cluster) and having too many clusters
- Nonparametric Bayesian Inference

What Is A Good Clustering?

- Internal criterion: A good clustering will produce high quality clusters in which:
  - the intra-class (that is, intra-cluster) similarity is high
  - the inter-class similarity is low
  - The measured quality of a clustering depends on both the obj representation and the similarity measure used

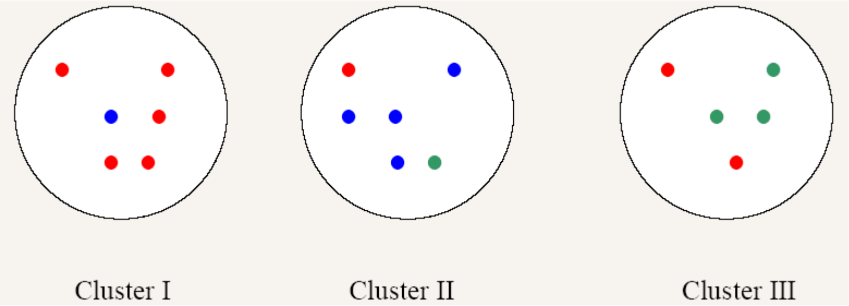
- External criteria for clustering quality
  - Quality measured by its ability to discover some or all of the hidden patterns or latent classes in gold standard data
  - Assesses a clustering with respect to ground truth
  - Example:
    - Purity
    - entropy of classes in clusters (or mutual information between classes and clusters)

External Evaluation of Cluster Quality

- Simple measure: **purity**, the ratio between the dominant class in the cluster and the size of cluster
  - Assume documents with C gold standard classes, while our clustering algorithms produce K clusters, $\omega_1, \omega_2, \ldots, \omega_K$ with n_i members.

$$Purity(w_i) = \frac{1}{n_i} \max_j (n_{ij}) \quad j \in C$$

  - Example



Cluster I: Purity = 1/6 (max(5, 1, 0)) = 5/6

Cluster II: Purity = 1/6 (max(1, 4, 1)) = 4/6

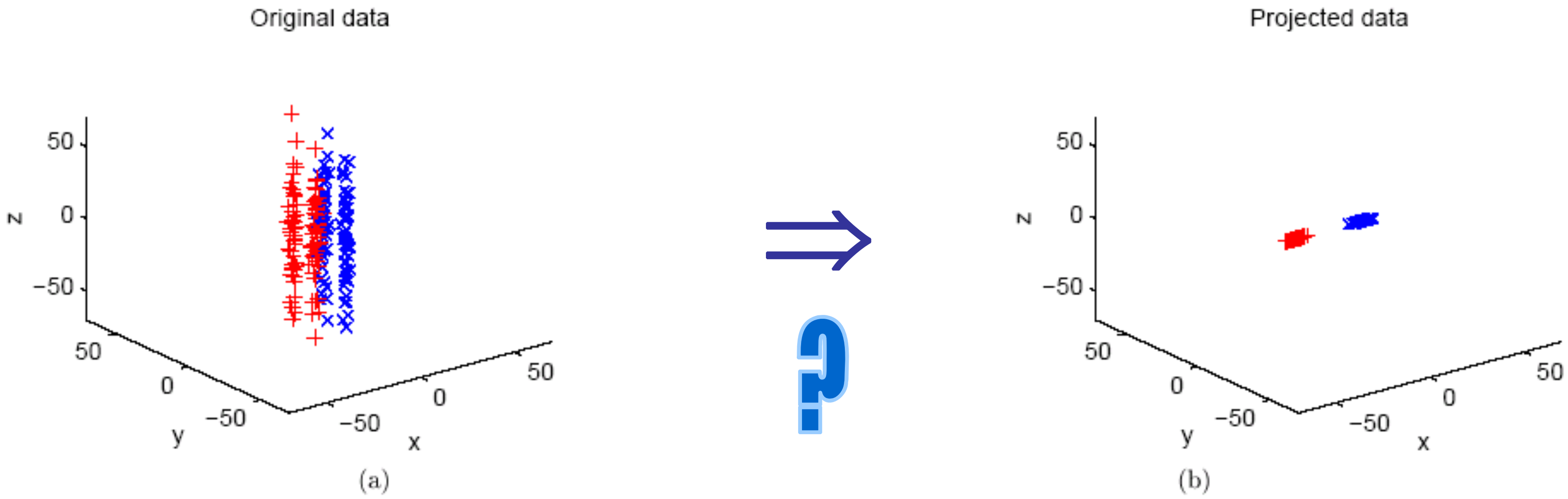
Cluster III: Purity = 1/5 (max(2, 0, 3)) = 3/5

