

Clustering

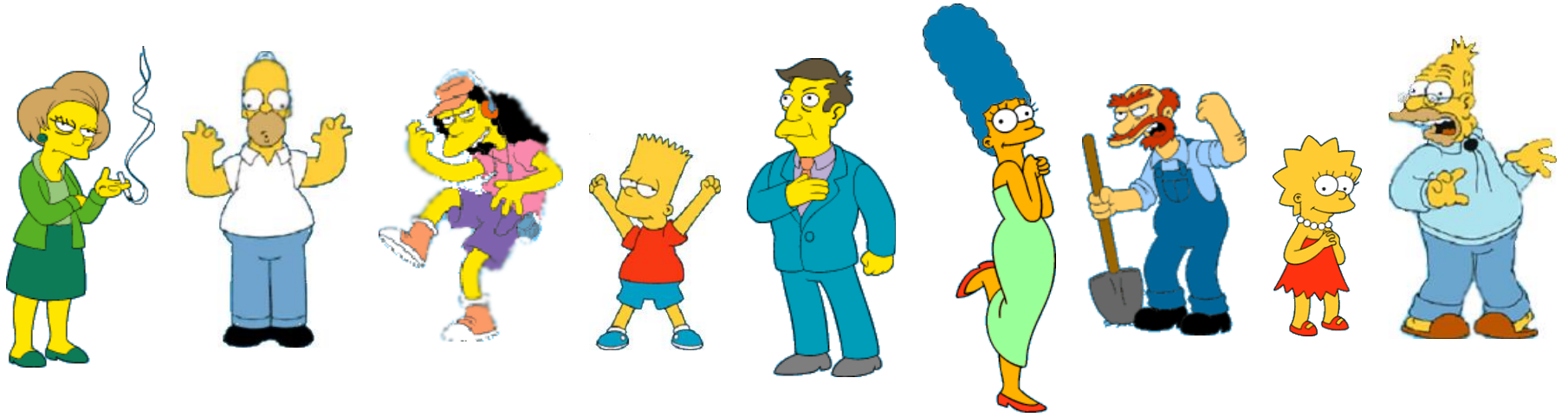
Most slides courtesy of
Eamonn Keogh
(UC Riverside)

What is Clustering?

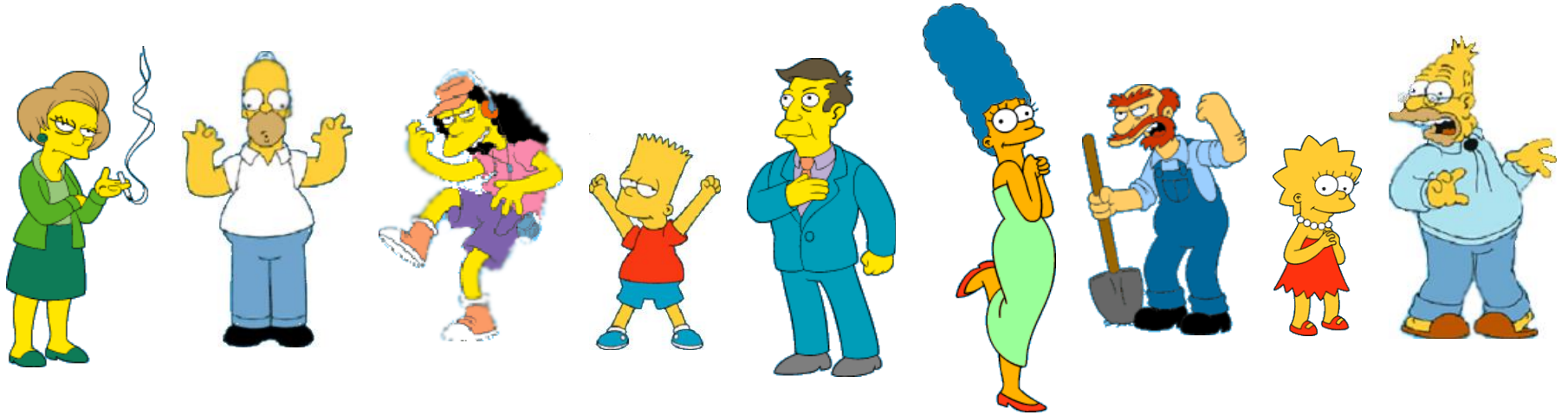
Also called *unsupervised learning*, sometimes called *classification* by statisticians and *sorting* by psychologists and *segmentation* by people in marketing

- Organizing data into classes such that there is
 - high intra-class similarity
 - low inter-class similarity
- Finding the class labels and the number of classes directly from the data (in contrast to classification).
- More informally, finding natural groupings among objects.

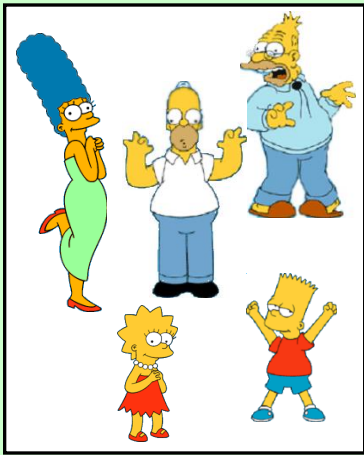
What is a natural grouping among these objects?



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

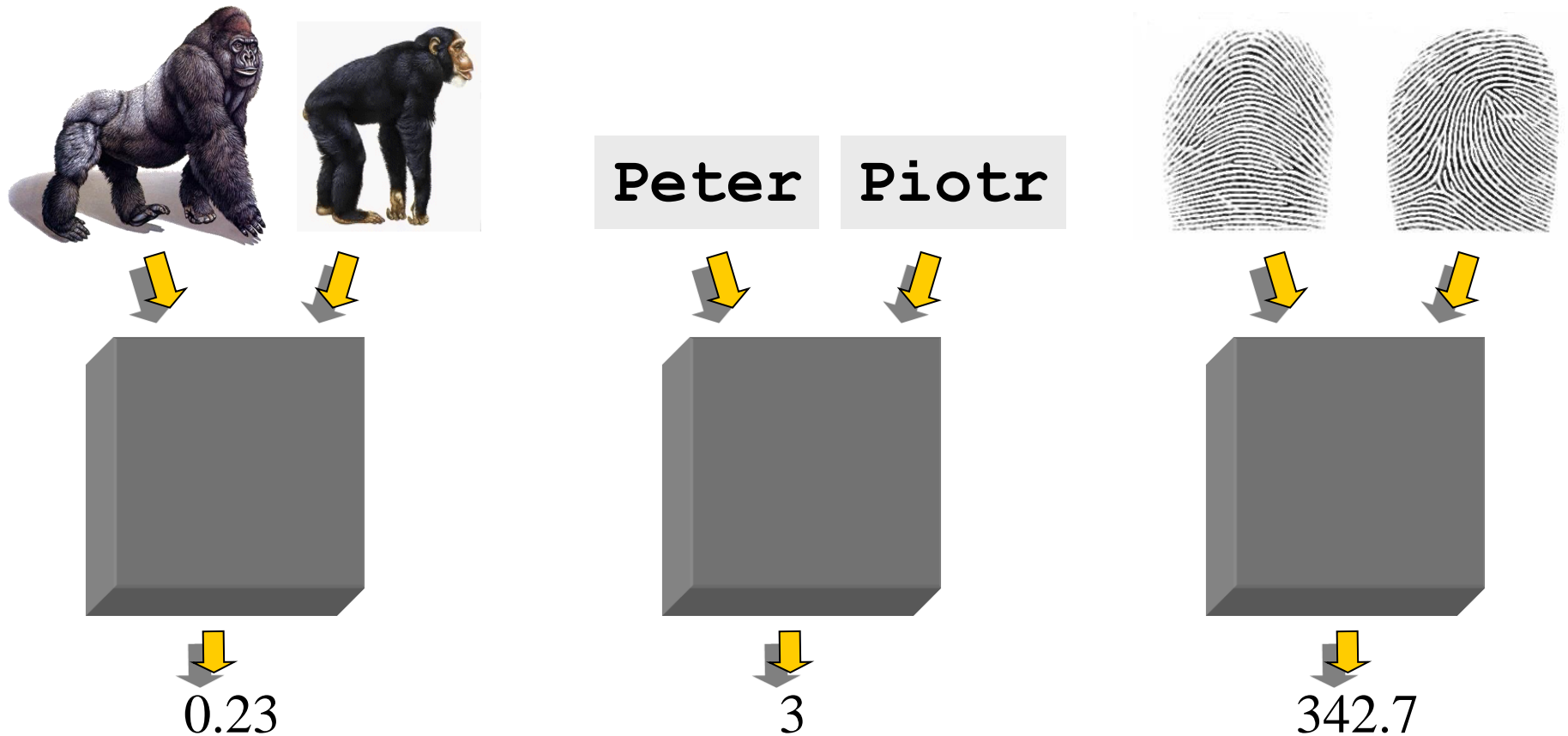


Similarity is hard to define, but...
“We know it when we see it”

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

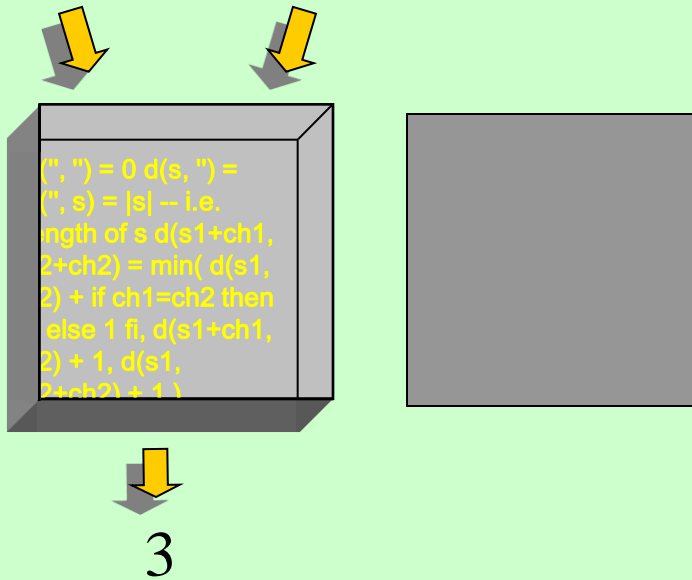
Defining Distance Measures

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 $D(O_1, O_2)$



Peter

Piotr



When we peek inside one of these black boxes, we see some function on two variables. These functions might very simple or very complex.

In either case it is natural to ask, what properties should these functions have?

What properties should a distance measure have?

- $D(A,B) = D(B,A)$
- $D(A,A) = 0$
- $D(A,B) = 0$ iff $A = B$
- $D(A,B) \leq D(A,C) + D(B,C)$

Symmetry

Constancy of Self-Similarity

Positivity (Separation)

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.”

$$D(A,B) = 0 \text{ Iif } A=B$$

Positivity (Separation)

Otherwise there are objects in your world that are different, but you cannot tell apart.

$$D(A,B) \leq D(A,C) + D(B,C)$$

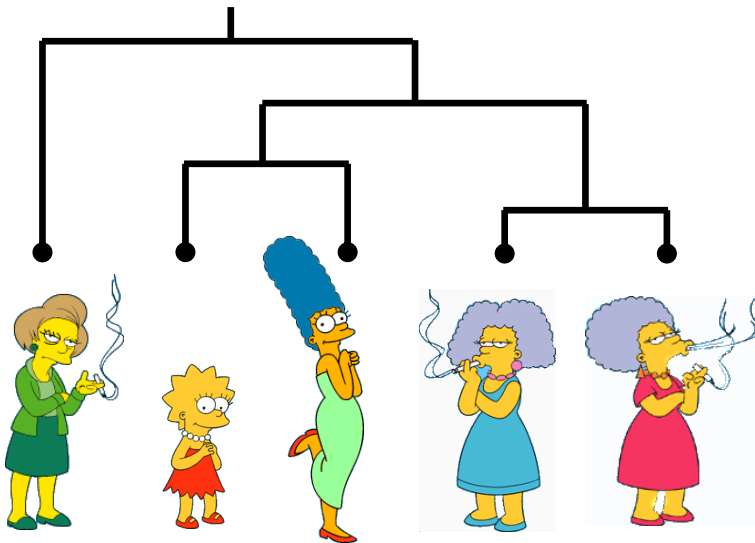
Triangular Inequality

Otherwise you could claim “Alex is very like Bob, and Alex is very like Carl, but Bob is very unlike Carl.”

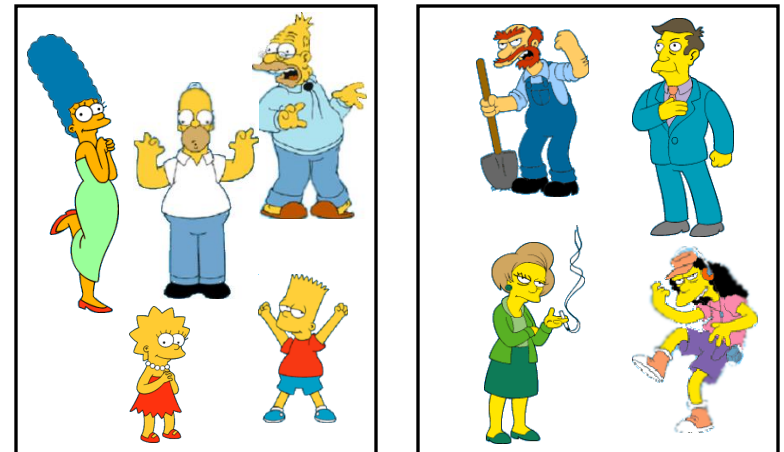
Two Types of Clustering

- **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

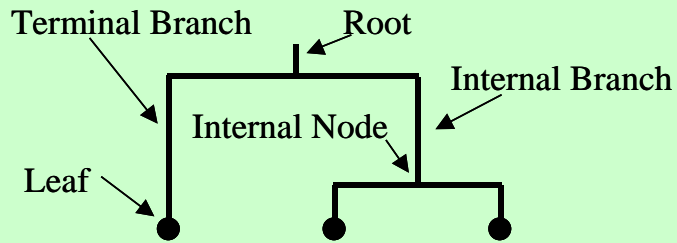


Desirable Properties of a Clustering Algorithm

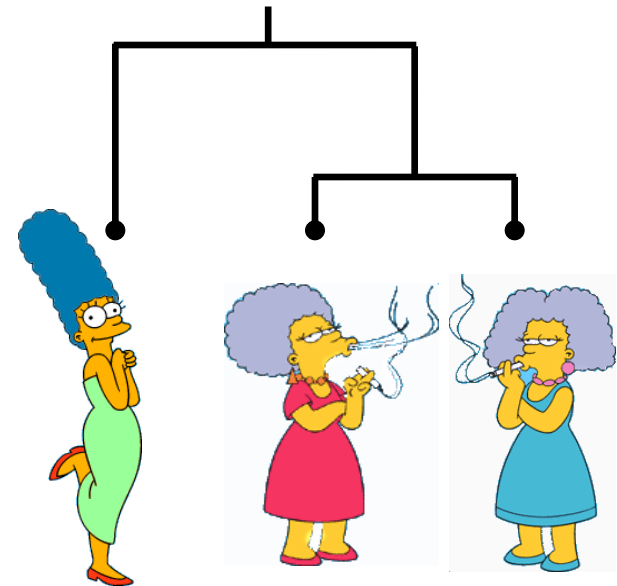
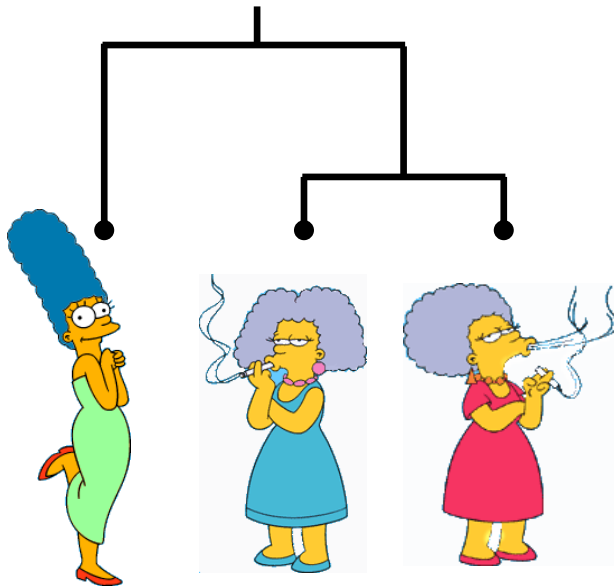
- Scalability (in terms of both time and space)
- Ability to deal with different data types
- Minimal requirements for domain knowledge to determine input parameters
- Able to deal with noise and outliers
- Insensitive to order of input records
- Incorporation of user-specified constraints
- Interpretability and usability

A Useful Tool for Summarizing Similarity Measurements

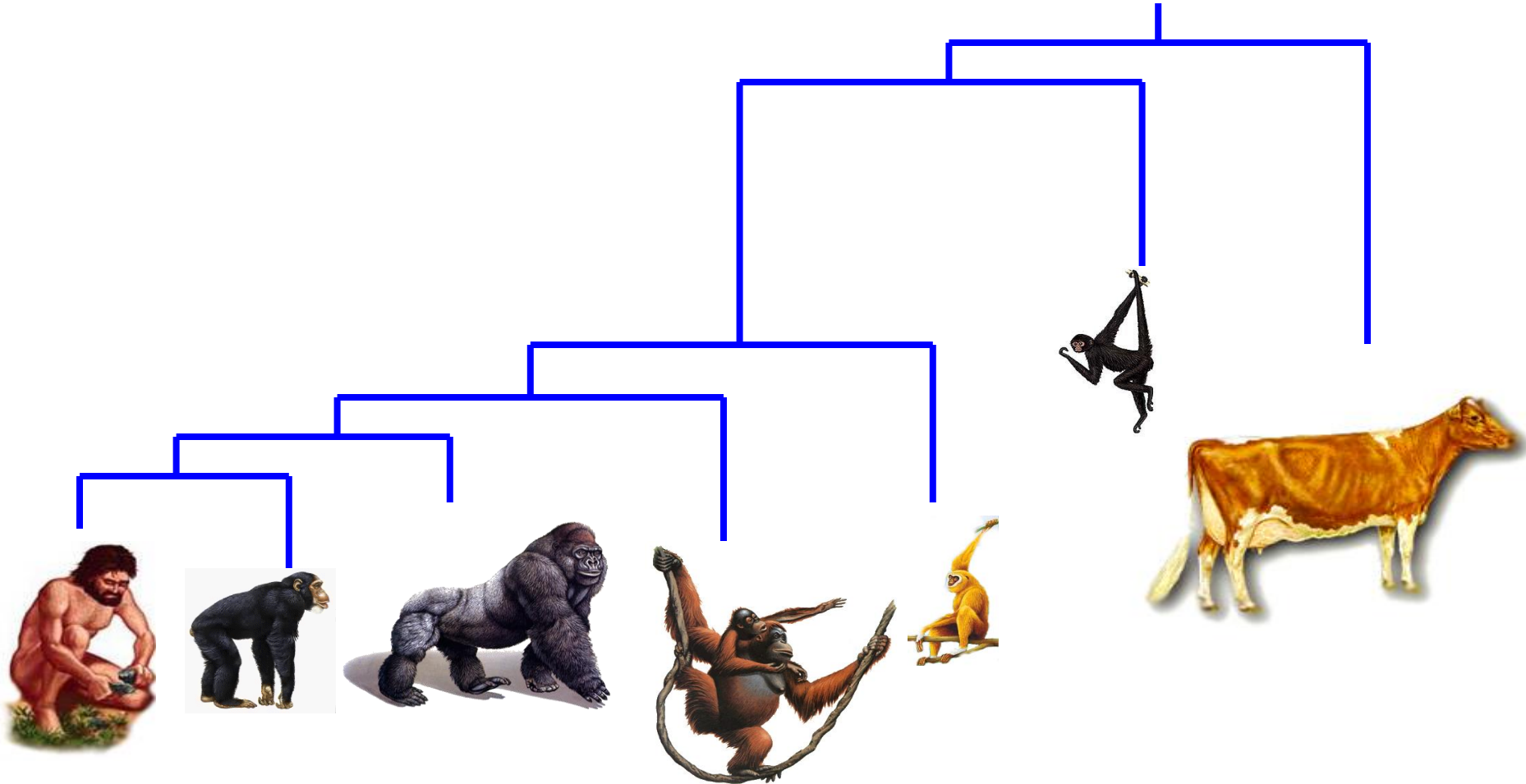
In order to better appreciate and evaluate the examples given in the early part of this talk, we will now introduce the *dendrogram*.



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



There is only one dataset that can be perfectly clustered using a hierarchy...



(Bovine:0.69395, (Spider Monkey 0.390, (Gibbon:0.36079,(Orang:0.33636,(Gorilla:0.17147,(Chimp:0.19268, Human:0.11927):0.08386):0.06124):0.15057):0.54939);

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.

Web Site Directory - Sites organized by subject

[Suggest your site](#)

Business & Economy

[B2B](#), [Finance](#), [Shopping](#), [Jobs](#)...

Regional

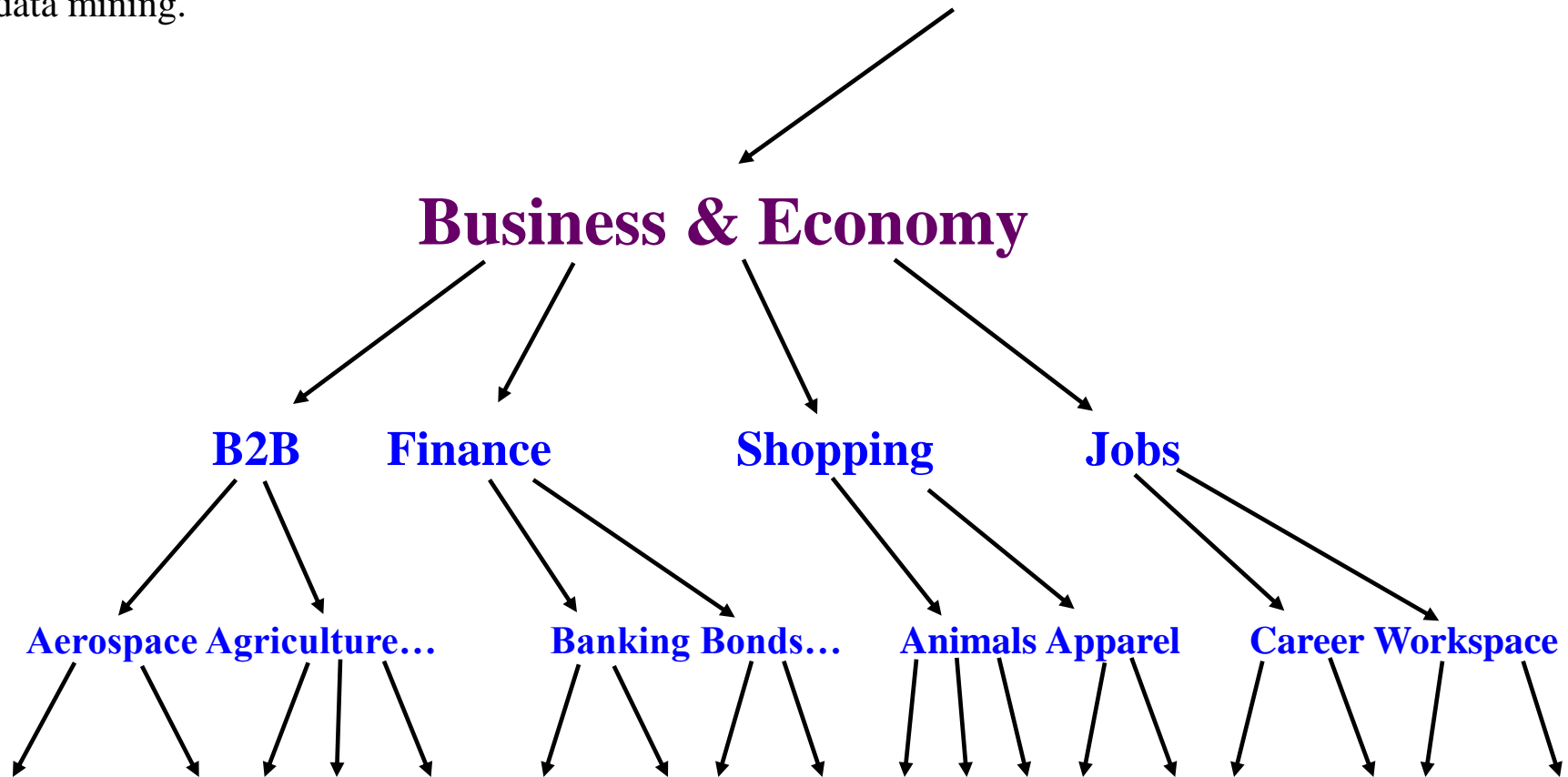
[Countries](#), [Regions](#), [US States](#)...

Computers & Internet

[Internet](#), [WWW](#), [Software](#), [Games](#)...

Society & Culture

[People](#), [Environment](#), [Religion](#)...



