ISE 5103 Intelligent Data Analytics

Clustering

Charles Nicholson, Ph.D.
cnicholson@ou.edu

University of Oklahoma
Gallogly College of Engineering
School of Industrial and Systems Engineering

Outline

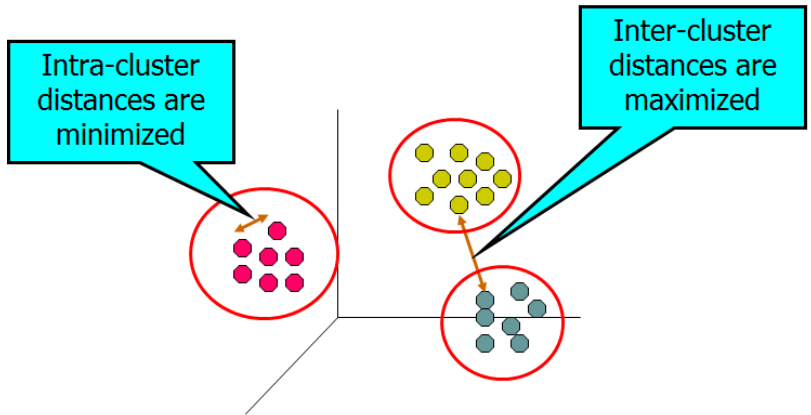
Supervised learning vs. unsupervised learning

- Supervised learning: discover patterns in the data that relate data attributes with a target (class) attribute.
  - These patterns are then utilized to predict the values of the target attribute in future data instances.
  - e.g. LDA, kNN, linear regression (OLS, PLS, ridge, lasso, elastic net), logistic regression (including penalized variants), decision trees (and forests, and bagging and boosting), etc.
- Unsupervised learning: There is no target variable.
  - We want to explore the data to find some intrinsic structures in them.
  - e.g. PCA, MDS, Association Mining, Clustering

What is clustering?

sometimes called *segmentation*

- Organizing data into classes such that there is
  - high intra-class similarity
  - low inter-class similarity



What is clustering?

- 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 are clustering applications?

- Biology: taxonomy of living things: kingdom, phylum, class, order, family, genus and species
- Finance: group stocks with similar price fluctuations

	Discovered Clusters	*Industry Group*
1	Applied-Matl-DOWN,Bay-Network-Down,3-COM-DOWN, Cabletron-Sys-DOWN,CISCO-DOWN,HP-DOWN, DSC-Comm-DOWN,INTEL-DOWN,LSI-Logic-DOWN, Micron-Tech-DOWN,Texas-Inst-Down,Tellabs-Inc-Down, Natl-Semiconduct-DOWN,Oracl-DOWN,SGI-DOWN, Sun-DOWN	Technology1-DOWN
2	Apple-Comp-DOWN,Autodesk-DOWN,DEC-DOWN, ADV-Micro-Device-DOWN,Andrew-Corp-DOWN, Computer-Assoc-DOWN,Circuit-City-DOWN, Compaq-DOWN, EMC-Corp-DOWN, Gen-Inst-DOWN, Motorola-DOWN,Microsoft-DOWN,Scientific-Atl-DOWN	Technology2-DOWN
3	Fannie-Mae-DOWN,Fed-Home-Loan-DOWN, MBNA-Corp-DOWN,Morgan-Stanley-DOWN	Financial-DOWN
4	Baker-Hughes-UP,Dresser-Inds-UP,Halliburton-HLD-UP, Louisiana-Land-UP,Phillips-Petro-UP,Unocal-UP, Schlumberger-UP	Oil-UP

- Information retrieval: document clustering
- Land use: Identification of areas of similar land use in an earth observation database
- Marketing: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs. *Birds of a feather, flock together*
- City-planning: Identifying groups of houses according to their house type, value, and

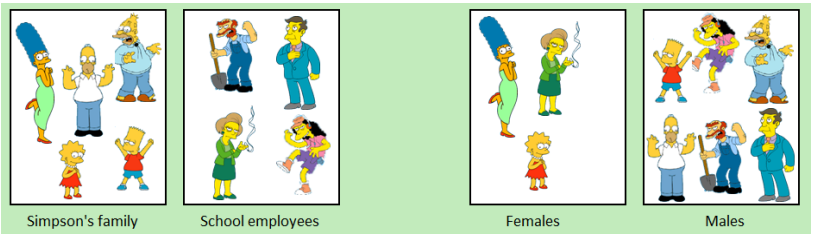
What are clustering applications?

In fact, clustering is one of the most utilized data mining techniques.

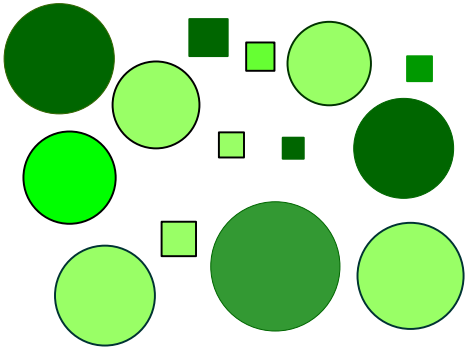
- It has a long history, and used in almost every field, e.g., medicine, psychology, botany, genetics, sociology, biology, archeology, marketing, insurance, libraries, etc.
- Recently, due to the rapid increase of online documents, text clustering becomes important.

What is a natural grouping among these objects?

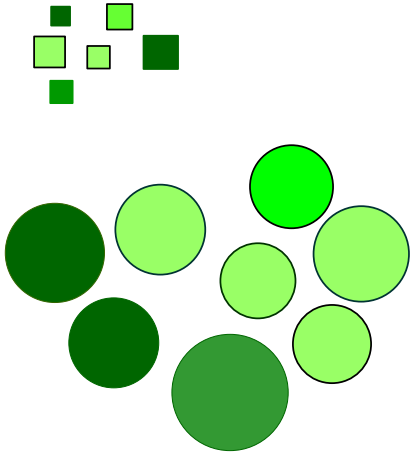
Clustering is subjective



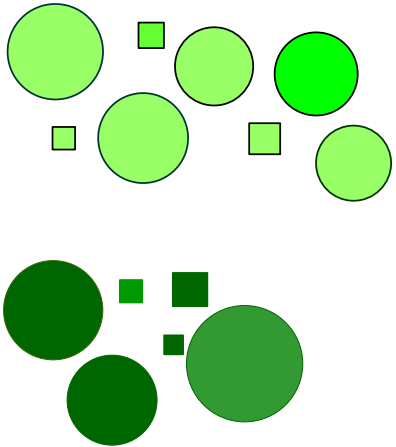
How to cluster these objects?



How to cluster these objects?



How to cluster these objects?



What is similarity?

