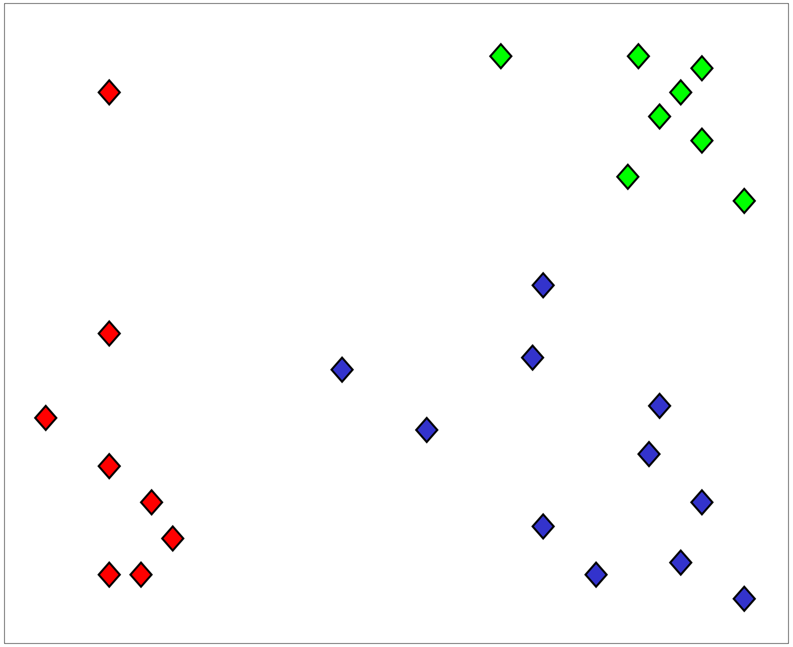
What is Clustering?

- Organizing data into *clusters* such that there is
  - high intra-cluster similarity
  - low inter-cluster similarity

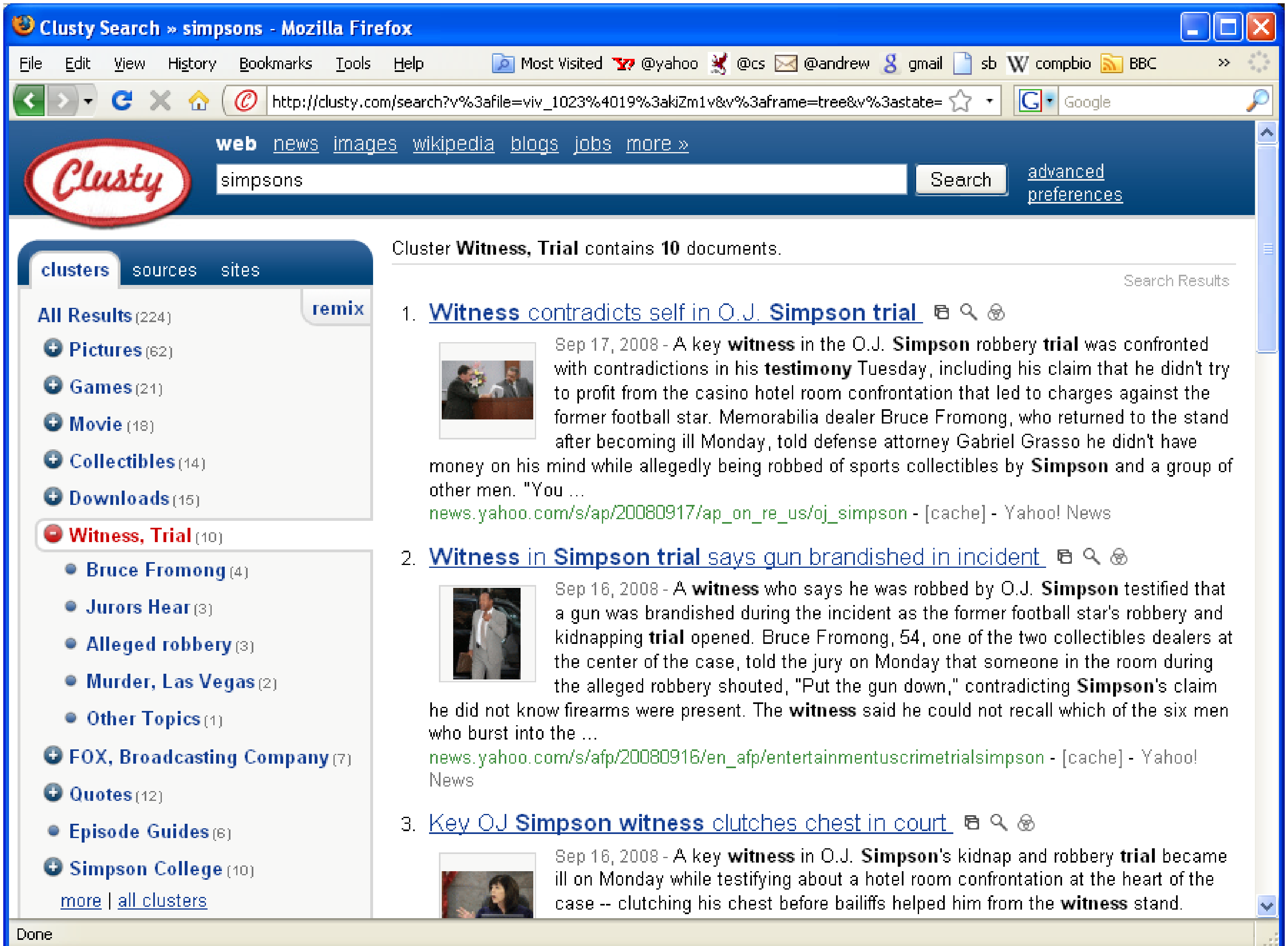
•Informally, finding natural groupings among objects.



•Why do we want to do that?

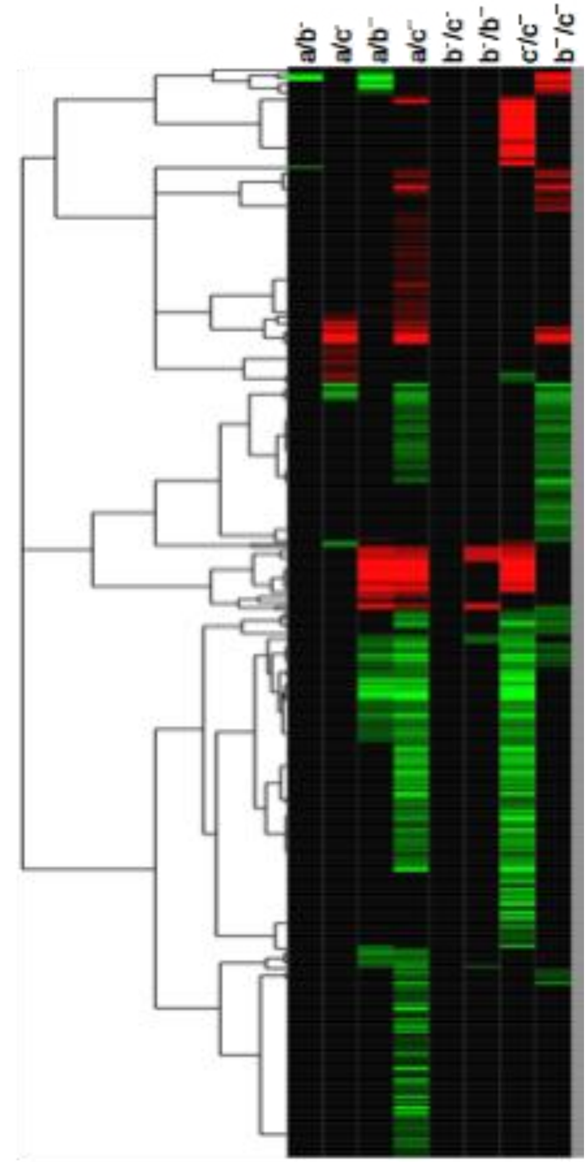
•Any REAL application?

Example: clusty



Example: clustering genes

- Microarrays measures the activities of all genes in different conditions
- Clustering genes can help determine new functions for unknown genes
- An early "killer application" in this area
  - The most cited (11,591) paper in PNAS!



Why clustering?

- Organizing data into clusters provides information about the internal structure of the data
  - Ex. Clusty and clustering genes above
- Sometimes the partitioning is the goal
  - Ex. Image segmentation
- Knowledge discovery in data
  - Ex. Underlying rules, reoccurring patterns, topics, etc.

Unsupervised learning

• Clustering methods are unsupervised learning techniques

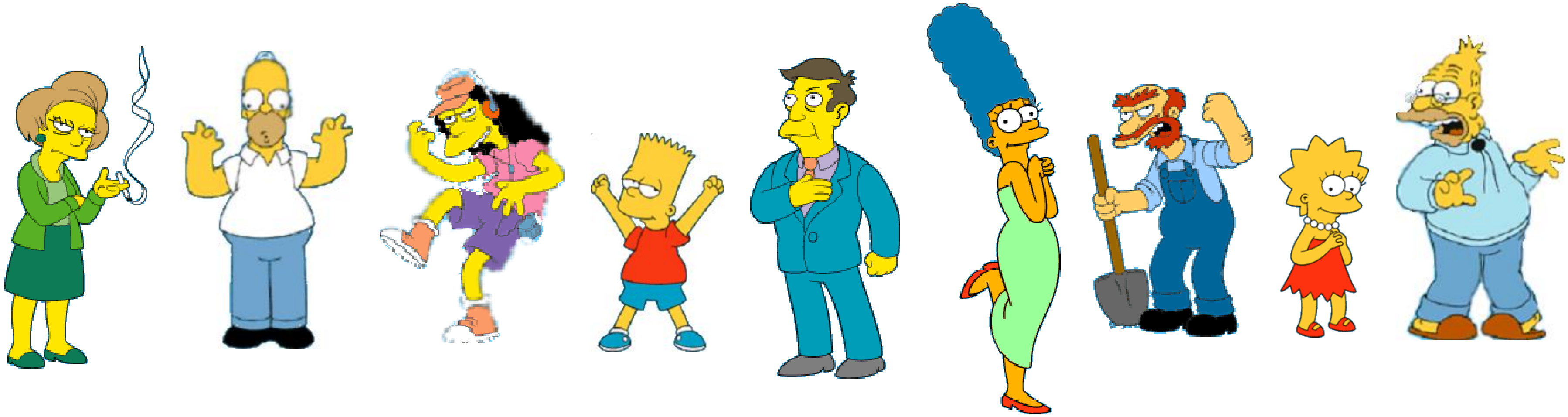
- We do not have a teacher that provides examples with their labels

• We will also discuss dimensionality reduction, another unsupervised learning method later in the course

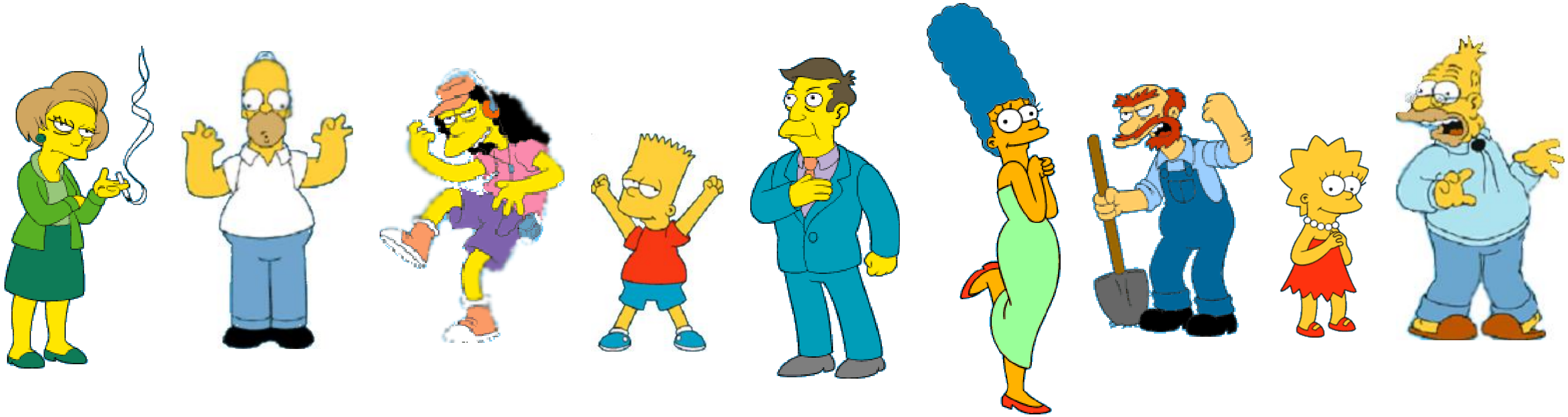
Outline

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

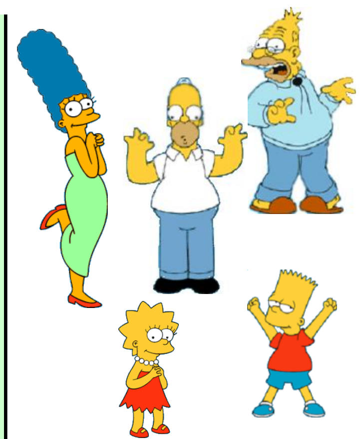
What is a natural grouping among these objects?



What is a natural grouping among these objects?

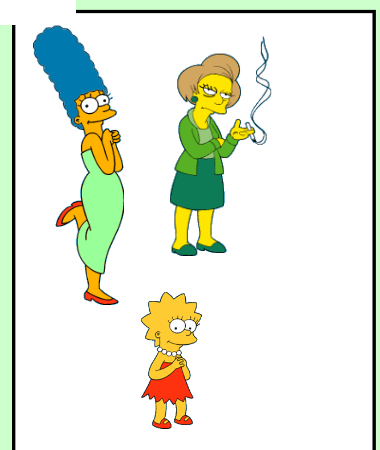


Clustering is subjective

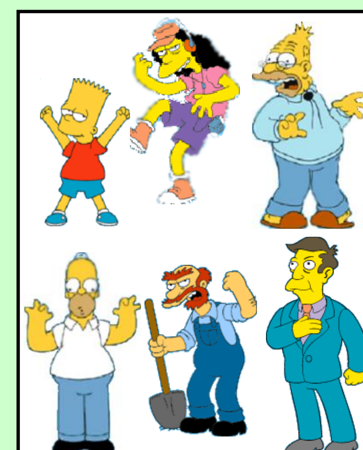


Simpson's Family

School Employees



Females



Males

What is Similarity?