A Demonstration of Hierarchical Clustering using String Edit Distance

Pedro (Portuguese)

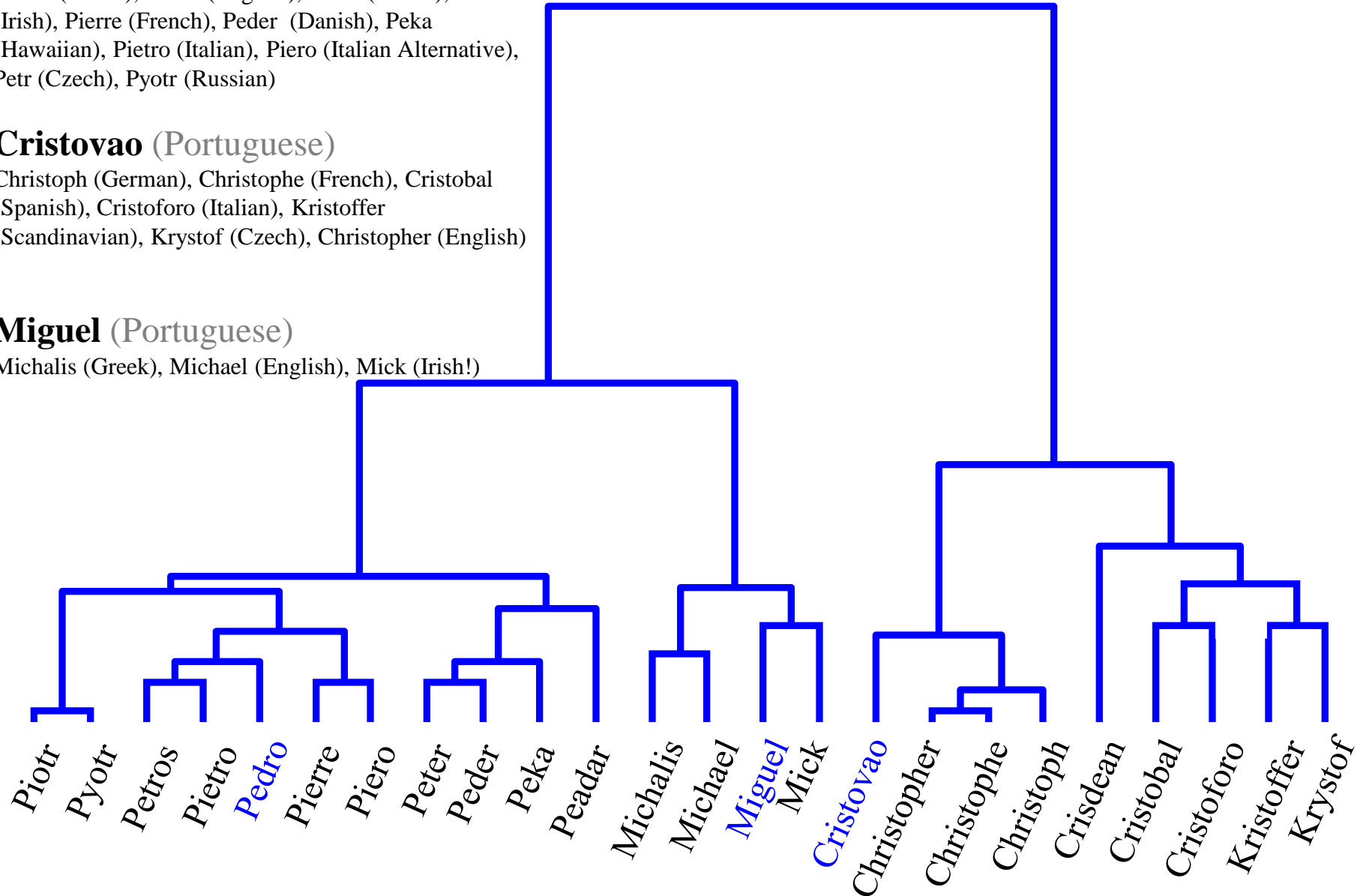
Petros (Greek), Peter (English), Piotr (Polish), Peadar (Irish), Pierre (French), Peder (Danish), Peka (Hawaiian), Pietro (Italian), Piero (Italian Alternative), Petr (Czech), Pyotr (Russian)

Cristovao (Portuguese)

Christoph (German), Christophe (French), Cristobal (Spanish), Cristoforo (Italian), Kristoffer (Scandinavian), Krystof (Czech), Christopher (English)

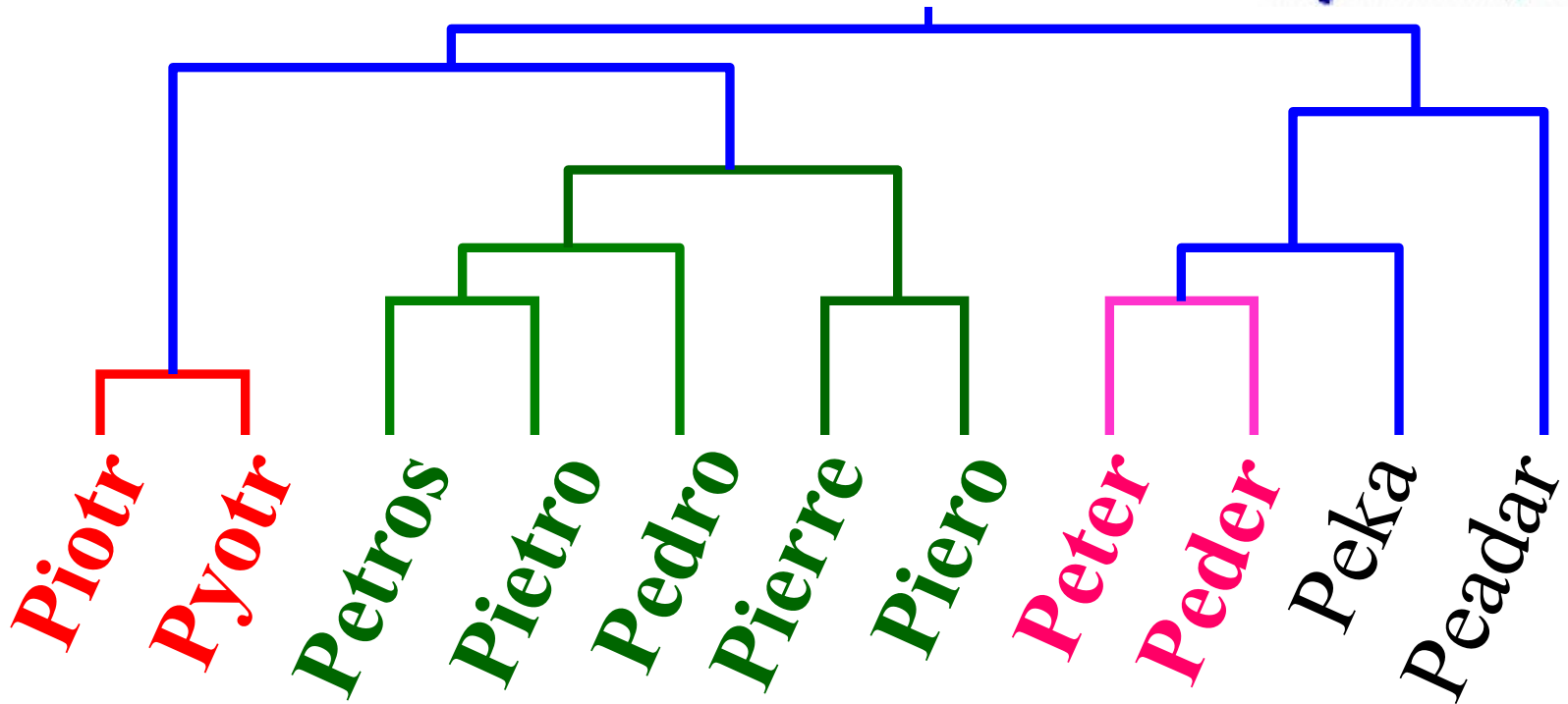
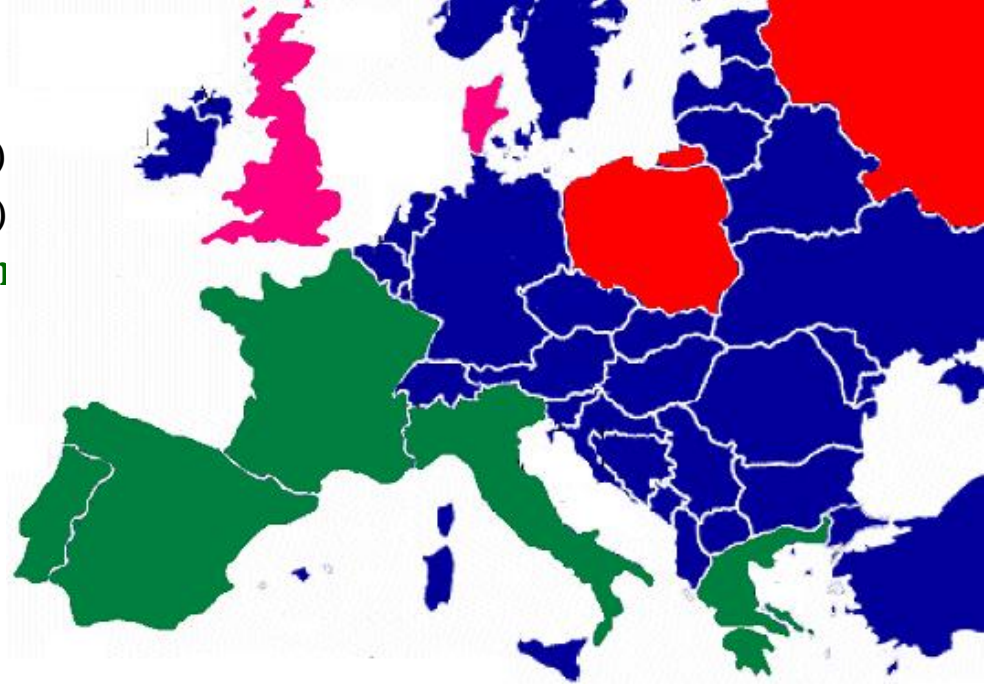
Miguel (Portuguese)

Michalis (Greek), Michael (English), Mick (Irish!)



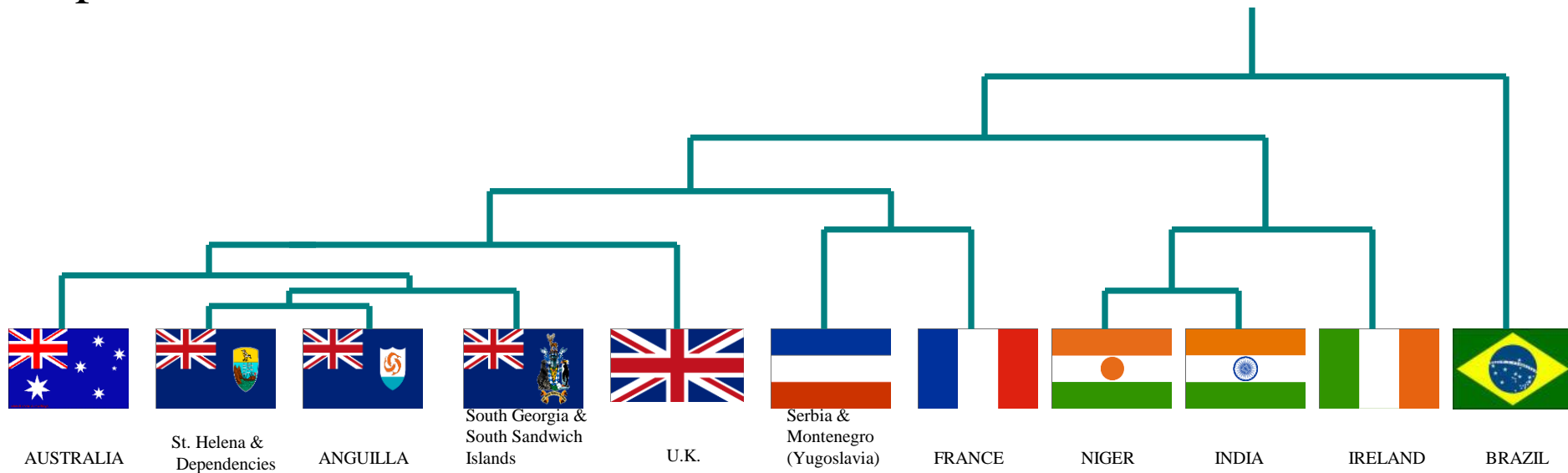
Pedro (Portuguese/Spanish)

Petros (**Greek**), Peter (**English**), Piotr (**Polish**)
Peadar (Irish), Pierre (**French**), Peder (**Danish**)
Peka (Hawaiian), Pietro (**Italian**), Piero (**Italian Alternative**), Petr (Czech), Pyotr (**Russian**)



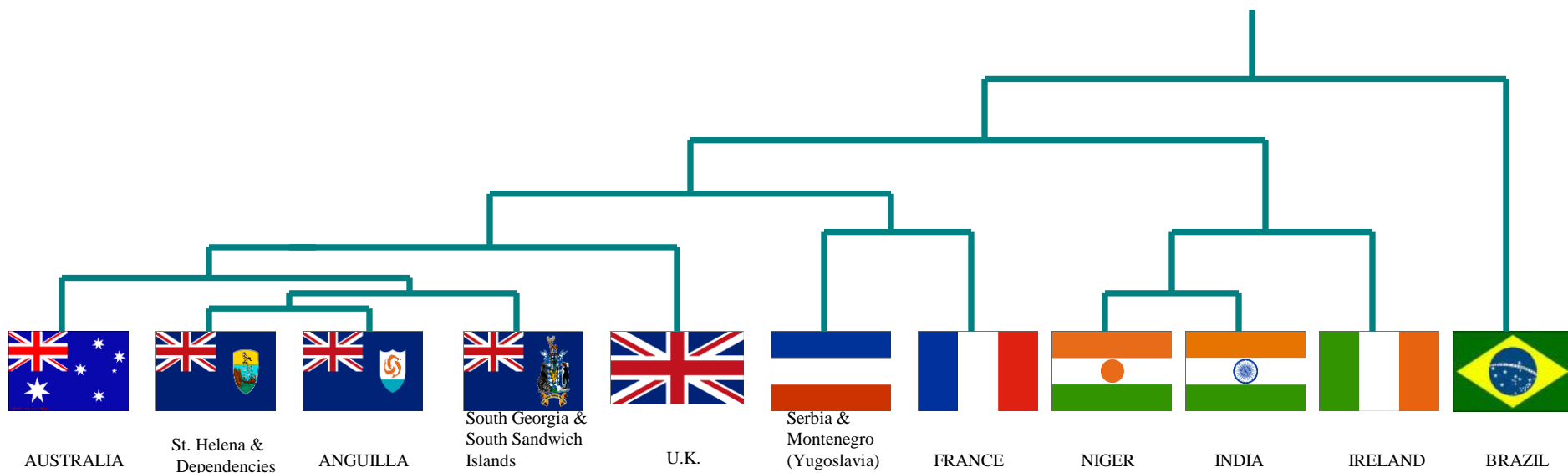
Hierarchical clustering can sometimes show patterns that are meaningless or spurious

- For example, in this clustering, the tight grouping of Australia, Anguilla, St. Helena etc is meaningful, since all these countries are former UK colonies.
- However the tight grouping of Niger and India is completely spurious, there is no connection between the two.

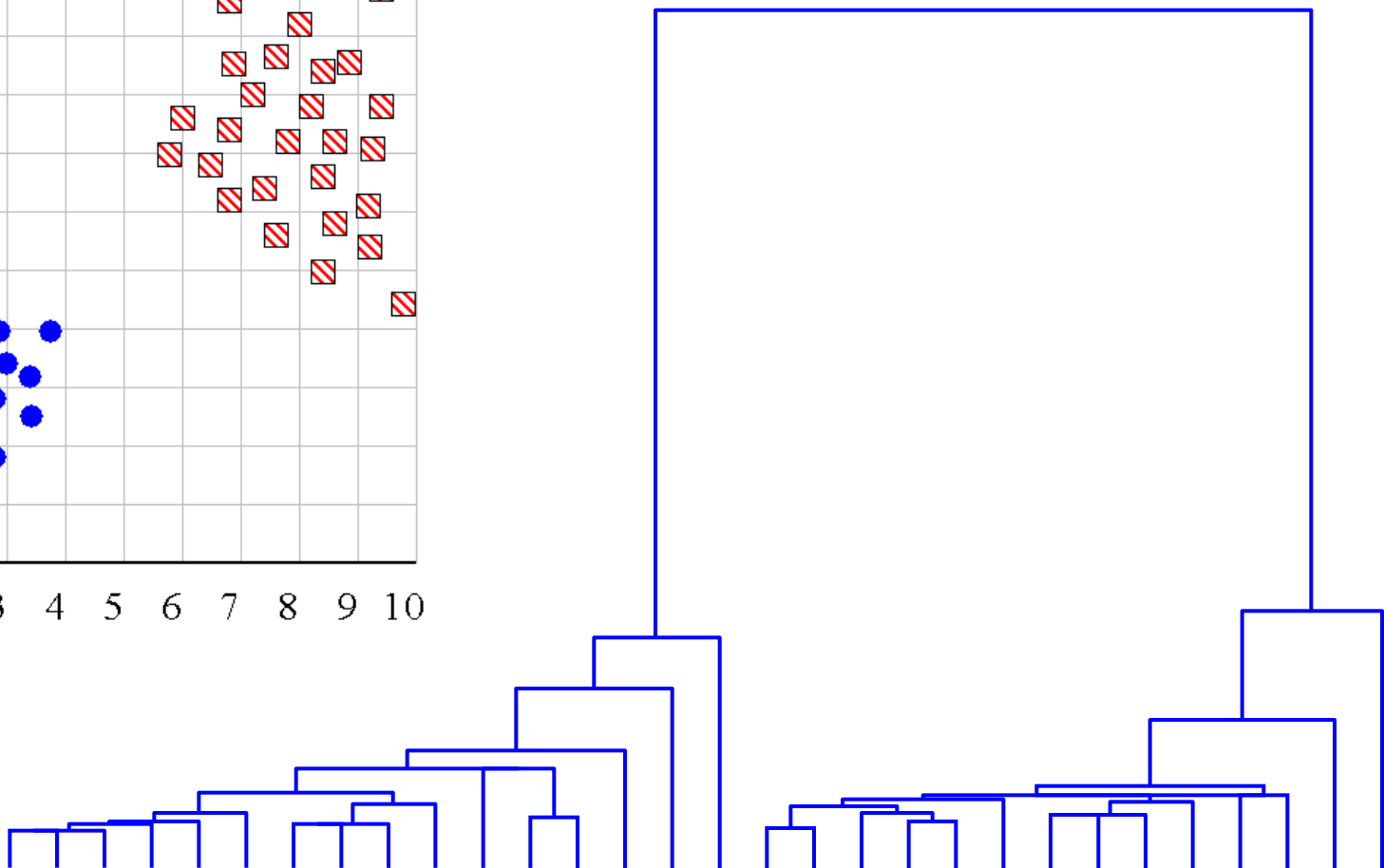
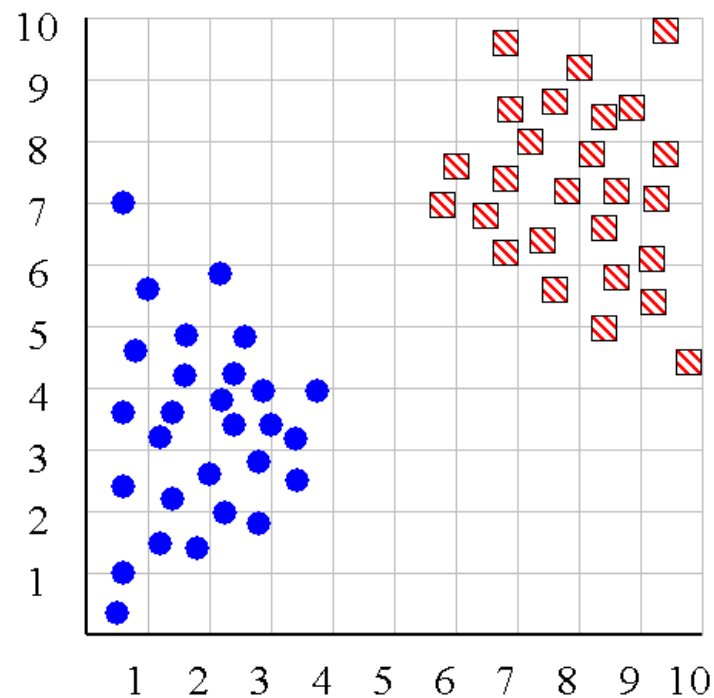


- The flag of Niger is orange over white over green, with an orange disc on the central white stripe, symbolizing the sun. The orange stands the Sahara desert, which borders Niger to the north. Green stands for the grassy plains of the south and west and for the River Niger which sustains them. It also stands for fraternity and hope. White generally symbolizes purity and hope.

- The Indian flag is a horizontal tricolor in equal proportion of deep saffron on the top, white in the middle and dark green at the bottom. In the center of the white band, there is a wheel in navy blue to indicate the Dharma Chakra, the wheel of law in the Sarnath Lion Capital. This center symbol or the 'CHAKRA' is a symbol dating back to 2nd century BC. The saffron stands for courage and sacrifice; the white, for purity and truth; the green for growth and auspiciousness.

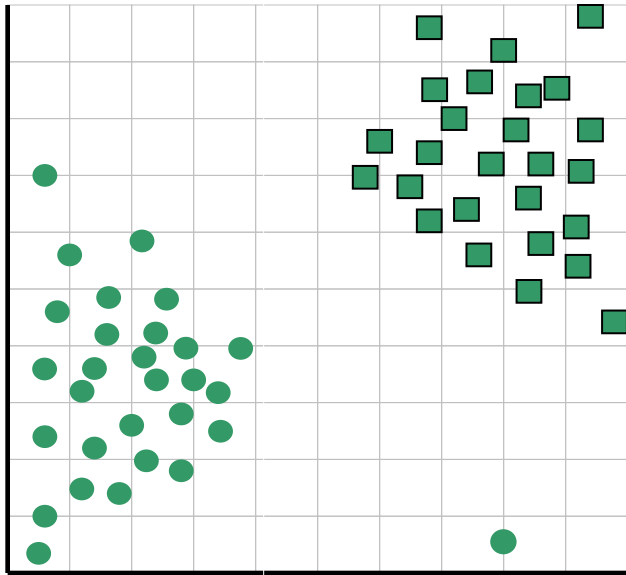


We can look at the dendrogram to determine the “correct” number of clusters. In this case, the two highly separated subtrees are highly suggestive of two clusters. (Things are rarely this clear cut, unfortunately)

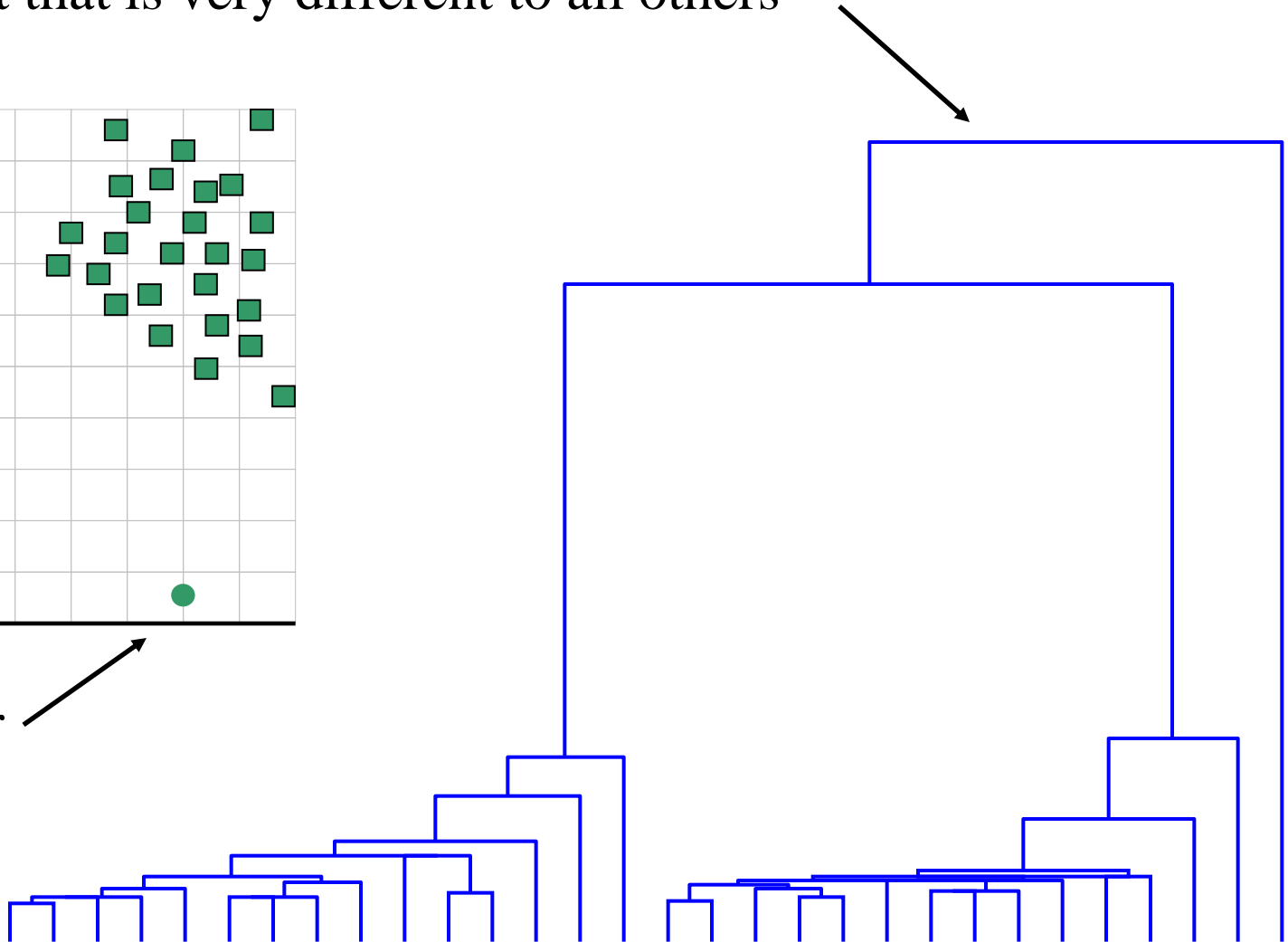


One potential use of a dendrogram is to detect outliers

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



Outlier



(How-to) Hierarchical Clustering

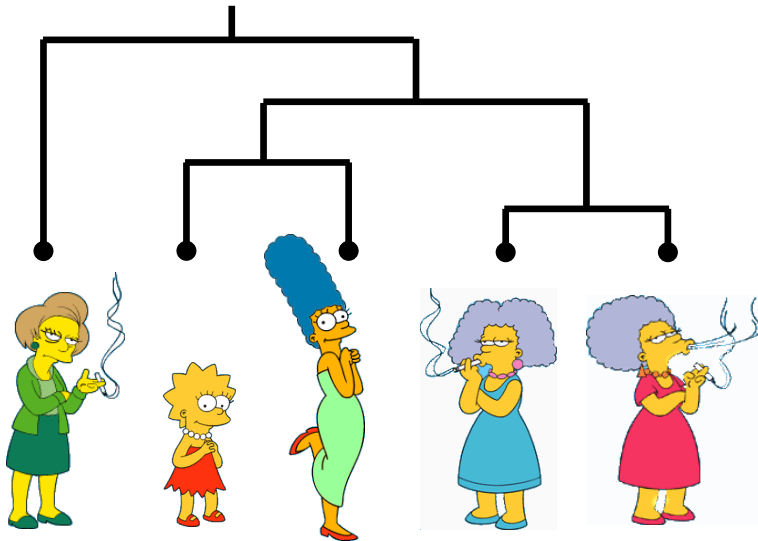
The number of dendrograms with n leaves = $(2n - 3)! / [(2^{(n-2)}) (n - 2)!]$

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.