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

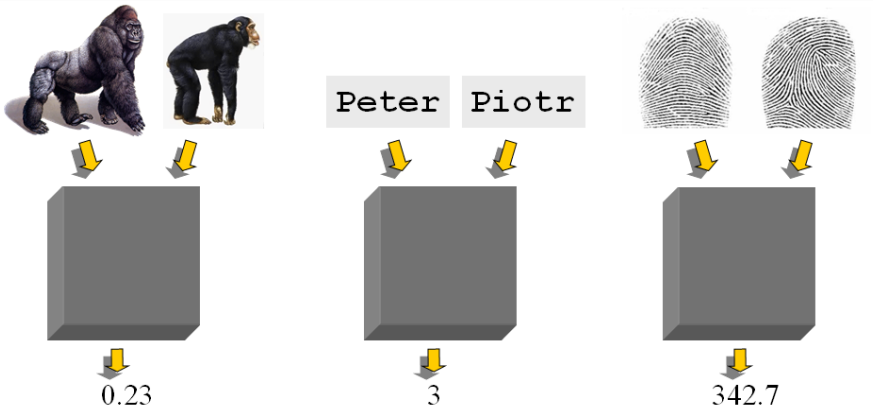
Similarity is hard to define, but... *we know it when we see it*

The real meaning of similarity is a philosophical question.

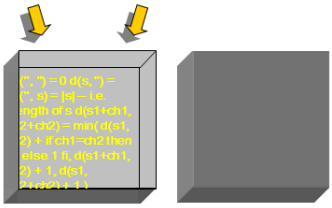
We will take a more pragmatic approach.

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



3

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?

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

Intuitions behind desirable distance measure properties

$D(A, B) \geq 0$ — *Non-negativity*

Identical is as good as it gets!

$D(A, B) = D(B, A)$ — *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."

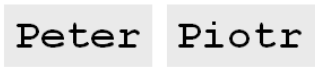
$D(A, B) = 0 \iff A = B$ — *Positivity (Separation)*

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

$D(A, B) \leq D(A, B) + D(B, C)$ — *Triangular Inequality*

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

How do we measure similarity?