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

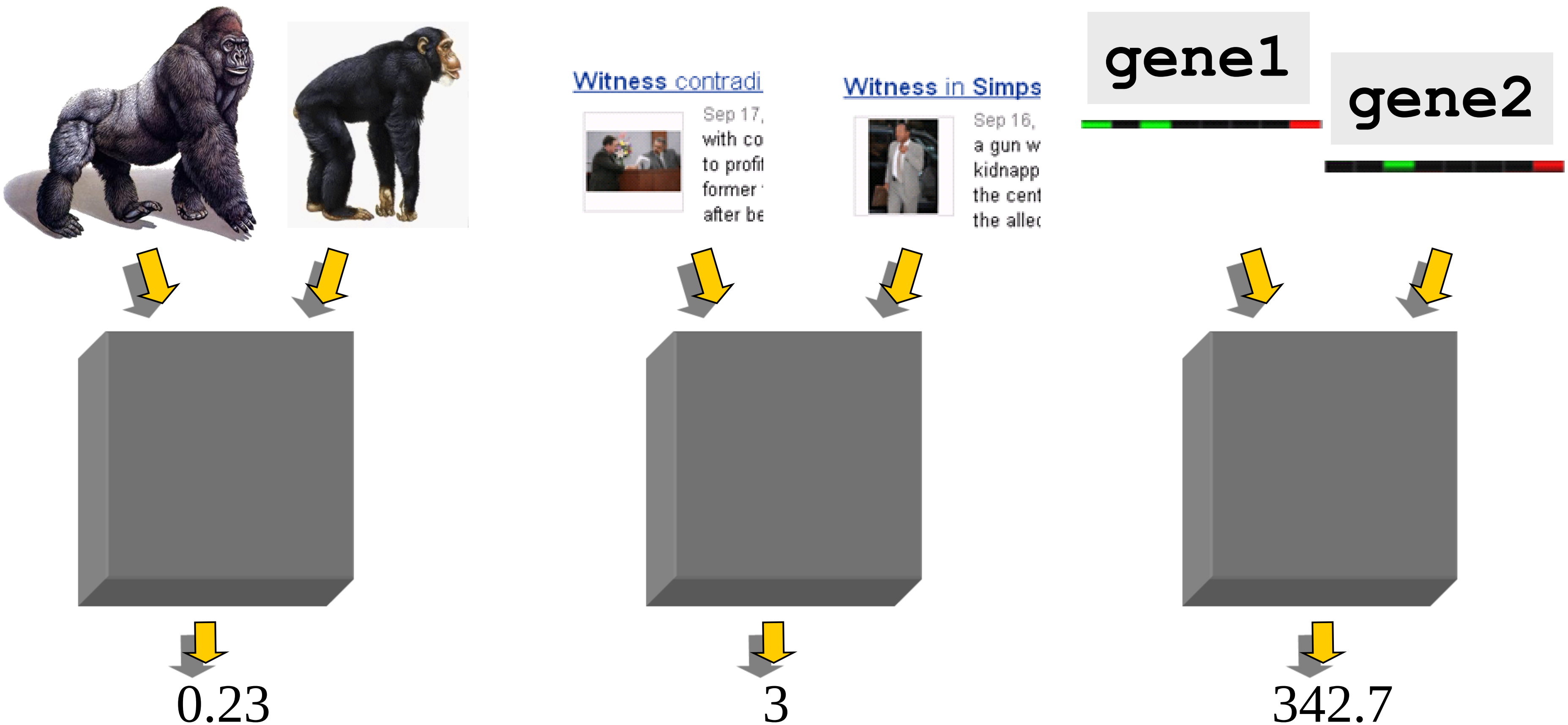
Webster's Dictionary

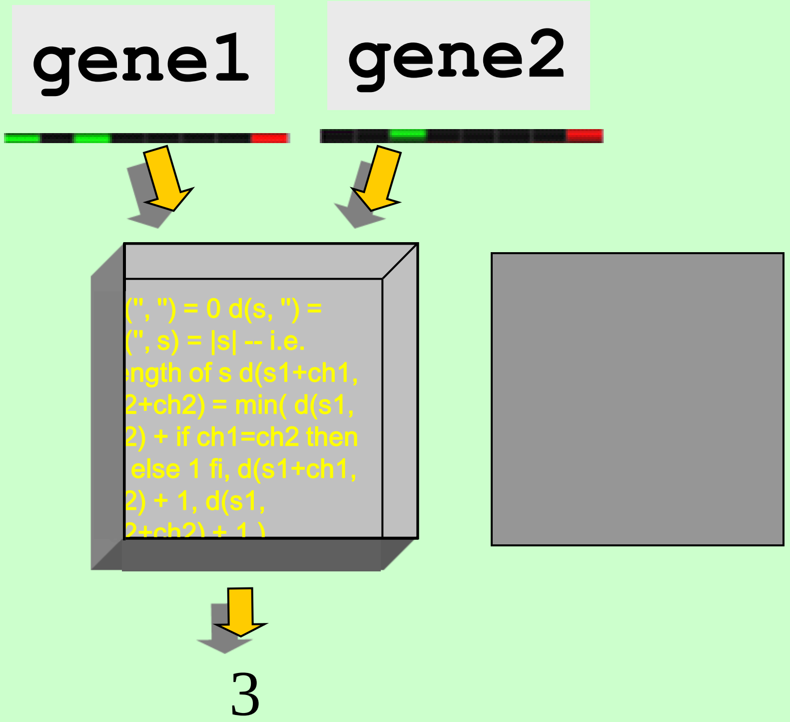
Similarity is hard to define, but… "*We know it when we see it*"

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)

A few examples:

- Euclidian distance

$$d(x,y) = \sqrt{\sum_i (x_i - y_i)^2}$$

- Correlation coefficient

$$s(x,y) = \frac{\sum_i (x_i - \mu_x)(y_i - \mu_y)}{\sigma_x \sigma_y}$$

- Similarity rather than distance
- Can determine similar trends

Outline

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

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
- Interpretability and usability

Optional

- Incorporation of user-specified constraints

Two Types of Clustering

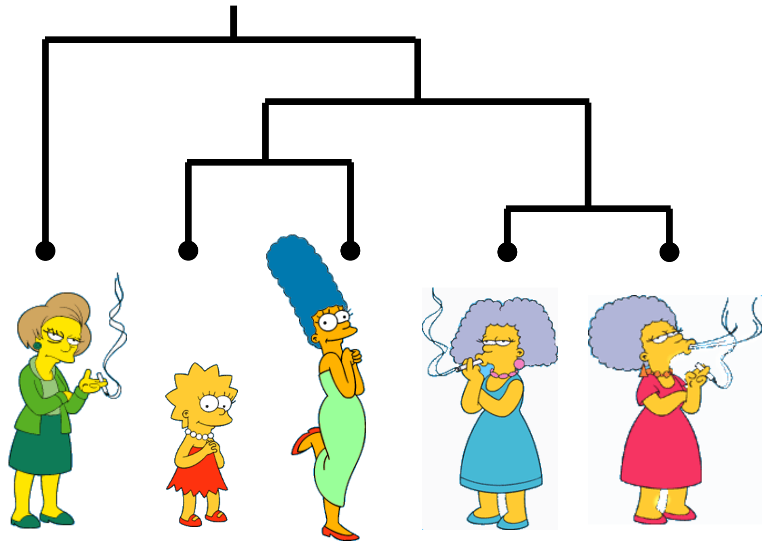
• **Partitional algorithms:** Construct various partitions and then evaluate them by some criterion

• **Hierarchical algorithms:** Create a hierarchical decomposition of the set of objects using some criterion (focus of this class)

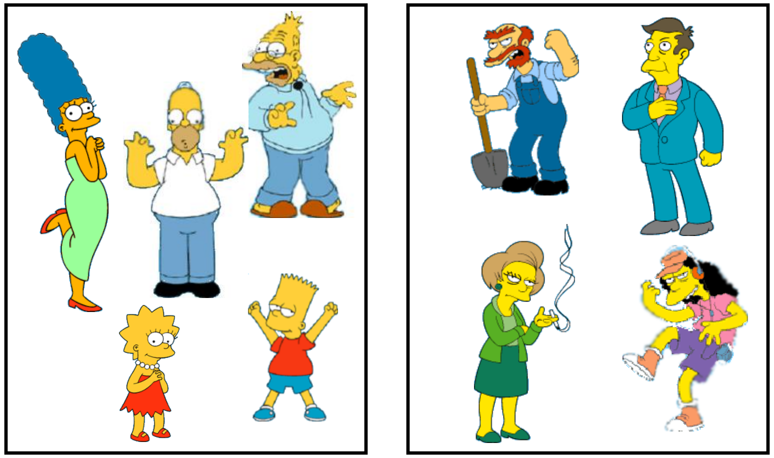
Bottom up or top down

Top down

Hierarchical



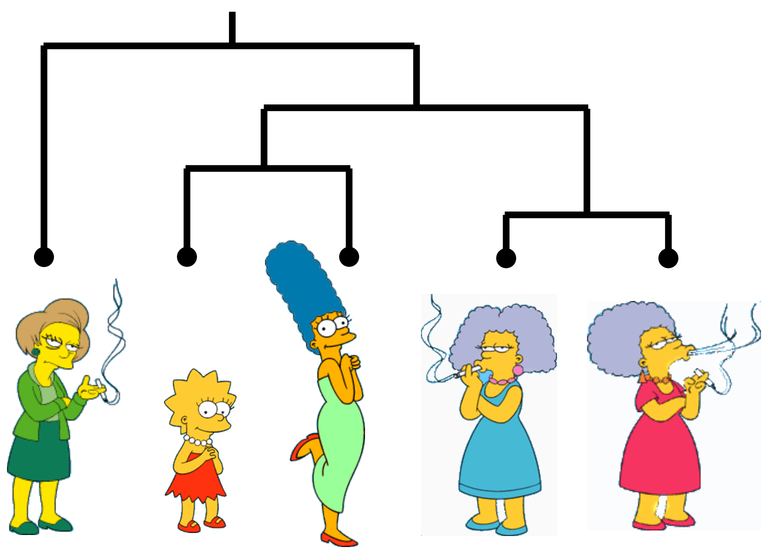
Partitional



(How-to) Hierarchical Clustering

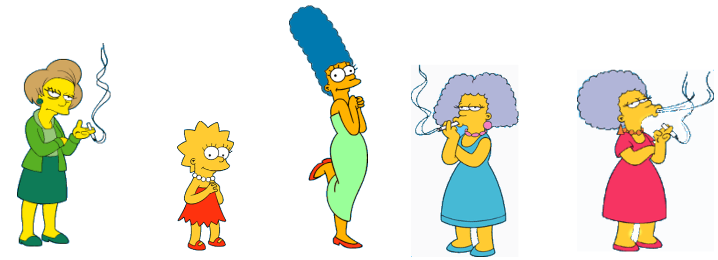
The number of dendrograms with n leafs $= (2n-3)!/[(2^{(n-2)})(n-2)!]$

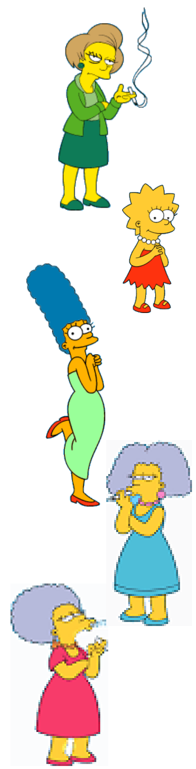
Number of Leafs	Number of Possible Dendrograms
2	1
3	3
4	15
5	105
...	…
10	34,459,425



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.

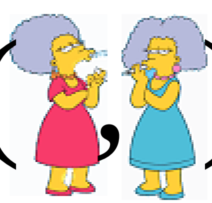
We begin with a distance matrix which contains the distances between every pair of objects in our database.





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

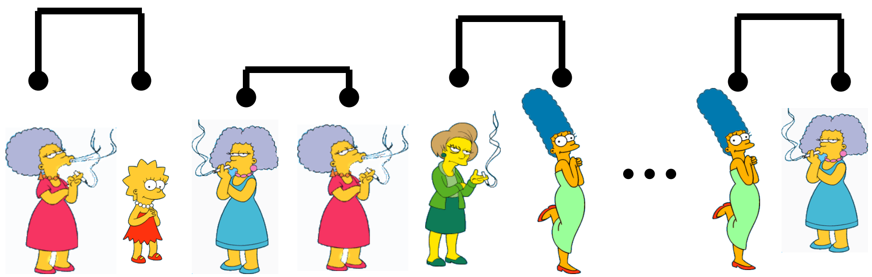
D() = 8

D() = 1

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…

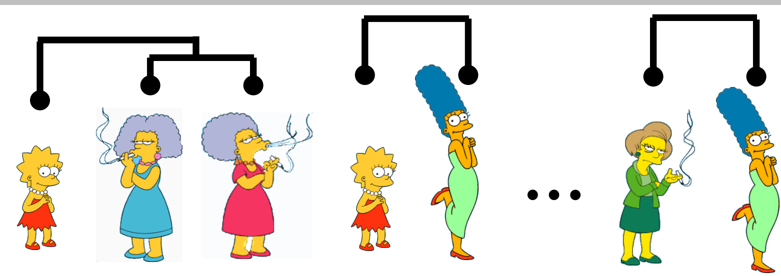


Choose the best

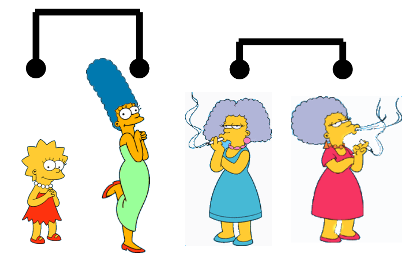
Bottom-Up (agglomerative):

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.

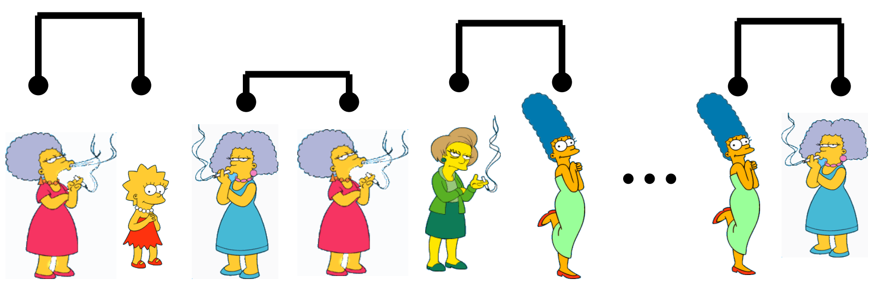
Consider all possible merges…



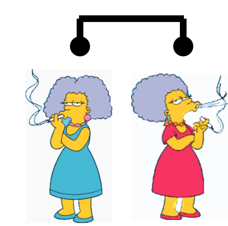
Choose the best



Consider all possible merges…



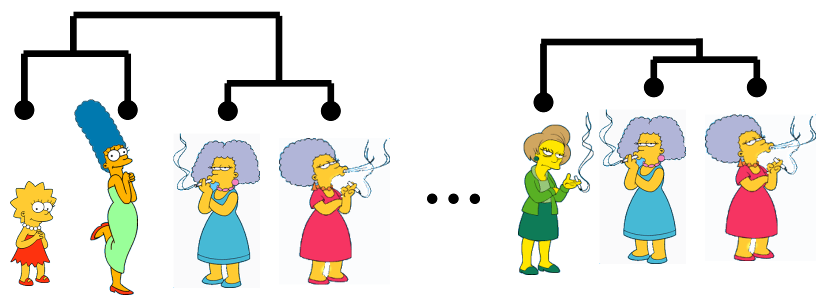
Choose the best



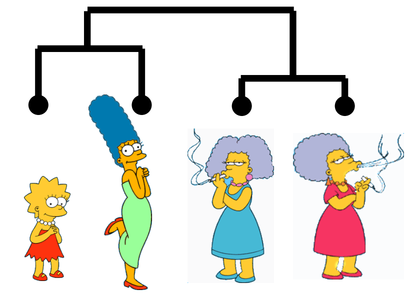
Bottom-Up (agglomerative):

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.

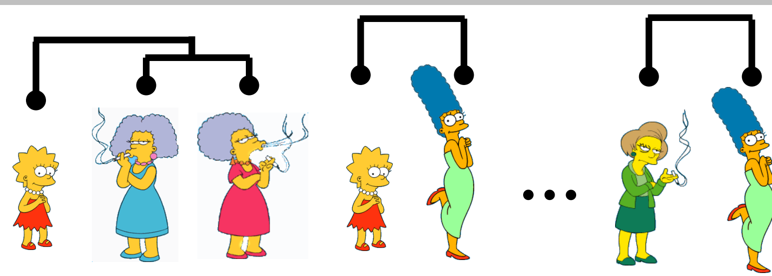
Consider all possible merges…



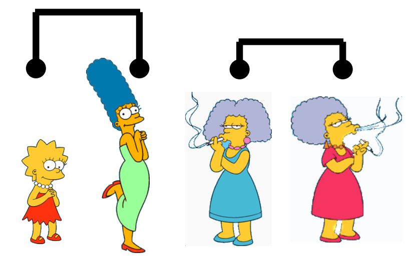
Choose the best



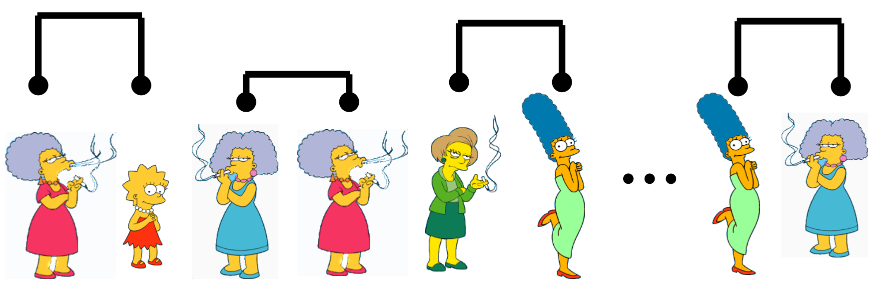
Consider all possible merges…



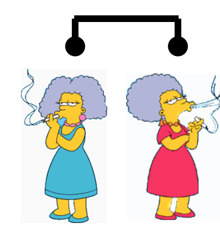
Choose the best



Consider all possible merges…

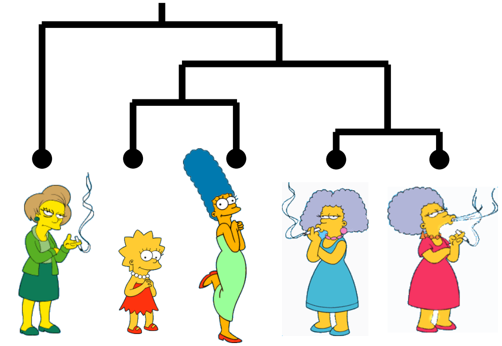


Choose the best

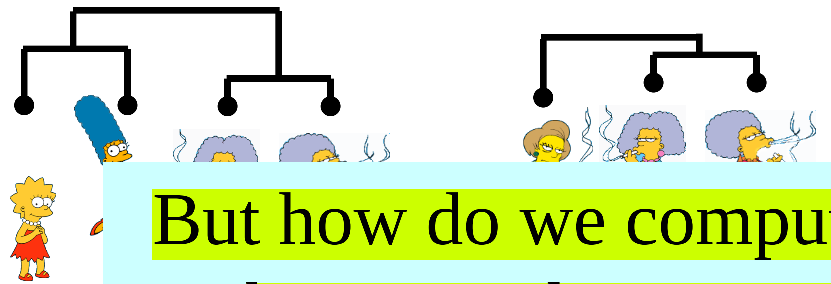


Bottom-Up (agglomerative):

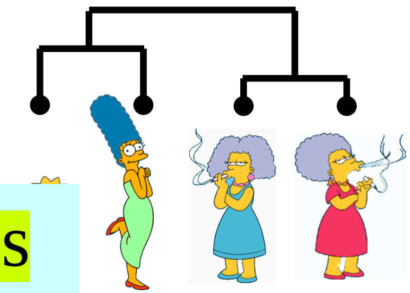
Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.