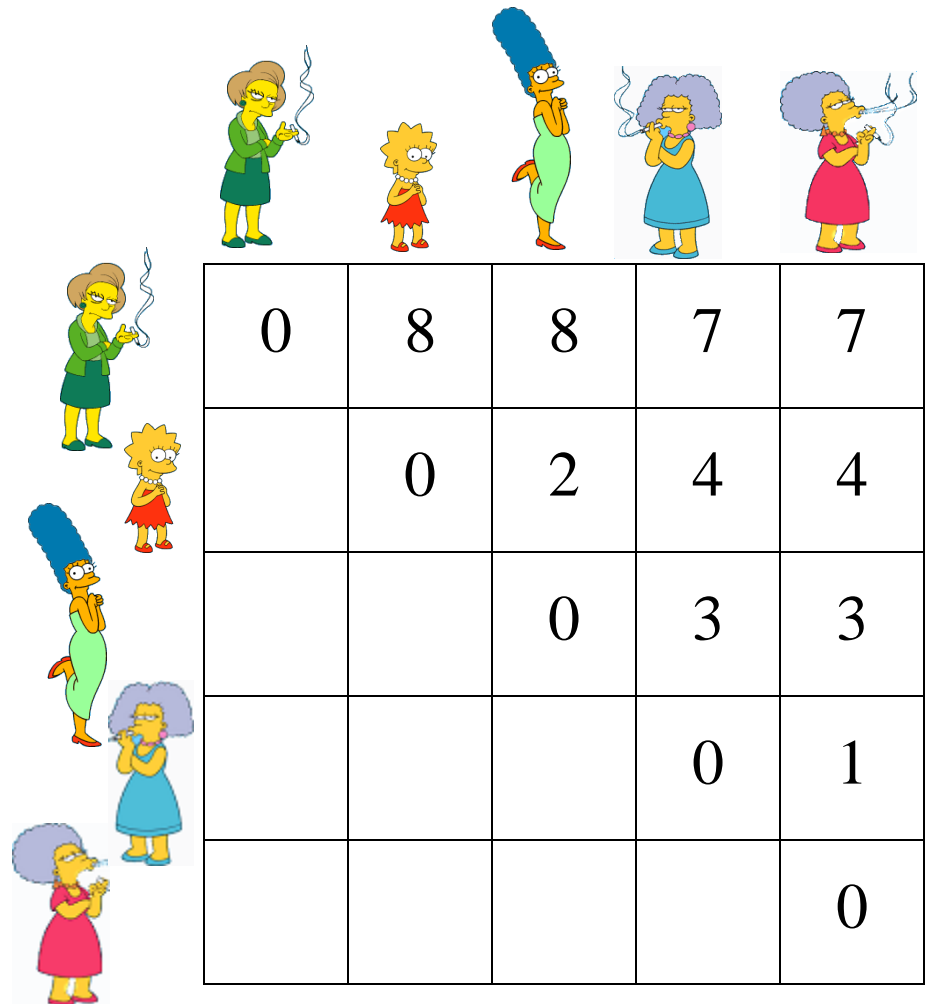
Top-Down (divisive): Starting with all the data in a single cluster, consider every possible way to divide the cluster into two. Choose the best division and recursively operate on both sides.















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

$$D(\text{Mrs. Krabappel, Lisa Simpson}) = 8$$

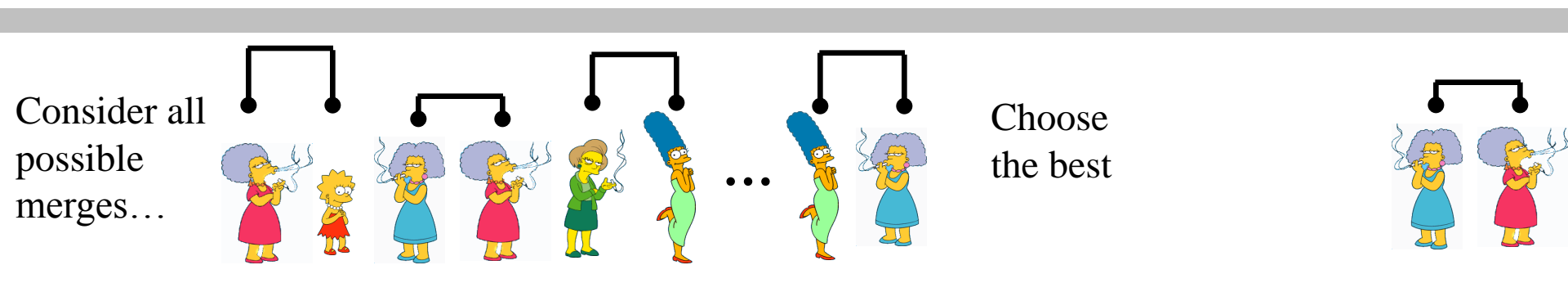
$$D(\text{Marge Simpson, Edna Krabappel}) = 1$$



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

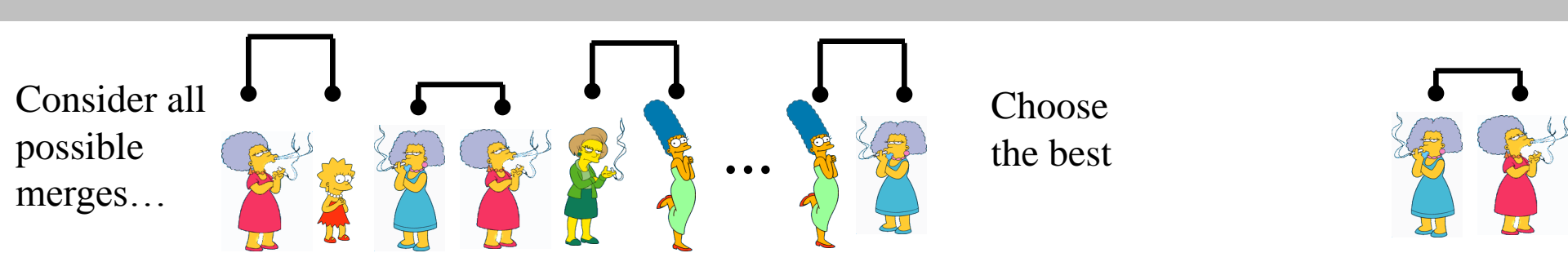
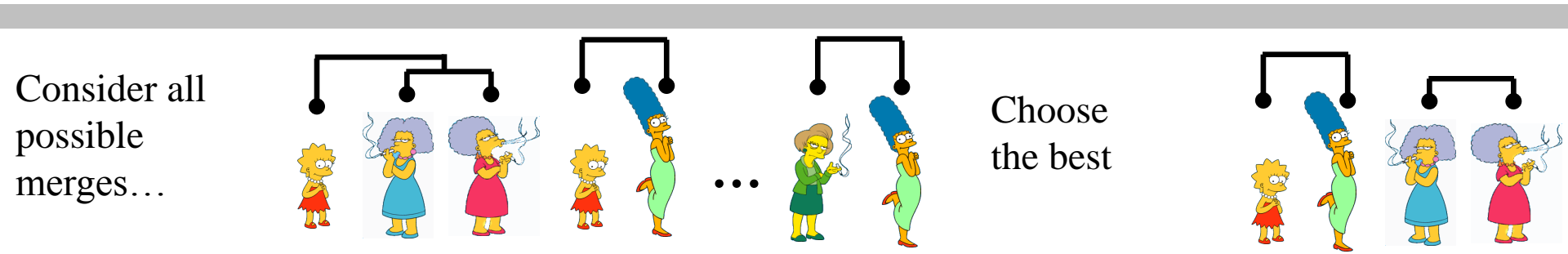
Bottom-Up (agglomerative):

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.



Bottom-Up (agglomerative):

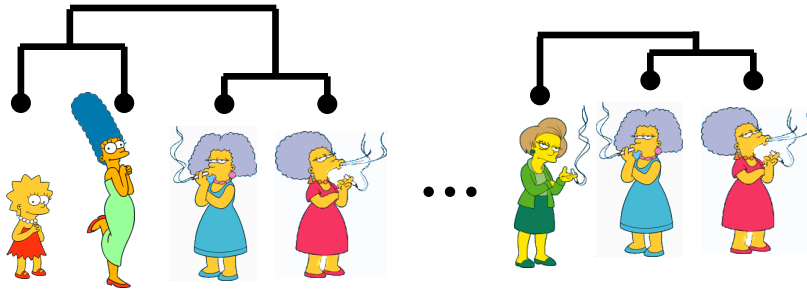
Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.



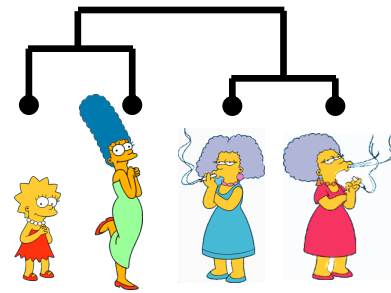
Bottom-Up (agglomerative):

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.

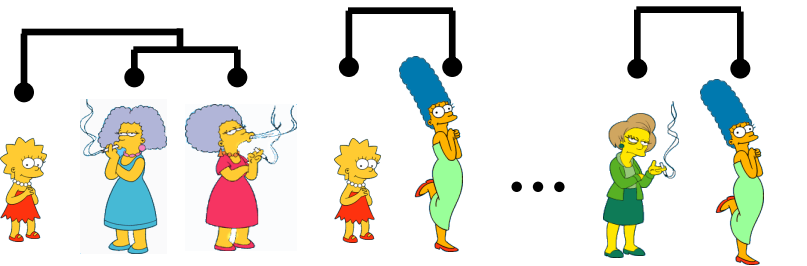
Consider all possible merges...



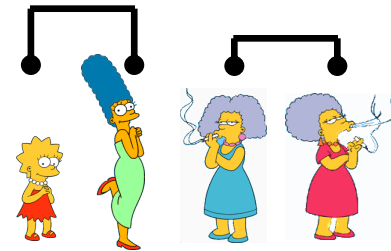
Choose the best



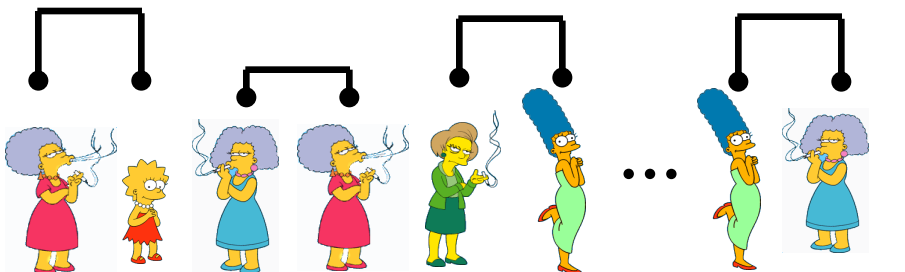
Consider all possible merges...



Choose the best



Consider all possible merges...

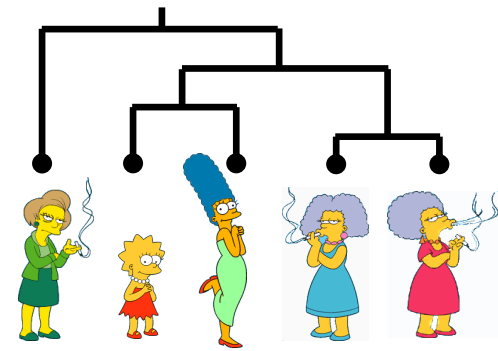


Choose the best

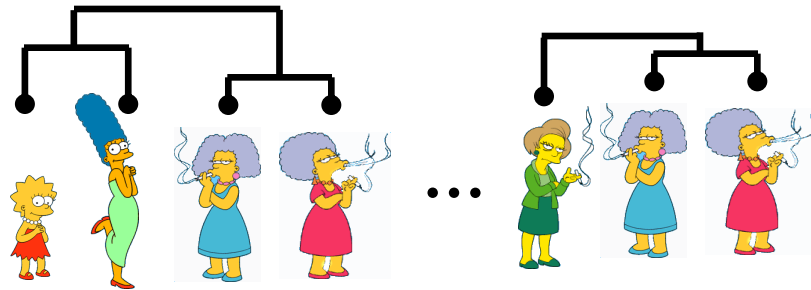


Bottom-Up (agglomerative):

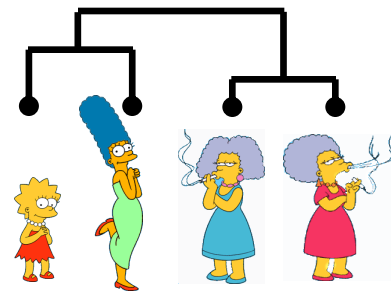
Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.



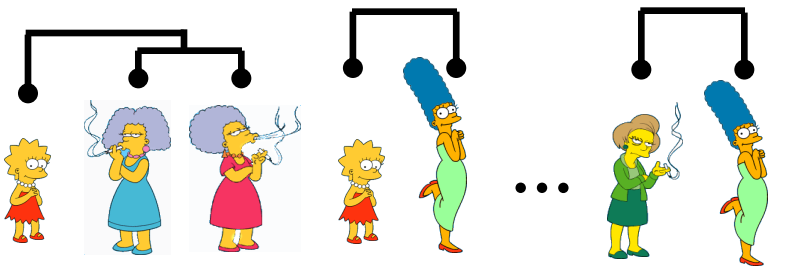
Consider all possible merges...



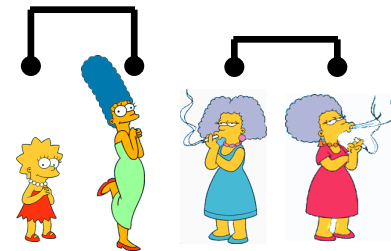
Choose the best



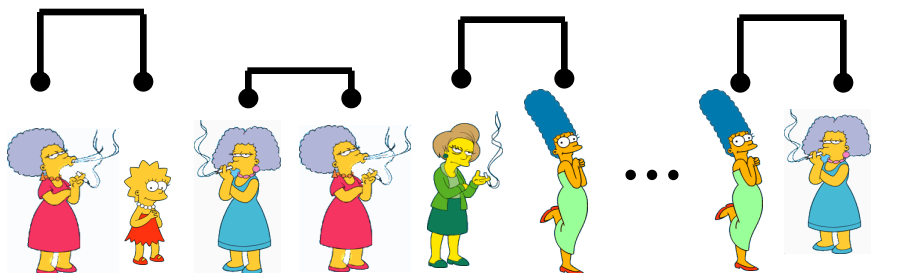
Consider all possible merges...



Choose the best



Consider all possible merges...

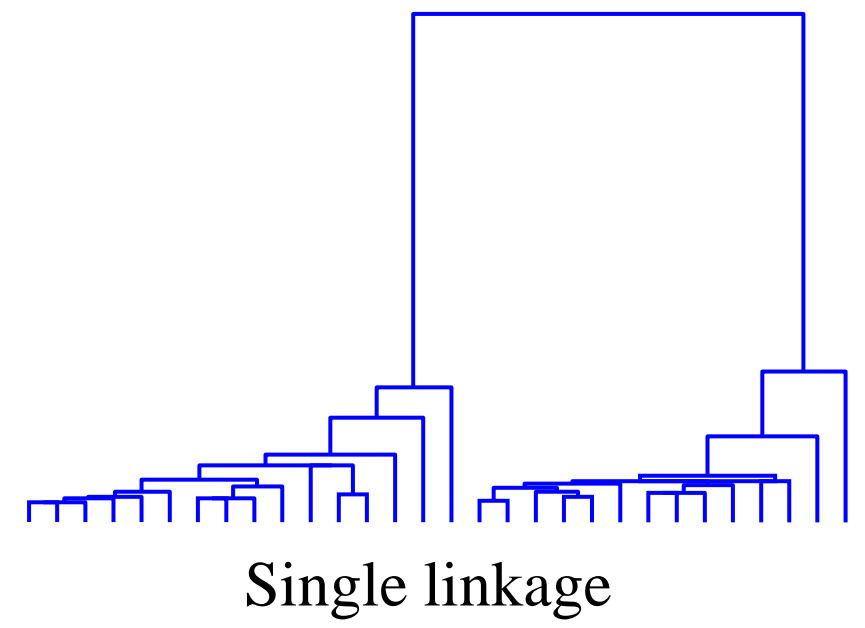
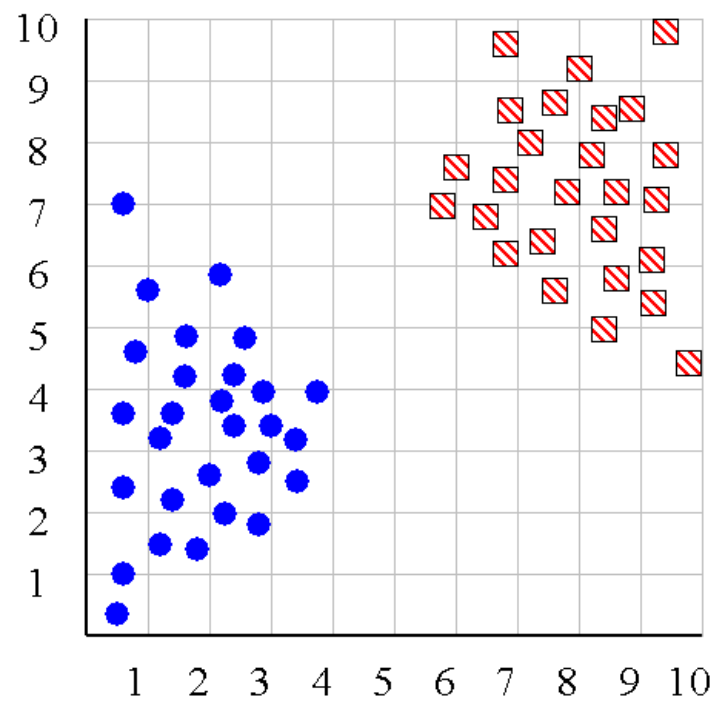


Choose the best

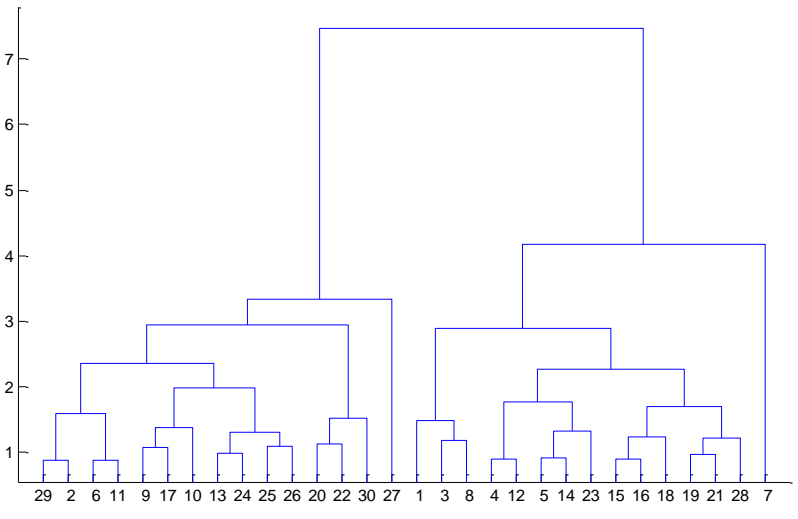


We know how to measure the distance between two objects, but defining the distance between an object and a cluster, or defining the distance between two clusters is non obvious.

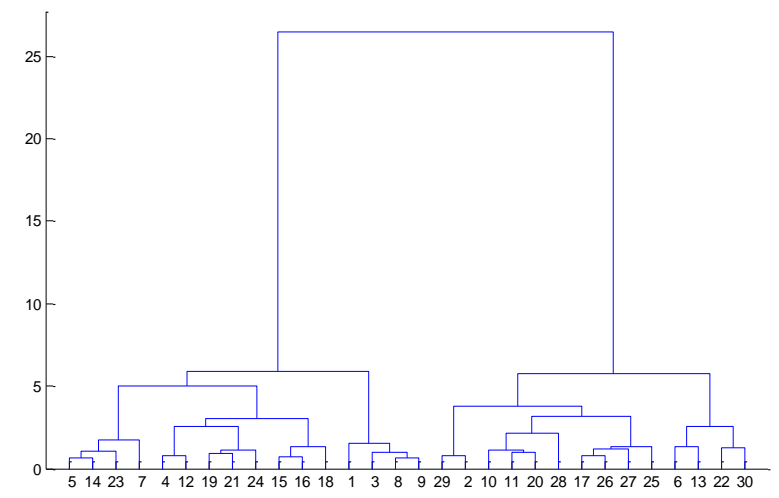
- **Single linkage (nearest neighbor):** In this method the distance between two clusters is determined by the distance of the two closest objects (nearest neighbors) in the different clusters.
- **Complete linkage (furthest neighbor):** In this method, the distances between clusters are determined by the greatest distance between any two objects in the different clusters (i.e., by the "furthest neighbors").
- **Group average linkage:** In this method, the distance between two clusters is calculated as the average distance between all pairs of objects in the two different clusters.
- **Wards Linkage:** In this method, we try to minimize the variance of the merged clusters



Single linkage



Average linkage



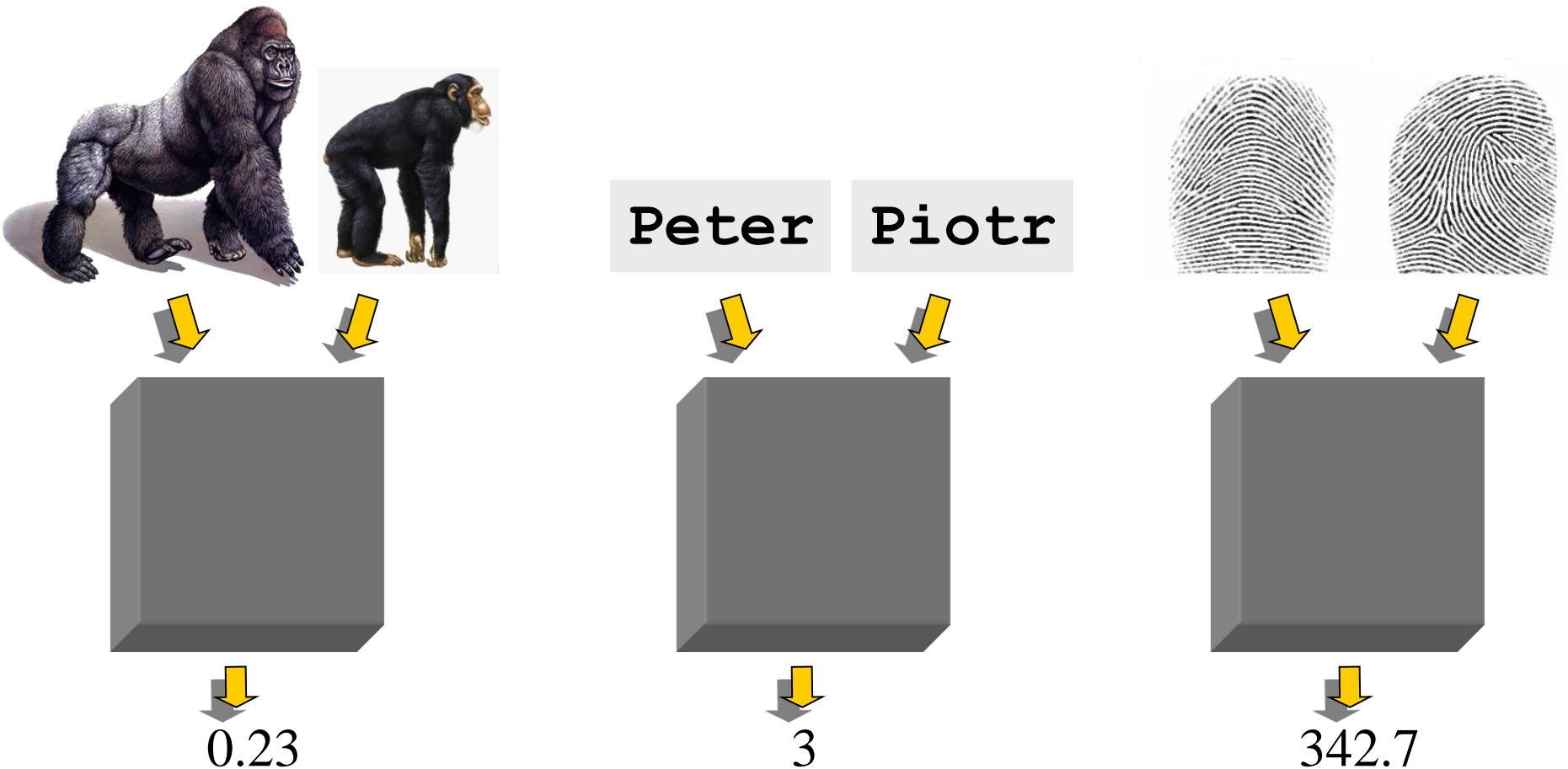
Wards linkage

Summary of Hierarchical Clustering Methods

- No need to specify the number of clusters in advance.
- Hierarchical nature 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.

Up to this point we have simply assumed that we can measure similarity, but

How do we measure similarity?



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.

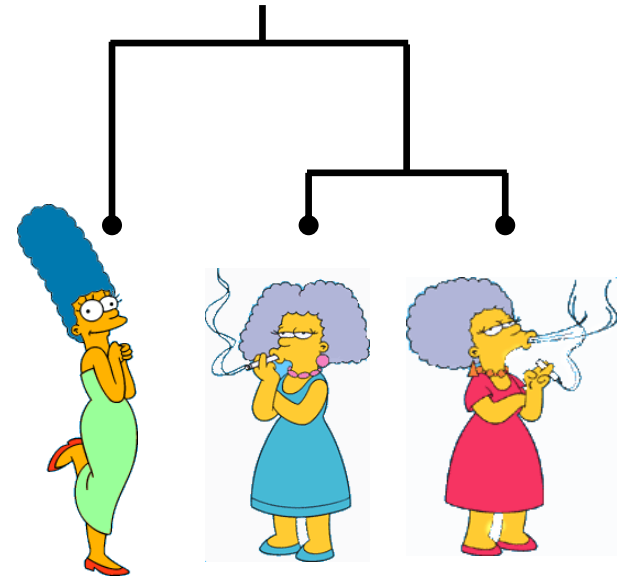
- Change dress color, 1 point
- Change earring shape, 1 point
- Change hair part, 1 point

$$D(\text{Patty}, \text{Selma}) = 3$$

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

$$D(\text{Marge}, \text{Selma}) = 5$$



This is called the “edit distance” or the “transformation distance”

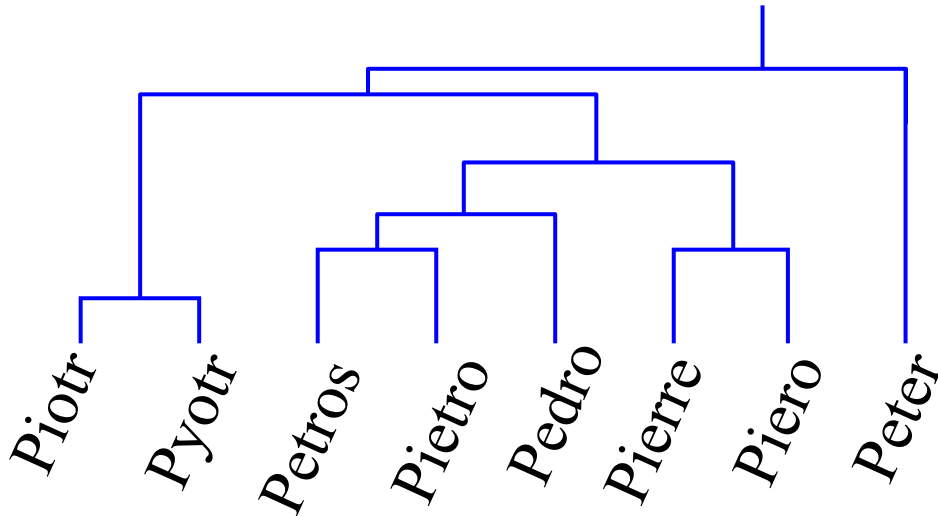
Edit Distance Example

It is possible to transform any string Q into string C , using only *Substitution*, *Insertion* and *Deletion*.