Other measures

Other partitioning Methods

- Partitioning around medioids (PAM): instead of averages, use multidim medians as centroids (cluster "prototypes"). Dudoit and Freedland (2002).
- Self-organizing maps (SOM): add an underlying "topology" (neighboring structure on a lattice) that relates cluster centroids to one another. Kohonen (1997), Tamayo et al. (1999).
- Fuzzy k-means: allow for a "gradation" of points between clusters; soft partitions. Gash and Eisen (2002).
- Mixture-based clustering: implemented through an EM (Expectation-Maximization)algorithm. This provides soft partitioning, and allows for modeling of cluster centroids and shapes. Yeung et al. (2001), McLachlan et al. (2002)

Semi-supervised Metric Learning



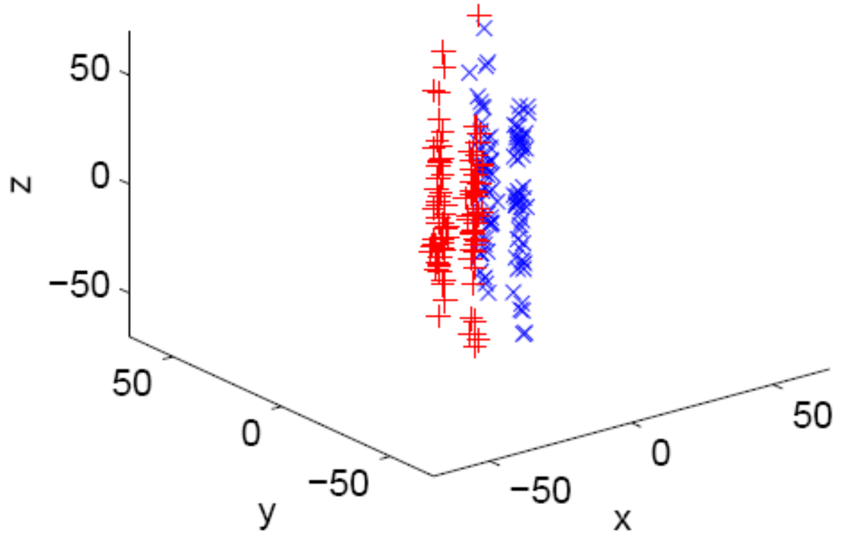
(a)

(b)

Xing et al, NIPS 2003

What is a good metric?

- What is a good metric over the input space for learning and data-mining



- How to convey metrics sensible to a human user (e.g., dividing traffic along highway lanes rather than between overpasses, categorizing documents according to writing style rather than topic) to a computer data-miner using a systematic mechanism?

Issues in learning a metric

- Data distribution is self-informing (E.g., lies in a sub-manifold)
  - Learning metric by finding an embedding of data in some space.
    - Con: does not reflect (changing) human subjectiveness.
- Explicitly labeled dataset offers clue for critical features
  - Supervised learning
    - Con: needs sizable homogeneous training sets.
- What about side information? (E.g., x and y look (or read) similar ...)
  - Providing small amount of qualitative and less structured side information is often much easier than stating explicitly a metric (what should be the metric for writing style?) or labeling a large set of training data.
- Can we learn a distance metric more informative than Euclidean distance using a small amount of side information?