Consider all possible merges…

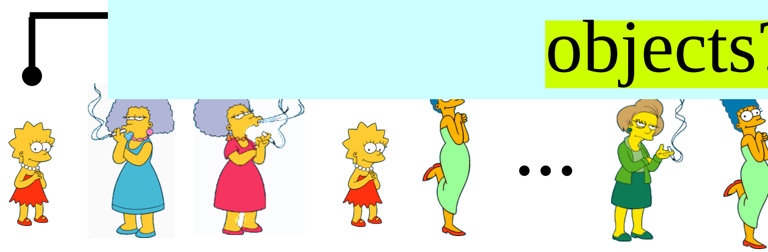


Choose

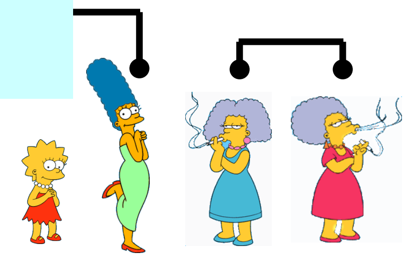


But how do we compute distances between clusters rather than objects?

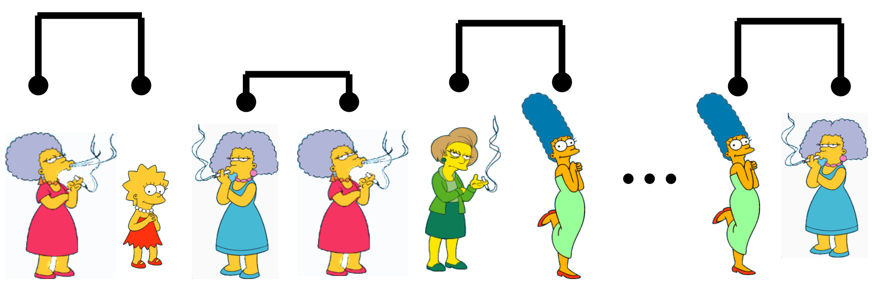
Consider all possible merges…



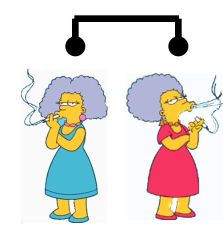
the best



Consider all possible merges…

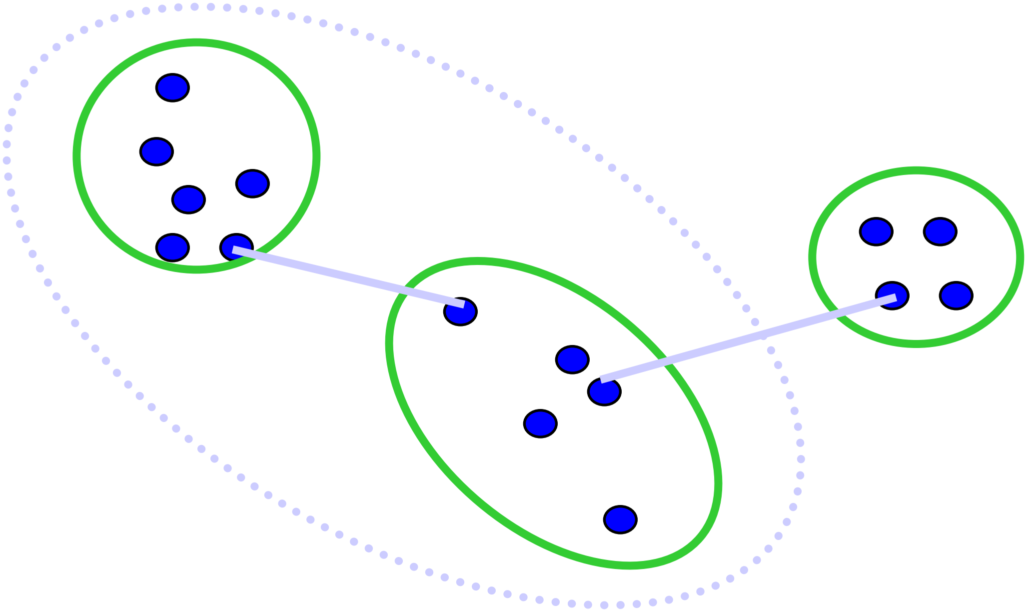


Choose the best



Computing distance between clusters: Single Link

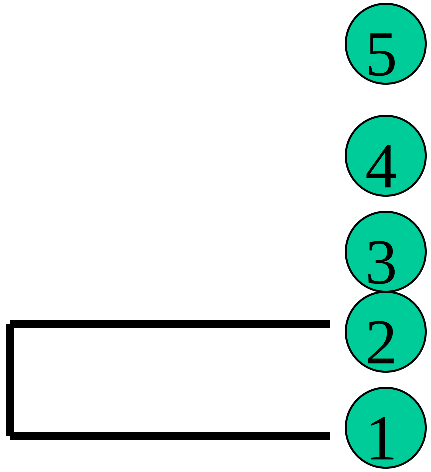
- cluster distance = distance of two closest members in each class



- Potentially long and skinny clusters

Example: single link

$$
\begin{array}{c|ccccc|}
 & 1 & 2 & 3 & 4 & 5 \\
\hline
1 & 0 & & & & \\
2 & 2 & 0 & & & \\
3 & 6 & 3 & 0 & & \\
4 & 10 & 9 & 7 & 0 & \\
5 & 9 & 8 & 5 & 4 & 0
\end{array}
$$



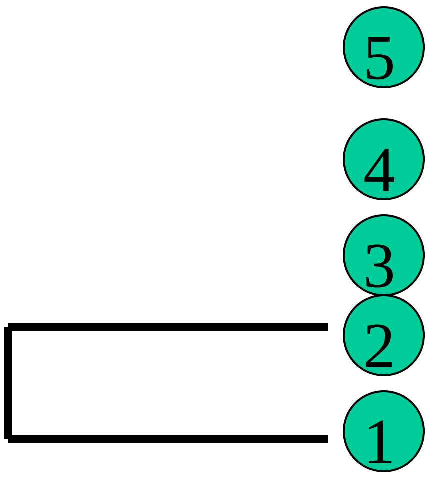
Example: single link

$$
\begin{array}{c|ccccc}
 & 1 & 2 & 3 & 4 & 5 \\
\hline
1 & 0 & & & & \\
2 & 2 & 0 & & & \\
3 & 6 & 3 & 0 & & \\
4 & 10 & 9 & 7 & 0 & \\
5 & 9 & 8 & 5 & 4 & 0
\end{array}
\Rightarrow
\begin{array}{c|cccc}
 & (1,2) & 3 & 4 & 5 \\
\hline
(1,2) & 0 & & & \\
3 & 3 & 0 & & \\
4 & 9 & 7 & 0 & \\
5 & 8 & 5 & 4 & 0
\end{array}
$$

$$d_{(1,2),3} = \min\{d_{1,3}, d_{2,3}\} = \min\{6,3\} = 3$$

$$d_{(1,2),4} = \min\{d_{1,4}, d_{2,4}\} = \min\{10,9\} = 9$$

$$d_{(1,2),5} = \min\{d_{1,5}, d_{2,5}\} = \min\{9,8\} = 8$$

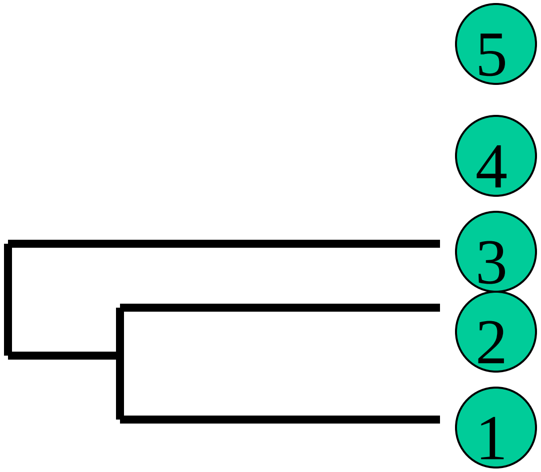


Example: single link

$$
\begin{array}{c|ccccc|}
 & 1 & 2 & 3 & 4 & 5 \\
1 & 0 & & & & \\
2 & 2 & 0 & & & \\
3 & 6 & 3 & 0 & & \\
4 & 10 & 9 & 7 & 0 & \\
5 & 9 & 8 & 5 & 4 & 0
\end{array}
\Rightarrow
\begin{array}{c|cccc|}
 & (1,2) & 3 & 4 & 5 \\
(1,2) & 0 & & & \\
3 & 3 & 0 & & \\
4 & 9 & 7 & 0 & \\
5 & 8 & 5 & 4 & 0
\end{array}
\Rightarrow
\begin{array}{c|ccc|}
 & (1,2,3) & 4 & 5 \\
(1,2,3) & 0 & & \\
4 & 7 & 0 & \\
5 & 5 & 4 & 0
\end{array}
$$

$$d_{(1,2,3),4} = \min\{d_{(1,2),4}, d_{3,4}\} = \min\{9,7\} = 7$$

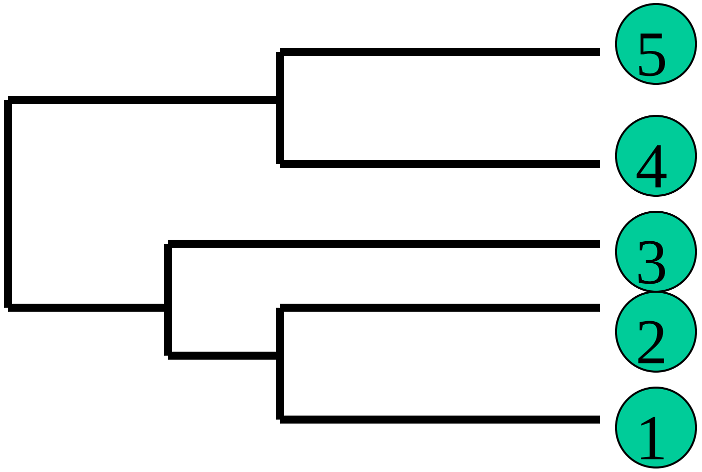
$$d_{(1,2,3),5} = \min\{d_{(1,2),5}, d_{3,5}\} = \min\{8,5\} = 5$$



Example: single link

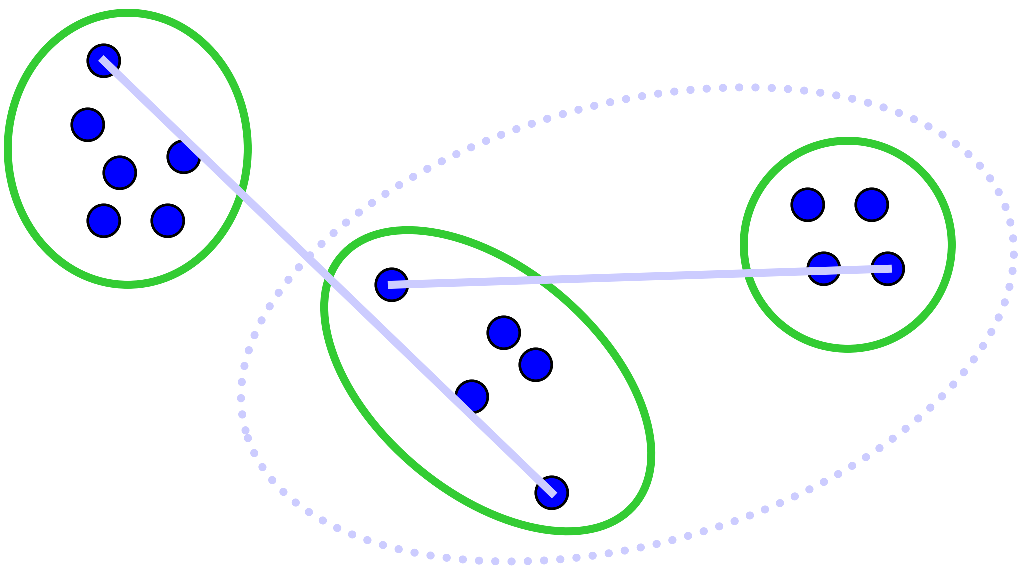
$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}\begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & & & & \\ 2 & 0 & & & \\ 6 & 3 & 0 & & \\ 10 & 9 & 7 & 0 & \\ 9 & 8 & 5 & 4 & 0 \end{bmatrix} \end{array} \Rightarrow \begin{array}{c} \\ (1,2) \\ 3 \\ 4 \\ 5 \end{array}\begin{array}{c} \begin{array}{cccc} (1,2) & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & & & \\ 3 & 0 & & \\ 9 & 7 & 0 & \\ 8 & 5 & 4 & 0 \end{bmatrix} \end{array} \Rightarrow \begin{array}{c} \\ (1,2,3) \\ 4 \\ 5 \end{array}\begin{array}{c} \begin{array}{ccc} (1,2,3) & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & & \\ 7 & 0 & \\ 5 & 4 & 0 \end{bmatrix} \end{array}$$

$$d_{(1,2,3),(4,5)} = \min\{d_{(1,2,3),4}, d_{(1,2,3),5}\} = 5$$



Computing distance between clusters: : Complete Link

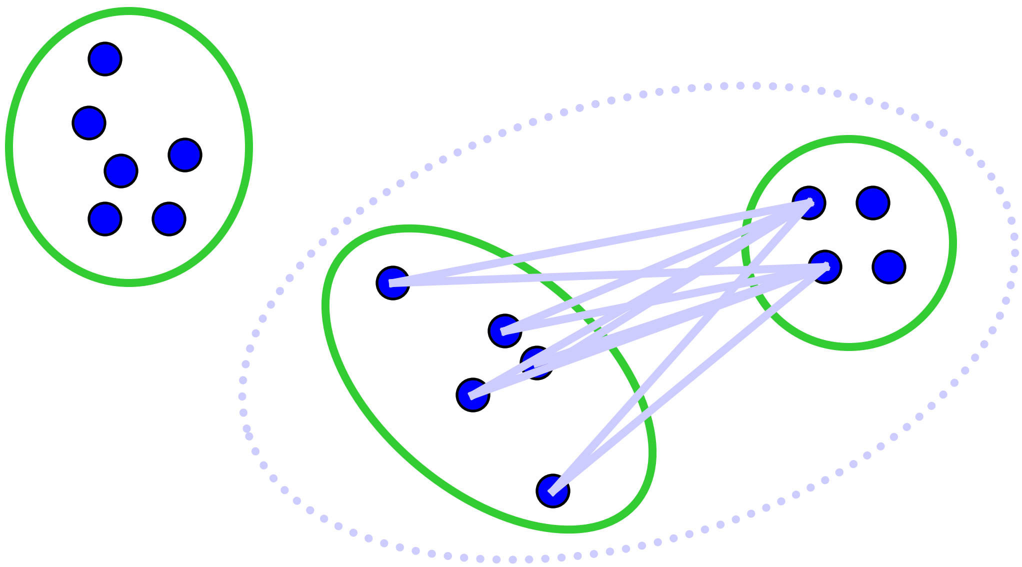
- cluster distance = distance of two farthest members



+ tight clusters

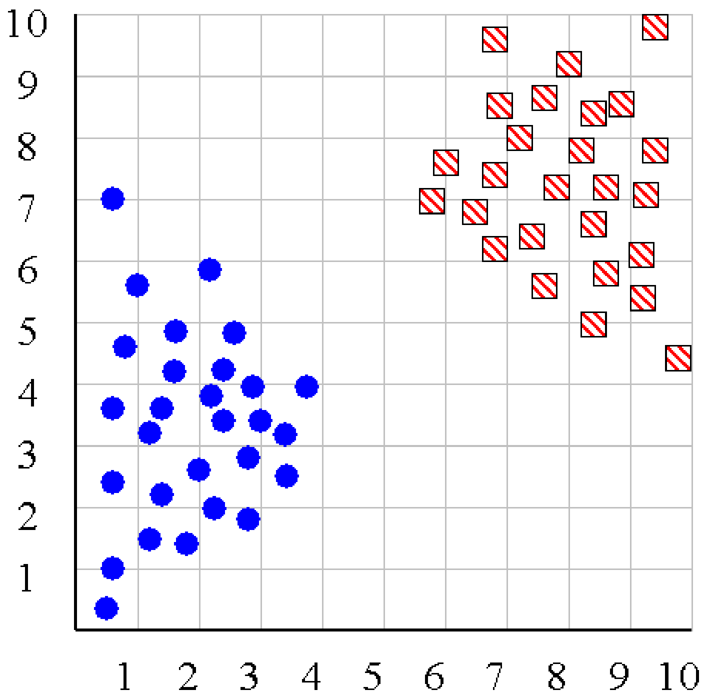
Computing distance between clusters: Average Link