0.23

3

342.7

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
```

D(Marge,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*.

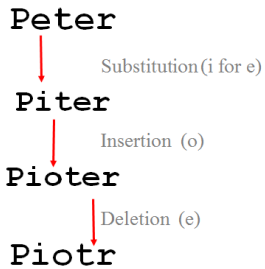
Note that for now we have ignored the issue of how we can find this cheapest transformation

How similar are the names "Peter" and "Piotr"?

Assume the following cost function

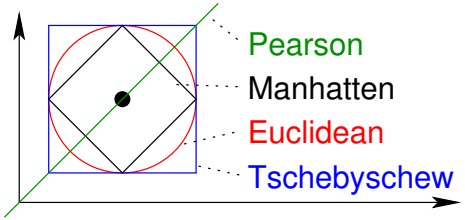
Substitution	1 Unit
Insertion	1 Unit
Deletion	1 Unit

D(Peter,Piotr) is 3



notion of (dis-)similarity: numerical attributes

Various choices for dissimilarities between two numerical vectors:



Minkowksi	L_p	$D_p(x,y) = \sqrt[p]{\sum_{i=1}^{n} \lvert x_i - y_i \rvert^p}$
Euclidean	L_2	$D_E(x,y) = \sqrt{(x_1 - y_1)^2 + \ldots + (x_n - y_n)^2}$
Manhattan	L_1	$D_M(x,y) = \lvert x_1 - y_1 \rvert + \ldots + \lvert x_n - y_n \rvert$
Tschebyschew	L_∞	$d_\infty(x,y) = \max\{\lvert x_1 - y_1 \rvert, \ldots, \lvert x_n - y_n \rvert\}$
Pearson		related to Euclidean of z-score transformed $\mathbf{x}$, $\mathbf{y}$

notion of (dis-)similarity: binary attributes

The two values (e.g. 0 and 1) of a binary attribute can be interpreted as some property being absent (0) or present (1).

In this sense, a vector of binary attribute can be interpreted as a set of properties that the corresponding object has.

Example

- The binary vector $(0, 1, 1, 0, 1)$ corresponds to the set of properties $\{a_2, a_3, a_5\}$.
- The binary vector $(0, 0, 0, 0, 0)$ corresponds to the empty set.
- The binary vector $(1, 1, 1, 1, 1)$ corresponds to the set $\{a_1, a_2, a_3, a_4, a_5\}$.

notion of (dis-)similarity: binary attributes

Each data object is represent by the corresponding set of properties that are present.

simple match	$d_M = 1 - \frac{b+n}{b+n+x}$
Russel & Rao	$d_R = 1 - \frac{b}{b+n+x}$
Jaccard	$d_J = 1 - \frac{b}{b+x}$
Dice	$d_D = 1 - \frac{2b}{2b+x}$

	no. of predicates that...
$b =$	...hold in both records
$n =$	...do not hold in both records
$x =$	...hold in only one of both records

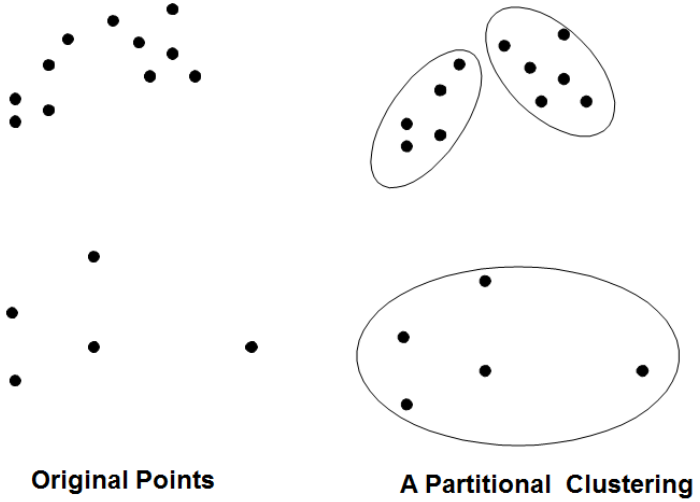
x	**y**	set X	set Y	b	n	x	d_M	d_R	d_J	d_D
101000	111000	$\{a_1, a_3\}$	$\{a_1, a_2, a_3\}$	2	3	1	$0.1\bar{6}$	$0.6\bar{6}$	$0.3\bar{3}$	0.20

Outline

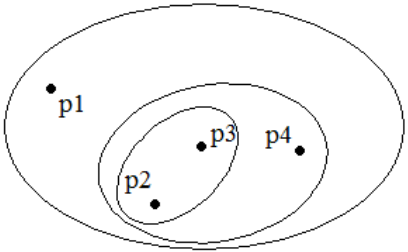
types of clustering approaches

- Clustering by Partitioning
  - data are grouped in an exclusive way, so that if a certain datum belongs to a definite cluster then it could not be included in another cluster
- Hierarchical Clustering
  - a set of nested clusters organized as a hierarchical tree
- Density Based Clustering
  - probabilistic approach guided by connectivity and density functions

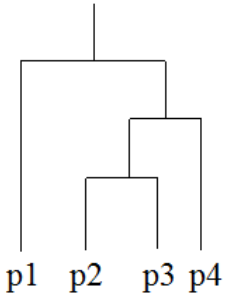
partitioning



hierarchical



Traditional Hierarchical Clustering



Traditional Dendrogram

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.

Goal: Given a K, find a partition of K clusters that optimizes the chosen partitioning criterion

→ Global optimal: exhaustively enumerate all partitions

→ k-means (MacQueen, 1967): Each cluster is represented by the center of the cluster

→ k-medoids or PAM (Partition around medoids) (Kaufman & Rousseeuw, 1987): Each cluster is represented by one of the objects in the cluster

k-means Clustering

- Let the set of data points (or instances) D be $\{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_n}\}$, where $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ is a vector in a real-valued space $X \subseteq \mathbb{R}^p$, and p is the number of attributes (dimensions) in the data.
- The k-means algorithm partitions the given data into k clusters.