Assume that each of these operators has a cost associated with it.

The similarity between two strings can be defined as the cost of the cheapest transformation from Q to C .

Note that for now we have ignored the issue of how we can find this cheapest transformation



How similar are the names “Peter” and “Piotr”?

Assume the following cost function

<i>Substitution</i>	1 Unit
<i>Insertion</i>	1 Unit
<i>Deletion</i>	1 Unit

$D(\text{Peter}, \text{Piotr})$ is 3

Peter



Substitution (i for e)

Piter



Insertion (o)

Pioter

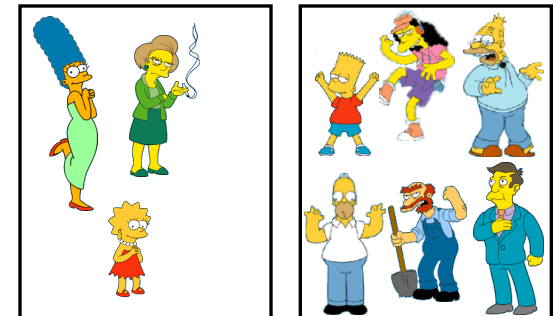
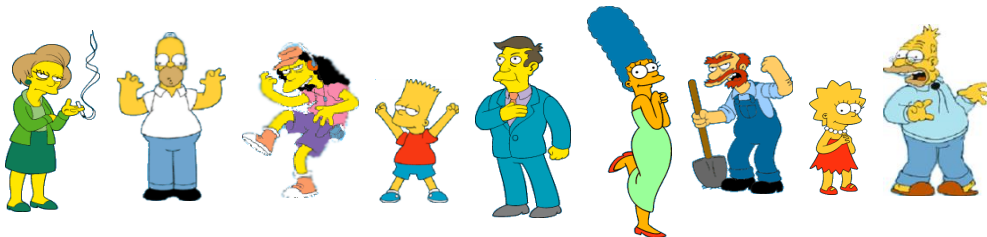


Deletion (e)

Piotr

Partitional Clustering

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



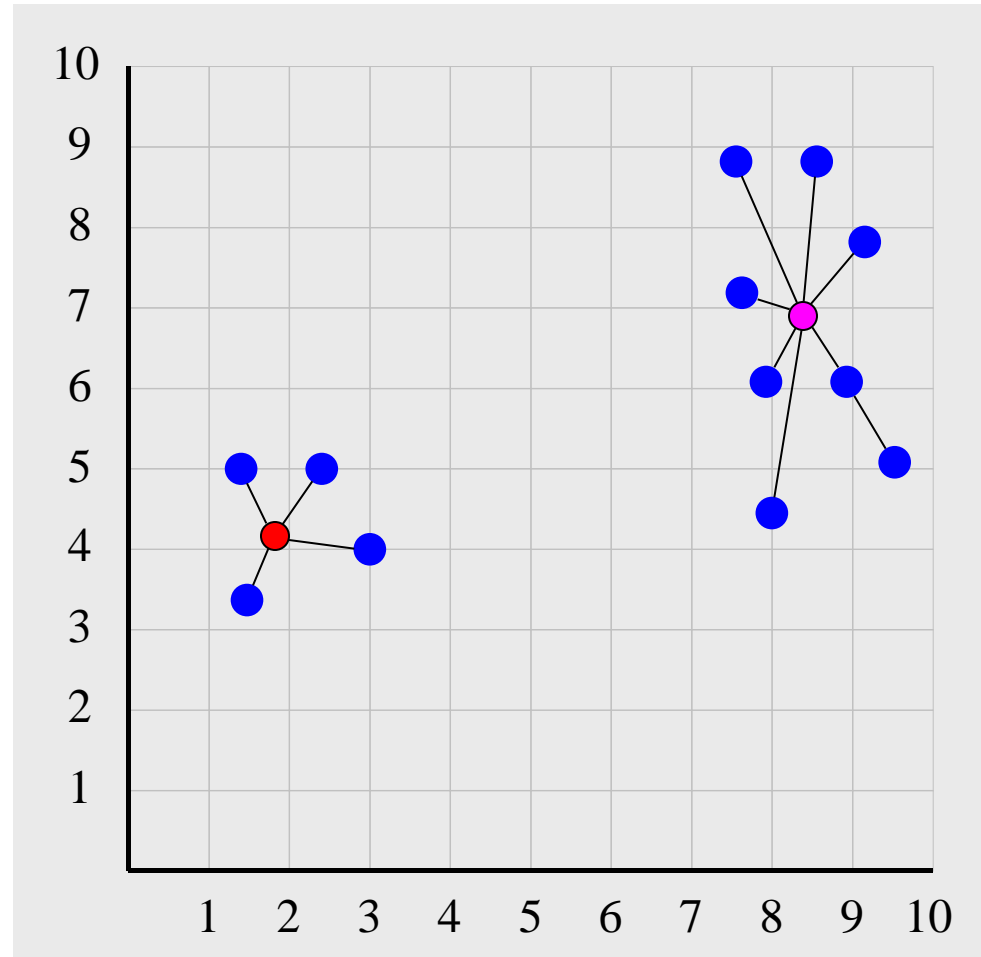
Squared Error

$$se_{K_i} = \sum_{j=1}^m \|t_{ij} - C_k\|^2$$

$$se_K = \sum_{j=1}^k se_{K_j}$$



Objective Function

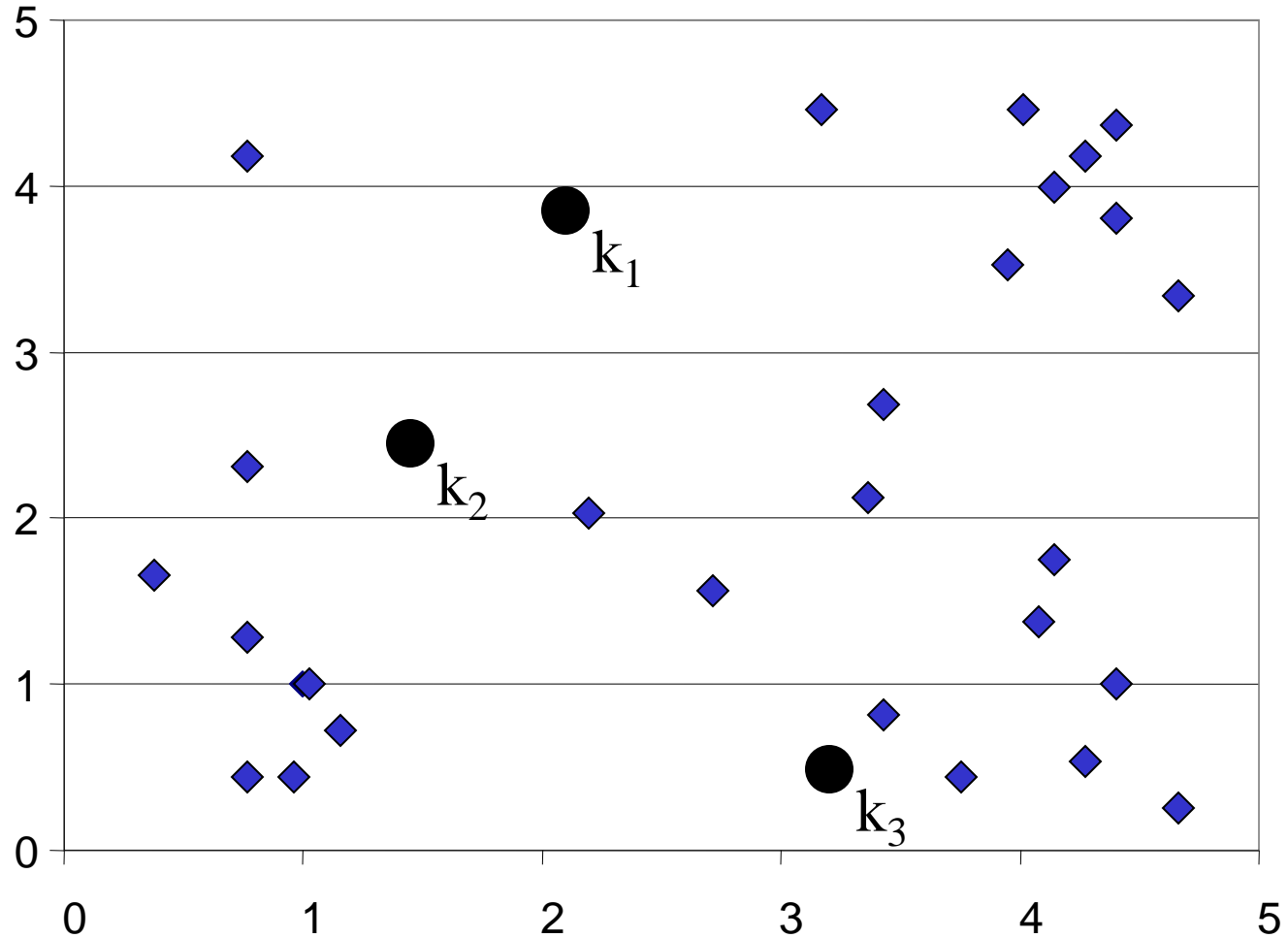


Algorithm *k-means*

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 center.
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 goto 3.

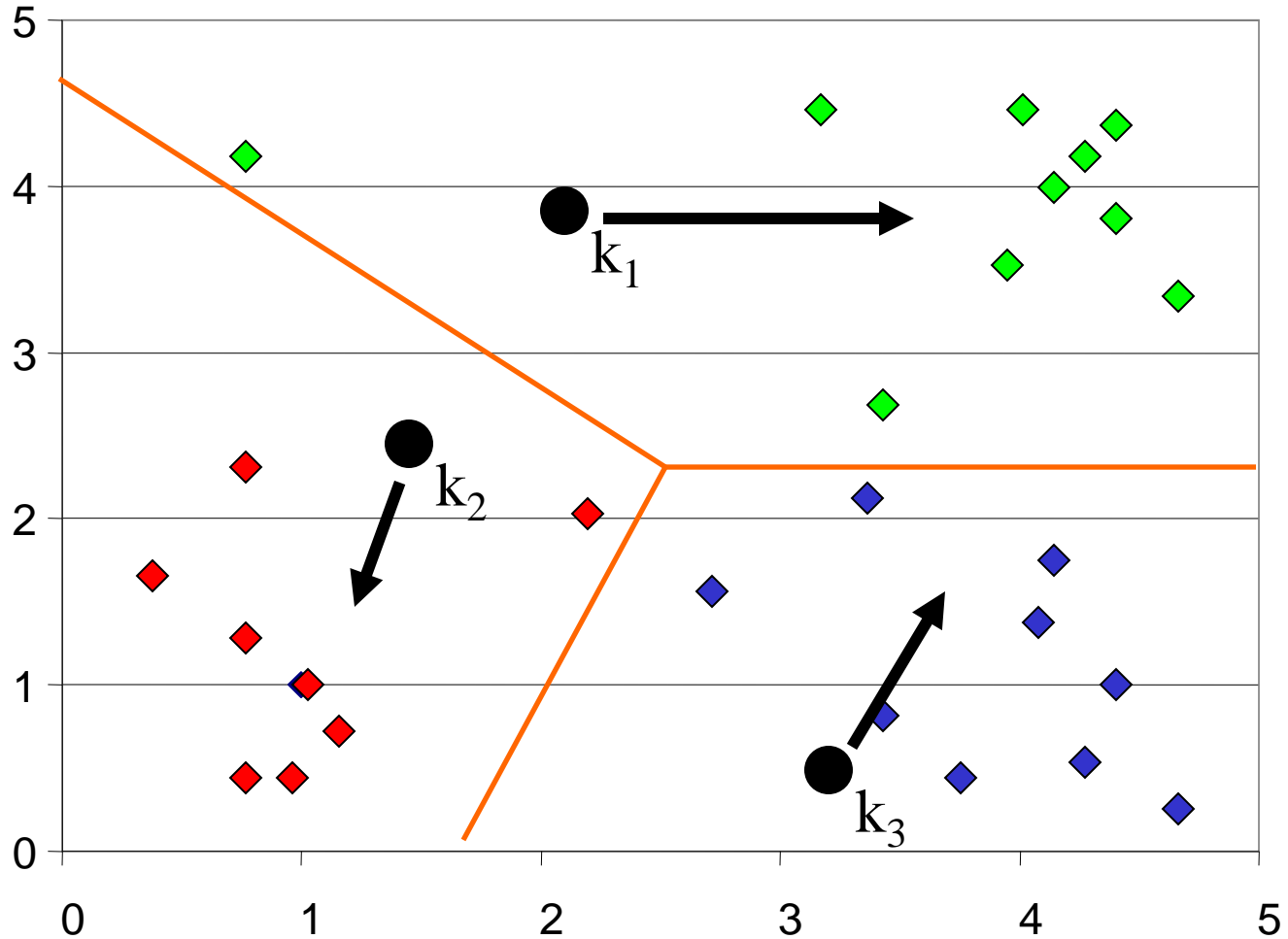
K-means Clustering: Step 1

Algorithm: k-means, Distance Metric: Euclidean Distance



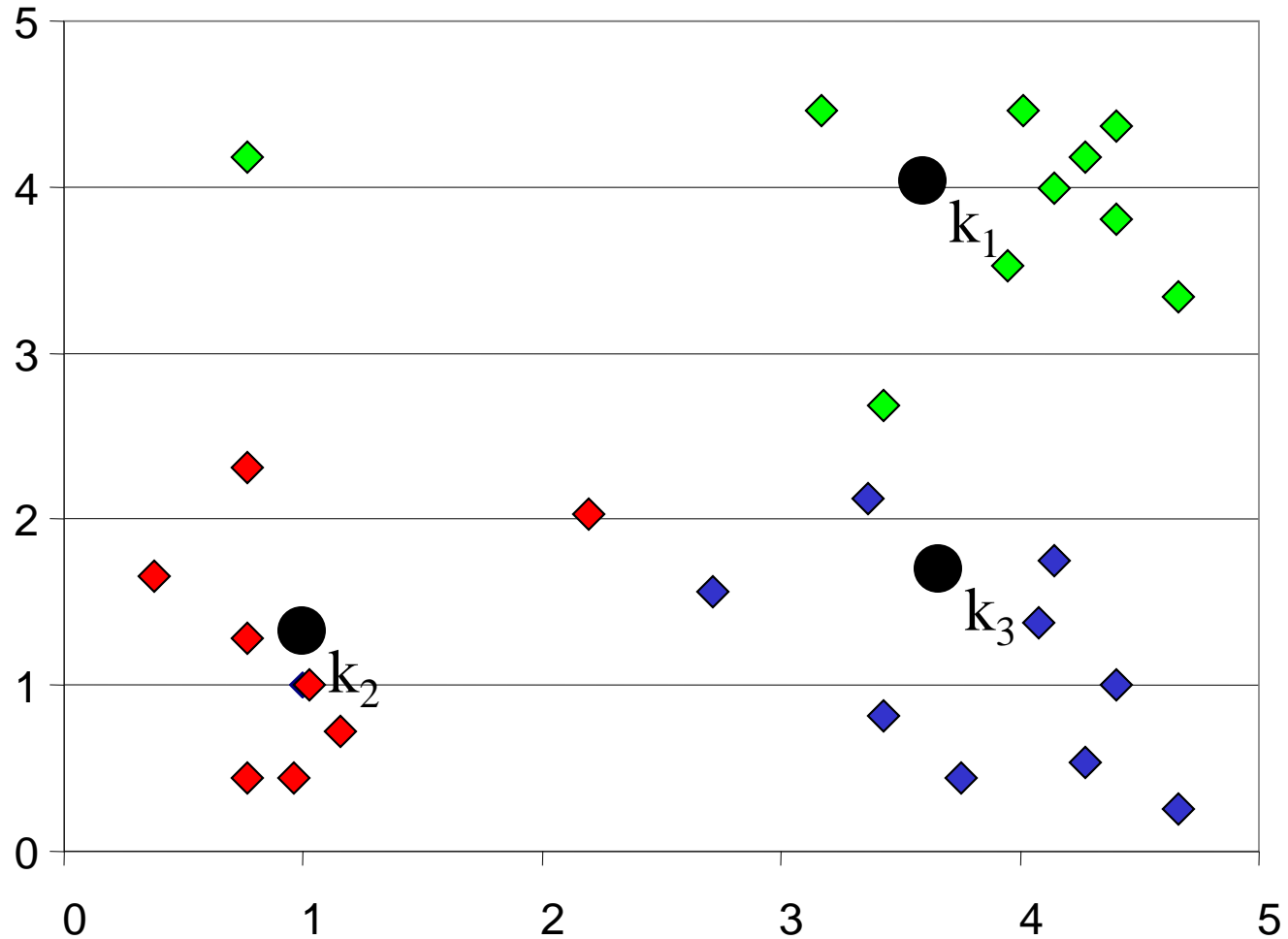
K-means Clustering: Step 2

Algorithm: k-means, Distance Metric: Euclidean Distance



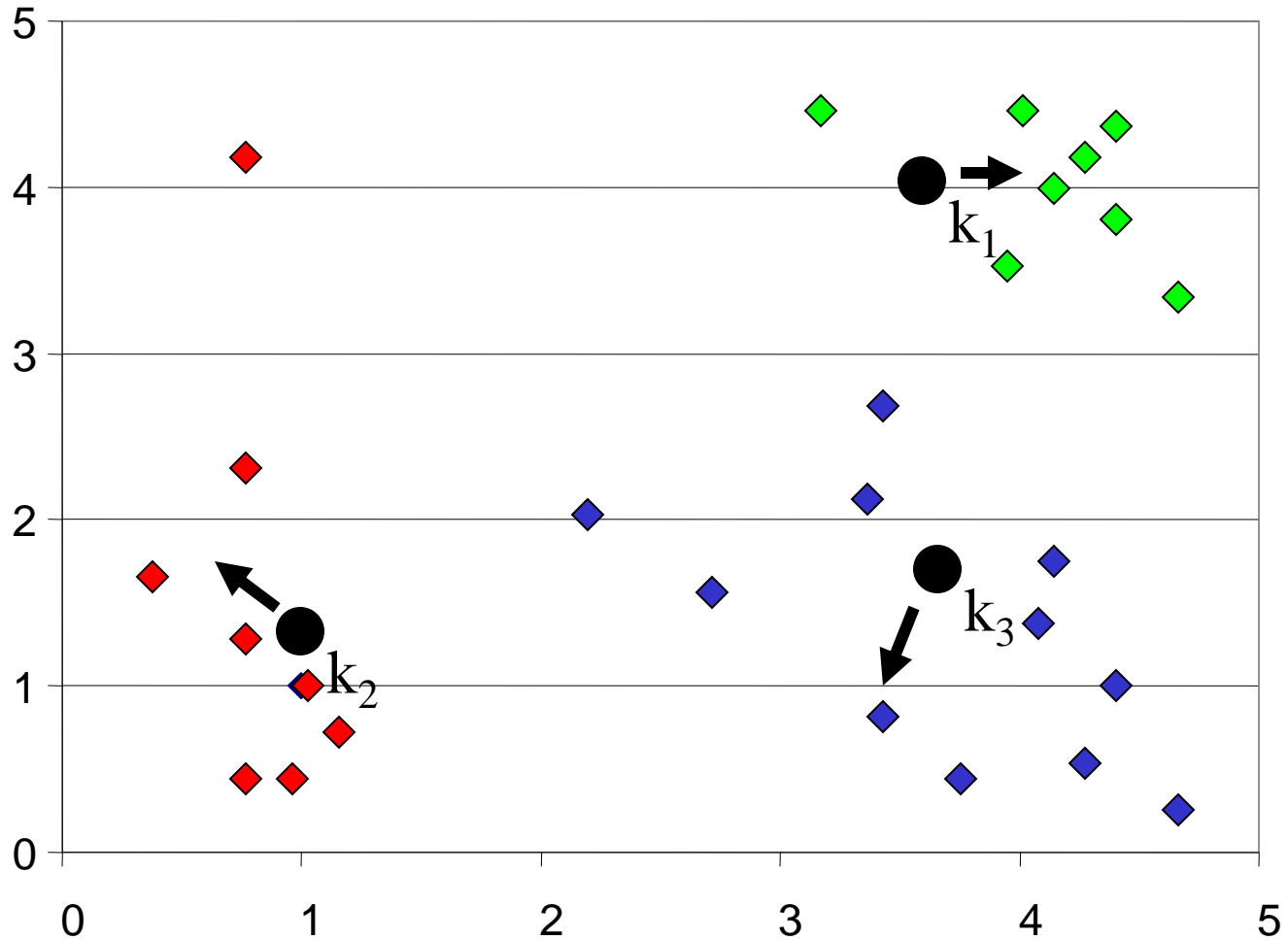
K-means Clustering: Step 3

Algorithm: k-means, Distance Metric: Euclidean Distance



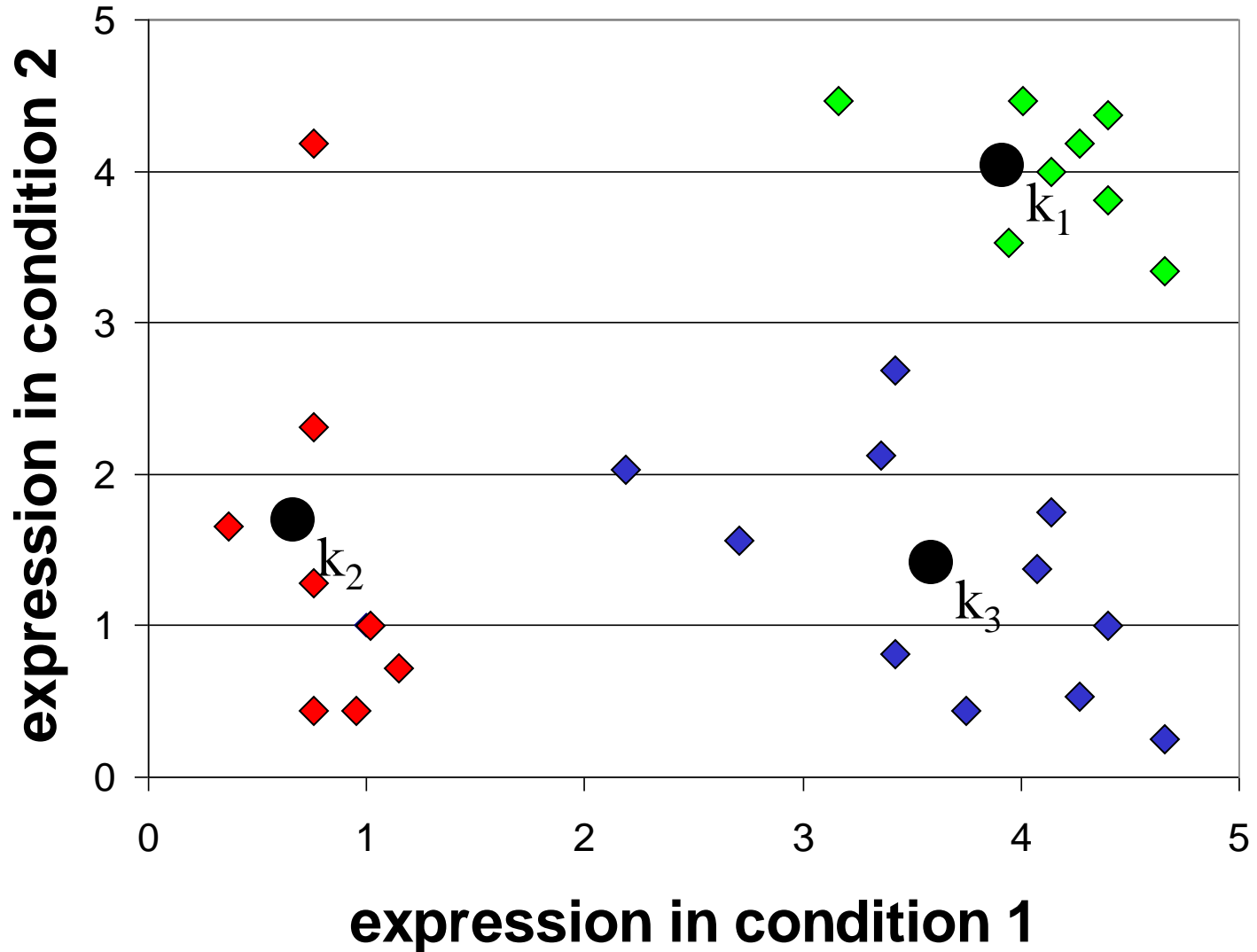
K-means Clustering: Step 4

Algorithm: k-means, Distance Metric: Euclidean Distance



K-means Clustering: Step 5

Algorithm: k-means, Distance Metric: Euclidean Distance



Comments on the *K-Means* Method

- Strength

- *Relatively efficient: $O(tkn)$, where n is # objects, k is # clusters, and t is # iterations. Normally, $k, t \ll n$.*
- Often terminates at a *local optimum*. The *global optimum* may be found using techniques such as: *deterministic annealing* and *genetic algorithms*

- Weakness

- Applicable only when *mean* is defined, then what about categorical data?
- 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*

The *K-Medoids* Clustering Method

- Find *representative* objects, called medoids, in clusters
- *PAM* (Partitioning Around Medoids, 1987)
 - starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering
 - *PAM* works effectively for small data sets, but does not scale well for large data sets