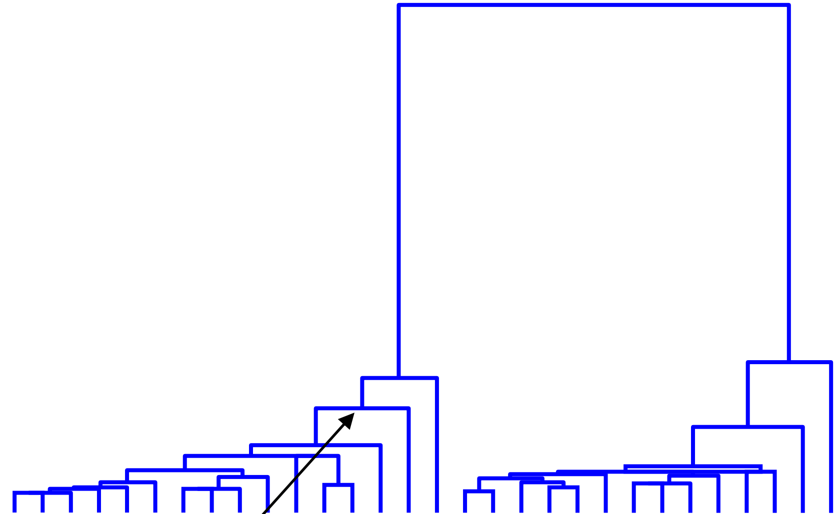
- cluster distance = average distance of all pairs



the most widely used measure

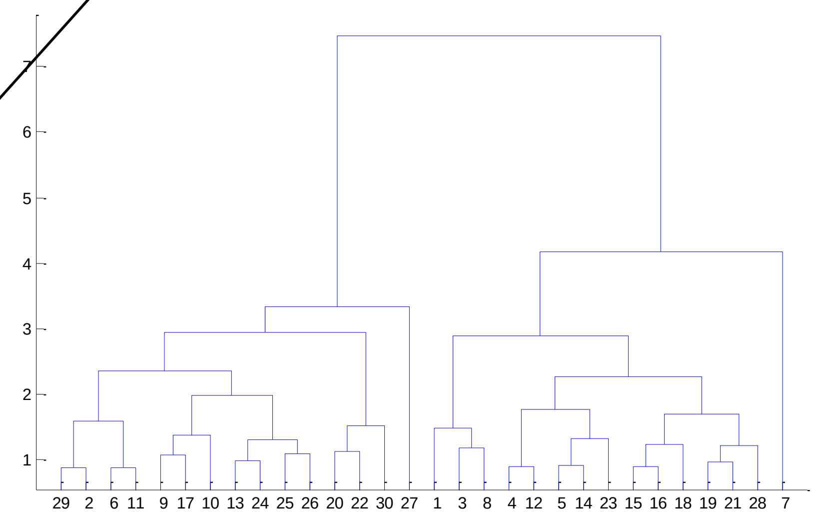
Robust against noise





Single linkage

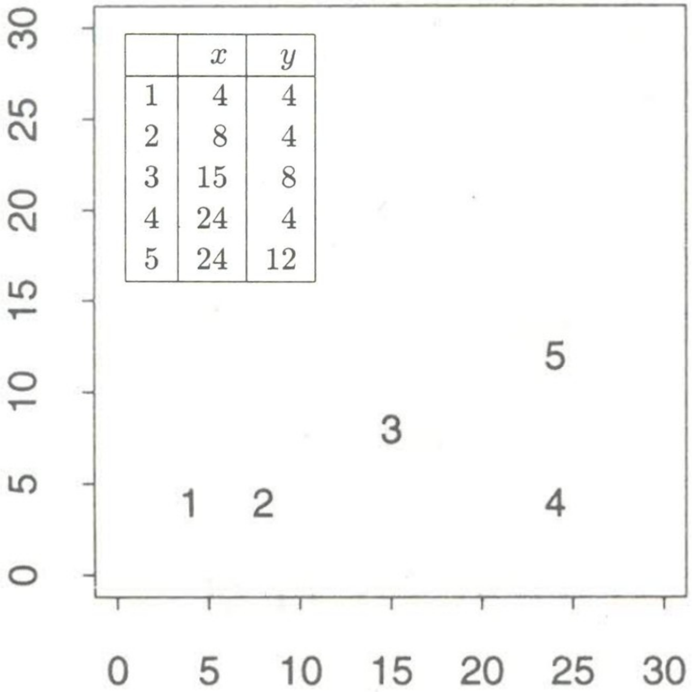
Height represents distance between objects / clusters



Average linkage

Ward's Method

minimum-variance method.



	x	y
1	4	4
2	8	4
3	15	8
4	24	4
5	24	12

consider merging $\{1\}$ and $\{2\}$.

The squared error for cluster $\{1,2\}$ is

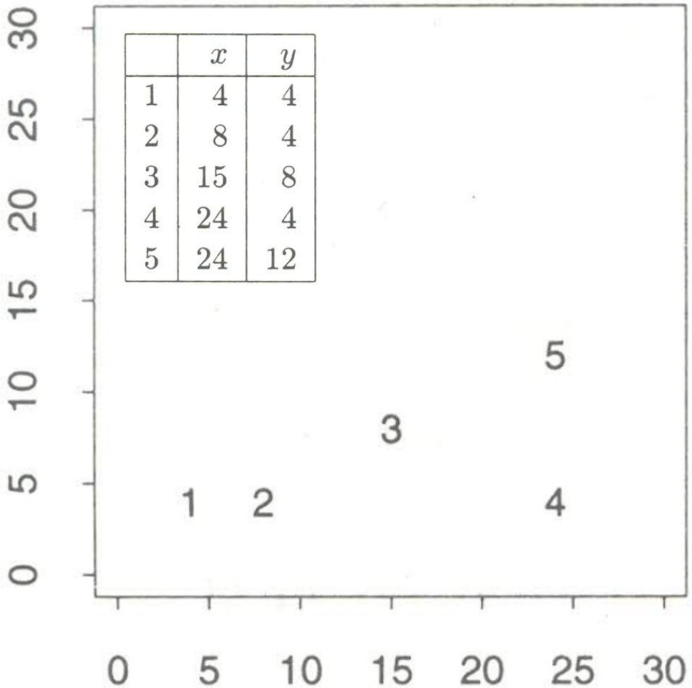
$$(4-6)^2 + (8-6)^2 + (4-4)^2 + (4-4)^2 = 8.$$

The squared error for each of the other clusters $\{3\}$, $\{4\}$, and $\{5\}$ is 0. Thus the total squared error for the clusters $\{1,2\}$, $\{3\}$,$\{4\}$,$\{5\}$ is

$$8+0+0+0=8.$$

The squared error E for the entire cluster is the sum of the squared errors of the samples

$$E = \sum_{i=1}^{m}\sum_{j=1}^{d}(x_{ij}-\mu_j)^2 = m\sigma^2.$$



	x	y
1	4	4
2	8	4
3	15	8
4	24	4
5	24	12

Ward's Method

minimum-variance method.

consider merging $\{1\}$ and $\{2\}$.

The squared error for cluster $\{1, 2\}$ is

$$(4-6)^2 + (8-6)^2 + (4-4)^2 + (4-4)^2 = 8.$$

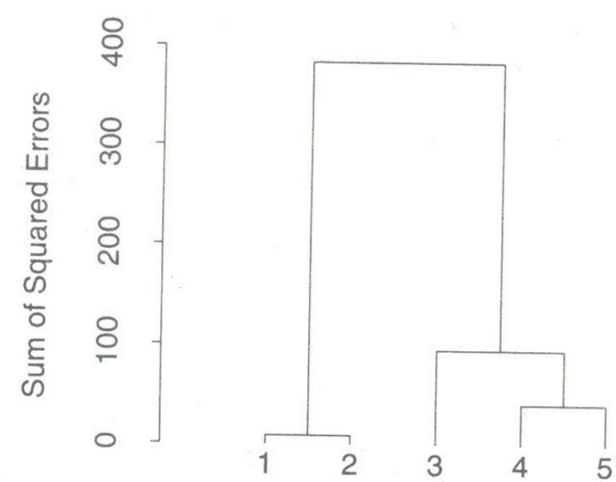
The squared error for each of the other clusters $\{3\}$, $\{4\}$, and $\{5\}$ is 0. Thus the total squared error for the clusters $\{1, 2\}$, $\{3\}$,$\{4\}$,$\{5\}$ is

$$8 + 0 + 0 + 0 = 8.$$

Clusters	Squared Error, E
{1,2},{3},{4},{5}	8.0
{1,3},{2},{4},{5}	68.5
{1,4},{2},{3},{5}	200.0
{1,5},{2},{3},{4}	232.0
{2,3},{1},{4},{5}	32.5
{2,4},{1},{3},{5}	128.0
{2,5},{1},{3},{4}	160.0
{3,4},{1},{2},{5}	48.5
{3,5},{1},{2},{4}	48.5
{4,5},{1},{2},{3}	32.0

Clusters	Squared Error, E
{1,2,3},{4},{5}	72.7
{1,2,4},{3},{5}	224.0
{1,2,5},{3},{4}	266.7
{1,2},{3,4},{5}	56.5
{1,2},{3,5},{4}	56.5
{1,2},{4,5},{3}	40.0

Clusters	Squared Error, E
{1,2,3},{4,5}	104.7
{1,2,4,5},{3}	380.0
{1,2},{3,4,5}	94.0

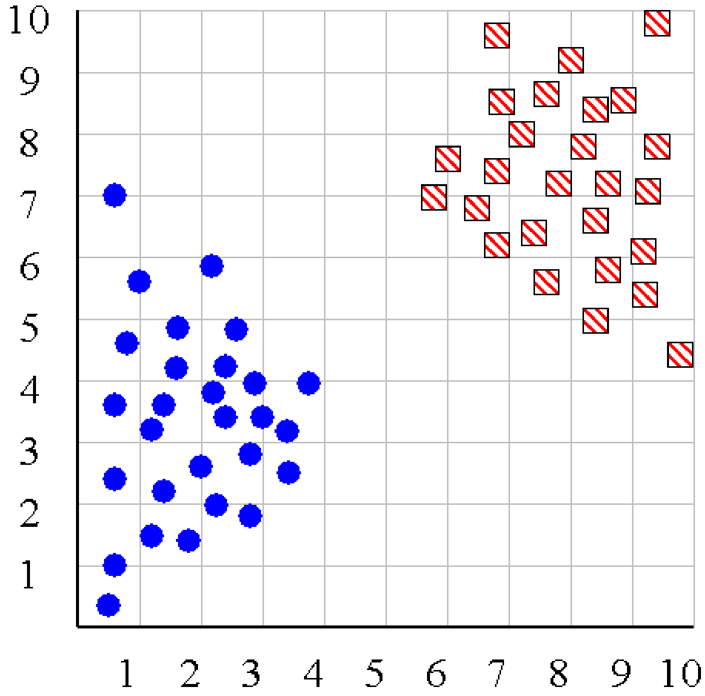


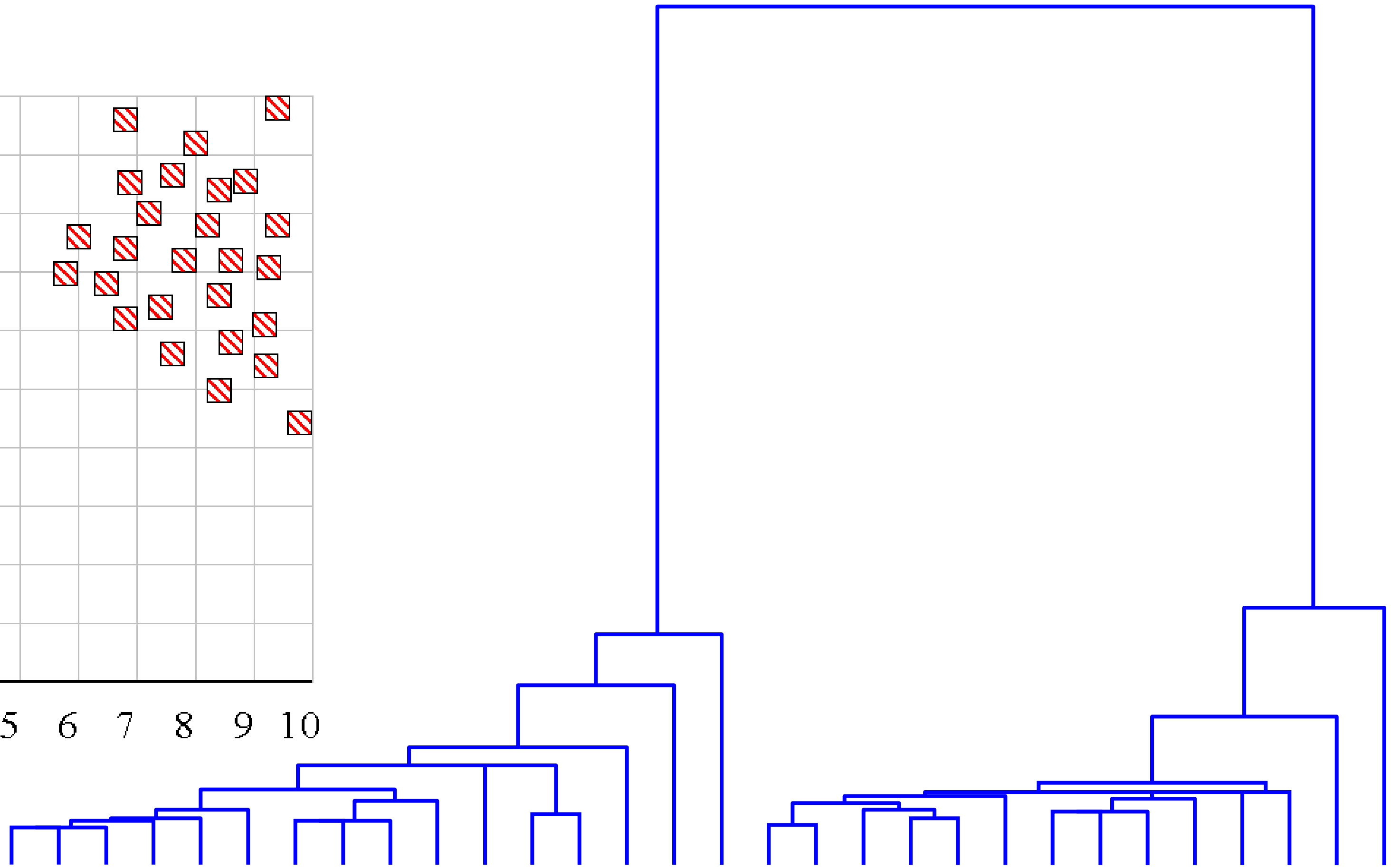
Summary of Hierarchal Clustering Methods

- 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.

But what are the clusters?

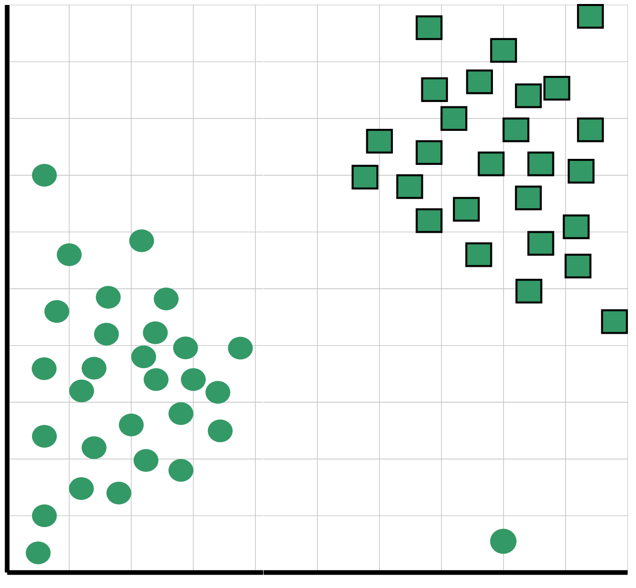
In some cases we can determine the “correct” number of clusters. However, things are rarely this clear cut, unfortunately.