- Each cluster has a cluster **center**, called **centroid**.
- k is specified by the user

k-means algorithm

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 *stopping criterion* met, exit. Otherwise goto 3.

stopping criterion

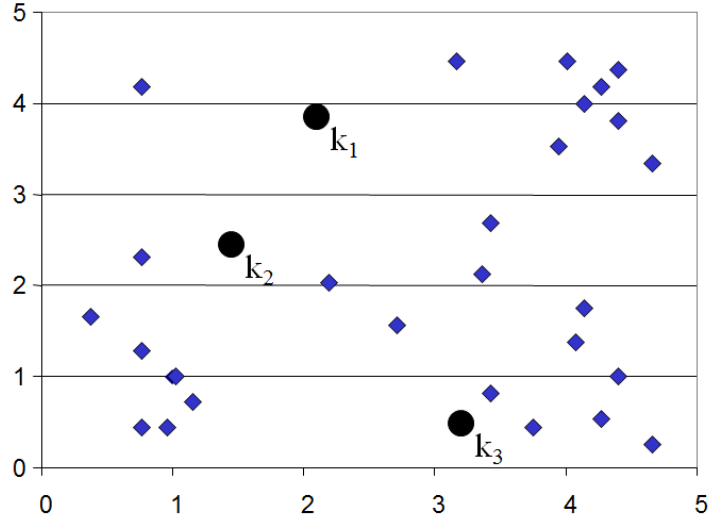
- no (or minimum) re-assignments of data points to different clusters, or
- no (or minimum) change of centroids, or
- minimum decrease in the sum of squared error (SSE),

$$SSE = \sum_{j=i}^{k} \sum_{x \in C_j} D(\mathbf{x}, \mathbf{m}_j)^2$$

where C_j is the j^{th} cluster, $\mathbf{m}_j$ is the centroid of cluster C_j (the mean vector of all the data points in C_j), and $D(\mathbf{x}, \mathbf{m}_j)$ is the distance between data point $\mathbf{x}$ and centroid $\mathbf{m}_j$.

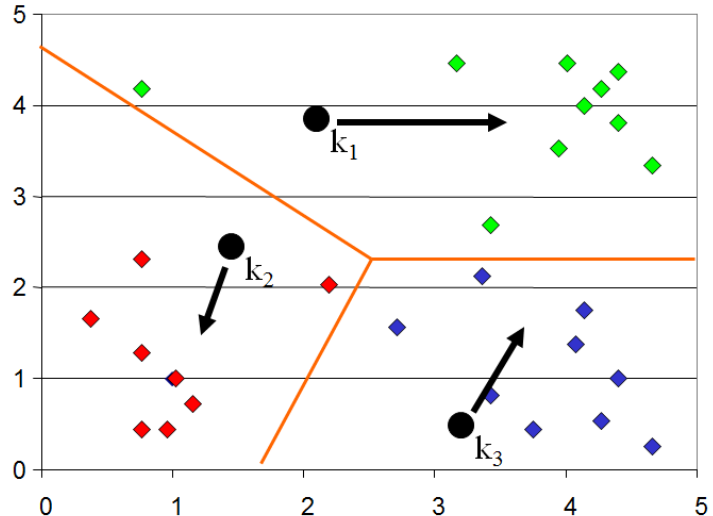
K-means Clustering: Step 1

Algorithm: k-means, Distance Metric: Euclidean Distance



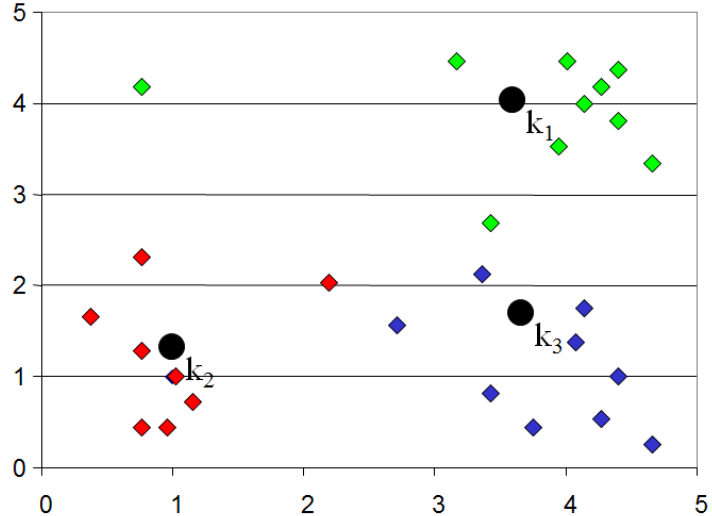
K-means Clustering: Step 2

Algorithm: k-means, Distance Metric: Euclidean Distance



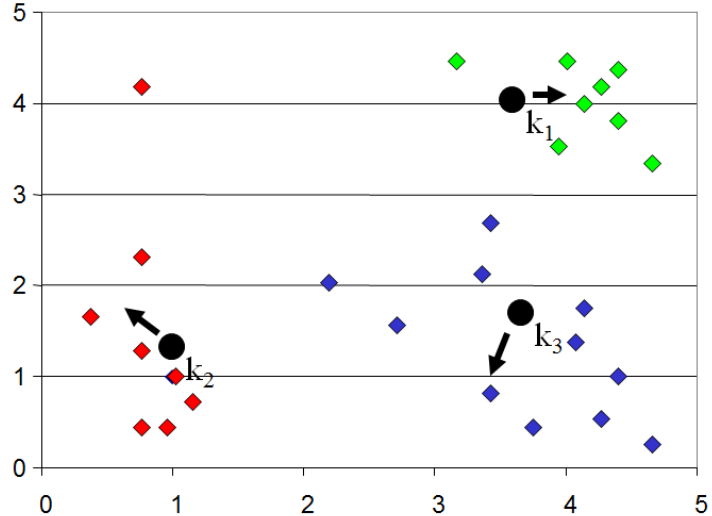
K-means Clustering: Step 3

Algorithm: k-means, Distance Metric: Euclidean Distance



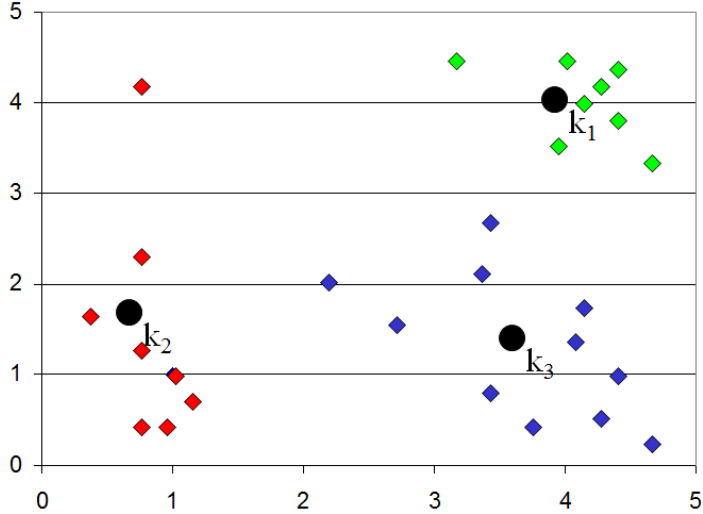
K-means Clustering: Step 4

Algorithm: k-means, Distance Metric: Euclidean Distance



K-means Clustering: Step 5

Algorithm: k-means, Distance Metric: Euclidean Distance

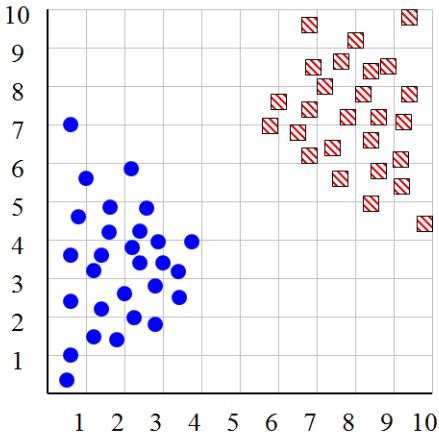


k-means clustering details

- Initial centroids are often chosen randomly.
  - Clusters produced vary from one run to another.
- The centroid is the *p*-dimensional vector of means for the points in the cluster.
- Distance is measured by Gowers, Euclidean, cosine similarity, correlation, or other approporiate distance measure
- K-means will converge for common similarity measures mentioned above.
- Most of the convergence happens in the first few iterations.
  - Often the stopping condition is changed from complete *convergence* to near convergence, i.e. *relatively few points change clusters*

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.

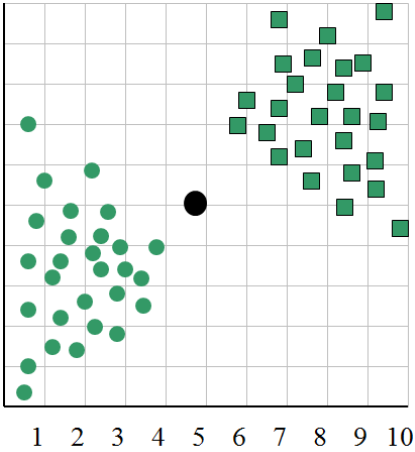


For our example, we will use the dataset on the left.

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.

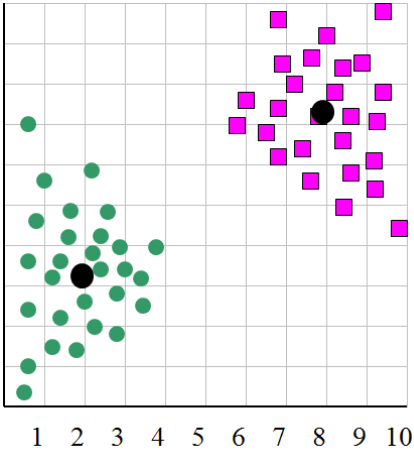
determining the right number of clusters

When k = 1, the objective function is 873.0



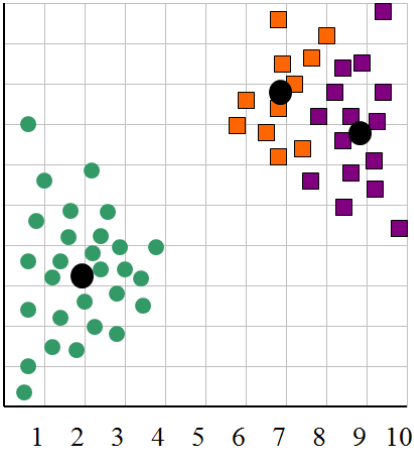
determining the right number of clusters

When k = 2, the objective function is 173.1



determining the right number of clusters

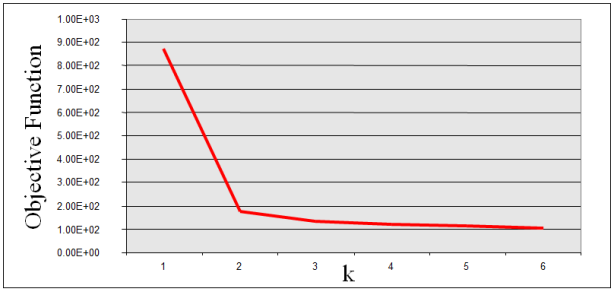
When k = 3, the objective function is 133.6



determining the right number of clusters

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

The abrupt change at k = 2, is suggestive of two clusters. This technique is known as “elbow finding”.



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

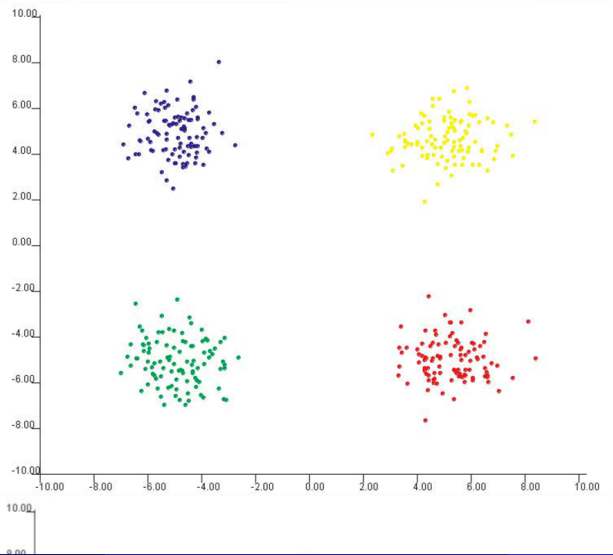
strengths of *k*-means clustering

- Simple: easy to understand and to implement
- Efficient: k-means is considered a linear algorithm.
- K-means is by far the most popular clustering algorithm.

weaknesses of *k*-means clustering

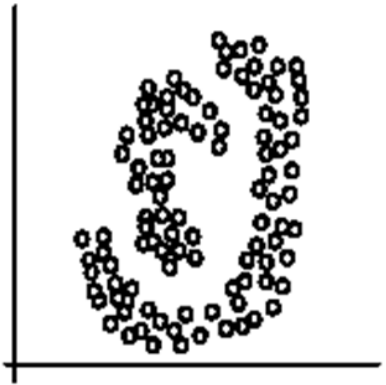
- The user needs to specify k.
- Sensitivity to scale
- Not suitable for clusters which are not hyper-spherical
- Sensitivity to initial seeds
- The algorithm is sensitive to outliers
- The algorithm is only applicable if the mean is defined.

sensitivity to scale

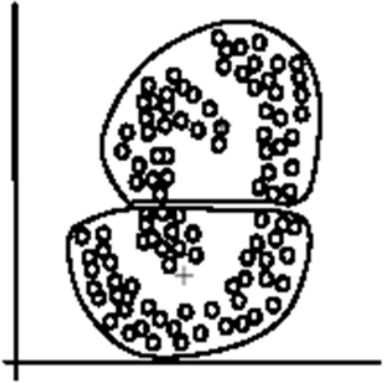


weaknesses of *k*-means clustering