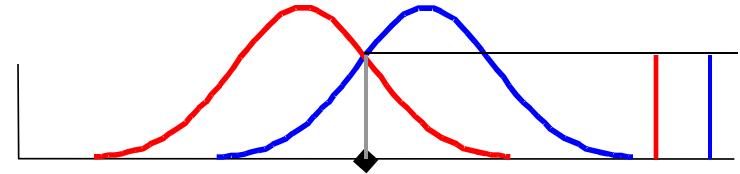
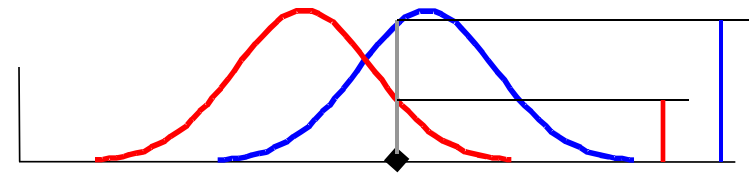
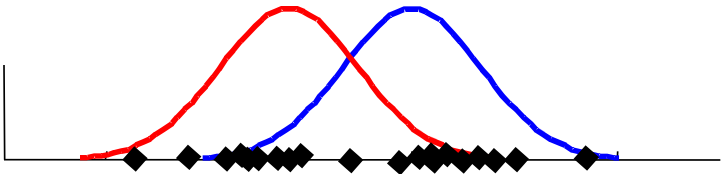
EM Algorithm

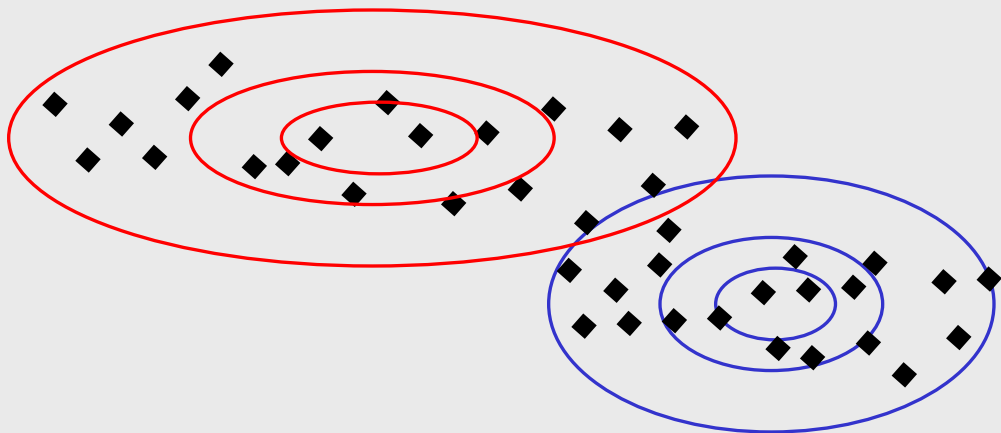
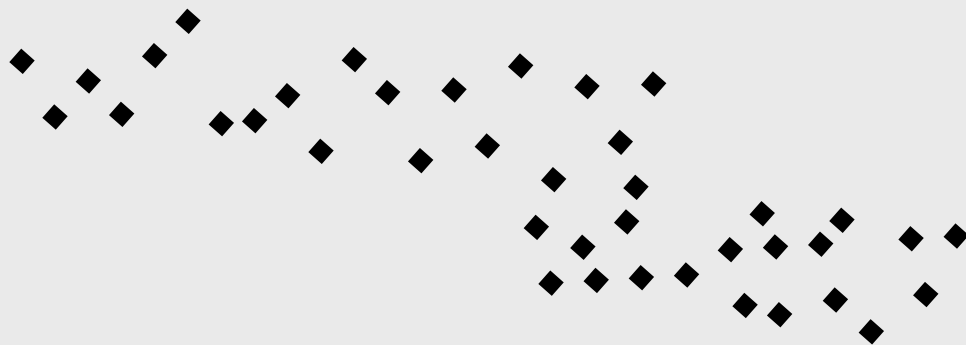
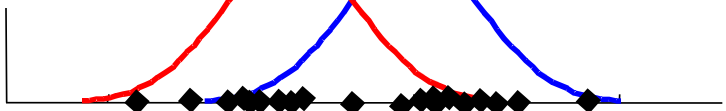
- Initialize K cluster centers
- Iterate between two steps
 - **E**xpectation step: assign points to clusters

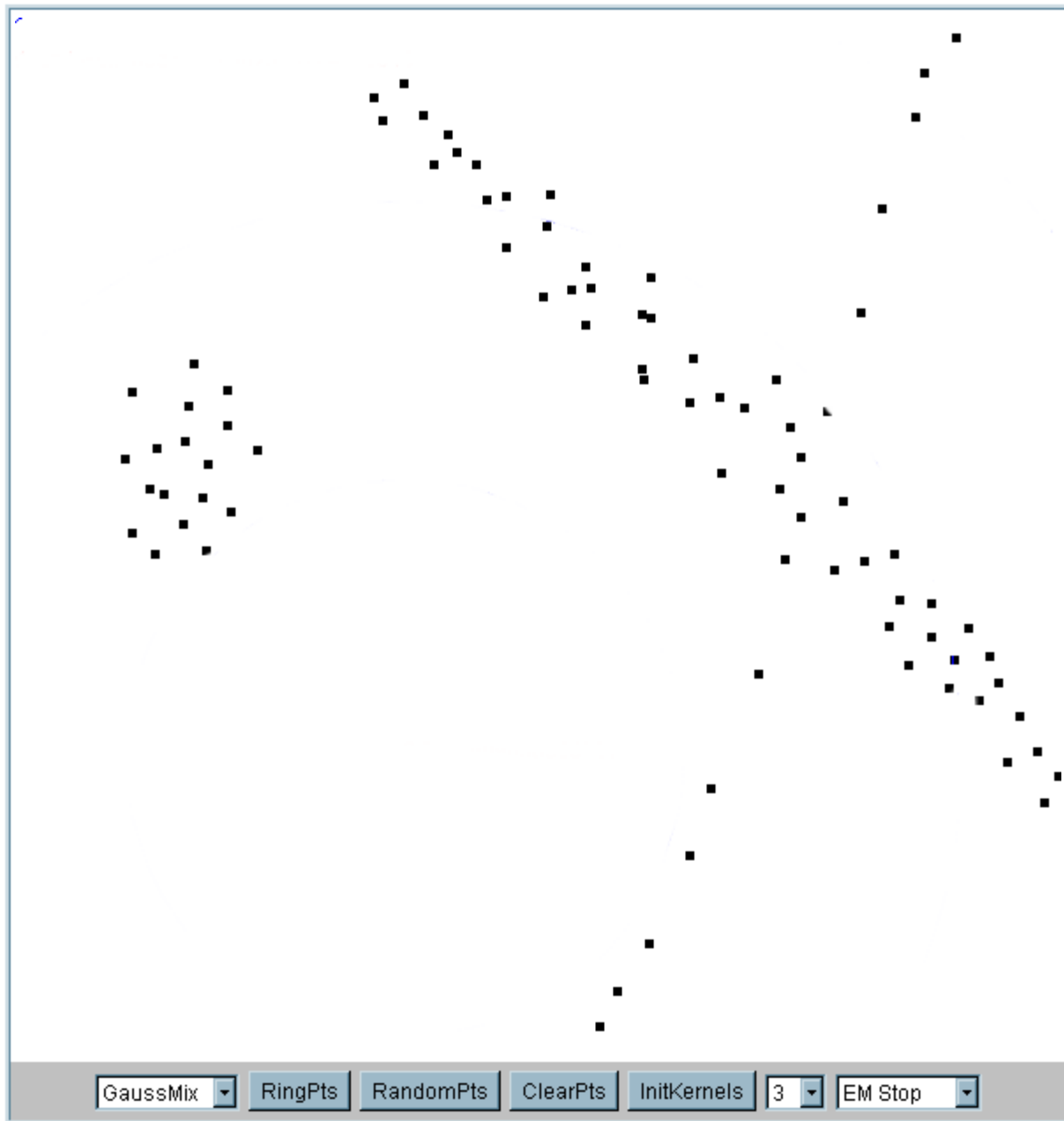
$$P(d_i \in c_k) = \frac{w_k \Pr(d_i | c_k)}{\sum_j w_j \Pr(d_i | c_j)}$$
$$w_k = \frac{\sum_i \Pr(d_i \in c_k)}{N}$$

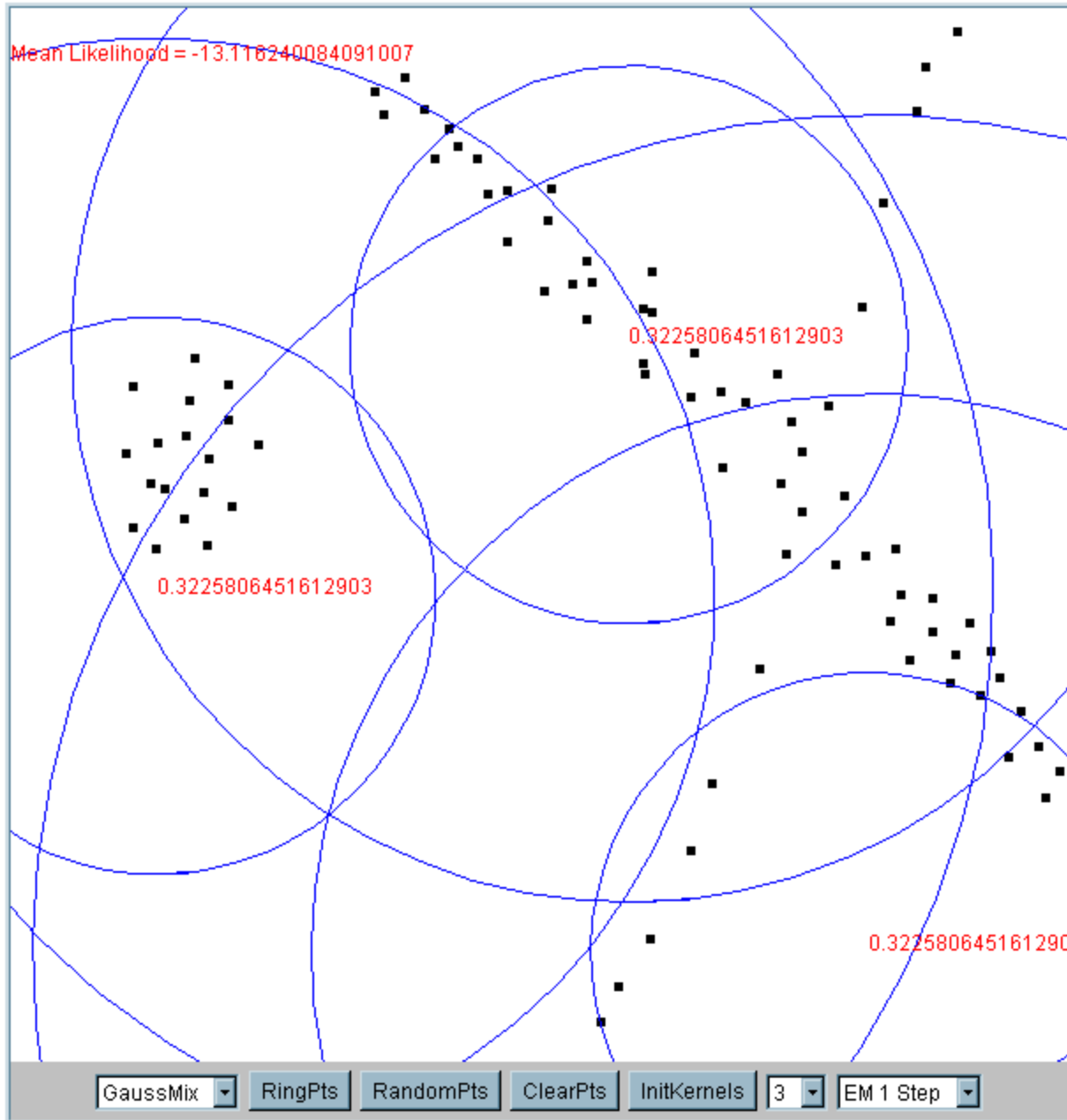
- **M**aximation step: estimate model parameters

$$\mu_k = \frac{1}{m} \sum_{i=1}^m \frac{d_i P(d_i \in c_k)}{\sum_k P(d_i \in c_k)}$$





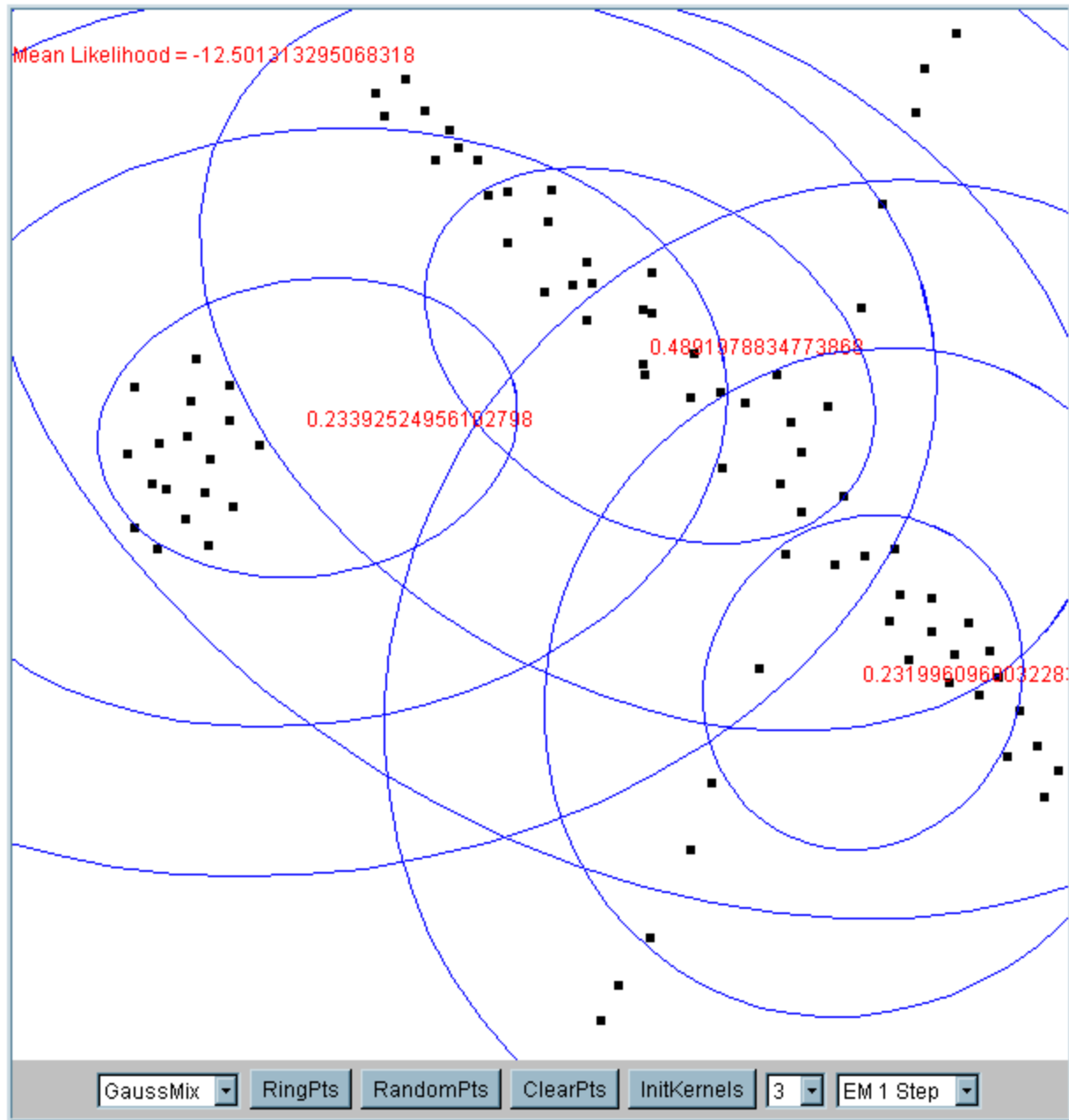




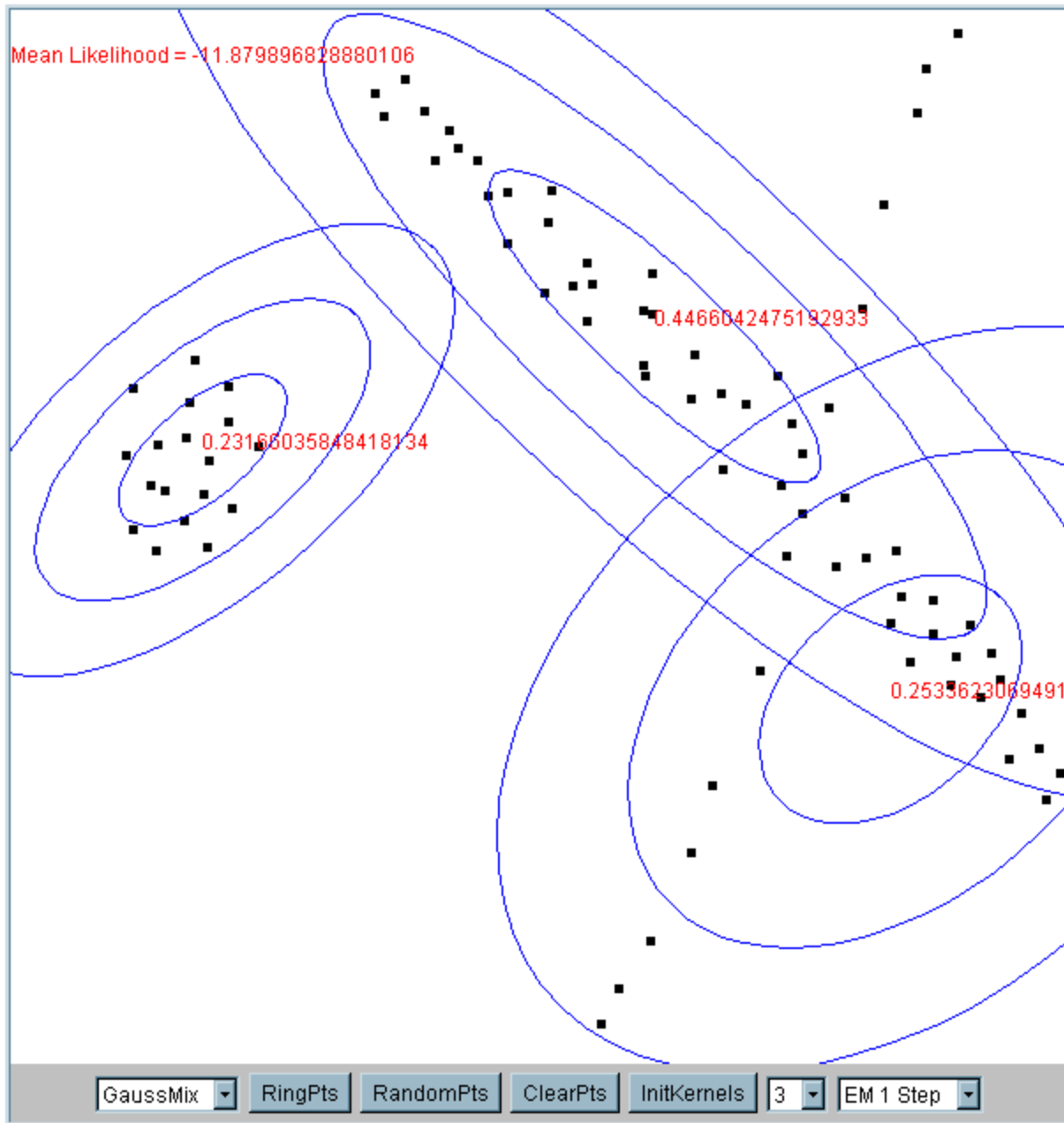
Iteration 1

The cluster means are randomly assigned

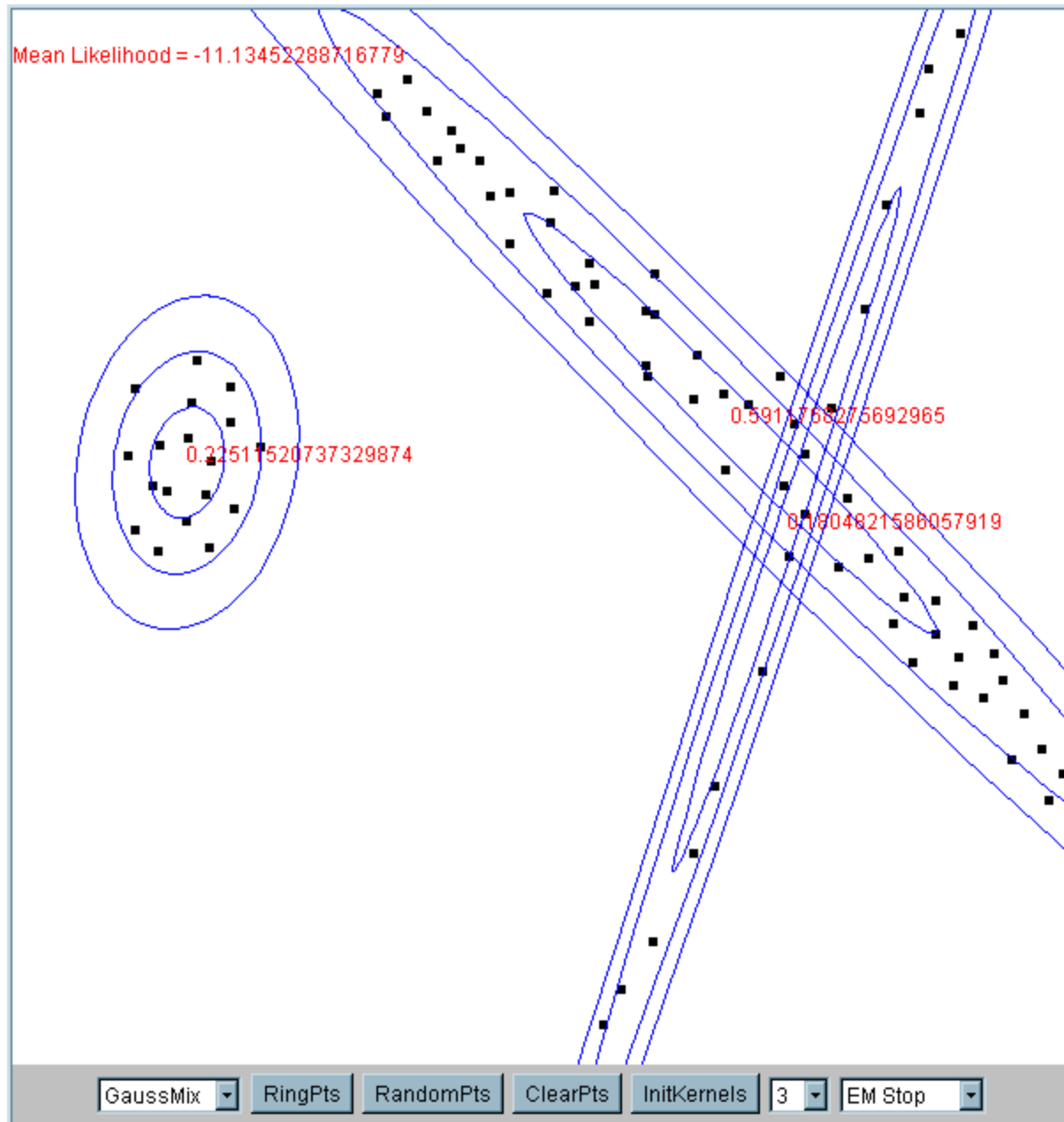
Iteration 2



Iteration 5



Iteration 25



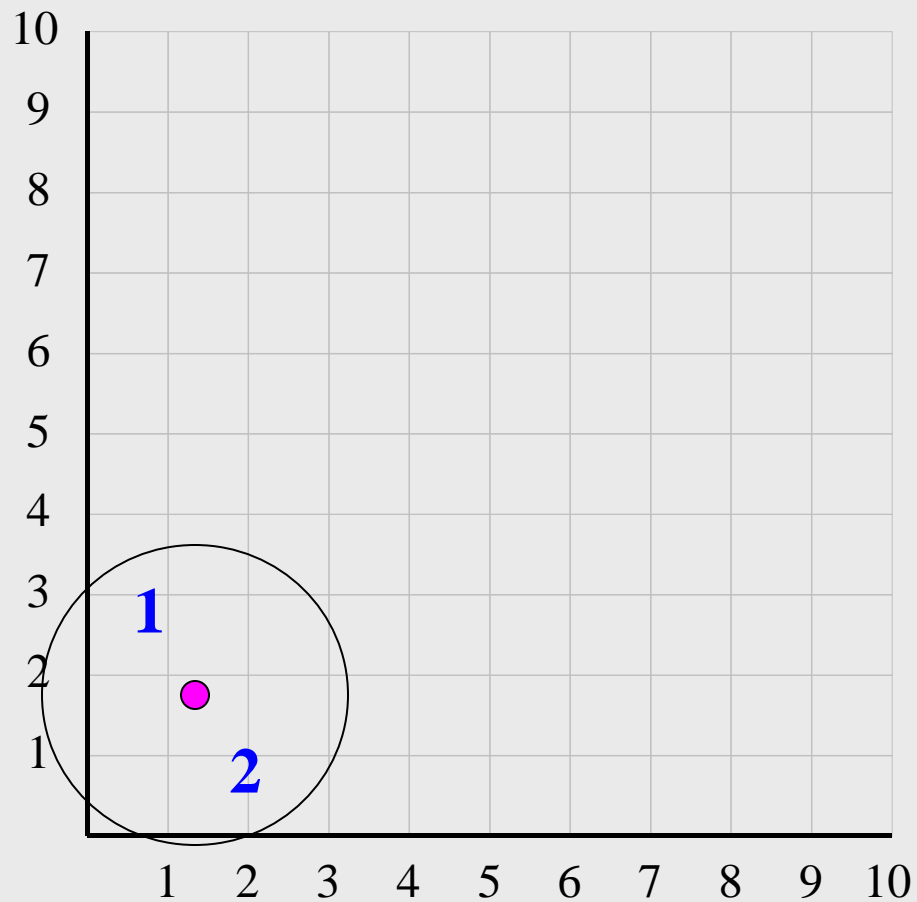
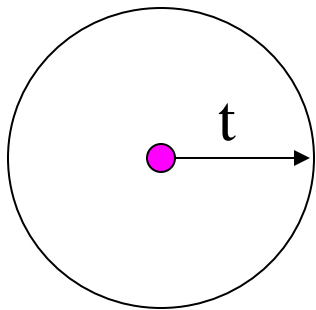
What happens if the data is streaming...