Distance Metric Learning

Side information:

Suppose for some set of points $\{x_i\}_{i=1}^m \subseteq \mathbb{R}^n$, we are given:

$$\begin{array}{lll} \mathcal{S}: & (x_i, x_j) \in \mathcal{S} & \text{if } x_i \text{ and } x_j \text{ are similar} \\ \mathcal{D}: & (x_i, x_j) \in \mathcal{D} & \text{if } x_i \text{ and } x_j \text{ are dissimilar} \end{array}$$

Distance metric learning:

Learn a distance metric of the form

$$d(x, y) = d_A(x, y) = ||x - y||_A = \sqrt{(x - y)^T A (x - y)},$$

such that pairs of points (x_i, x_j) in $\mathcal{S}$ have small squared distance.

- In general, A parameterizes a family of Mahalanobis distances over $\mathbb{R}^n$.
- Learning A is equivalent to finding a rescaling of a data: $x \to A^{1/2}x$.

Optimal Distance Metric

- Learning an optimal distance metric with respect to the side-information leads to the following optimization problem:

$$\min_{A} \quad \sum_{(x_i,x_j)\in\mathcal{S}} ||x_i - x_j||_A^2 \qquad (1)$$

$$\text{s.t.} \sum_{(x_i,x_j)\in\mathcal{D}} ||x_i - x_j||_A \geq 1, \qquad (2)$$

$$A \geq 0. \qquad (3)$$

- This optimization problem is *convex*. Local-minima-free algorithms exist.
- Xing et al 2003 provided an efficient gradient descent + iterative constraint-projection method

Examples of learned distance metrics

- Distance metrics learned on three-cluster artificial data:

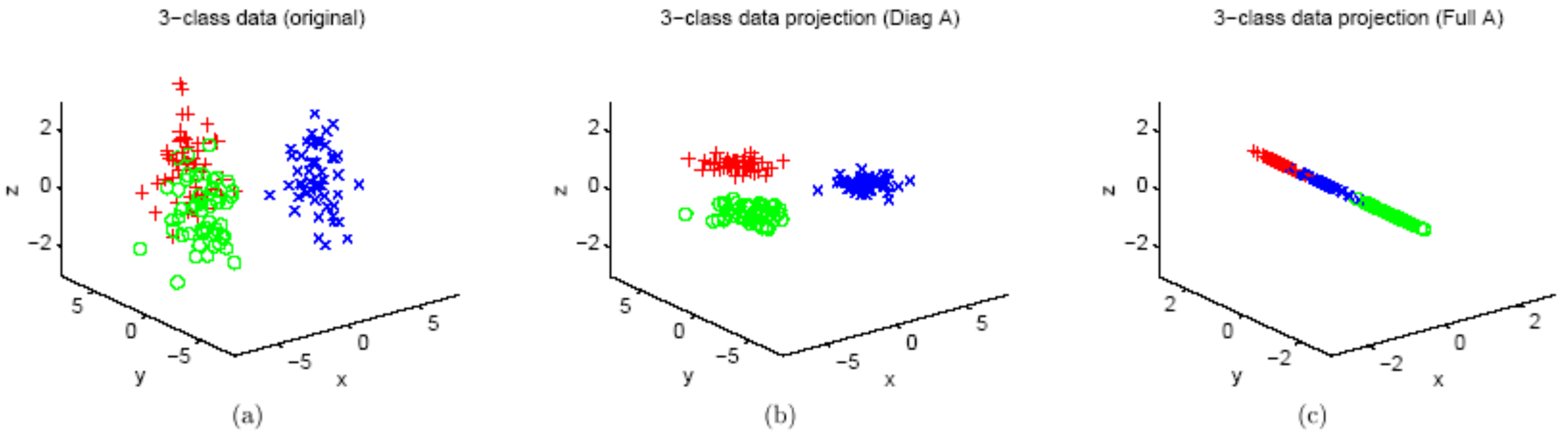
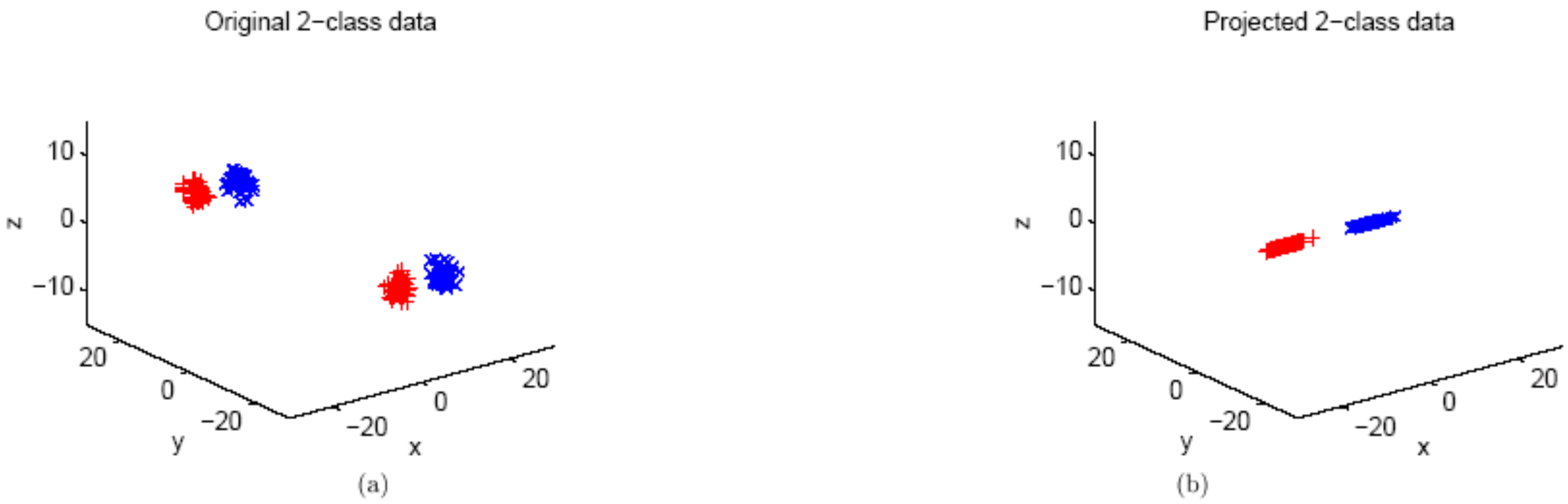


Figure 2: (a) Original data. (b) Rescaling corresponding to learned diagonal A. (c) Rescaling corresponding to full A.

Application to Clustering

- Artificial Data I: a difficult two-class dataset



Original 2−class data

Projected 2−class data

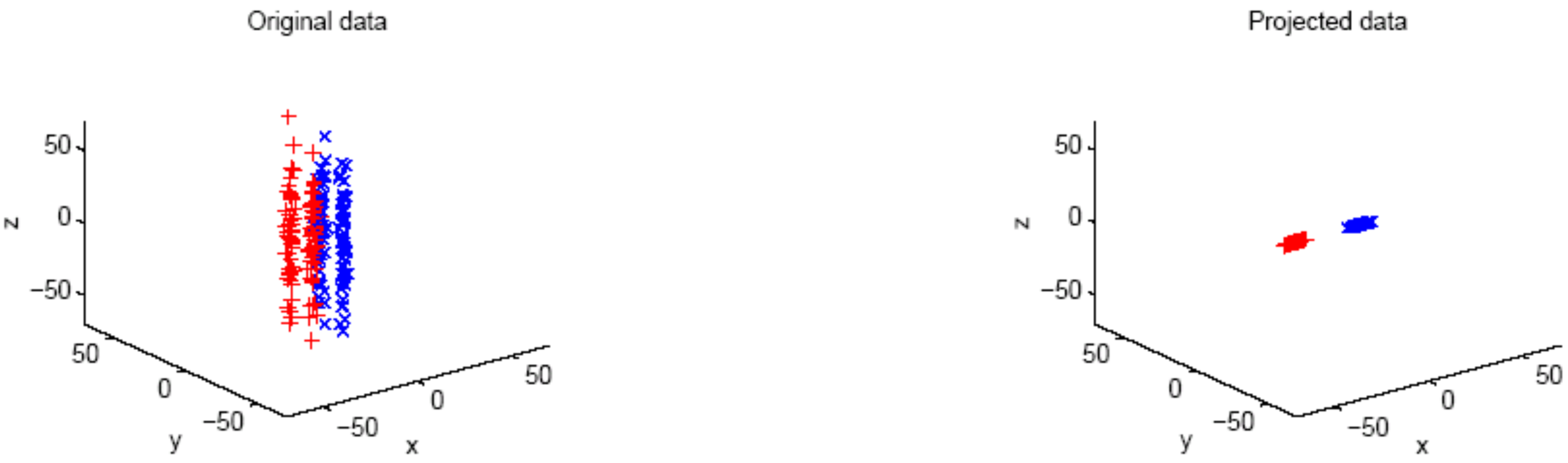
(a)