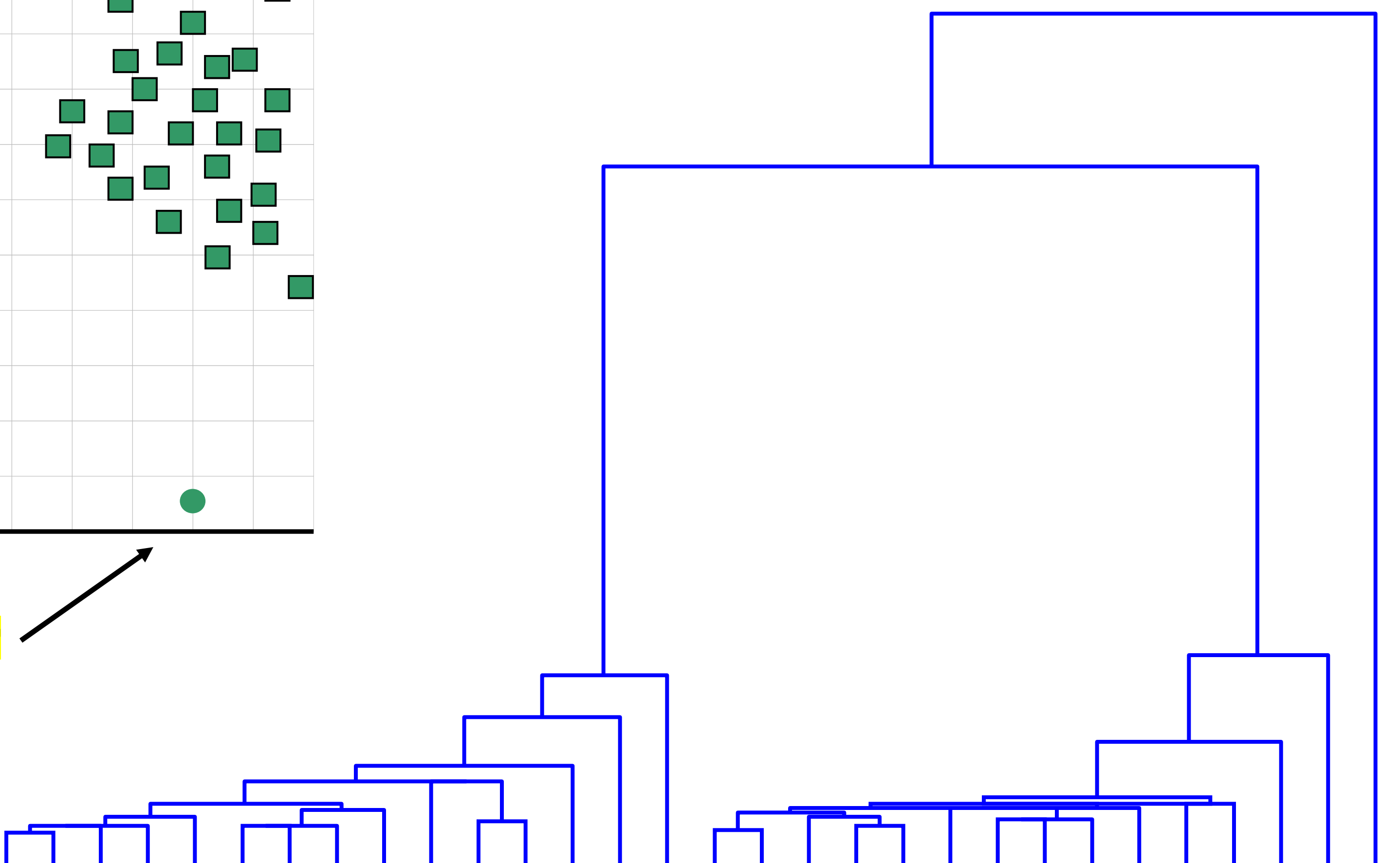


One potential use of a dendrogram is to detect outliers

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



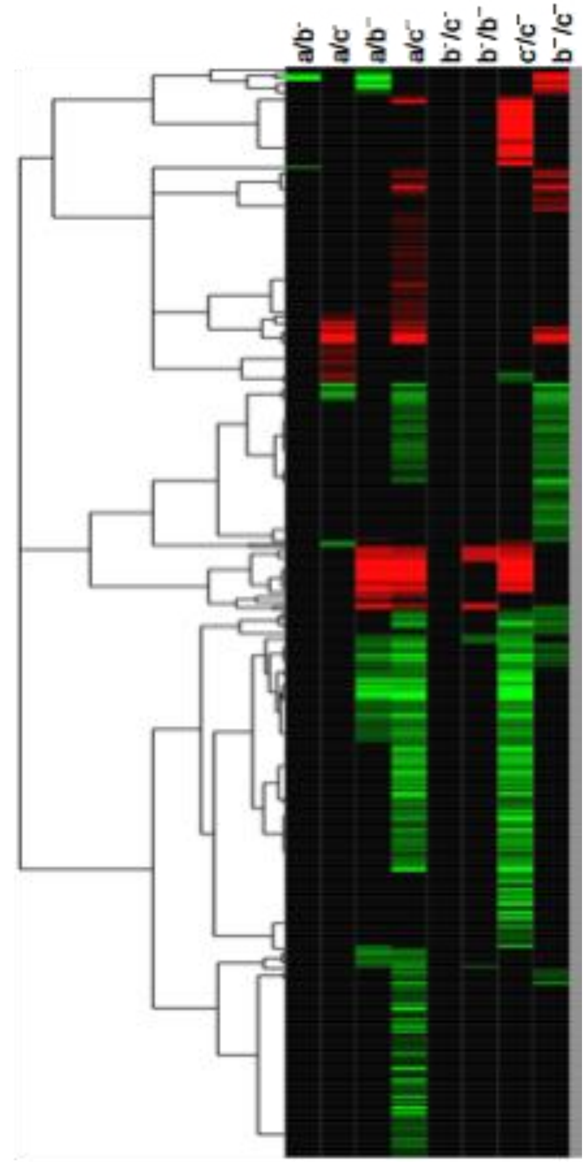
Outlier



Example: clustering genes

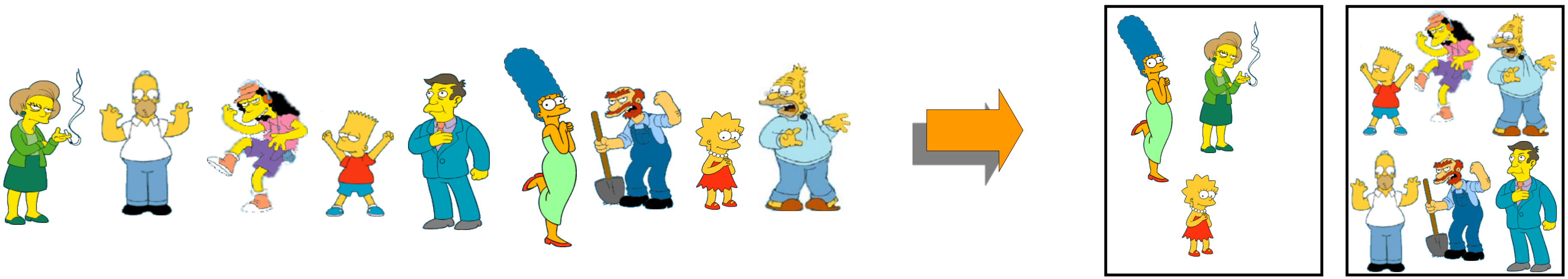
- Microarrays measures the activities of all genes in different conditions

- Clustering genes can help determine new functions for unknown genes



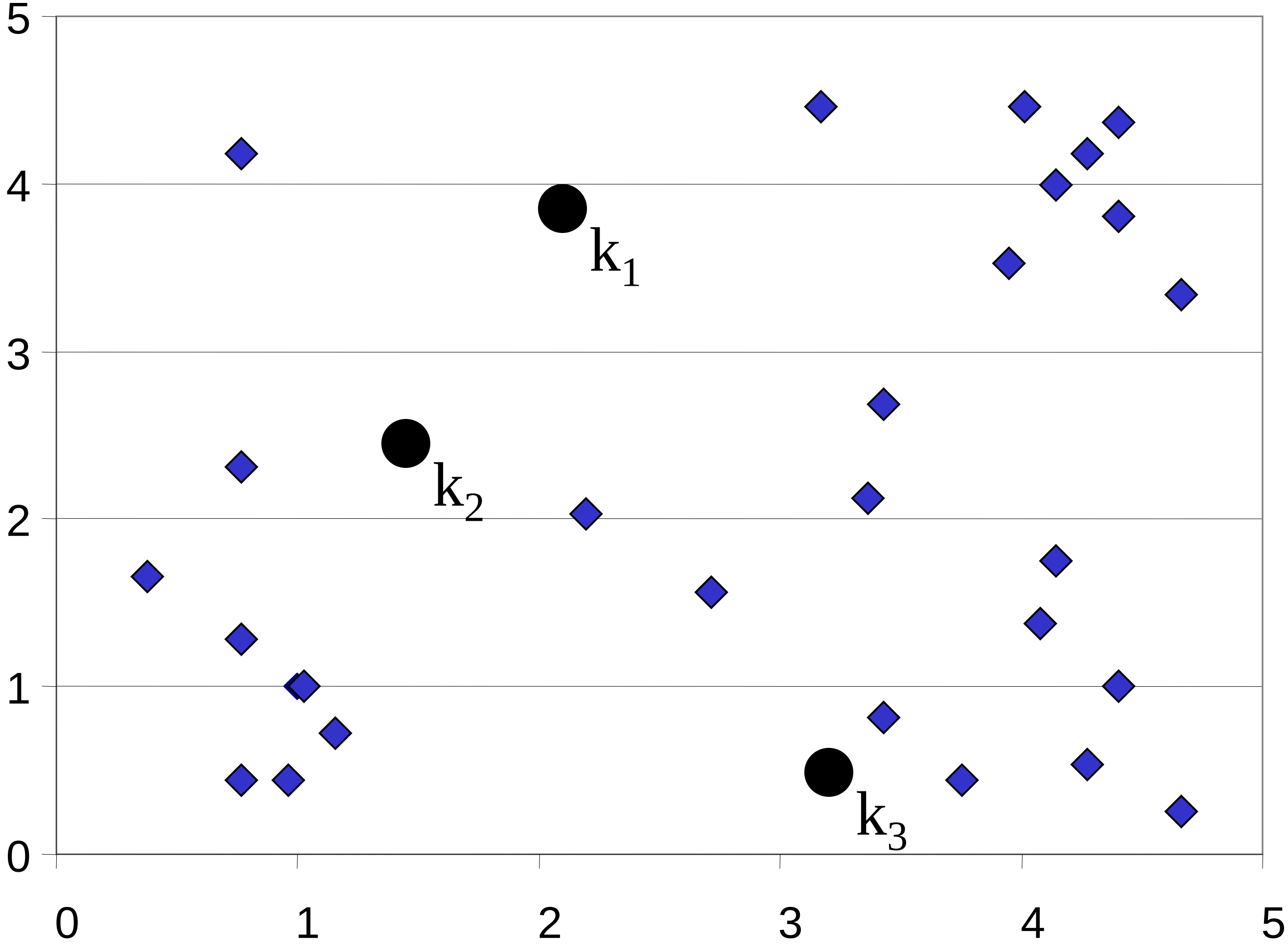
Partitional Clustering

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



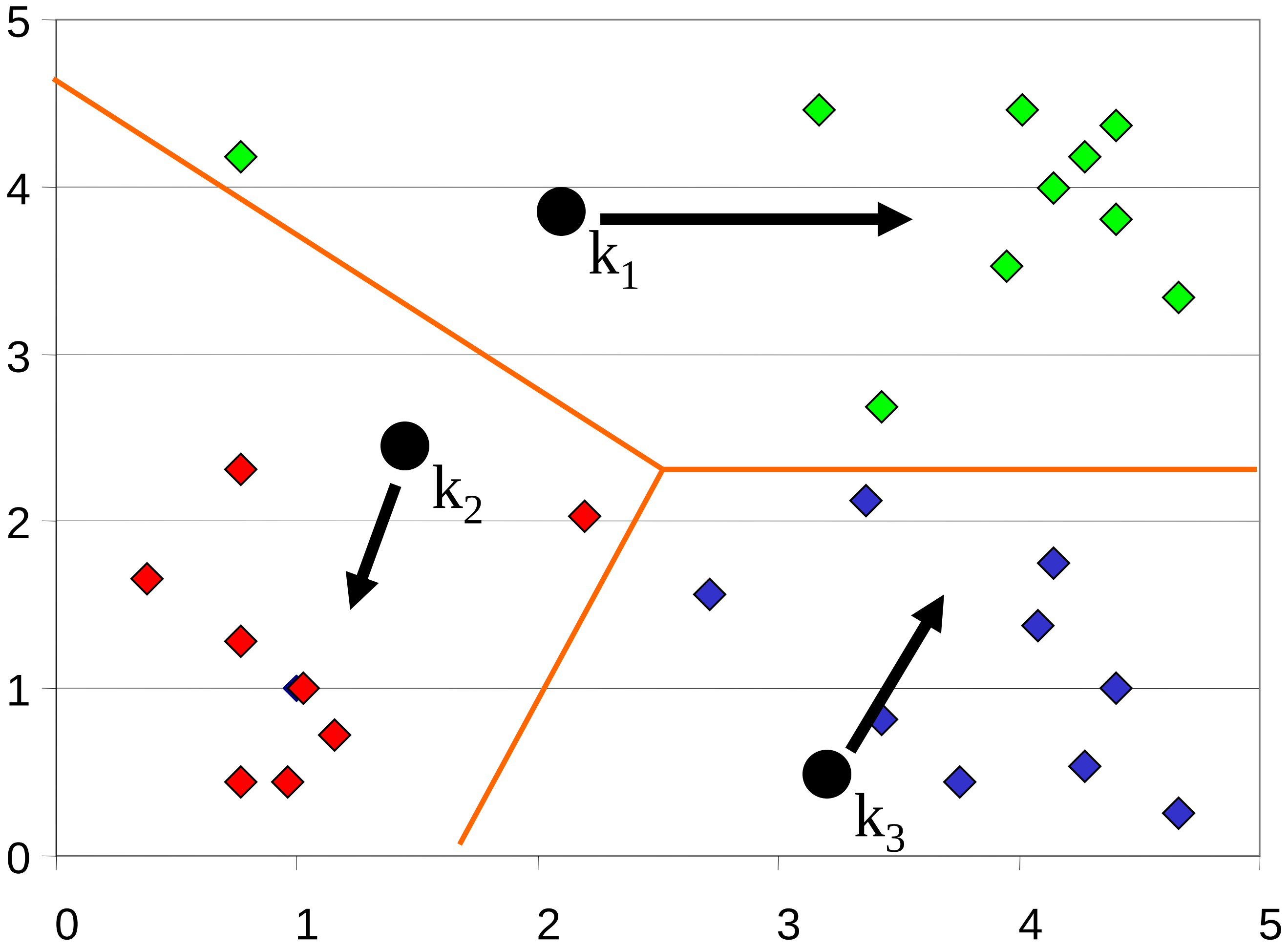
K-means Clustering: Initialization

Decide K, and initialize K centers (randomly)