- The user needs to specify k.
- Sensitivity to scale
- Not suitable for clusters which are not hyper-spherical
- Sensitivity to initial seeds
- The algorithm is sensitive to outliers
- The algorithm is only applicable if the mean is defined.



(A): Two natural clusters



(B): k-means clusters

weaknesses of *k*-means clustering

- The user needs to specify k.
- Sensitivity to scale
- Not suitable for clusters which are not hyper-spherical
- Sensitivity to initial seeds
- The algorithm is sensitive to outliers
- The algorithm is only applicable if the mean is defined.

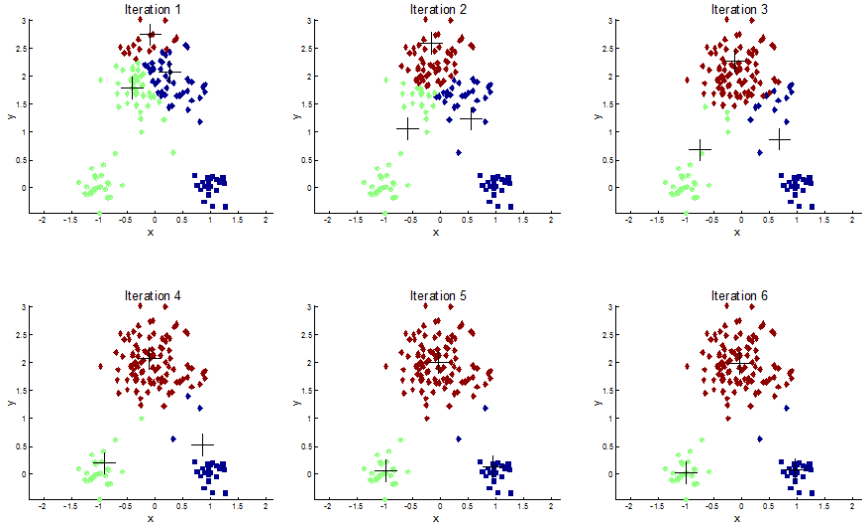
If there are K “real” clusters then the chance of selecting one centroid from each cluster is small:

- Chance is relatively small when K is large
- If clusters are the same size, n, then

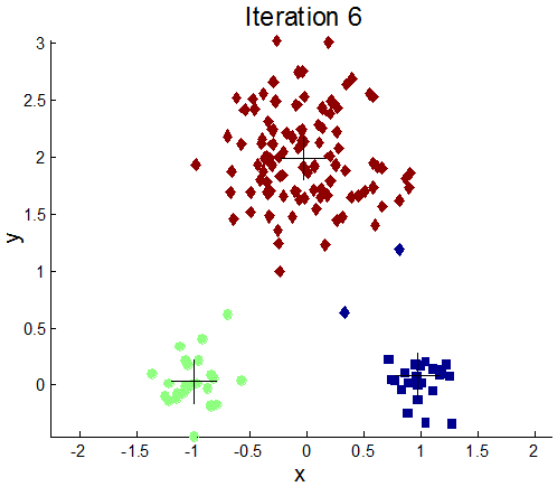
$$P = \frac{\text{num ways to select 1 centroid from each cluster}}{\text{num ways to select K centroids}} = \frac{K!}{K^K}$$

- For example, if K = 10, then probability = 10!/1010 = 0.00036
- Sometimes the initial centroids will readjust themselves in “right” way, and sometimes they don’t

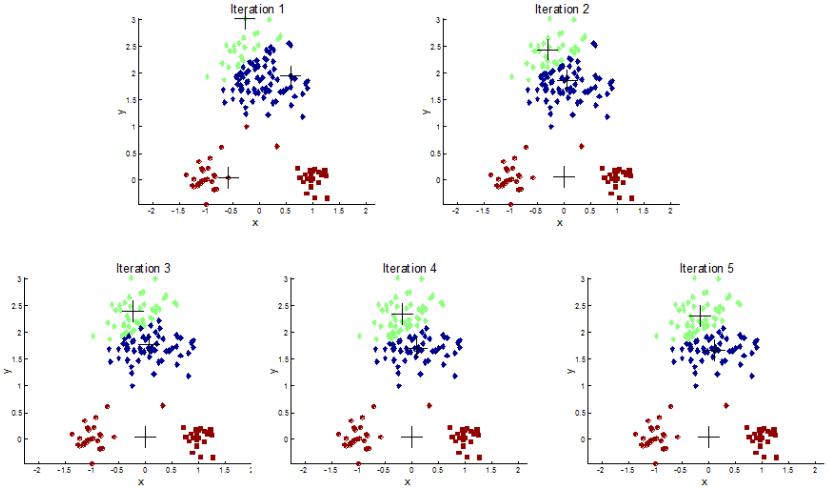
importance of choosing initial centroids



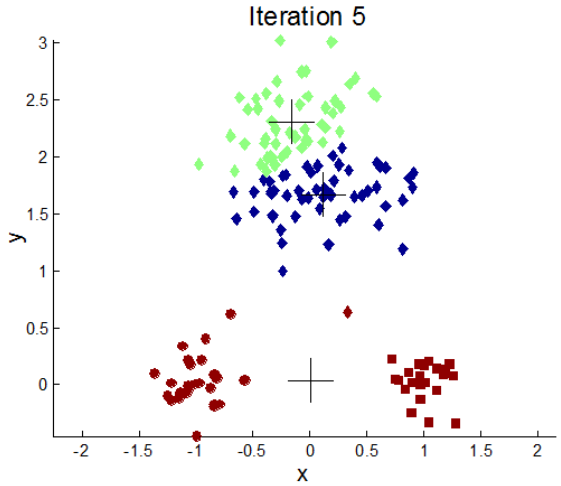
importance of choosing initial centroids



importance of choosing initial centroids

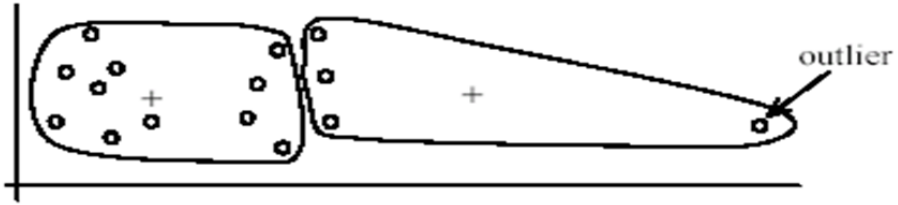


importance of choosing initial centroids

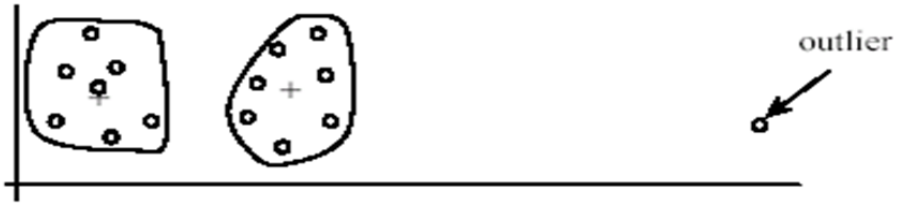


weaknesses of *k*-means clustering

- The user needs to specify k.
- Sensitivity to scale
- Not suitable for clusters which are not hyper-spherical
- Sensitivity to initial seeds
- The algorithm is sensitive to outliers
- The algorithm is only applicable if the mean is defined.



(A): Undesirable clusters



(B): Ideal clusters

weaknesses of *k*-means clustering

- The user needs to specify k.
- Sensitivity to scale
- Not suitable for clusters which are not hyper-spherical
- Sensitivity to initial seeds
- The algorithm is sensitive to outliers
- The algorithm is only applicable if the mean is defined.

solutions to k-means problems

- **Multiple runs** $\rightarrow$ helps with **initial seed** problem
- **Pre-processing**
  - Normalize the data $\rightarrow$ to address **scale** problem
  - Eliminate outliers $\rightarrow$ if you already know **outliers** exist, then ...
  - Random sampling $\rightarrow$ chance of selecting an **outlier** is very small
  - Hierarchical clustering $\rightarrow$ to help **determine** k
  - More help with determining k

    Milligan, G., M. Cooper. 1985. An examination of procedures for determining the number of clusters in a data set. Psychometrika 50:159-179
    Hastie, T., R. Tibshirani, R., G. Walther. 2001. Estimating the number of clusters in a dataset via the Gap statistic. Journal of Royal Statistical Society: Series B. 63:411-423
- **During processing**
  - **Outlier deletion**: remove points in the clustering process that are much further away from the centroids than other data points
- **Post-processing**
  - Eliminate small clusters that may represent **outliers**

variations of the *k*-means method

A few variants of the k-means which differ in:

- Selection of the initial k seeds
- Dissimilarity measures
- Strategies to calculate cluster means

Handling categorical data: **k-modes**

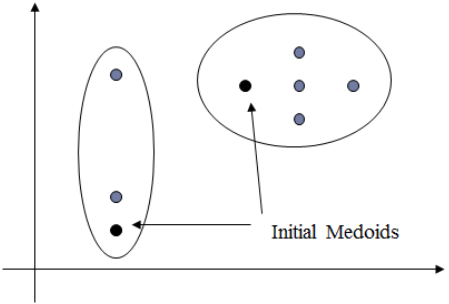
- Replacing means of clusters with modes
- Using new dissimilarity measures to deal with categorical objects
- Using a frequency based method to update modes of clusters