(b)

1. K-means: Accuracy = 0.4975
2. Constrained K-means: Accuracy = 0.5060
3. K-means + metric: Accuracy = 1
4. Constrained K-means + metric: Accuracy = 1

Application to Clustering

- Artificial Data II: two-class data with strong irrelevant feature



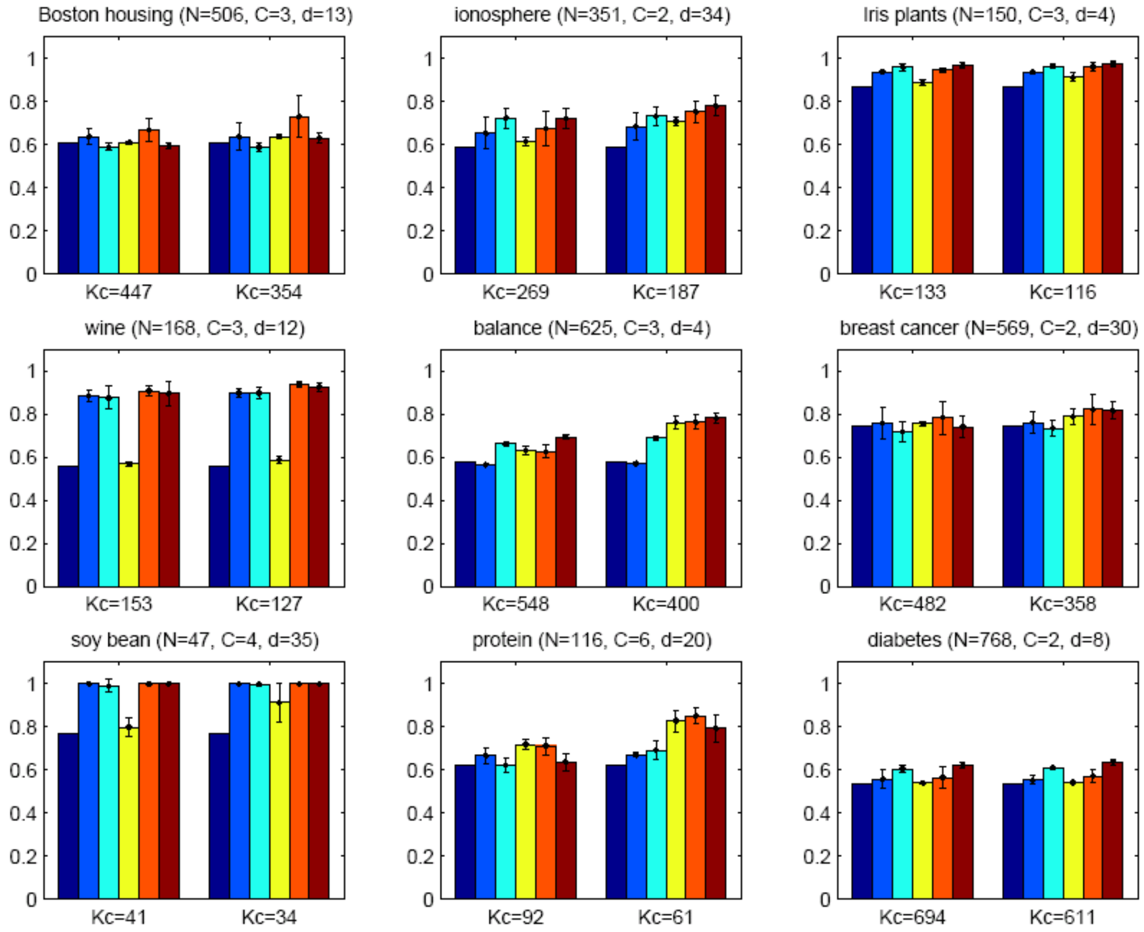
(a)