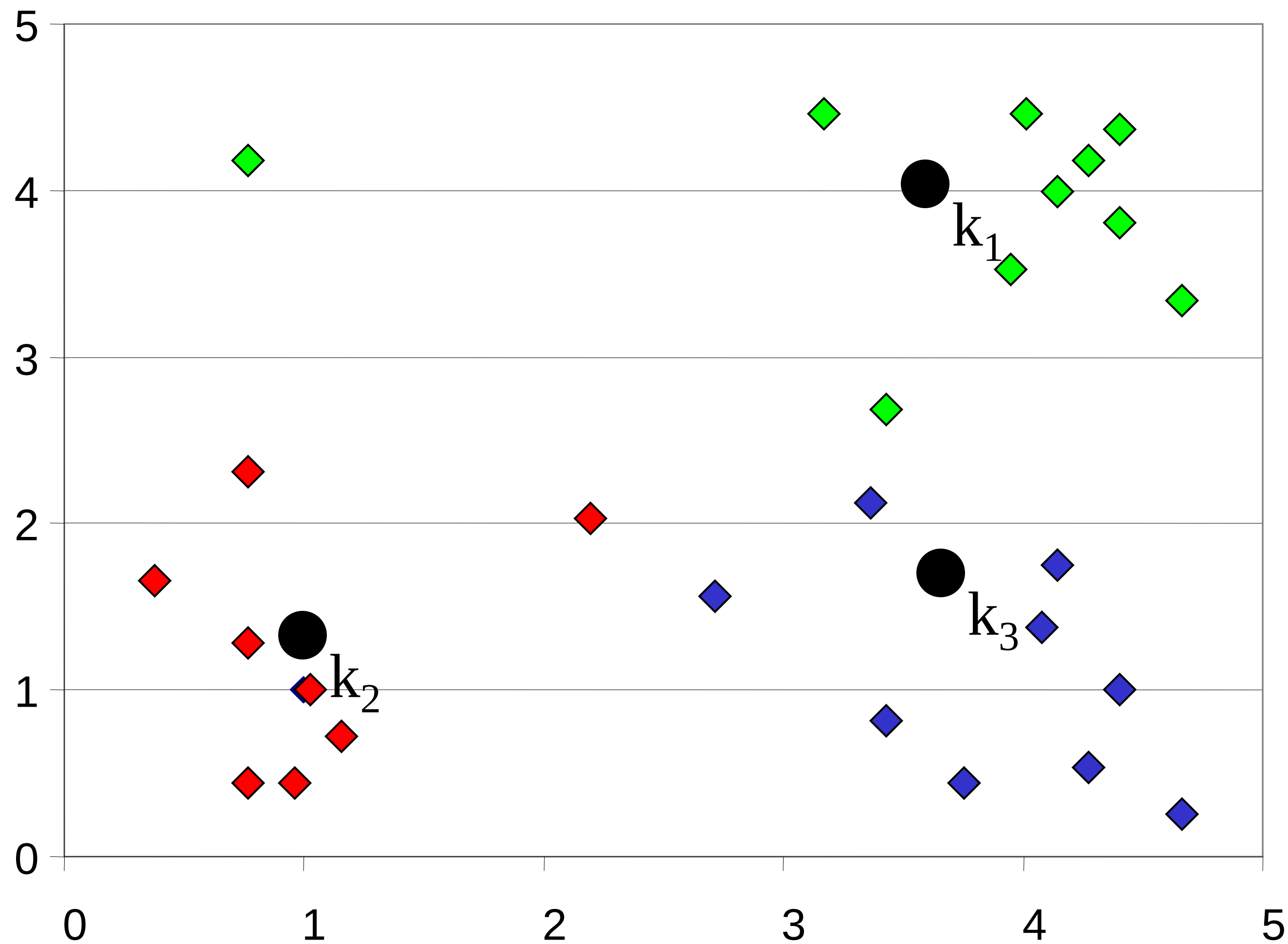
K-means Clustering: Iteration 1

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



K-means Clustering: Iteration 2

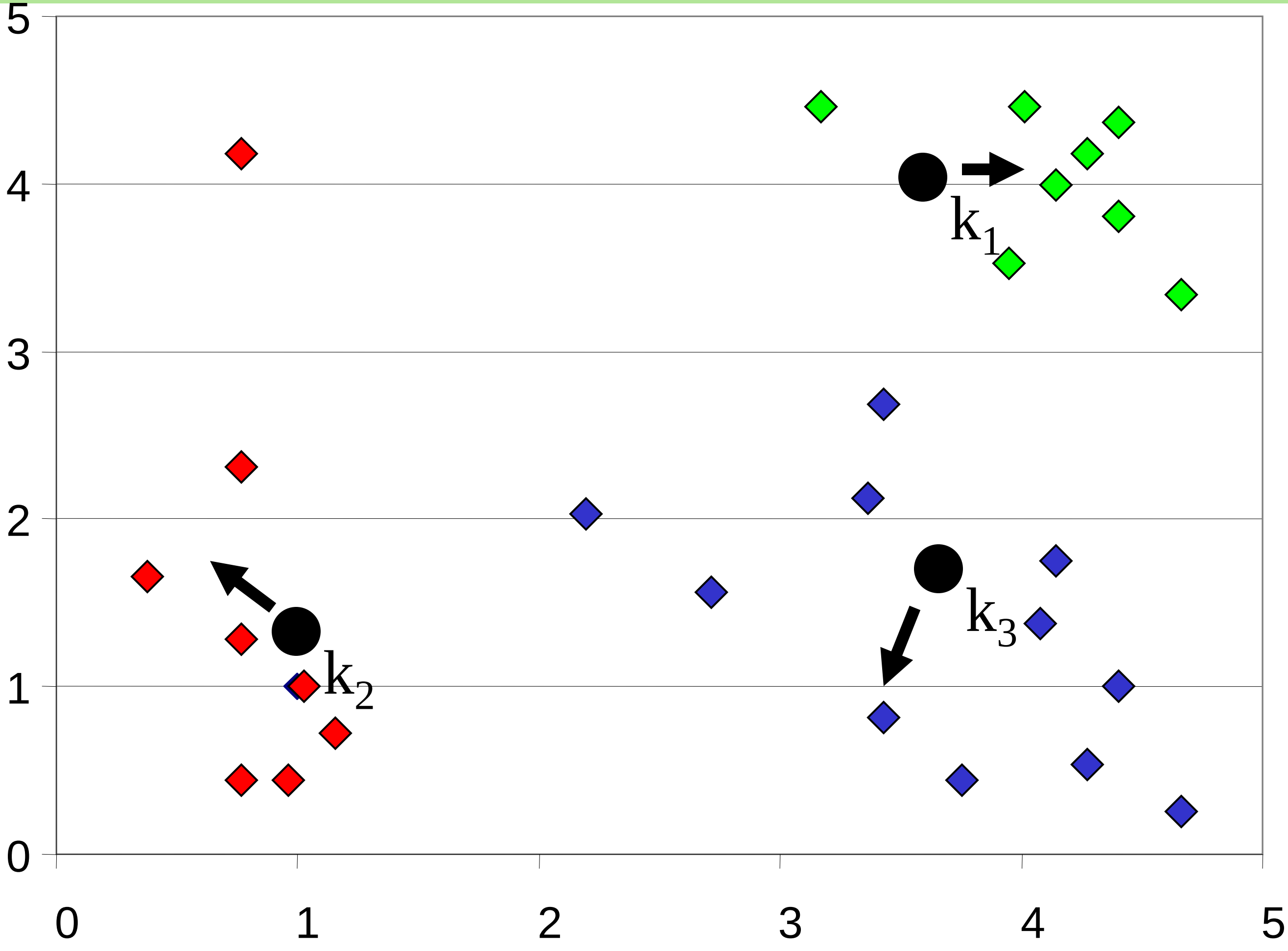
After moving centers, re-assign the objects…



K-means Clustering: Iteration 2

After moving centers, re-assign the objects to nearest centers.

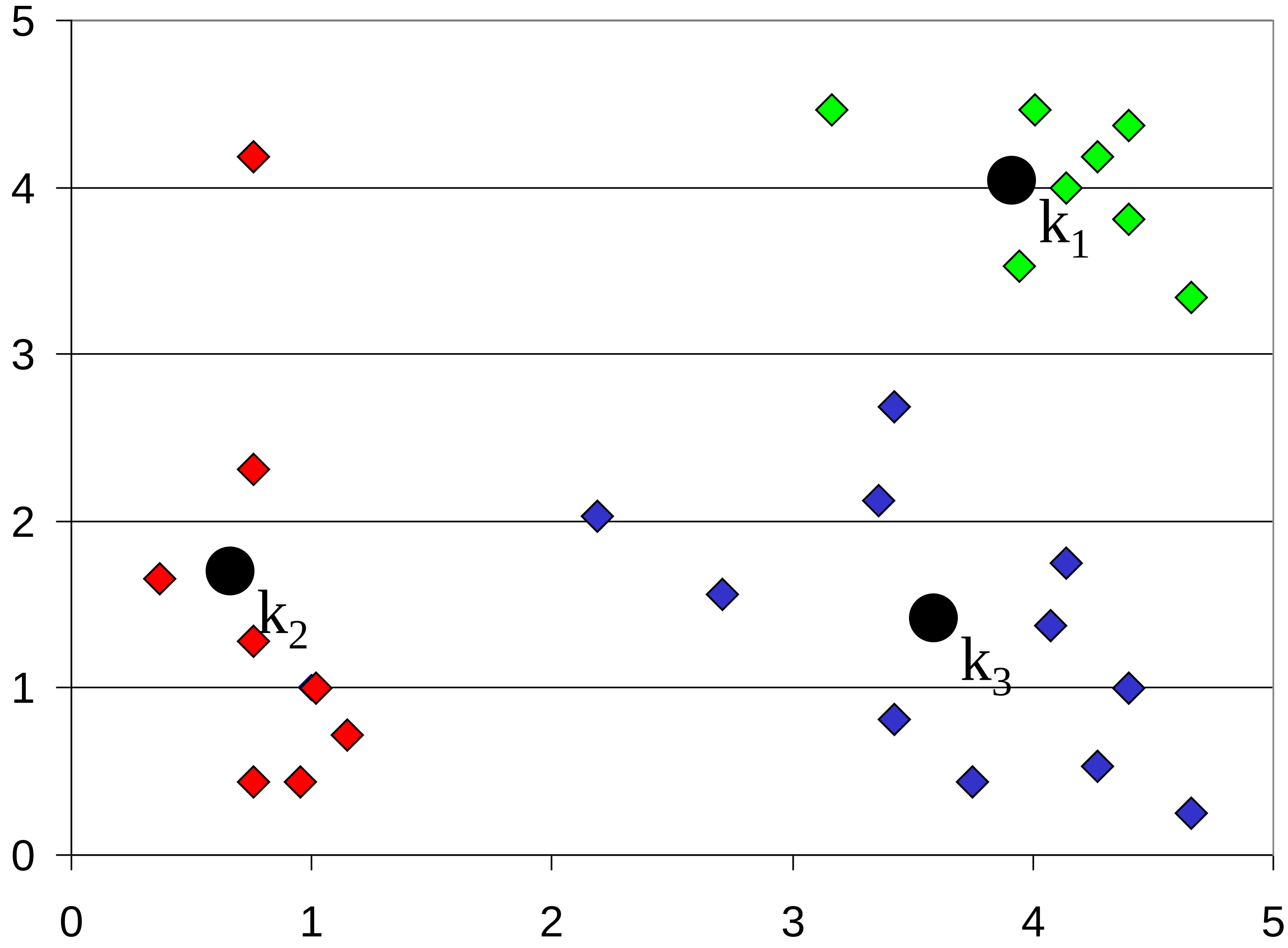
Move a center to the mean of its new members.



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.

Algorithm *k-means*

1. Decide on a value for K, th

2. Initialize the K cluster cent
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

Use one of the distance / similarity functions we discussed earlier

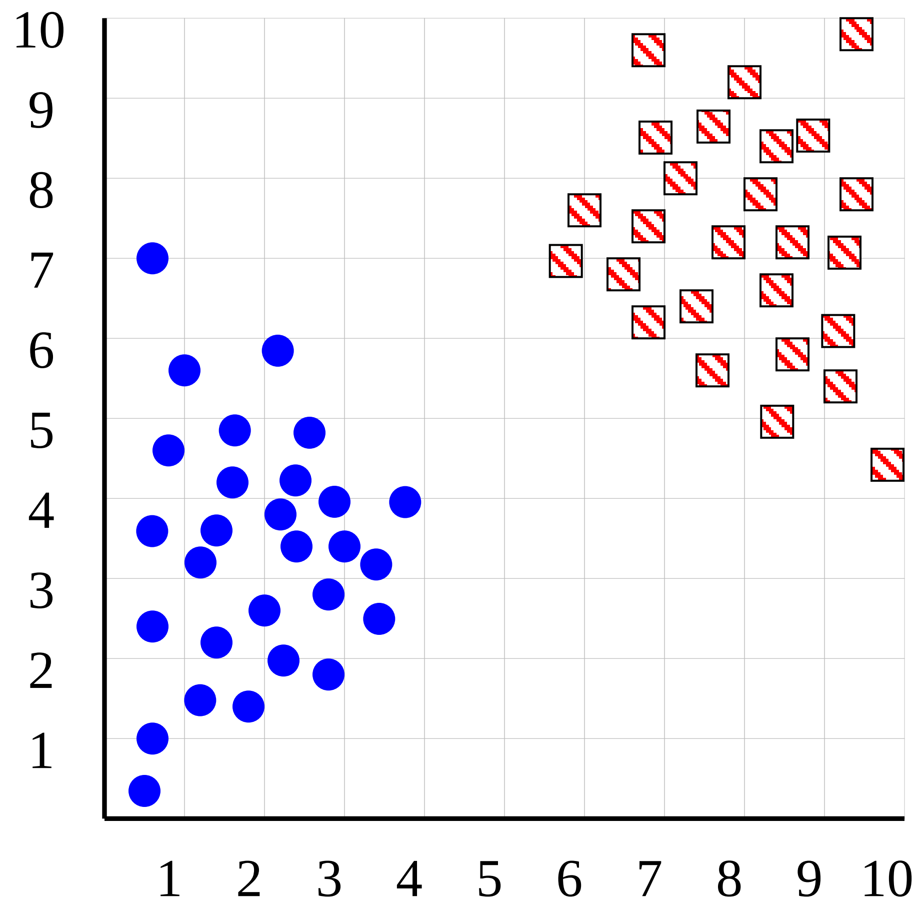
Average / median of class members

Summary: *K-Means*

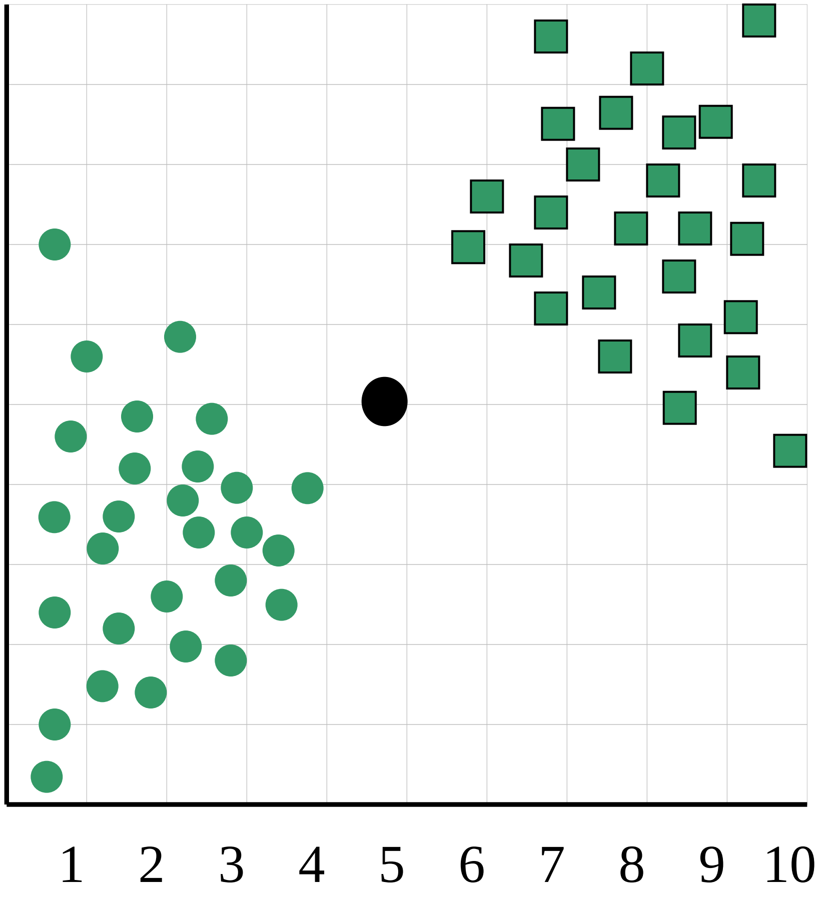
- Strength
  - Simple, easy to implement and debug
  - Intuitive objective function: optimizes intra-cluster similarity
  - *Relatively efficient*: $O(tkn)$, where n is # objects, k is # clusters, and t is # iterations. Normally, k, $t << n$.
- Weakness
  - Applicable only when *mean* is defined, what 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

How can we tell the right number of clusters?