k-medoids

k-means algorithm is sensitive to outliers

Since an object with an extremely large value may substantially distort the distribution of the data.

Medoid: the most centrally located point in a cluster, as a representative point of the cluster.

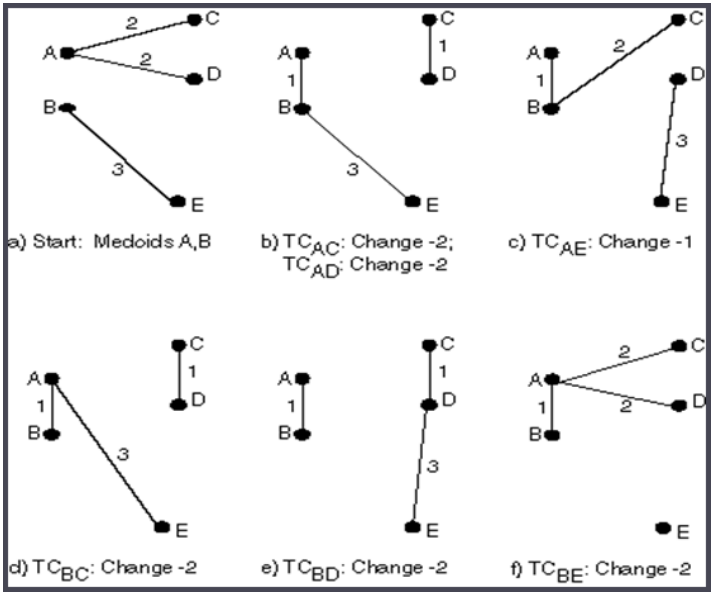


partition around medoids (PAM)

Algorithm PAM:

1. Given k from user
2. Randomly pick k instances as initial medoids
3. Assign each data point to the nearest medoid x
4. Calculate the objective function the sum of dissimilarities of all points to their nearest medoids.
5. Randomly select a point y (where y is not a medoid)
6. Swap x by y if the swap reduces the objective function
7. Repeat (3-6) until no change

PAM example


a) Start: Medoids A,B
b) TC_{AC}: Change -2; TC_{AD}: Change -2
c) TC_{AE}: Change -1
d) TC_{BC}: Change -2
e) TC_{BD}: Change -2
f) TC_{BE}: Change -2

hierarchical clustering

- Hierarchical clustering builds clusters step by step.
- Usually a bottom up strategy is applied by first considering each data object as a separate cluster and then step by step joining clusters together that are close to each other. This approach is called agglomerative hierarchical clustering.
- In contrast to agglomerative hierarchical clustering, divisive hierarchical clustering starts with the whole data set as a single cluster and then divides clusters step by step into smaller clusters.
- In order to decide which data objects should belong to the same cluster, a (dis-)similarity measure is needed.
- Note: We do need to have access to features, all that is needed for hierarchical clustering is an $n \times n$-matrix $[d_{i,j}]$, where $d_{i,j}$ is the (dis-)similarity of data objects i and j. (n is the number of data objects.)

agglomerative hierarchical clustering: algorithm

Input: $n \times n$ dissimilarity matrix $[d_{i,j}]$.

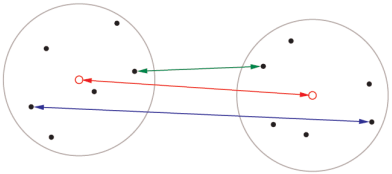
1. Start with n clusters, each data objects forms a single cluster.
2. Reduce the number of clusters by joining those two clusters that are most similar (least dissimilar).
3. Repeat step 3 until there is only one cluster left containing all data objects.

measuring dissimilarity between clusters

- The dissimilarity between two clusters containing only one data object each is simply the dissimilarity of the two data objects specified in the dissimilarity matrix $[d_{i,j}]$.
- But how do we compute the dissimilarity between clusters that contain more than one data object?

measuring dissimilarity between clusters

- **Centroid**
  Distance between the centroids (mean value vectors) of the two clusters[1]
- **Average Linkage**
  Average dissimilarity between all pairs of points of the two clusters.
- **Single Linkage**
  Dissimilarity between the two most similar data objects of the two clusters.
- **Complete Linkage**
  Dissimilarity between the two most dissimilar data objects of the two clusters.

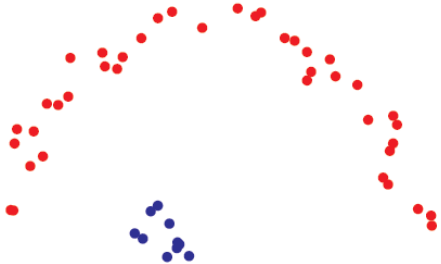


[1] Requires that we can compute the mean vector!

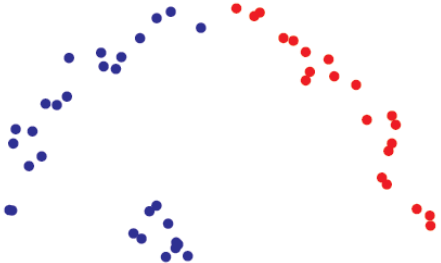
measuring dissimilarity between clusters

- **Single linkage** can “follow chains” in the data (may be desirable in certain applications).
- **Complete linkage** leads to very compact clusters.
- **Average linkage** also tends clearly towards compact clusters.

measuring dissimilarity between clusters



Single linkage



Complete linkage

measuring dissimilarity between clusters

Ward's method

- another strategy for merging clusters
- In contrast to single, complete or average linkage, it takes the number of data objects in each cluster into account.