(b)

1. K-means: Accuracy $= 0.4993$
2. Constrained K-means: Accuracy $= 0.5701$
3. K-means + metric: Accuracy $= 1$
4. Constrained K-means + metric: Accuracy $= 1$

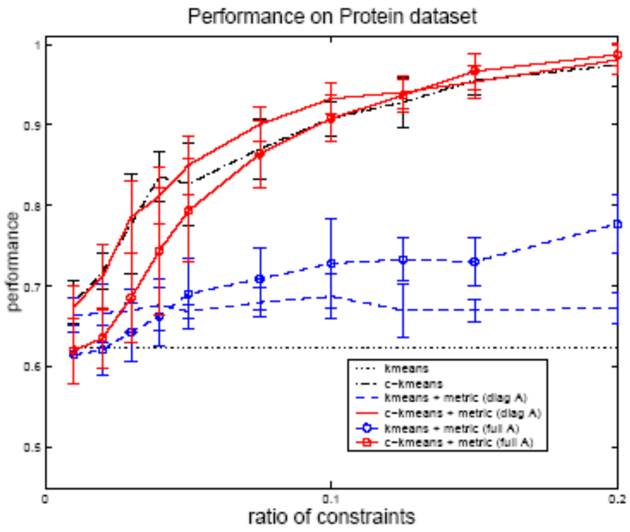
Application to Clustering

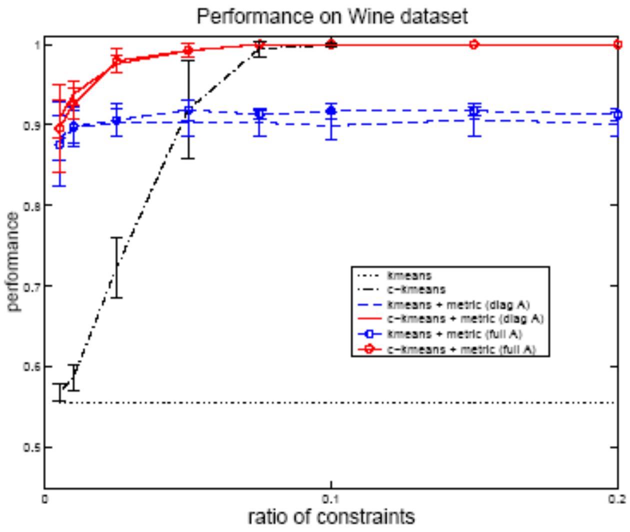
- 9 datasets from the UC Irvine repository



Accuracy vs. amount of side-information

- Two typical examples of how the quality of the clusters found increases with the amount of side-information.





Take home message

- Distance metric learning is an important problem in machine learning and data mining.
- A good distance metric can be learned from small amount of side-information in the form of similarity and dissimilarity constraints from data by solving a convex optimization problem.
- The learned distance metric can identify the most significant direction(s) in feature space that separates data well, effectively doing implicit Feature Selection.
- The learned distance metric can be used to improve clustering performance.