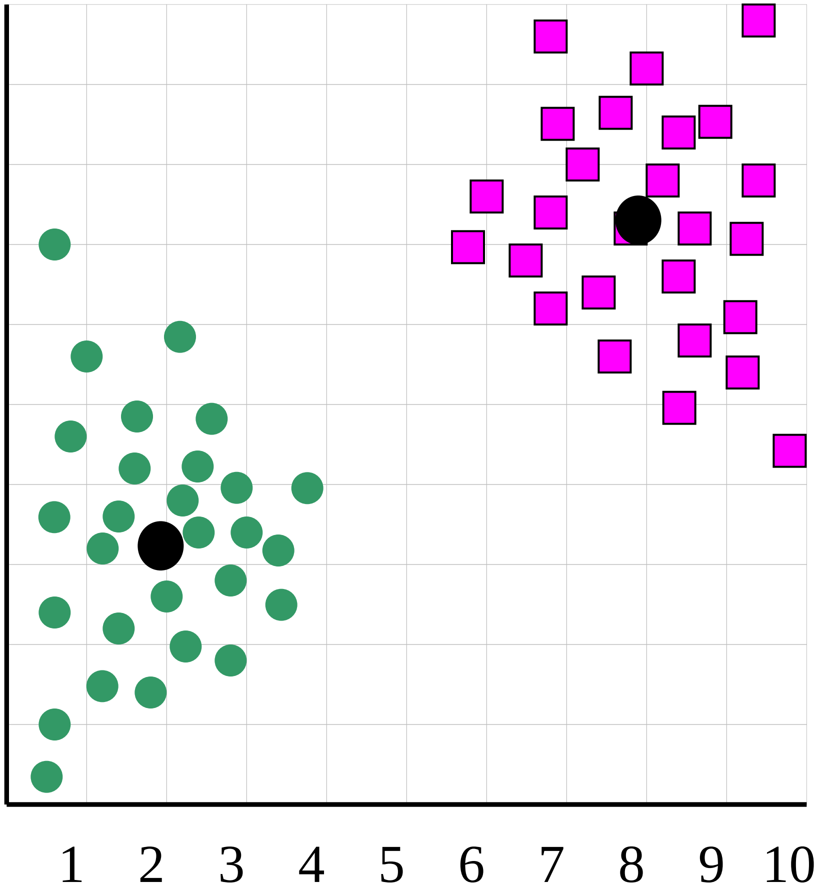
In general, this is a unsolved problem. However there are many approximate methods. In the next few slides we will see an example.



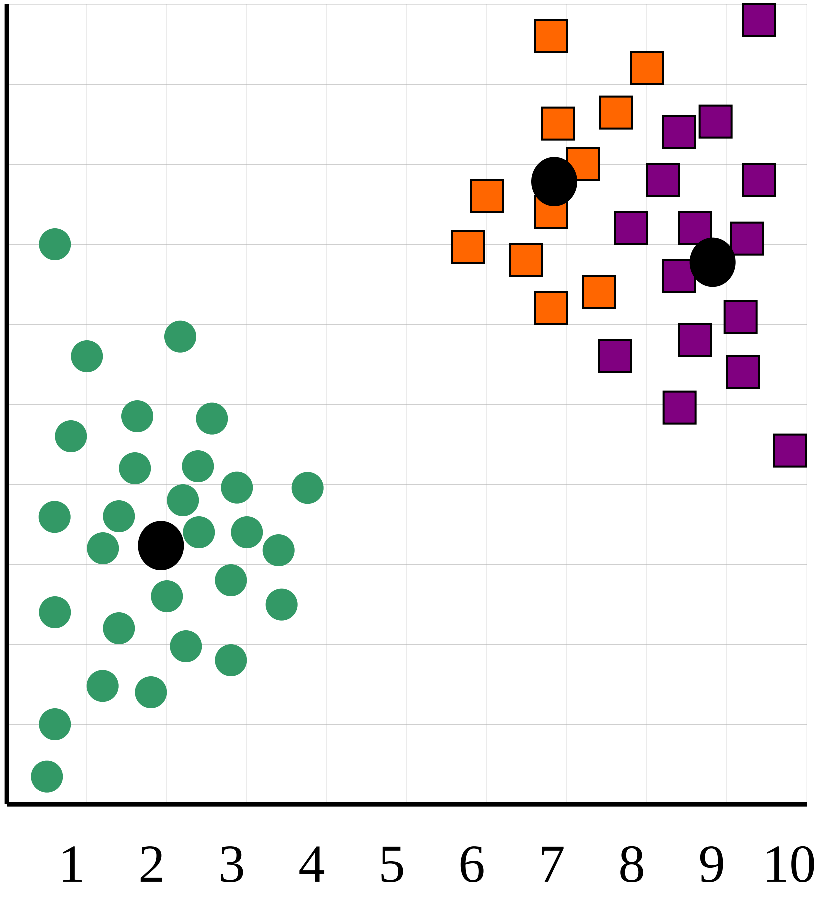
When k = 1, the objective function is 873.0



When k = 2, the objective function is 173.1

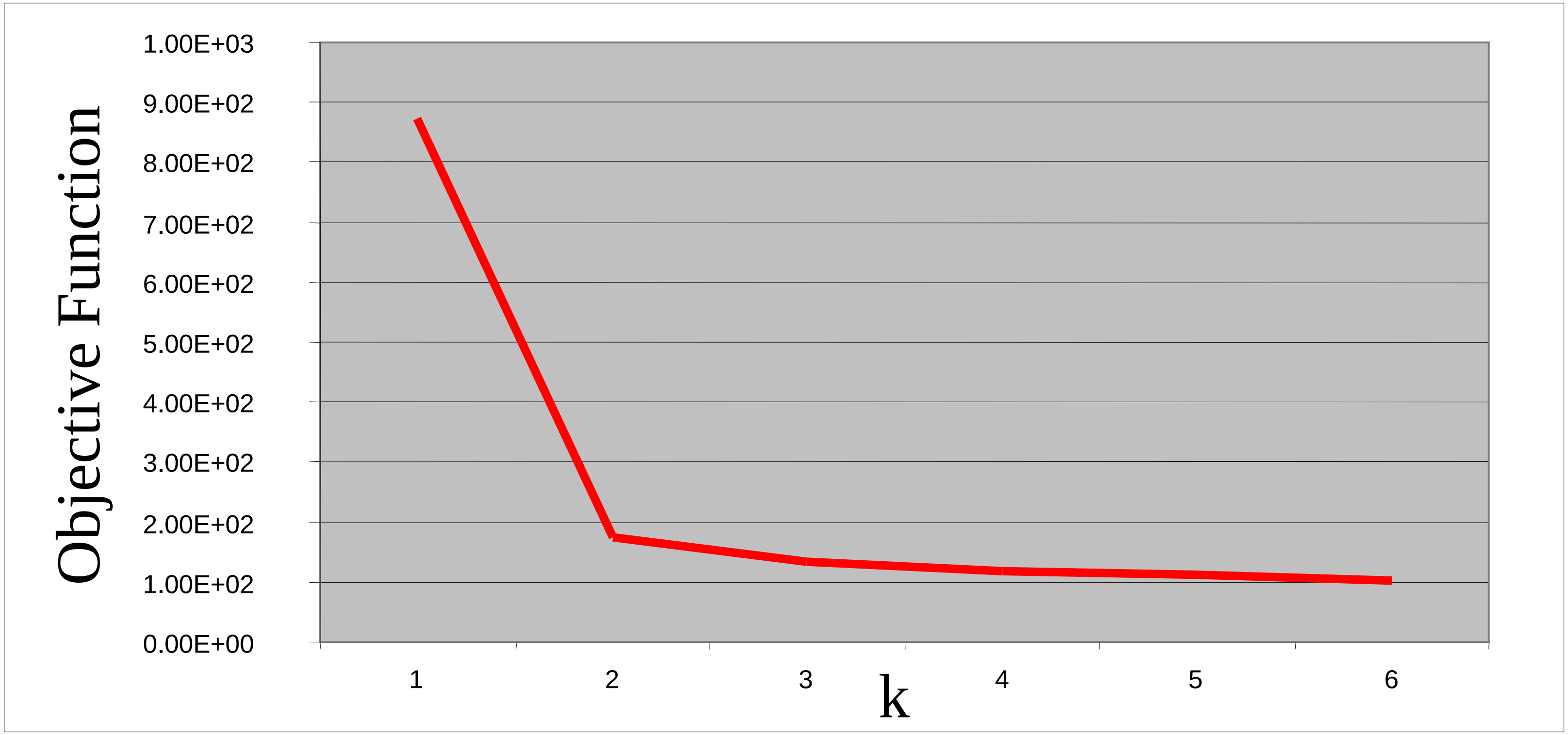


When k = 3, the objective function is 133.6



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

DBSCAN

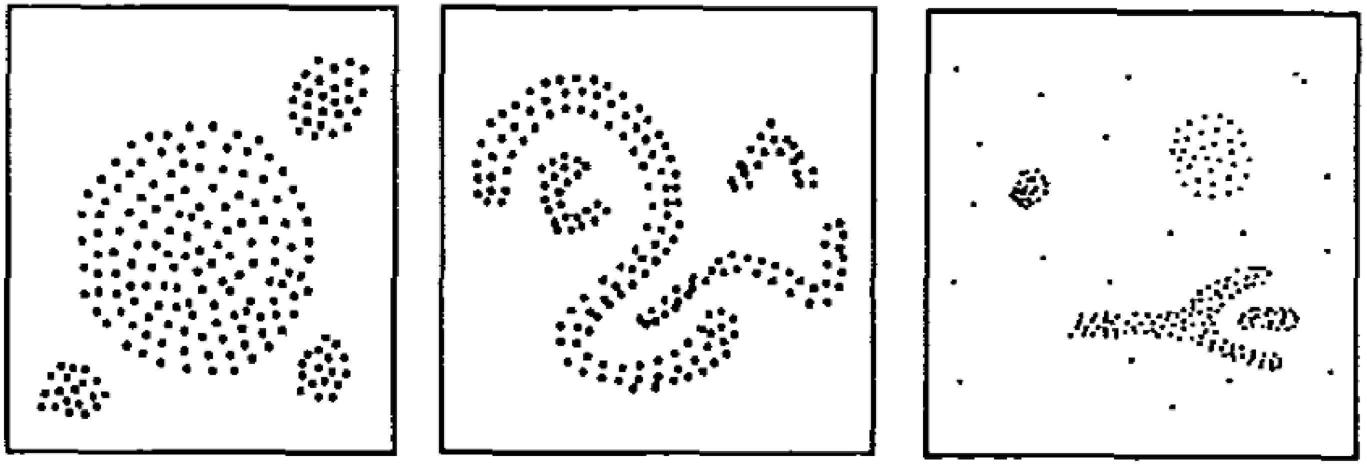
(Density-Based Spatial Clustering and Application with Noise)

Source: www.sthda.com

DBSCAN is a density-based clusering algorithm, introduced in Ester et al. 1996, which can be used to identify clusters of any shape in a data set containing noise and outliers.

The basic idea behind the density-based clustering approach is derived from a human intuitive clustering method. For instance, by looking at the figure below, one can easily identify four clusters along with several points of noise, because of the differences in the density of points.

The key idea is that for each point of a cluster, the neighborhood of a given radius has to contain at least a minimum number of points.



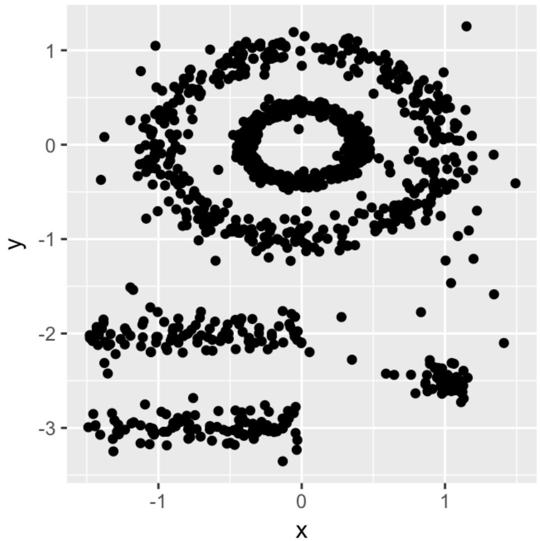
database 1 database 2 database 3

Why DBSCAN

Partitioning methods (K-means, PAM clustering) and hierarchical clustering are suitable for finding spherical-shaped clusters or convex clusters. In other words, they work well only for compact and well separated clusters. Moreover, they are also severely affected by the presence of noise and outliers in the data.

Unfortunately, real life data can contain: i) clusters of arbitrary shape such as those shown in the figure below (oval, linear and “S” shape clusters); ii) many outliers and noise.

The figure below shows a data set containing nonconvex clusters and outliers/noises.

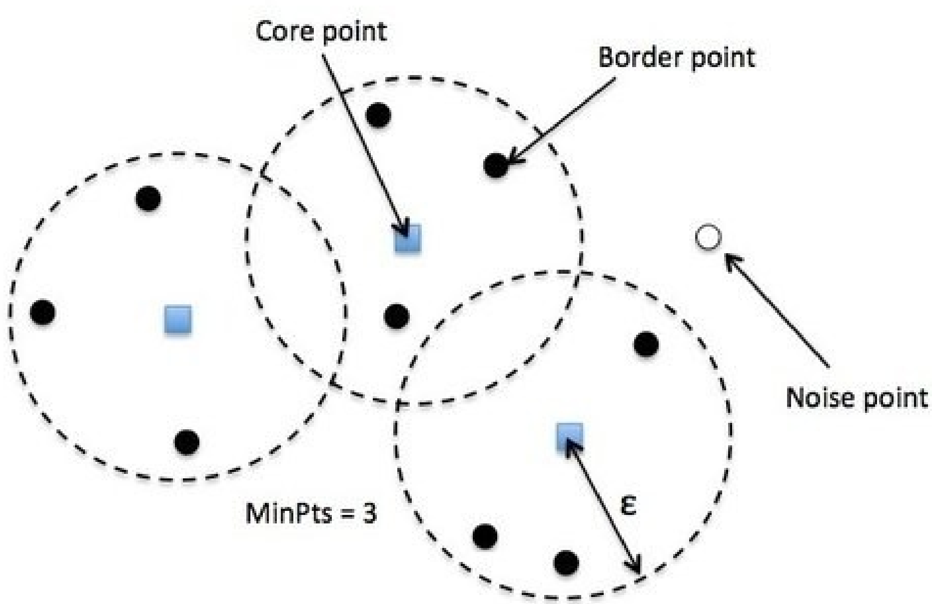


DBSCAN Algorithm

The goal is to identify dense regions, which can be measured by the number of objects close to a given point.

Two important parameters are required for DBSCAN: epsilon **("eps")** and minimum points **("MinPts")**. The parameter eps defines the radius of neighborhood around a point x. It's called called the eps-neighborhood of x. The parameter MinPts is the minimum number of neighbors within "eps" radius.

Any point x in the data set, with a neighbor count greater than or equal to MinPts, is marked as a **core point**. We say that x is **border point**, if the number of its neighbors is less than MinPts, but it belongs to the eps-neighborhood of some core point z. Finally, if a point is neither a core nor a border point, then it is called a **noise point** or an outlier.



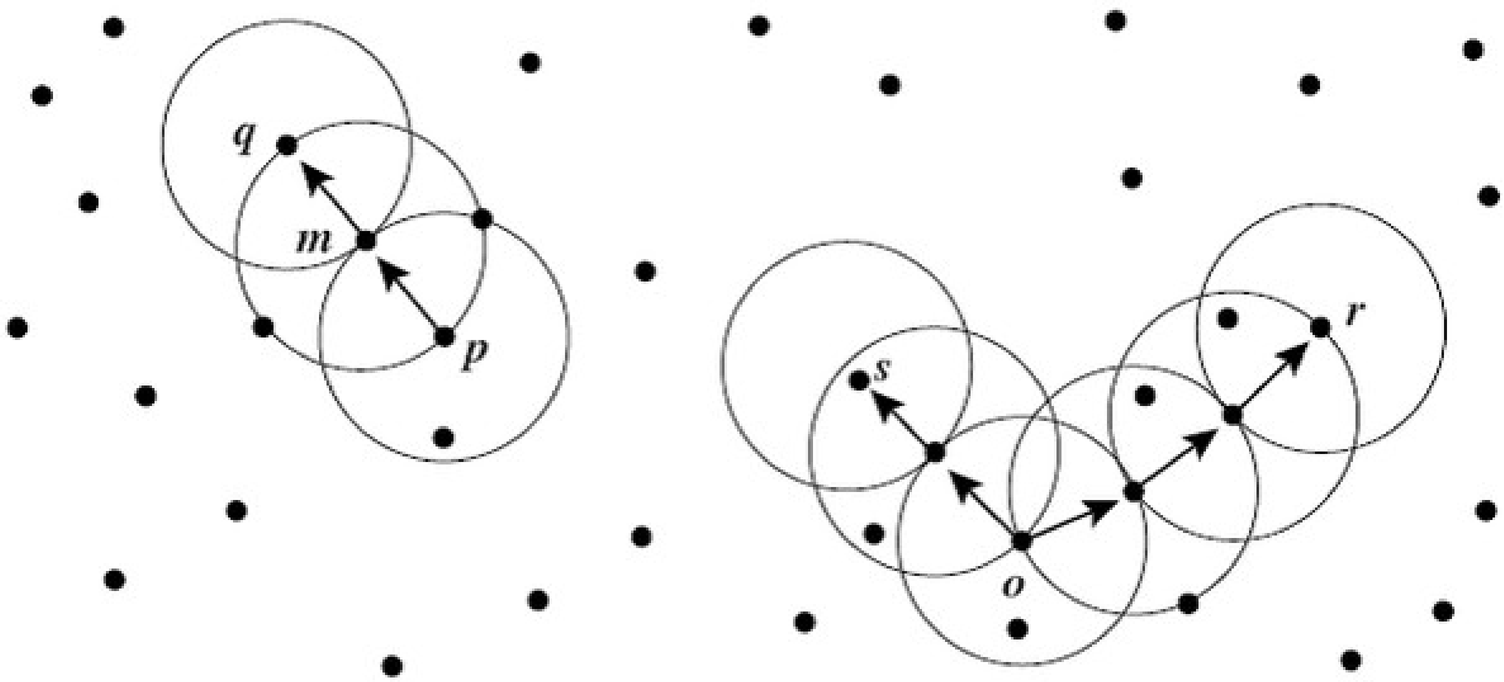
DBSCAN Algorithm

We start by defining 3 terms:

Direct density reachable: A point "A" is directly density reachable from another point "B" if: i) "A" is in the eps-neighborhood of "B" and ii) "B" is a core point. Example: **'p' and 'm'**; **'q' and 'm'**

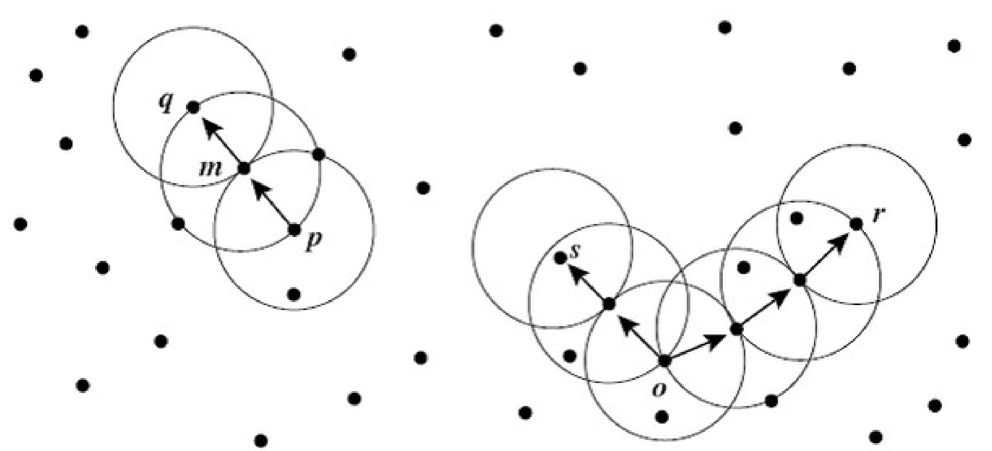
Density reachable: A point "A" is density reachable from "B" if there are a set of core points leading from "B" to "A. Example: **'s' and 'r'**.