Nearest Neighbor Clustering

Not to be confused with Nearest Neighbor **Classification**

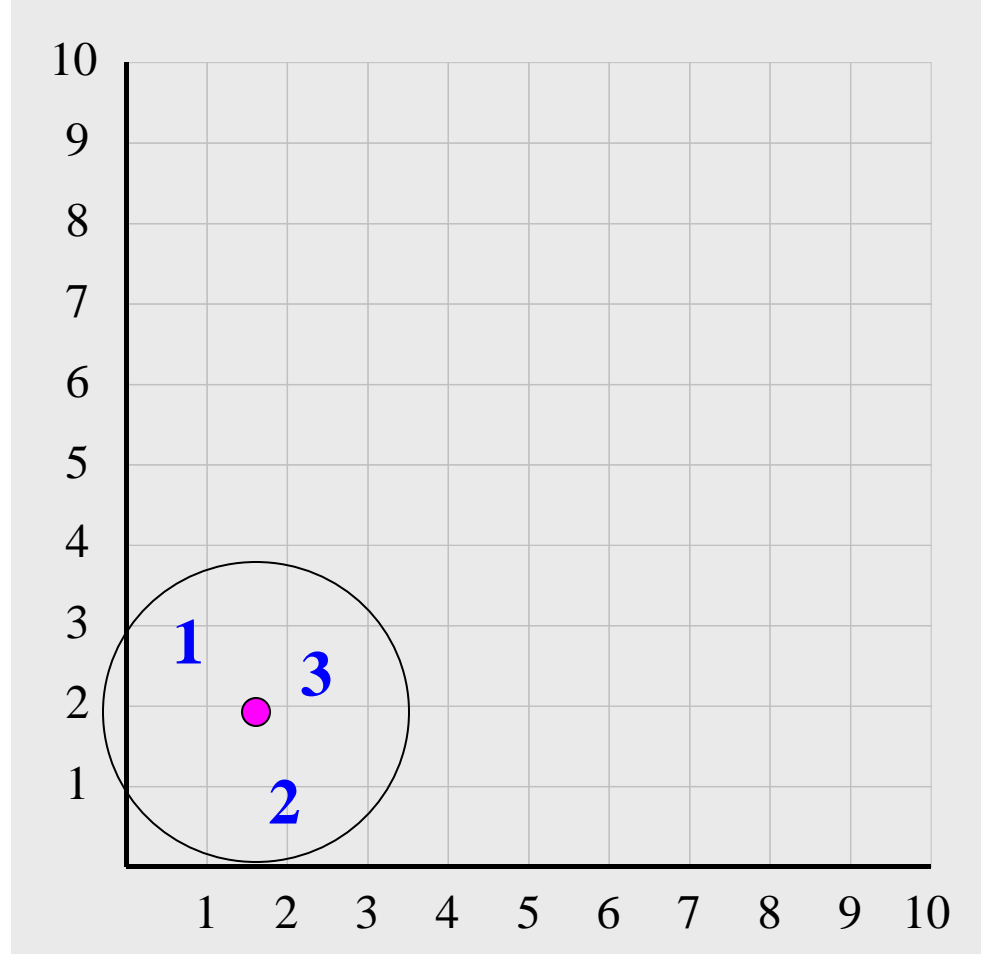
- Items are iteratively merged into the existing clusters that are closest.
- Incremental
- Threshold, t , used to determine if items are added to existing clusters or a new cluster is created.

Threshold t



New data point arrives...

It is within the threshold for cluster 1, so add it to the cluster, and update cluster center.

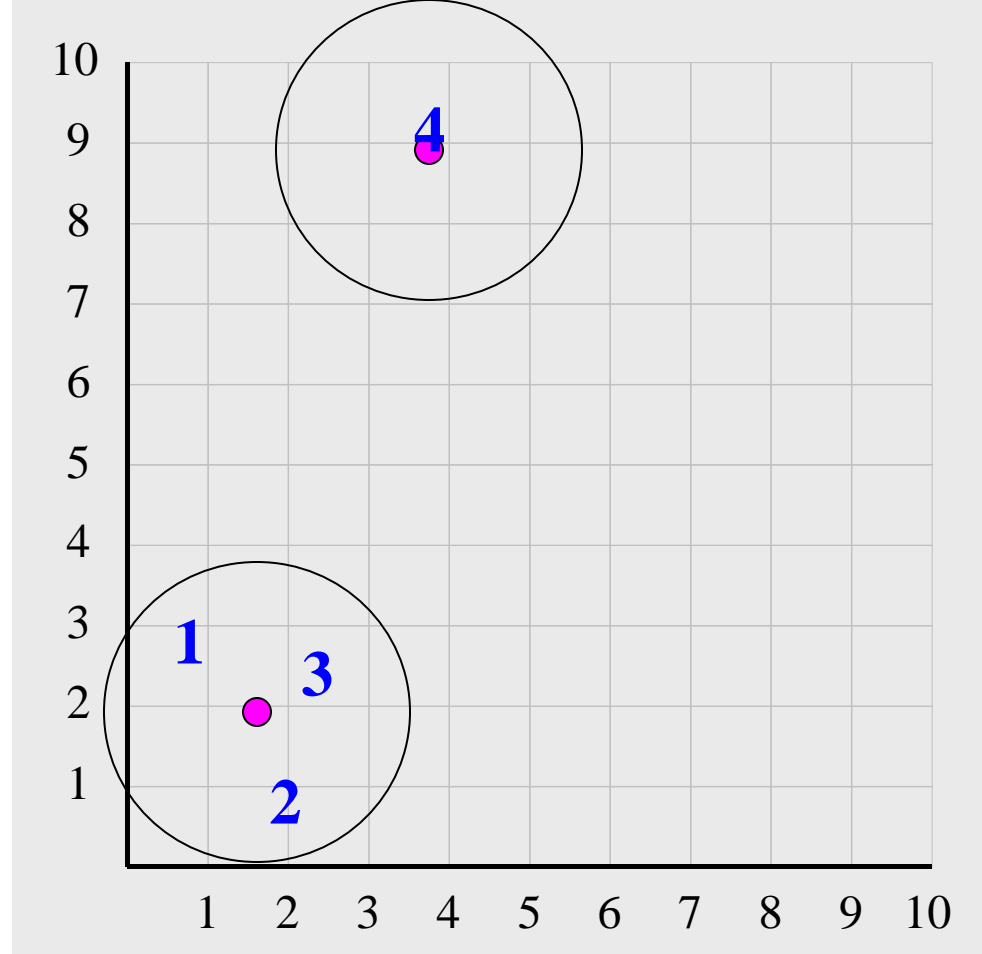


New data point arrives...

It is **not** within the threshold for cluster 1, so create a new cluster, and so on..

Algorithm is highly order dependent...

It is difficult to determine t in advance...



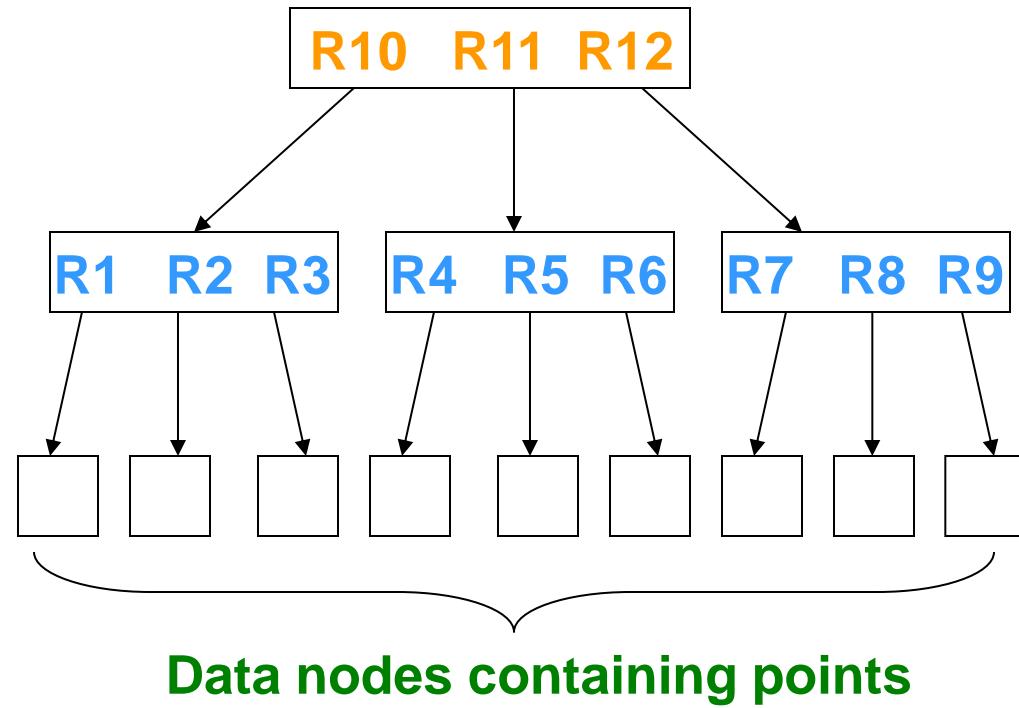
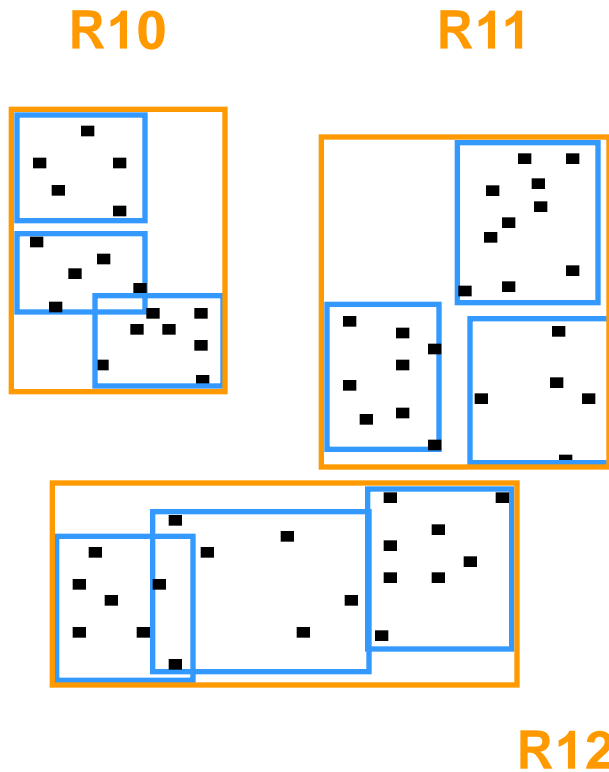
Partitional Clustering Algorithms

- Clustering algorithms have been designed to handle very large datasets
- E.g. the Birch algorithm
 - Main idea: use an in-memory R-tree to store points that are being clustered
 - Insert points one at a time into the R-tree, merging a new point with an existing cluster if it is less than some δ distance away
 - If there are more leaf nodes than fit in memory, merge existing clusters that are close to each other
 - At the end of first pass we get a large number of clusters at the leaves of the R-tree
 - Merge clusters to reduce the number of clusters

Partitional Clustering Algorithms

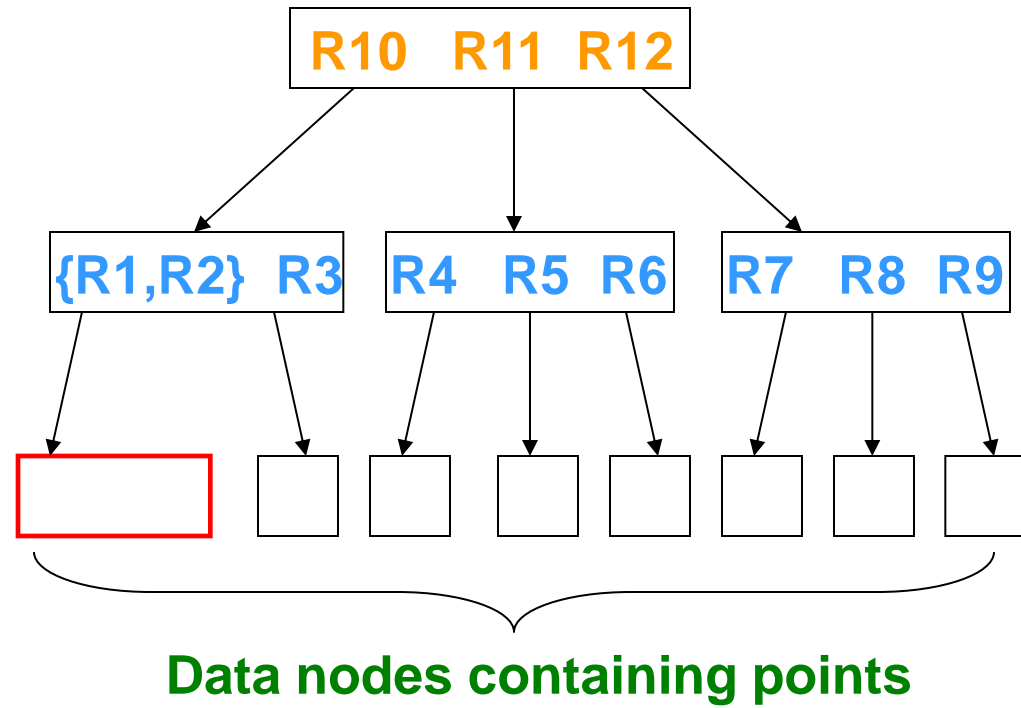
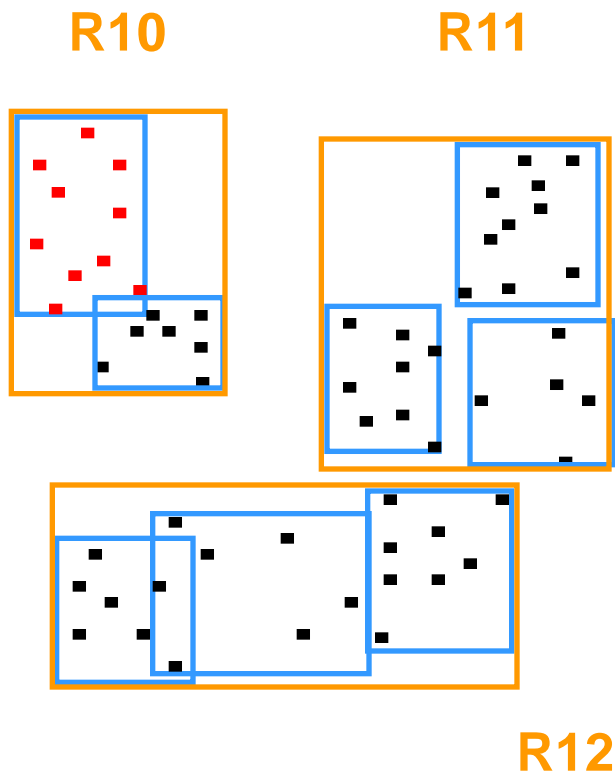
We need to specify the number of clusters in advance, I have chosen 2

- The Birch algorithm



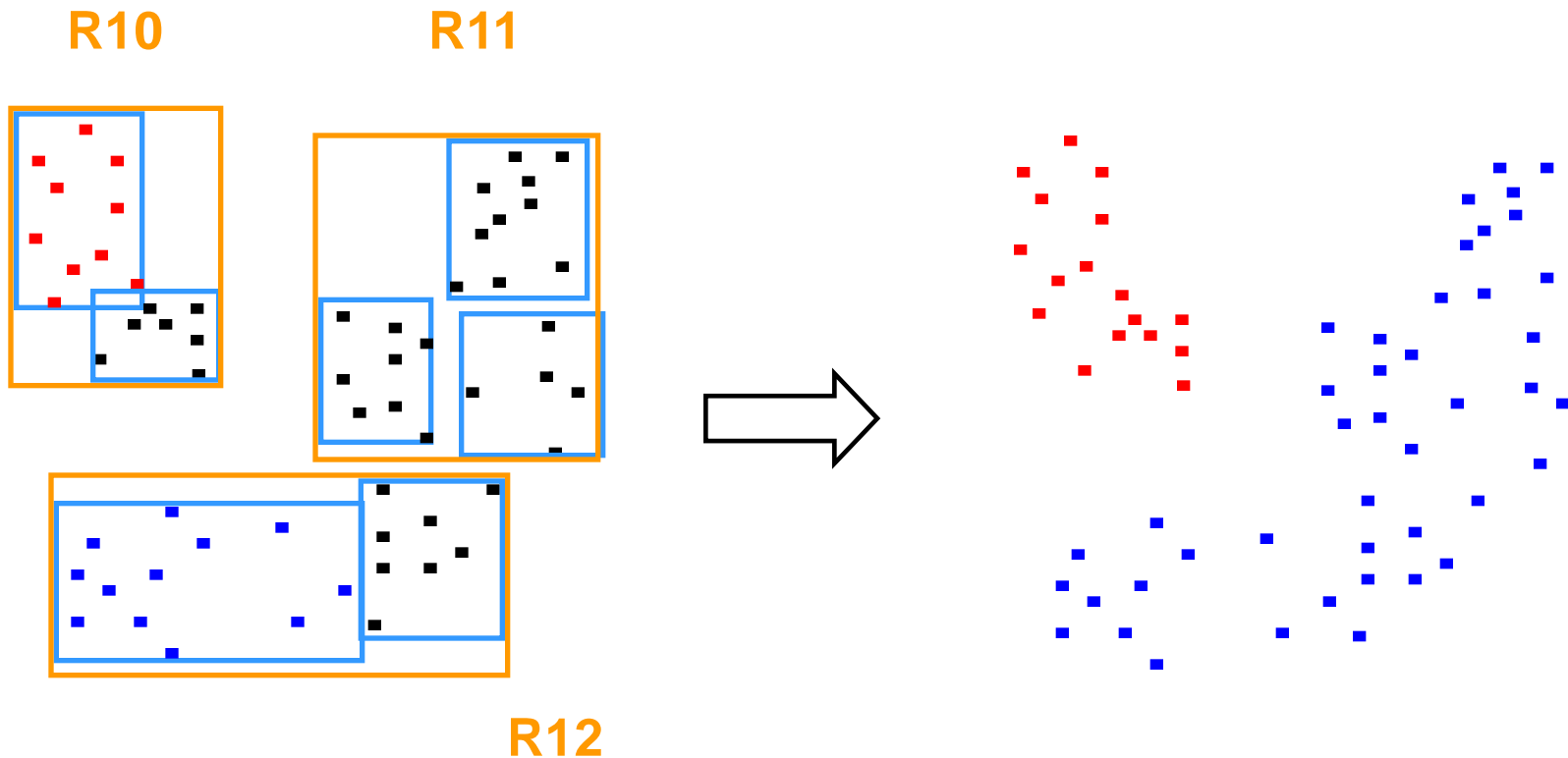
Partitional Clustering Algorithms

- The Birch algorithm



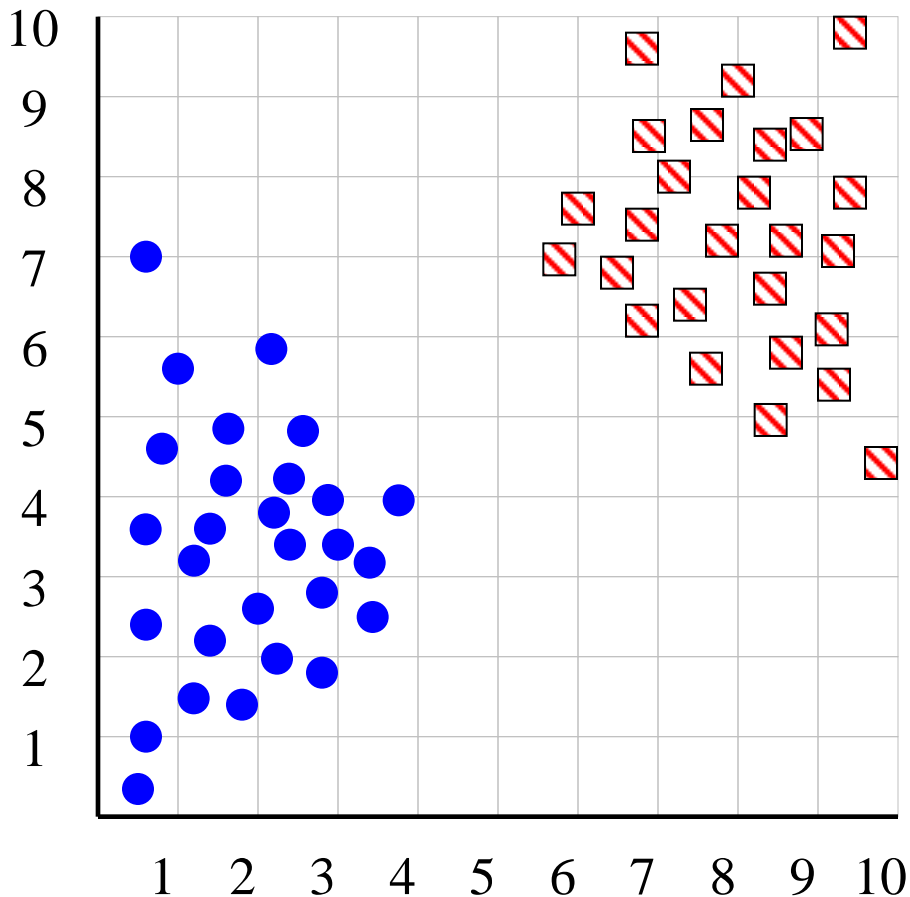
Partitional Clustering Algorithms

- The Birch algorithm



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

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



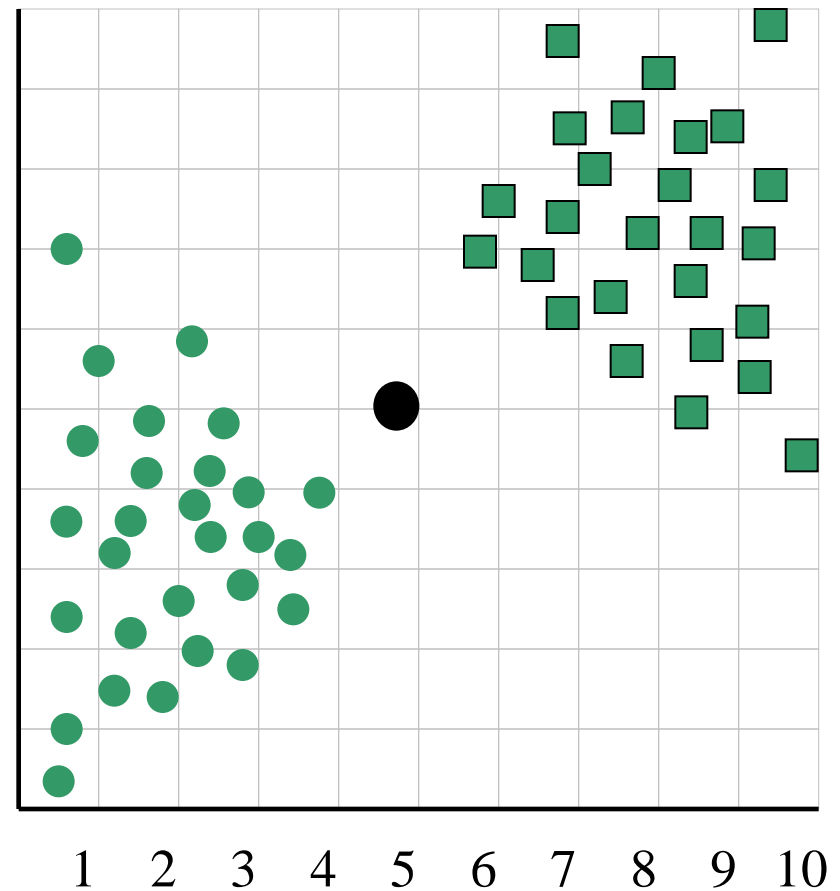
For our example, we will use the familiar **katydid**/**grasshopper** dataset.

However, in this case we are imagining that we do NOT know the class labels. We are only clustering on the X and Y axis values.