measuring dissimilarity between clusters

The updated dissimilarity between the newly formed cluster $\{\mathcal{C} \cup \mathcal{C}'\}$ and the cluster $\mathcal{C}''$ is computed in the follwing way.

$$d'(\{\mathcal{C} \cup \mathcal{C}'\}, \mathcal{C}'') = \ldots$$

$$\begin{aligned}
\text{single linkage} &= \min\{d'(\mathcal{C}, \mathcal{C}''), d'(\mathcal{C}', \mathcal{C}'')\} \\
\text{complete linkage} &= \max\{d'(\mathcal{C}, \mathcal{C}''), d'(\mathcal{C}', \mathcal{C}'')\} \\
\text{average linkage} &= \frac{|\mathcal{C}| d'(\mathcal{C}, \mathcal{C}'') + |\mathcal{C}'| d'(\mathcal{C}', \mathcal{C}'')}{|\mathcal{C}| + |\mathcal{C}'|} \\
\text{Ward} &= \frac{(|\mathcal{C}| + |\mathcal{C}''|) d'(\mathcal{C}, \mathcal{C}'') + (|\mathcal{C}'| + |\mathcal{C}''|) d'(\mathcal{C}', \mathcal{C}'') - |\mathcal{C}''| d'(\mathcal{C}, \mathcal{C}')}{|\mathcal{C}| + |\mathcal{C}'| + |\mathcal{C}''|} \\
\text{centroid}^2 &= \frac{1}{|\mathcal{C} \cup \mathcal{C}'||\mathcal{C}''|} \sum_{\mathbf{x} \in \mathcal{C} \cup \mathcal{C}'} \sum_{\mathbf{y} \in \mathcal{C}''} d(\mathbf{x}, \mathbf{y})
\end{aligned}$$

[2] If metric, usually mean vector needs to be computed!

dendrograms

- The cluster merging process arranges the data points in a binary tree.
- Draw the data tuples at the bottom or on the left
  (equally spaced if they are multi-dimensional).
- Draw a connection between clusters that are merged, with the distance to the data points representing the distance between the clusters.

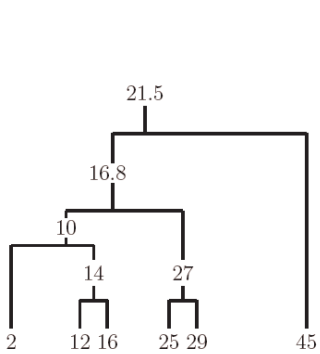
hierarchical clustering

Example

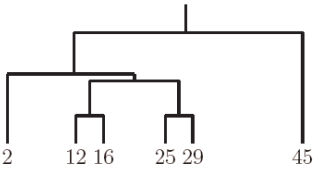
Clustering of the 1-dimensional data set $\{2, 12, 16, 25, 29, 45\}$.

All three approaches to measure the distance between clusters lead to different dendrograms.

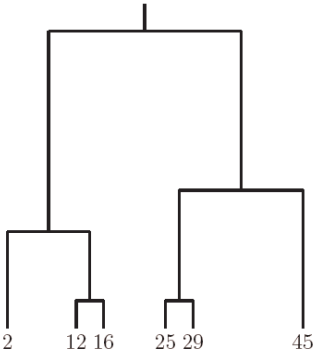
hierarchical clustering



Centroid

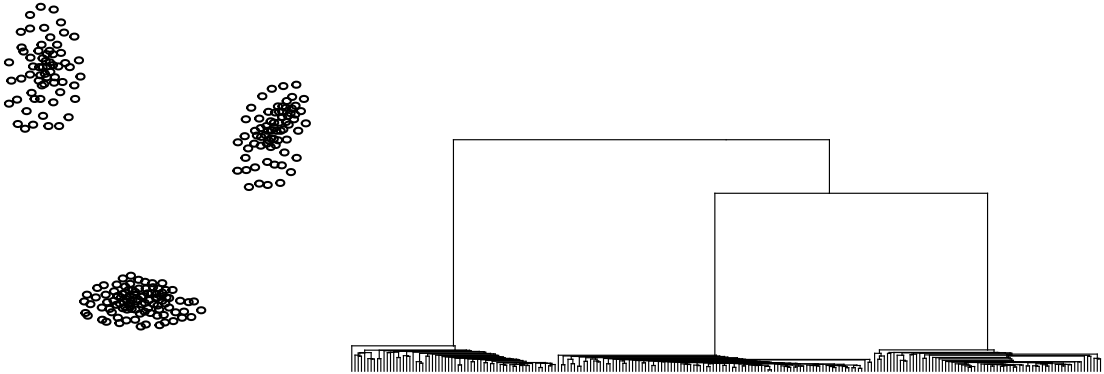


Single linkage

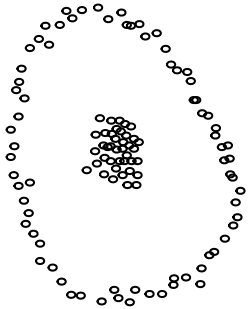


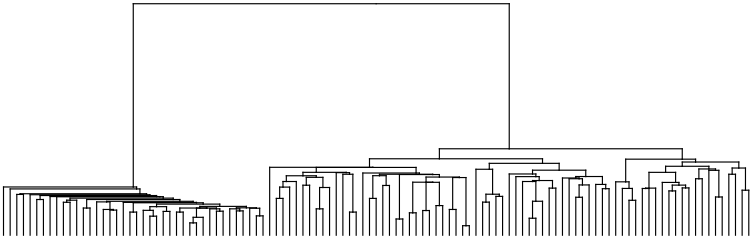
Complete linkage

dendrograms

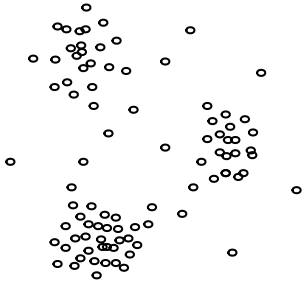


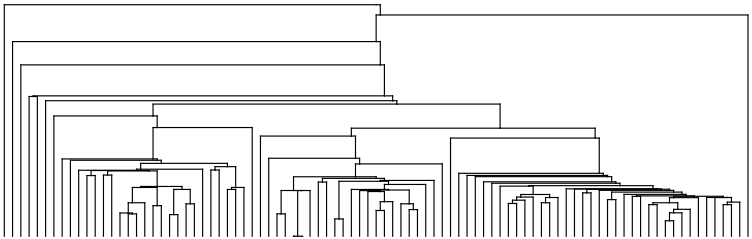
dendrograms





dendrograms





choosing the right clusters

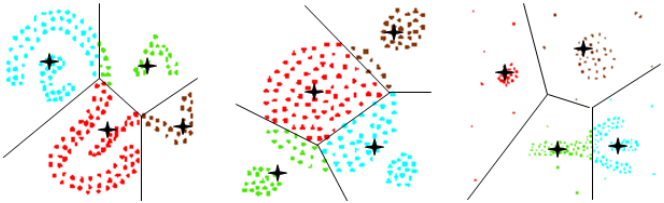
- **simplest approach:**
  - Specify a minimum desired distance between clusters.
  - Stop merging clusters if the closest two clusters are farther apart than this distance.
- **visual approach:**
  - Merge clusters until all data points are combined into one cluster.
  - Draw the dendrogram and find a good cut level.
  - Advantage: Cut needs not be strictly horizontal.
- **more sophisticated approaches:**
  - Analyze the sequence of distances in the merging process.
  - Try to find a step in which the distance between the two clusters merged is considerably larger than the distance of the previous step.
  - Several heuristic criteria exist for this step selection.

density based clustering

Basic idea: **dense regions** of data objects separated by regions of low density

Why Density-Based Clustering methods?

- Discover clusters of arbitrary shape



Results of a k-medoid algorithm for k=4

- Can handle noise
- Requires parameters (but not k)

density based clustering: algorithms

Algorithms

- **DBSCAN**: Density Based Spatial Clustering of Applications with Noise (Ester, et al. KDD'96)
- **DENCLUE**: DENsity-based CLUstEring (Hinneburg and D. Keim KDD'98/2006)
- **OPTICS**: Ordering points to identify the clustering structure (Ankerst, et al. SIGMOD'99).