Density connected: Two points "A" and "B" are density connected if there are a core point "C", such that both "A" and "B" are density reachable from "C". Example: **'p' and 'q'**;



DBSCAN Algorithm

1. Arbitrarily select a point P.
2. Retrieve all points directly density-reachable from P with respect to **ε**.
3. If P is a core point, a cluster is formed. Find recursively all its density connected points and assign them to the same cluster as P.
4. If P is not a core point, DBSCAN iterates through the remaining unvisited points in the dataset.



BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies)

??

DBSCAN

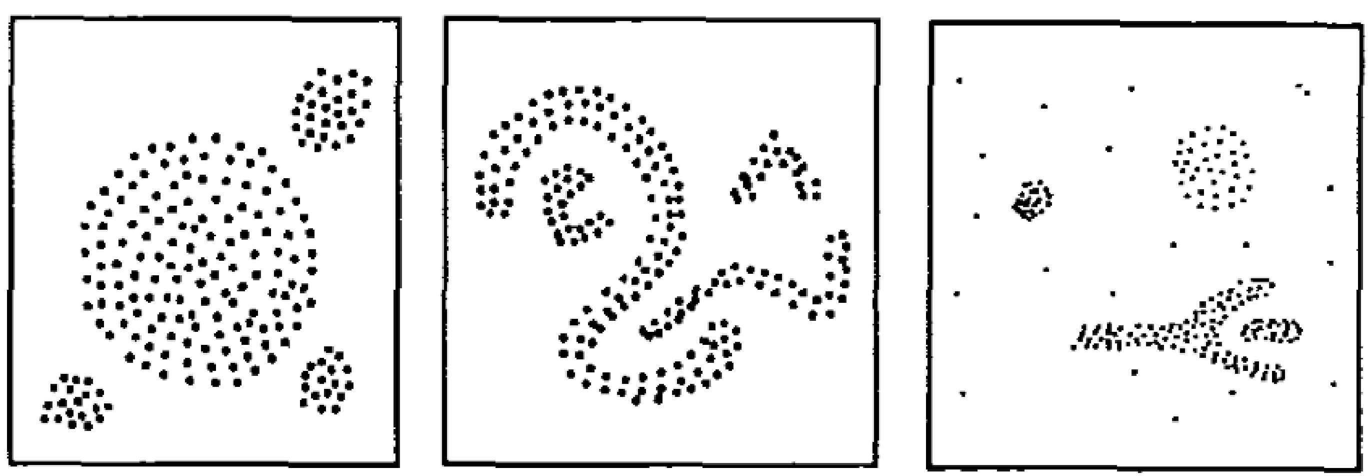
(Density-Based Spatial Clustering and Application with Noise)

Source: www.sthda.com

DBSCAN is a density-based clusering algorithm, introduced in Ester et al. 1996, which can be used to identify clusters of any shape in a data set containing noise and outliers.

The basic idea behind the density-based clustering approach is derived from a human intuitive clustering method. For instance, by looking at the figure below, one can easily identify four clusters along with several points of noise, because of the differences in the density of points.

The key idea is that for each point of a cluster, the neighborhood of a given radius has to contain at least a minimum number of points.



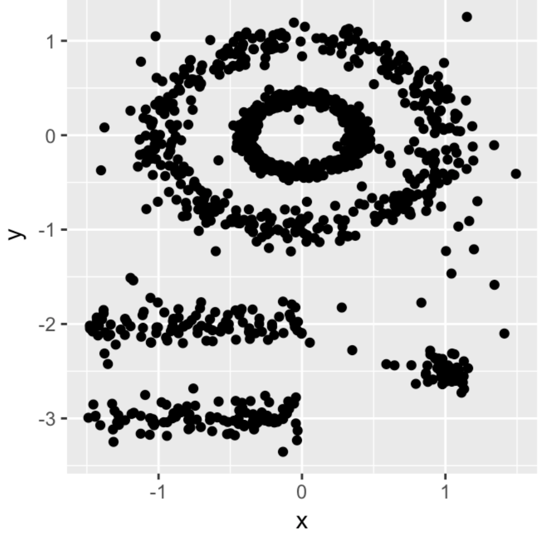
database 1 database 2 database 3

Why DBSCAN

Partitioning methods (K-means, PAM clustering) and hierarchical clustering are suitable for finding spherical-shaped clusters or convex clusters. In other words, they work well only for compact and well separated clusters. Moreover, they are also severely affected by the presence of noise and outliers in the data.

Unfortunately, real life data can contain: i) clusters of arbitrary shape such as those shown in the figure below (oval, linear and “S” shape clusters); ii) many outliers and noise.

The figure below shows a data set containing nonconvex clusters and outliers/noises.

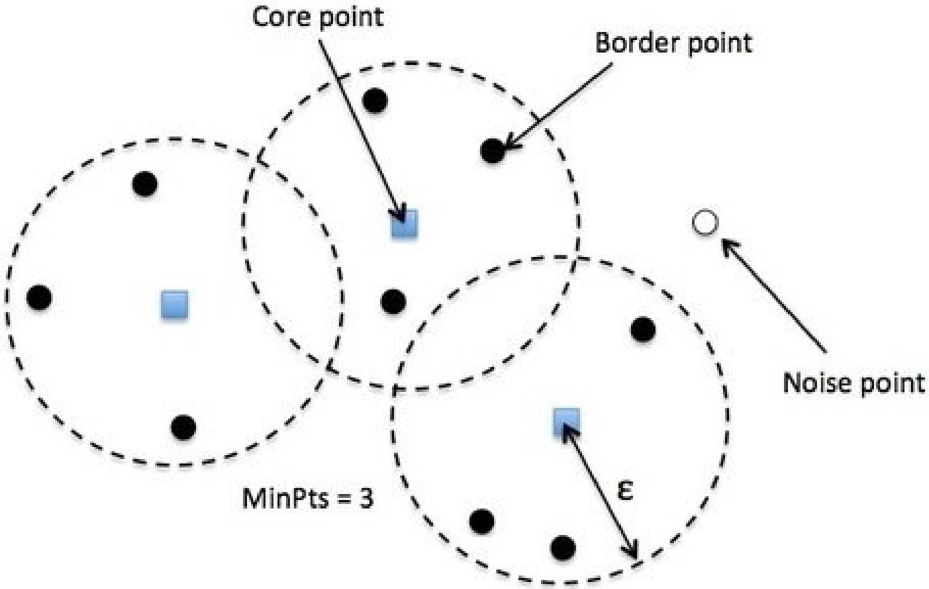


DBSCAN Algorithm

The goal is to identify dense regions, which can be measured by the number of objects close to a given point.

Two important parameters are required for DBSCAN: epsilon **("eps")** and minimum points **("MinPts")**. The parameter eps defines the radius of neighborhood around a point x. It's called called the eps-neighborhood of x. The parameter MinPts is the minimum number of neighbors within "eps" radius.

Any point x in the data set, with a neighbor count greater than or equal to MinPts, is marked as a **core point**. We say that x is **border point**, if the number of its neighbors is less than MinPts, but it belongs to the eps-neighborhood of some core point z. Finally, if a point is neither a core nor a border point, then it is called a **noise point** or an outlier.



Further reading >>
www.geeksforgeeks.org/ml-dbscan-reachability-and-connectivity/

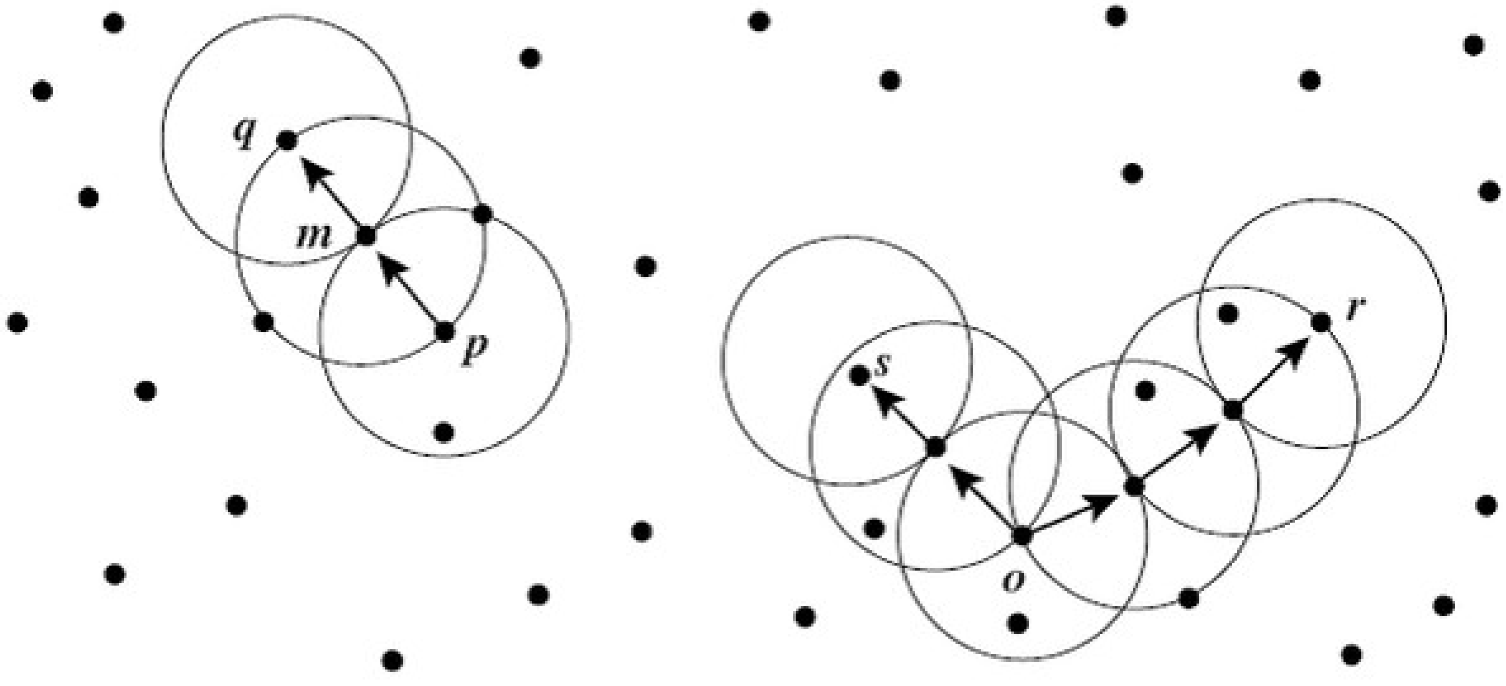
DBSCAN Algorithm

We start by defining 3 terms:

Direct density reachable: A point “A” is directly density reachable from another point “B” if: i) “A” is in the eps-neighborhood of “B” and ii) “B” is a core point. Example: **‘p’ and ‘m’**; **‘q’ and ‘m’**

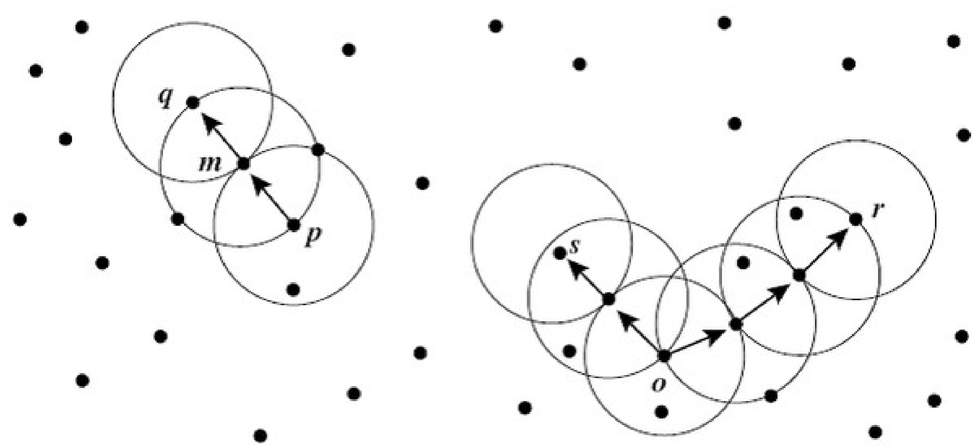
Density reachable: A point “A” is density reachable from “B” if there are a set of core points leading from “B” to “A. Example: **‘p’ and ‘q’**.

Density connected: Two points “A” and “B” are density connected if there are a core point “C”, such that both “A” and “B” are density reachable from “C”. Example: **‘s’ and ‘r’**;

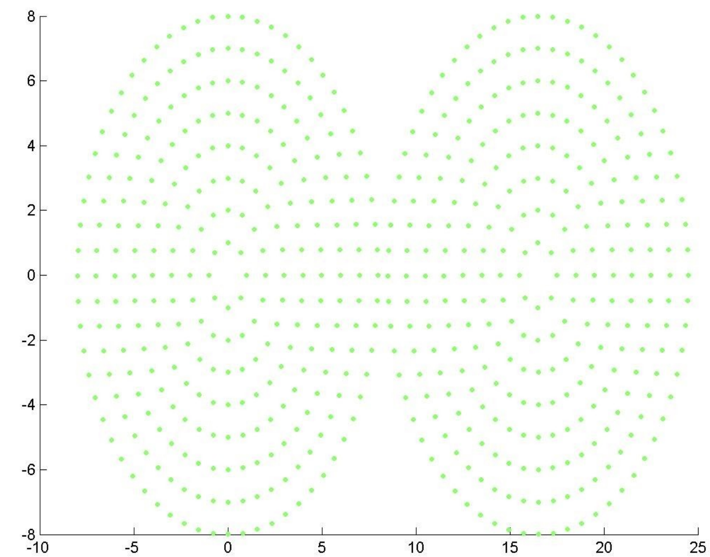


DBSCAN Algorithm

1. Arbitrarily select a point P.
2. Retrieve all points directly density-reachable from P with respect to **ε**.
3. If P is a core point, a cluster is formed. Find recursively all its density connected points and assign them to the same cluster as P.
4. If P is not a core point, DBSCAN iterates through the remaining unvisited points in the dataset.



DBSCAN Algorithm



Advantages

1) Does not require a-priori specification of number of clusters.

2) Able to identify noise data while clustering.

3) DBSCAN algorithm is able to find arbitrarily size and arbitrarily shaped clusters.

Disadvantages

1) DBSCAN algorithm fails in case of varying density clusters.

2) Fails in case of neck type of dataset.

3) Does not work well in case of high dimensional data.