When $k = 1$, the objective function is 873.0

$$se_{K_i} = \sum_{j=1}^m \|t_{ij} - C_k\|^2$$

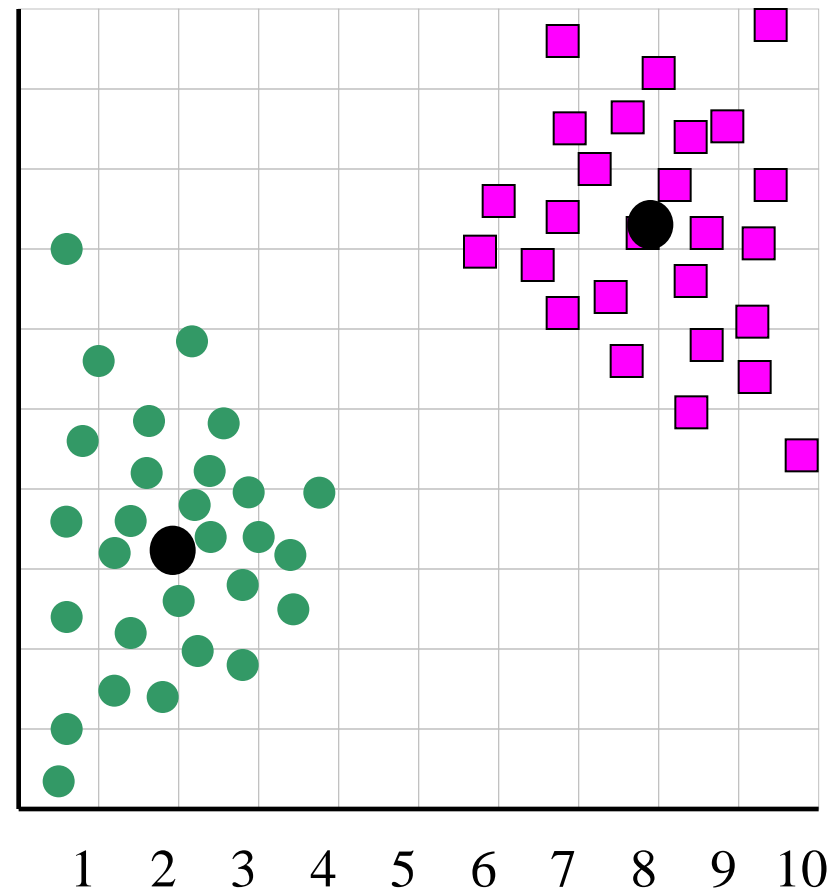
$$se_K = \sum_{j=1}^k se_{K_j}$$



When $k = 2$, the objective function is 173.1

$$se_{K_i} = \sum_{j=1}^m \|t_{ij} - C_k\|^2$$

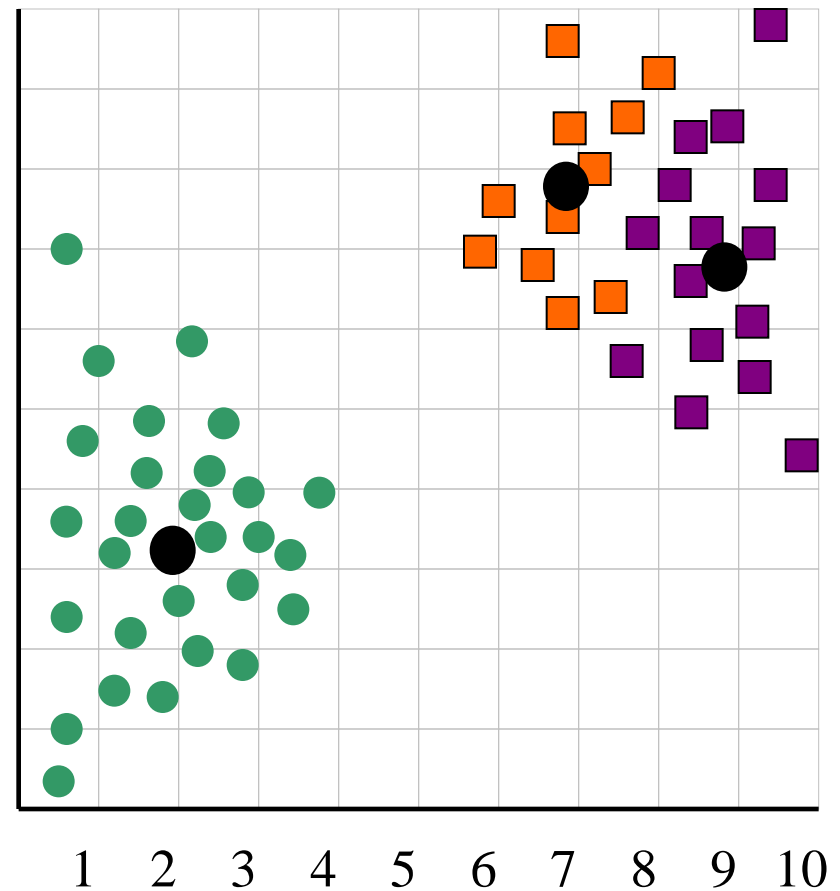
$$se_K = \sum_{j=1}^k se_{K_j}$$



When $k = 3$, the objective function is 133.6

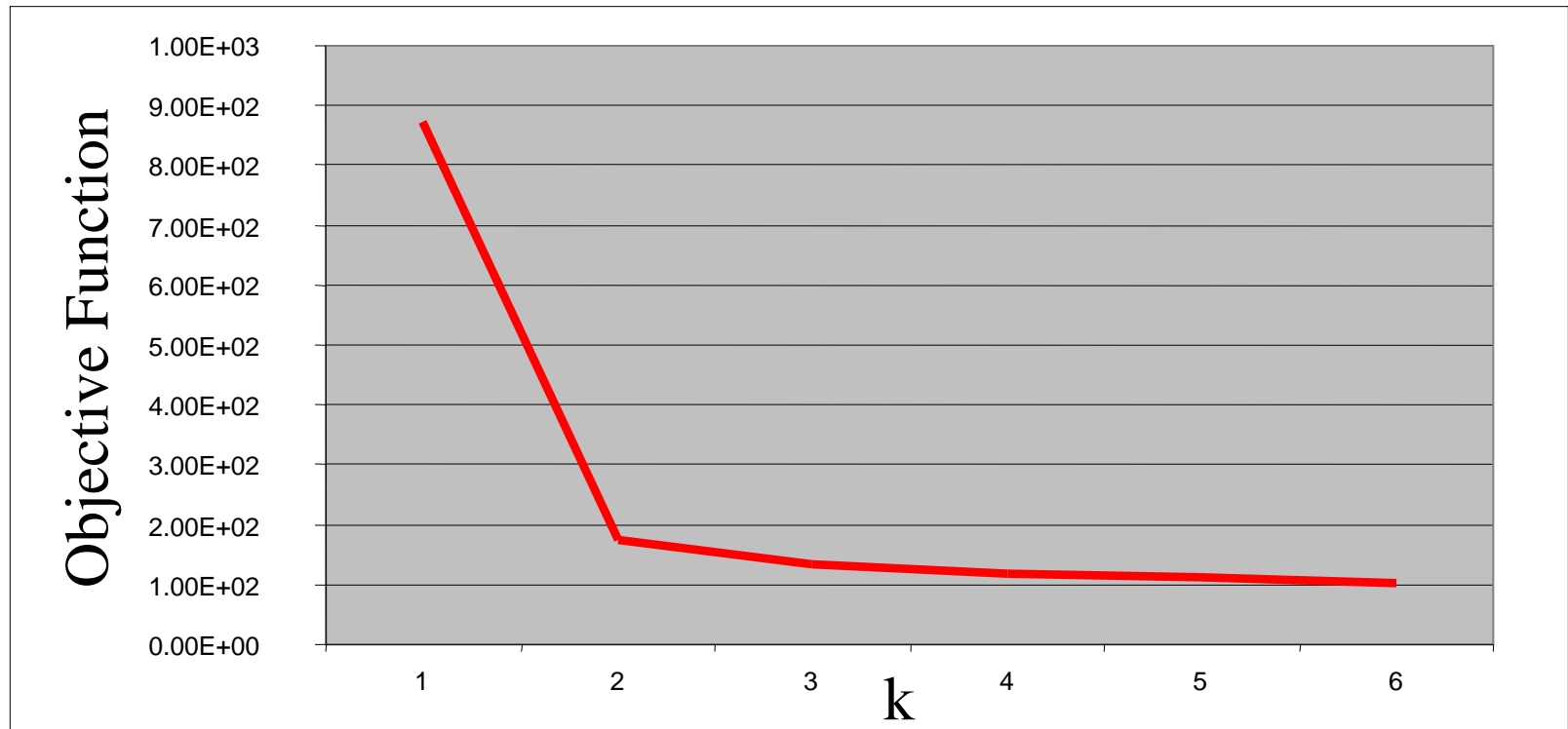
$$se_{K_i} = \sum_{j=1}^m \|t_{ij} - C_k\|^2$$

$$se_K = \sum_{j=1}^k se_{K_j}$$



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”.

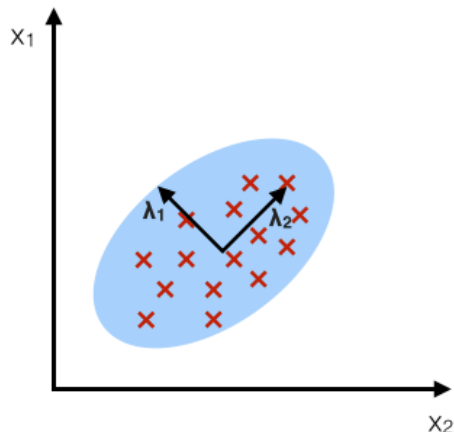


Note that the results are not always as clear cut as in this toy example

Linear Discriminant Analysis (LDA)

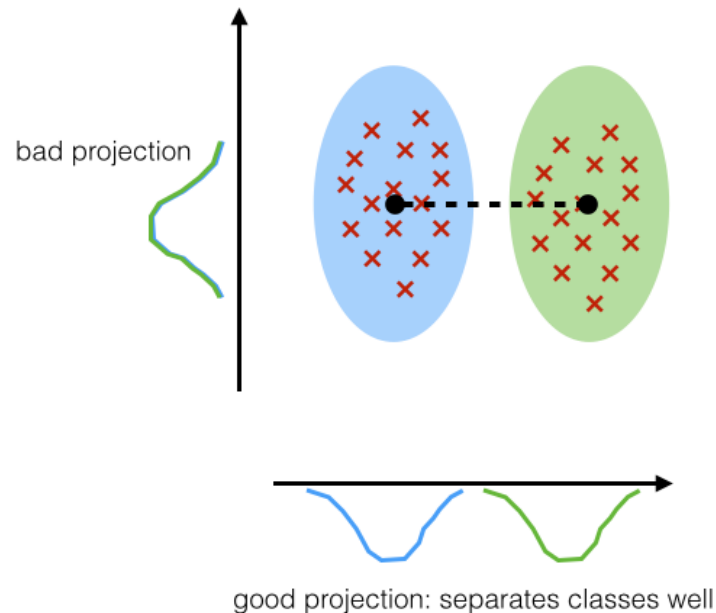
PCA:

component axes that maximize the variance



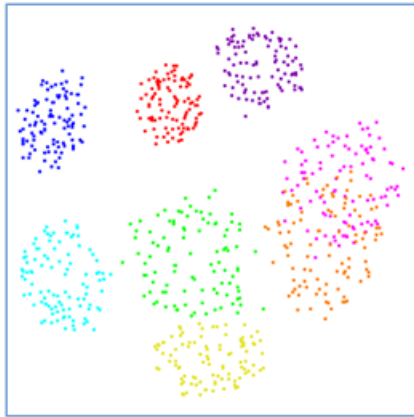
LDA:

maximizing the component axes for class-separation

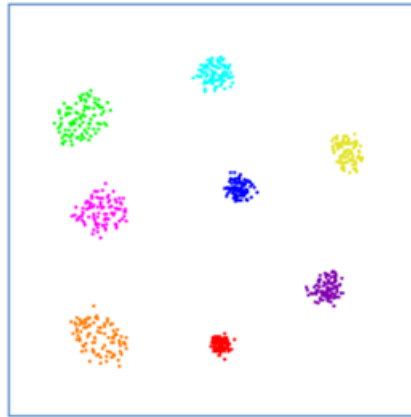


LDA

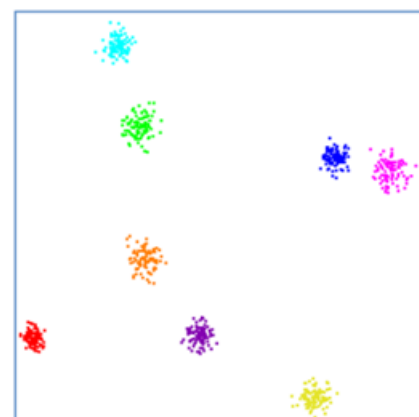
- Focuses on class separability and not on cluster shape
- If finding out about cluster shape is the goal then LDA is not desirable



Euclidian



SSIM



LDA

t-SNE

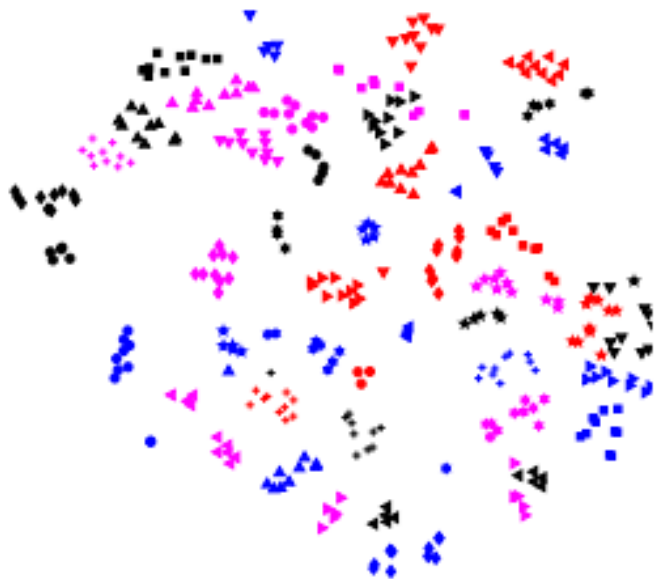
t-SNE = t-Distributed Stochastic Neighbor
Embedding

graph-based

locals statistics preserving

tends to isolate clusters well in the embedding

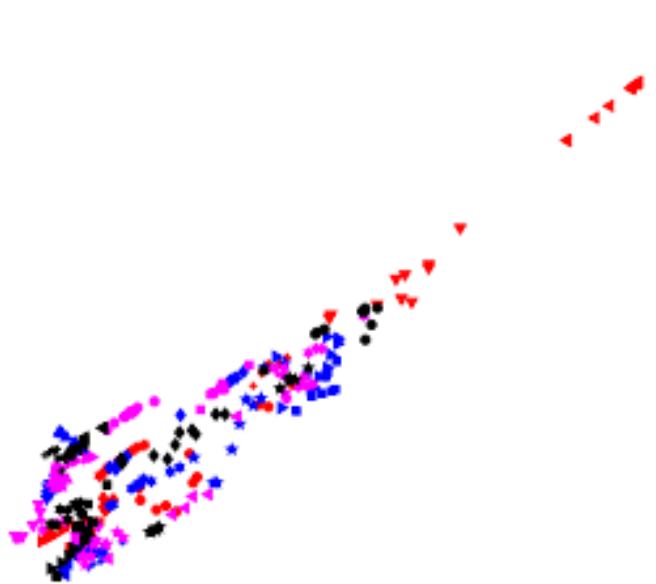
but compresses them



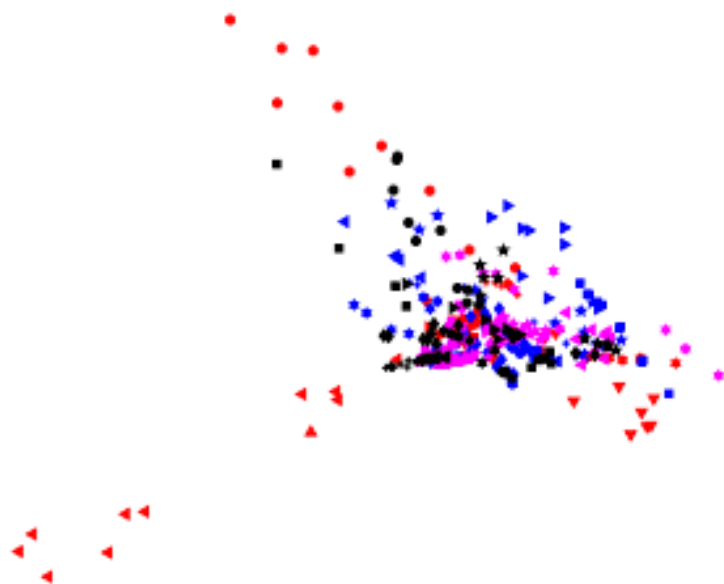
(a) Visualization by t-SNE.



(b) Visualization by Sammon mapping.



(c) Visualization by Isomap.



(d) Visualization by LLE.

Visualization

Kernel-Based and Spectral Clustering

Klaus Mueller

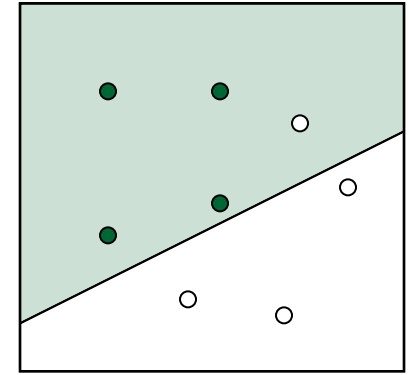
Computer Science Department

Stony Brook University

Definition: Support Vectors

- A regular linear discriminant function is:

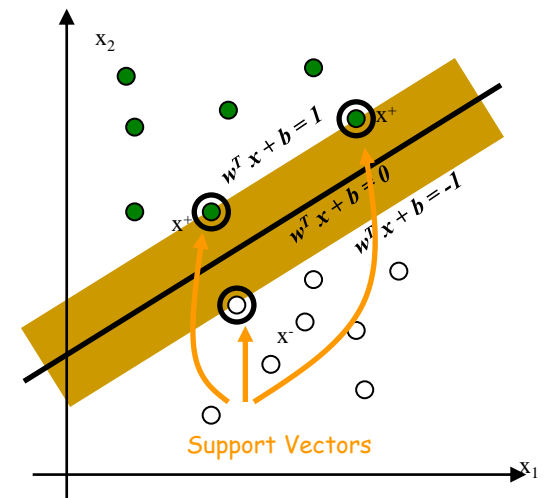
$$g(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$$



- A *support-vector* based linear discriminant function is:

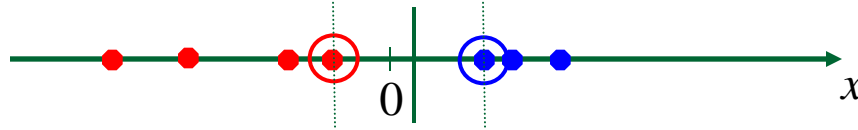
$$g(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b = \sum_{i \in \text{SV}} \alpha_i \mathbf{x}_i^T \mathbf{x} + b$$

- Here we represent \mathbf{w} by a set of support vectors \mathbf{x}_i



Non-Linear Cluster Separation

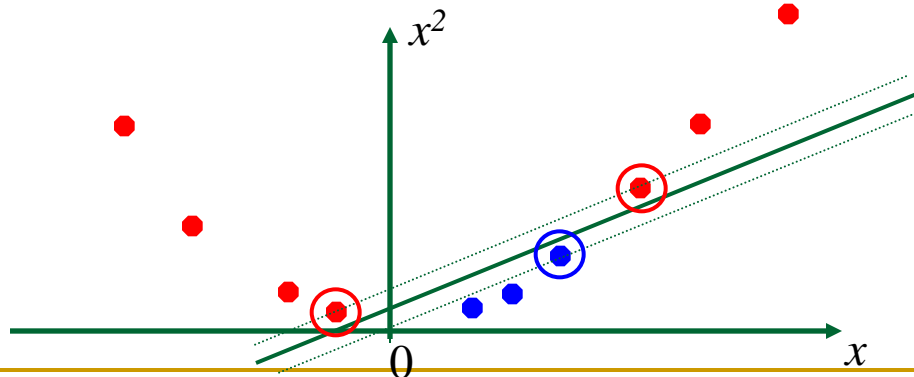
- Clusters that are linearly separable work out great:



- But what to do when the separation is non-linear?

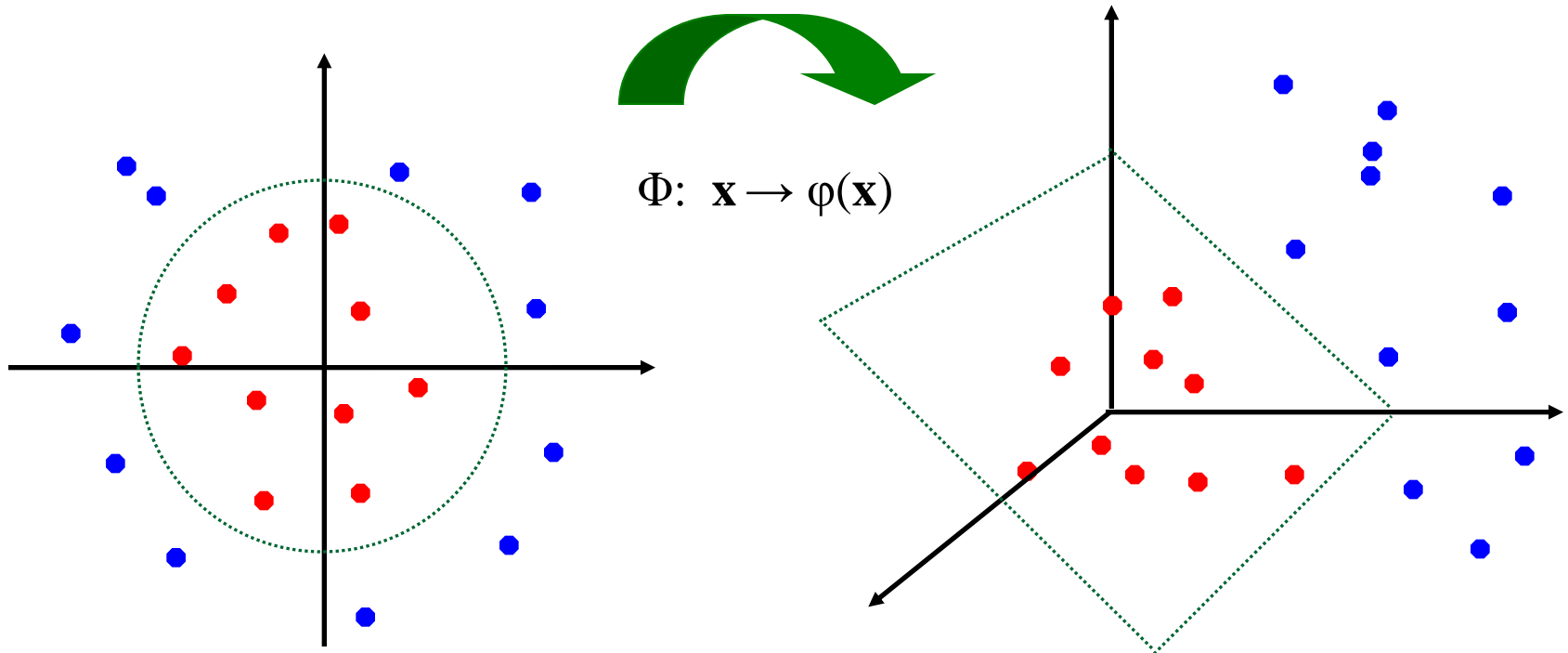


- How about... mapping data to a higher-dimensional space:



Non-linear Separation: Feature Space

- General idea: the original input space can be mapped to some higher-dimensional feature space where the training set is separable:



The Kernel Trick

- With this mapping, our discriminant function is now:

$$g(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b = \sum_{i \in SV} \alpha_i \phi(\mathbf{x}_i)^T \phi(\mathbf{x}) + b$$

- No need to know this mapping explicitly, because we only use the **dot product** of feature vectors.
- A **kernel function** is defined as a function that corresponds to a dot product of two feature vectors in some expanded feature space:

$$K(\mathbf{x}_i, \mathbf{x}_j) \equiv \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

The Kernel Trick

- An example:

2-dimensional vectors $\mathbf{x}=[x_1 \ x_2]$;

let $K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i^T \mathbf{x}_j)^2$,

Need to show that $K(\mathbf{x}_i, \mathbf{x}_j) = \boldsymbol{\varphi}(\mathbf{x}_i)^T \boldsymbol{\varphi}(\mathbf{x}_j)$:

$$\begin{aligned} K(\mathbf{x}_i, \mathbf{x}_j) &= (1 + \mathbf{x}_i^T \mathbf{x}_j)^2, \\ &= 1 + x_{i1}^2 x_{j1}^2 + 2 x_{i1} x_{j1} x_{i2} x_{j2} + x_{i2}^2 x_{j2}^2 + 2 x_{i1} x_{j1} + 2 x_{i2} x_{j2} \\ &= [1 \ x_{i1}^2 \ \sqrt{2} x_{i1} x_{i2} \ x_{i2}^2 \ \sqrt{2} x_{i1} \ \sqrt{2} x_{i2}]^T [1 \ x_{j1}^2 \ \sqrt{2} x_{j1} x_{j2} \ x_{j2}^2 \ \sqrt{2} x_{j1} \ \sqrt{2} x_{j2}] \\ &= \boldsymbol{\varphi}(\mathbf{x}_i)^T \boldsymbol{\varphi}(\mathbf{x}_j), \quad \text{where } \boldsymbol{\varphi}(\mathbf{x}) = [1 \ x_1^2 \ \sqrt{2} x_1 x_2 \ x_2^2 \ \sqrt{2} x_1 \ \sqrt{2} x_2] \end{aligned}$$

The Kernel Trick

- Examples of commonly-used kernel functions:

- Linear kernel: $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$

- Polynomial kernel: $K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i^T \mathbf{x}_j)^p$

- Gaussian (Radial-Basis Function (RBF)) kernel:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right)$$

- Sigmoid:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\beta_0 \mathbf{x}_i^T \mathbf{x}_j + \beta_1)$$

- In general, functions that satisfy *Mercer's condition* can be kernel functions.