KDD: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining
SIGMOD: Association for Computing Machinery's Special Interest Group on Management of Data – International Conference on Management of Data

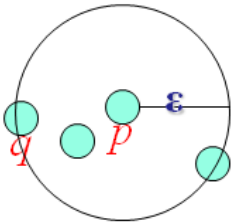
basic concept

- *local point density* – for point p to be in a cluster, the density around that point must exceed some threshold
- local point density at a point p defined by two parameters:
  - ϵ: radius for the neighborhood of point p
    i.e., $N_\epsilon(p) = \{q \text{ in data set } D | dist(p, q) \leq \epsilon\}$
  - MinPts: minimum number of points in the given neighbourhood $N_\epsilon(p)$

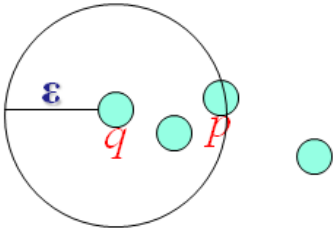
high density

High density: ϵ-Neighborhood of an object contains at least *MinPts* of objects

e.g. for MinPts = 4



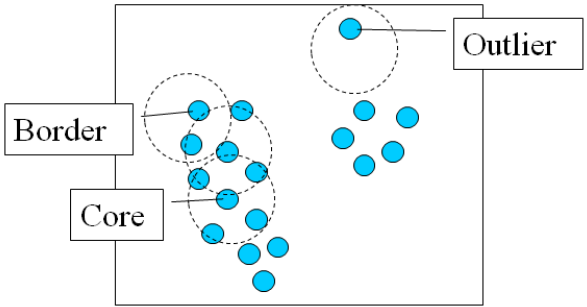
ϵ-neighborhood of *p*
Density of *p* is “high”



ϵ-neighborhood of *q*
Density of *q* is “low”

core, border, and outliers

e.g. ϵ = 1 unit; MinPts = 5



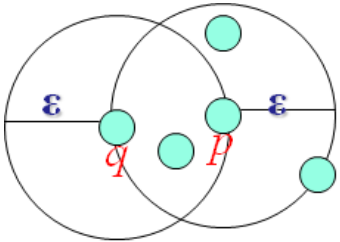
Core points have MinPts or more points within ϵ-radius; located at interior of a cluster.

Border points have fewer than MinPts within ϵ-radius, but are in the neighborhood of a core point.

Outlier points are neither core or border points.

density reachable

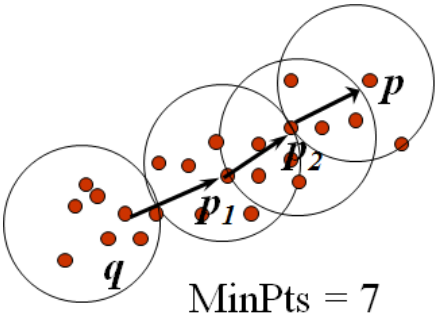
An object q is directly density-reachable from object p if p is a core object and $q \in N_\epsilon(p)$



- q is directly density-reachable from p
- p is not directly density-reachable from q
- density-reachability is asymmetric

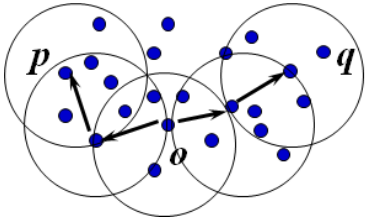
density reachable

- p is directly density-reachable from p_2
- p_2 is directly density-reachable from p_1
- p_1 is directly density-reachable from q
- $p \leftarrow p_2 \leftarrow p_1 \leftarrow q$ form a chain
- p is (indirectly) density-reachable from q
- q is not density-reachable from p



density connectivity

- density-reachability is not symmetric, therefore not good enough to describe clusters
- Two points p and q are density-connected if they are commonly density-reachable from a point o
- Density-connectivty is symmetric and is used to define clusters



density based cluster defined

Formal definition of a density-based cluster

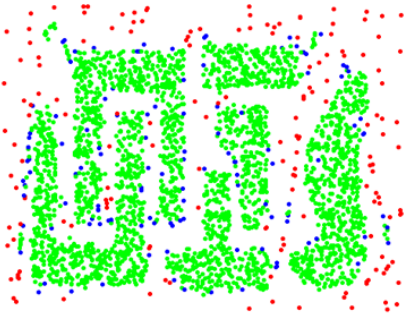
Given: data set D, parameters ϵ and *MinPts*.

A cluster C is a subset of objects satisfying two criteria:

- Connected: $\forall p, q \in C$: p and q are density-connected.
- Maximal: $\forall p, q$: if $p \in C$ and q is density-reachable from p, then $q \in C$.

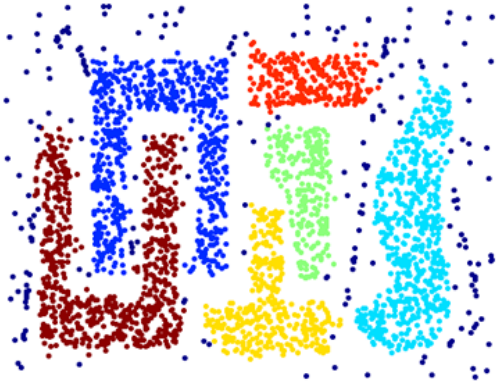
density based cluster examples

Original points



Core, border, and outlier points

density based cluster examples



Density-based 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
- Able to deal with noise and outliers
- Insensitive to order of input records
- Incorporation of user-specified constraints
- Interpretability and usability

summary

- Clustering has a long history and is still active
  - There are a huge number of clustering algorithms
  - More are still coming every year.
- We only introduced several main algorithms.

  There are many others, e.g., sub-space clustering, scale-up methods, neural networks based methods, fuzzy clustering, co-clustering, etc.
- Clustering is hard to evaluate, but very useful in practice. This partially explains why there are still a large number of clustering algorithms being devised every year.
- Clustering is highly application dependent and to some extent subjective.