Kernel k-Means

Standard k-means algorithm:

$$\mathcal{D}(\{\pi_c\}_{c=1}^k) = \sum_{c=1}^k \sum_{\mathbf{a}_i \in \pi_c} \|\mathbf{a}_i - \mathbf{m}_c\|^2, \text{ where } \mathbf{m}_c = \frac{\sum_{\mathbf{a}_i \in \pi_c} \mathbf{a}_i}{|\pi_c|}.$$

Kernel k-means:

$$\mathcal{D}(\{\pi_c\}_{c=1}^k) = \sum_{c=1}^k \sum_{\mathbf{a}_i \in \pi_c} \|\phi(\mathbf{a}_i) - \mathbf{m}_c\|^2, \text{ where } \mathbf{m}_c = \frac{\sum_{\mathbf{a}_i \in \pi_c} \phi(\mathbf{a}_i)}{|\pi_c|}.$$

Distance function (expanded):

$$\phi(\mathbf{a}_i) \cdot \phi(\mathbf{a}_i) - \frac{2 \sum_{\mathbf{a}_j \in \pi_c} \phi(\mathbf{a}_i) \cdot \phi(\mathbf{a}_j)}{|\pi_c|} + \frac{\sum_{\mathbf{a}_j, \mathbf{a}_l \in \pi_c} \phi(\mathbf{a}_j) \cdot \phi(\mathbf{a}_l)}{|\pi_c|^2}.$$

Note that these are all dot products, so we can use a kernel:

$$K_{ij} = \phi(\mathbf{a}_i) \cdot \phi(\mathbf{a}_j),$$

Computational Complexity

Computational complexity is higher than with standard k-means

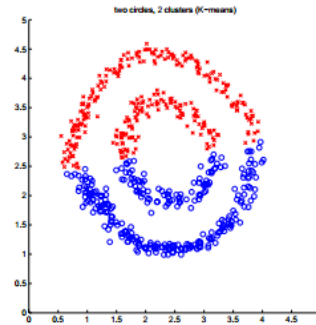
$$\phi(\mathbf{a}_i) \cdot \phi(\mathbf{a}_i) - \frac{2 \sum_{\mathbf{a}_j \in \pi_c} \phi(\mathbf{a}_i) \cdot \phi(\mathbf{a}_j)}{|\pi_c|} + \frac{\sum_{\mathbf{a}_j, \mathbf{a}_l \in \pi_c} \phi(\mathbf{a}_j) \cdot \phi(\mathbf{a}_l)}{|\pi_c|^2}.$$

- first term is constant for a given point \mathbf{a}_i
- second term is $O(n)$ for each data point, so we get $O(n^2)$ for all points
- third term is fixed per cluster
- if K is sparse then the cost can be lower
- also incur the cost for evaluation of the kernel
- so the total cost is $O(n^2(t+m))$

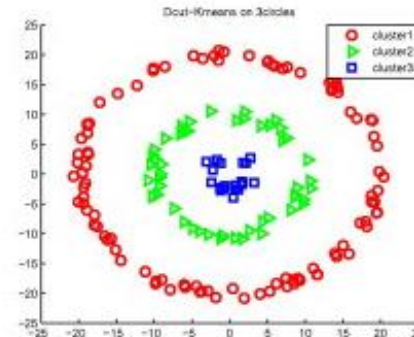
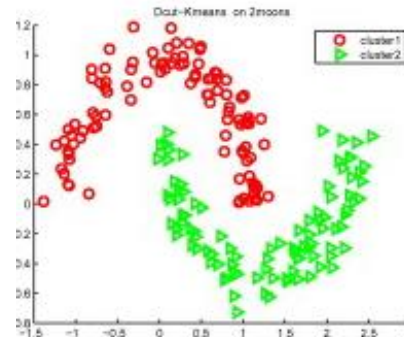
where t = number of iterations and m = number of dimensions

Results

Standard k-means



Kernel k-means



Note:

- the use of kernels is very general
- they can be used in many other clustering strategies

Kernel PCA

An extension of PCA to non-linear distributions

- instead of directly doing a PCA, the n data points x_i are mapped into a higher-dimensional feature space

$$x_i \rightarrow \varphi(x_i)$$

- then we solve the standard Eigenvalue problem:

$$\lambda w = Cw \quad , \quad (2.23)$$

with the covariance matrix $C = \frac{1}{n} \sum_{j=1}^n \varphi(x_j) \varphi(x_j)^T$. From the definition of C follows that Cw is a linear combination of the vectors $\varphi(x_i)$. Thus, w must lie in the span of $\varphi(x_1), \dots, \varphi(x_n)$. Hence, we can write

$$w = \sum_{i=1}^n \alpha_i \varphi(x_i) \quad . \quad (2.24)$$

Combining (2.23) and (2.24) gives

$$\lambda \sum_{i=1}^n \alpha_i \varphi(x_i) = \frac{1}{n} \sum_{i,j=1}^n \alpha_i \varphi(x_j) (\varphi(x_j)^T \varphi(x_i)) \quad , \quad (2.25)$$

Kernel PCA

This is equivalent to the set of n equations

$$\lambda \sum_{i=1}^n \alpha_i (\varphi(\mathbf{x}_i)^T \varphi(\mathbf{x}_l)) = \frac{1}{n} \sum_{i,j=1}^n \alpha_i (\varphi(\mathbf{x}_j)^T \varphi(\mathbf{x}_l)) (\varphi(\mathbf{x}_j)^T \varphi(\mathbf{x}_i)) \quad \forall l.$$

- it defines all n projections in the span

Again we can use the kernel trick

$$n\lambda \mathbf{K} \alpha = \mathbf{K}^2 \alpha$$

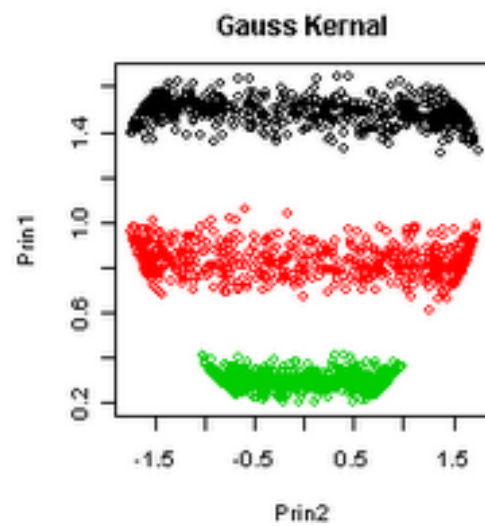
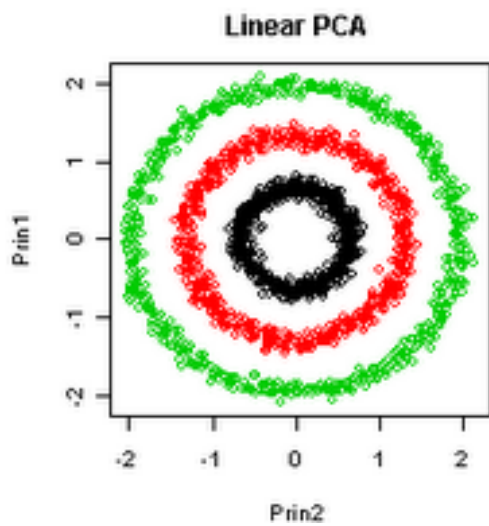
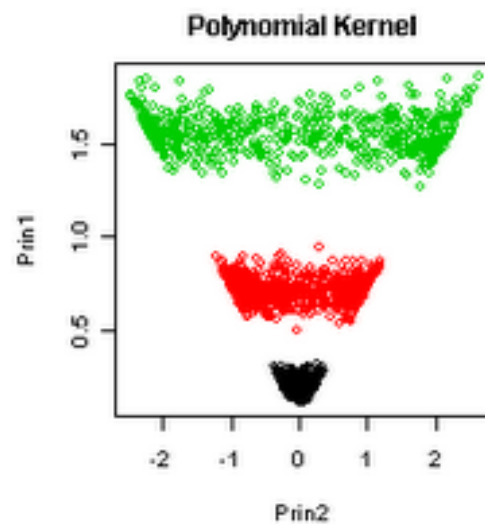
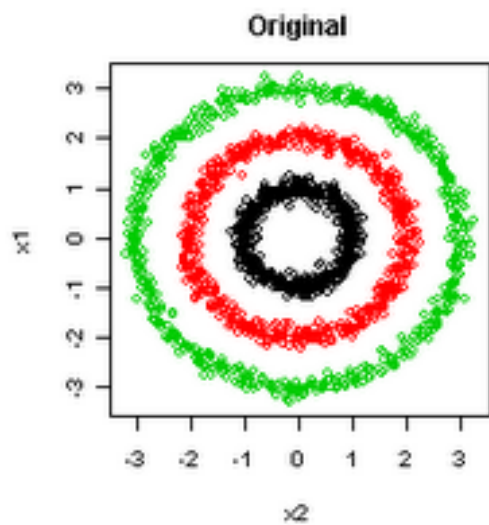
- which we can simplify to:

$$n\lambda \alpha = \mathbf{K} \alpha$$

The vector α for each principal component can be then obtained by extracting the eigenvectors of \mathbf{K}

- $K_{ij} = k(x_i, x_j)$

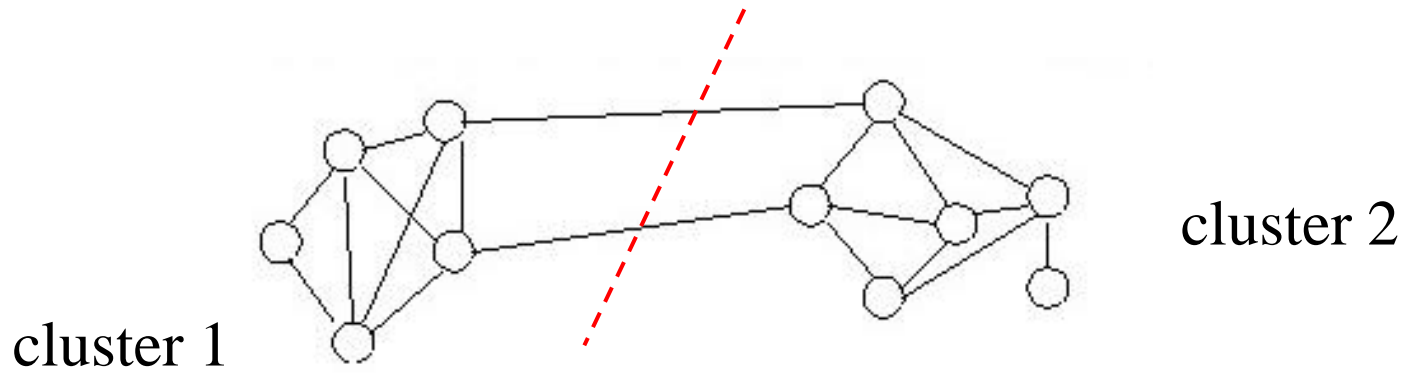
Results



Graph Representations

We may represent a dataset as a graph

Clusters then form sub-graphs that can be separated by *cuts*



A number of graph cutting algorithms have been devised for the segmentation of images

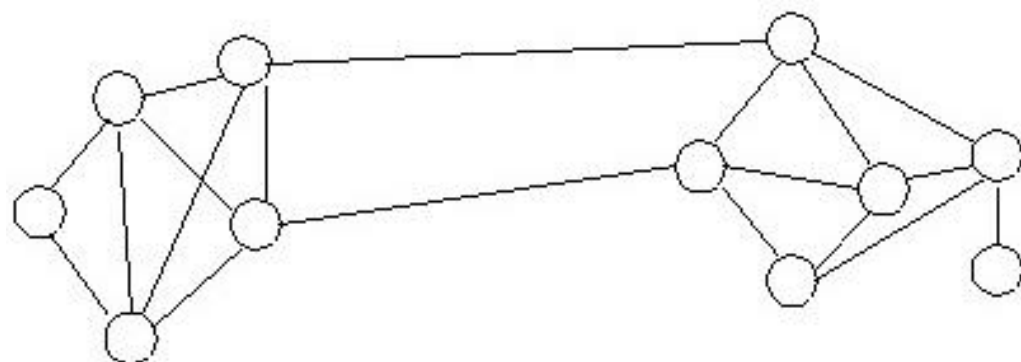


- Mincut, RatioCut, Ncut

We will be using a spectral method to reduce complexity

Some Graph Notations

- Given: data points X_1, \dots, X_n , pairwise similarities $w_{ij} = s(X_i, X_j)$
- $W = (w_{ij})$ adjacency matrix of the graph
- $d_i = \sum_j w_{ij}$ degree of a vertex
- $D = \text{diag}(d_1, \dots, d_n)$ degree matrix
- $|A| =$ number of vertices in A
- $\text{vol}(A) = \sum_{i \in A} d_i$



In the following: vector $f = (f_1, \dots, f_n)$ interpreted as function on the graph with $f(X_i) = f_i$.

Graph Laplacian

There are many options for the components of W

- use all neighbors \rightarrow fully connected graph \rightarrow dense W
- use k nearest neighbors only \rightarrow sparse W
- use a kernel to weight distances \rightarrow also sparse W
this is also called the *Affinity Matrix*

Build the Graph Laplacian

- many metrics have been defined

$$L = D - W$$

$$L = D^{-1/2} W D^{-1/2}$$

Find the Eigenvectors and values of L

- some also use the adjacency matrix directly

Eigenvectors of the Graph Laplacian

The first Eigenvalues are zero

- the number of zero-values indicates the number of clusters, k

The next k Eigenvalues are non-zero

- a small gap between these values corresponds to the graph having small cuts
- a large gap means that the graph has no cuts

Set a matrix E with the Eigenvectors forming the columns

- an eigenvector can be interpreted as associating values (the coordinate entries of the eigenvector) with the vertices of the graph
- these vertices correspond to rows in the matrix E
 - thus k-means clustering of these rows will cluster similar vertices

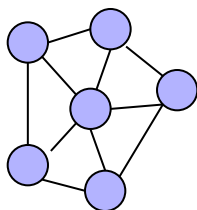
Eigenvectors of the Graph Laplacian

Cluster E into k clusters

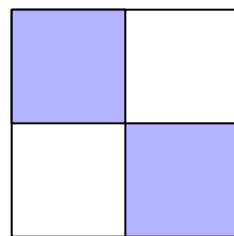
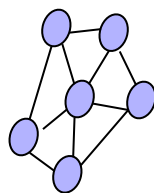
- assign a data point i to cluster j only if row i of E was assigned to cluster j
- this finalizes the spectral clustering
- in practice need to do some normalizations (omitted here)

Illustrative case:

- two isolated clusters



Kernel-linked graph of
2 isolated clusters



Its adjacency matrix
after clustering (stylized)

Toy Example

In the following slide:

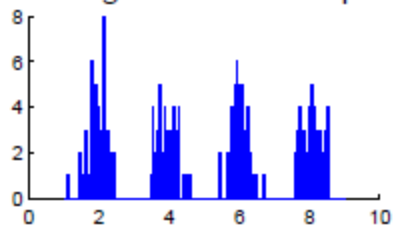
- 200 points sampled from 4 Gaussians
- Gaussian kernel with $\sigma=1$
- top two rows 10-connected graph, bottom two rows fully connected
- in each two rows see un-normalized and normalized Laplacian

Last row:

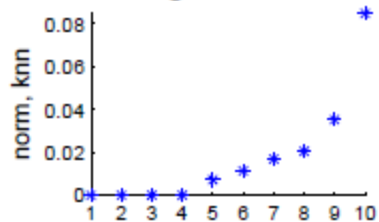
- threshold the second eigenvector at 0, then the part below 0 corresponds to clusters 1, 2, and the part above 0 to clusters 3, 4
- thresholding the third eigenvector separates clusters 1, 4 from clusters 2, 3
- thresholding the fourth eigenvector separates clusters 1, 3 from clusters 2, 4.
- the first 4 eigenvectors carry all the information about the 4 clusters.

Spectral clustering using k-means on the first 4 eigenvectors easily detects the correct 4 clusters

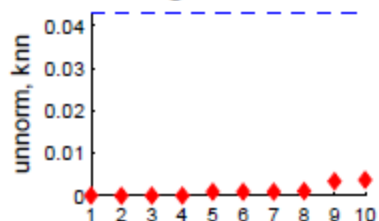
Histogram of the sample



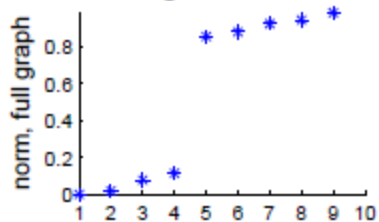
Eigenvalues



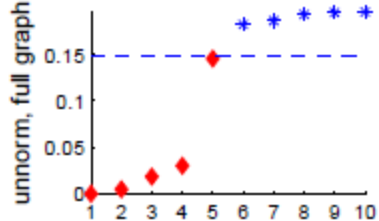
Eigenvalues



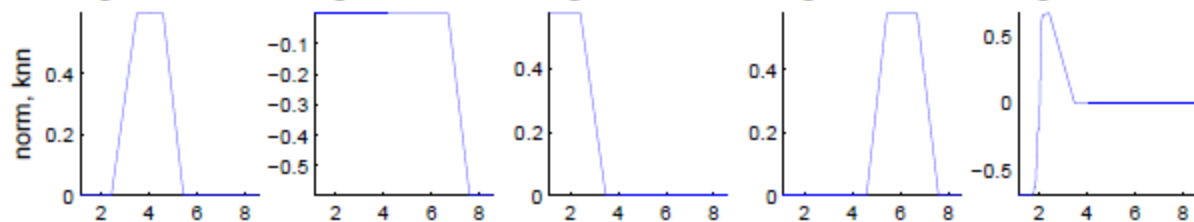
Eigenvalues



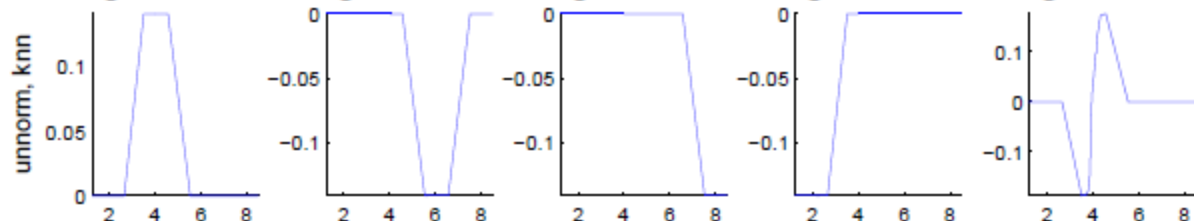
Eigenvalues



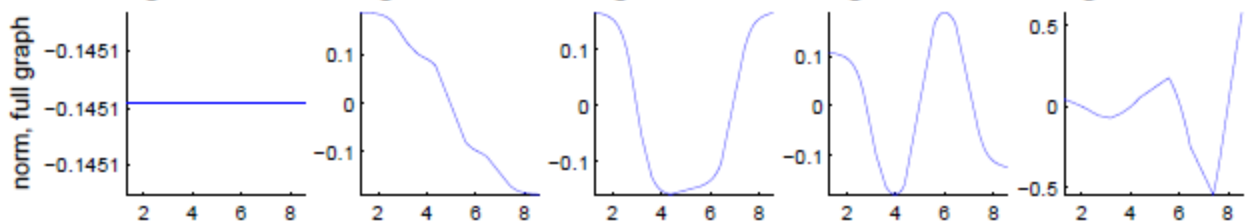
Eigenvector 1 Eigenvector 2 Eigenvector 3 Eigenvector 4 Eigenvector 5



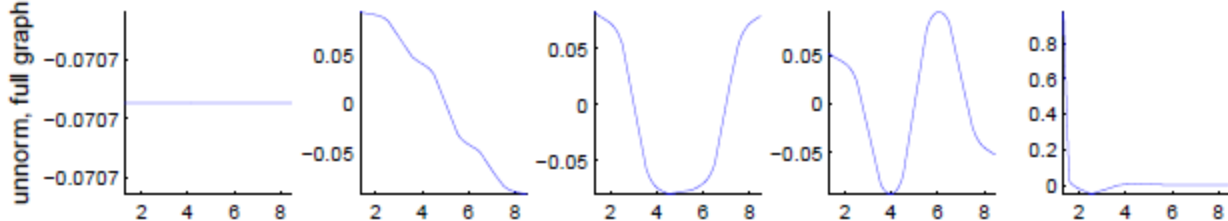
Eigenvector 1 Eigenvector 2 Eigenvector 3 Eigenvector 4 Eigenvector 5



Eigenvector 1 Eigenvector 2 Eigenvector 3 Eigenvector 4 Eigenvector 5

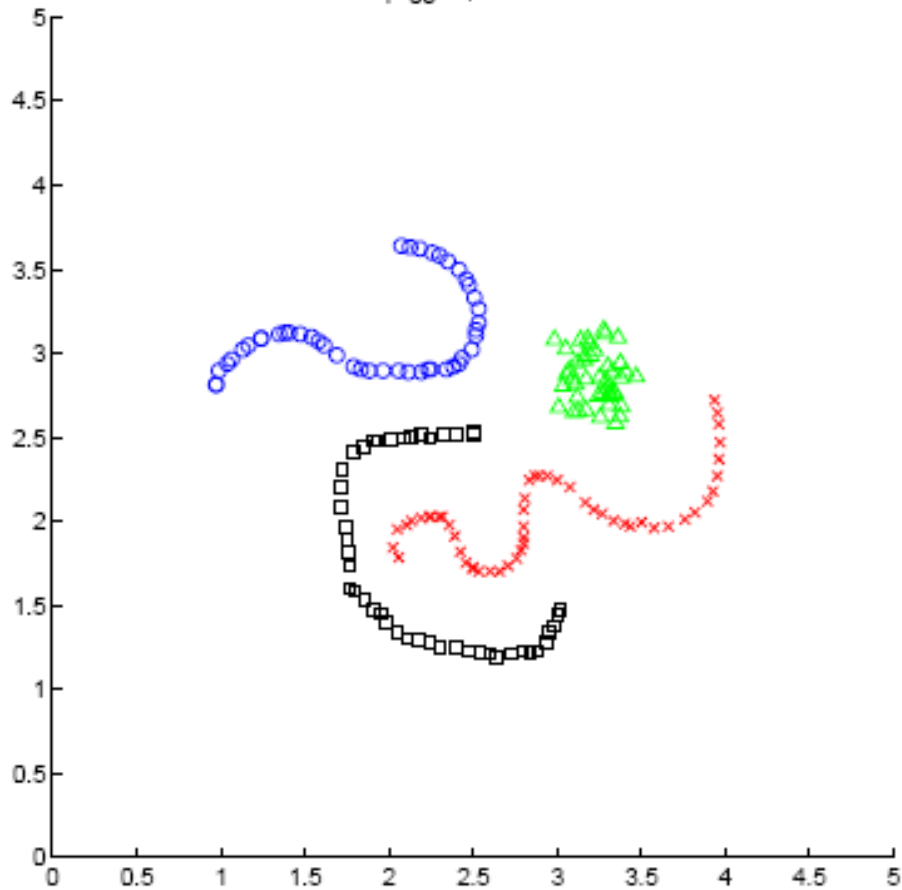


Eigenvector 1 Eigenvector 2 Eigenvector 3 Eigenvector 4 Eigenvector 5

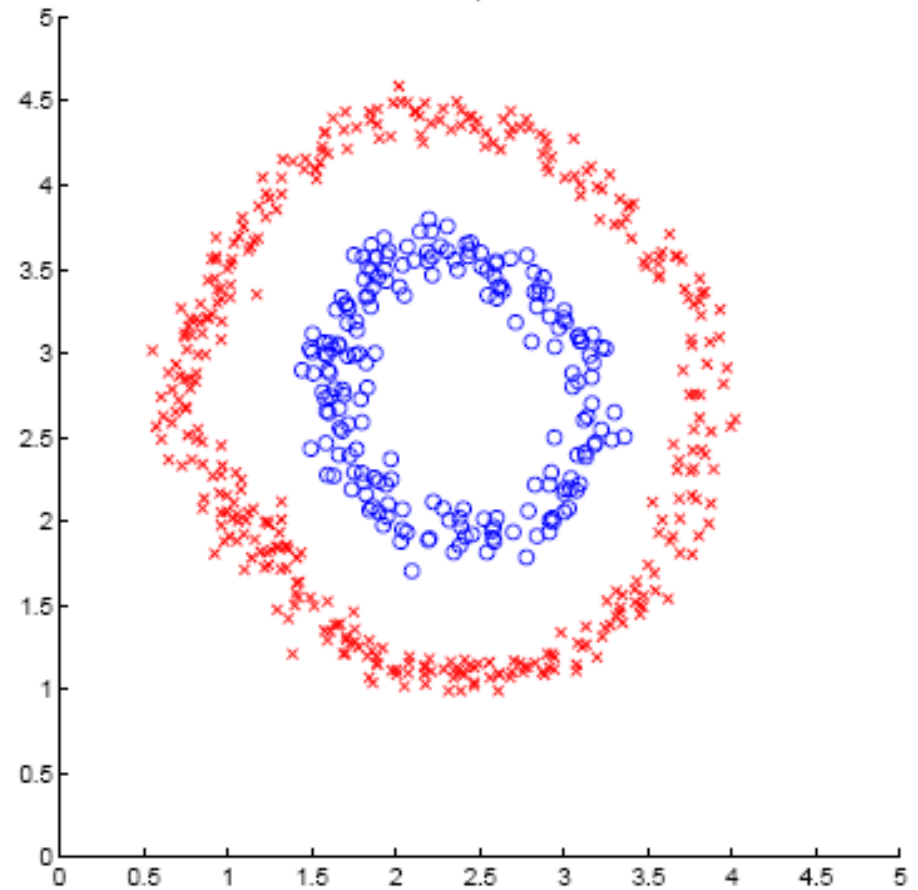


Examples

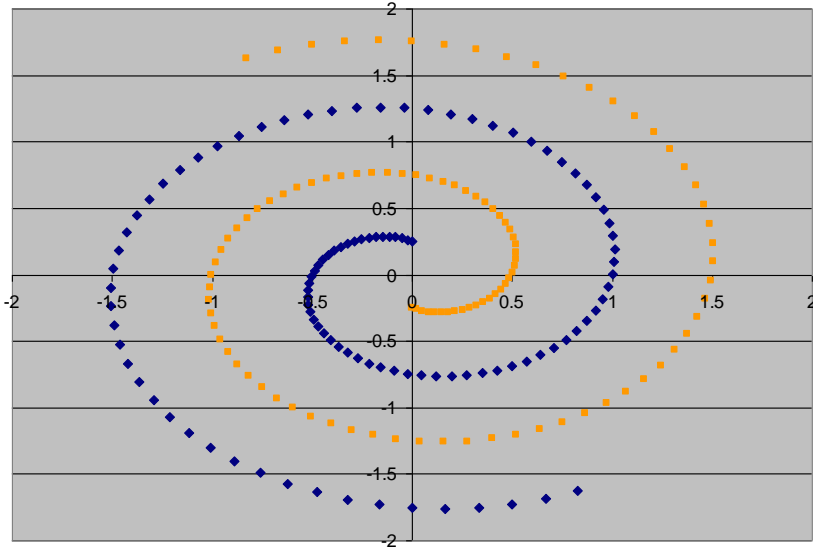
squiggles, 4 clusters



twocircles, 2 clusters

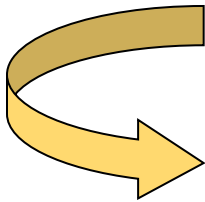


Examples



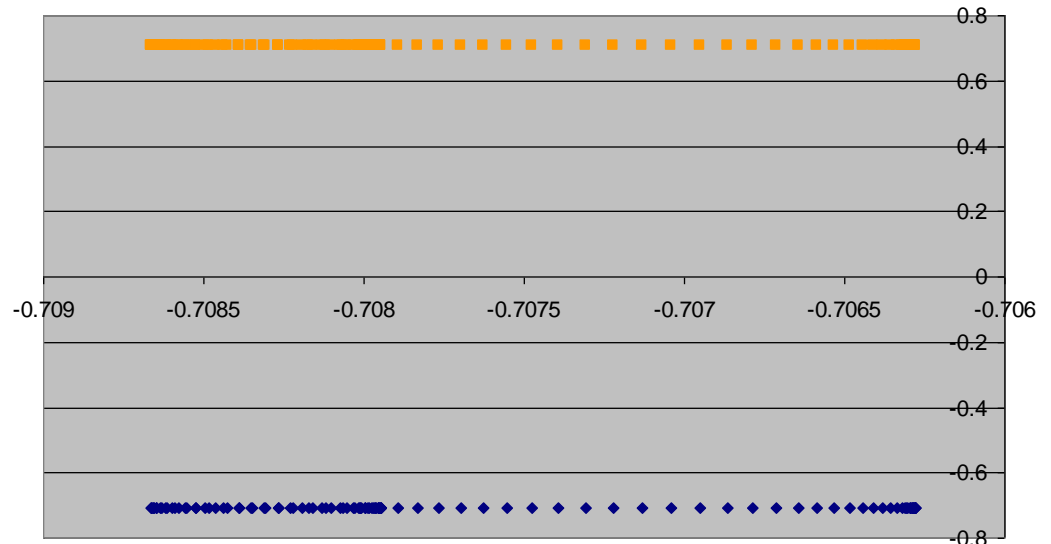
Dataset exhibits complex cluster shapes

⇒ K-means performs very poorly in this space due bias toward dense spherical clusters.



In the embedded space given by two leading eigenvectors, clusters are trivial to separate.

Spectral Clustering - *Derek Greene*



How to Select k?

Eigengap: the difference between two consecutive eigenvalues.

Most stable clustering is generally given by the value k that maximizes the expression

$$\Delta_k = |\lambda_k - \lambda_{k-1}|$$

Largest eigenvalues
of Cisi/Medline data

$$\max \Delta_k = |\lambda_2 - \lambda_1|$$

⇒ Choose $k=2$

