Data Analysis and Knowledge Discovery

Unsupervised Learning 1

Jari Björne

University of Turku
Department of Computing

jari.bjorne@utu.fi

Outline

Section 1

Finding Patterns

Finding patterns

- Patterns describe or summarize the data set or parts of it.
- Finding patterns is an exploratory data analysis task. There is no specific target attribute whose values should be predicted as in supervised learning
- Often called unsupervised learning

Methods that have already been discussed in the context of data understanding:

- Visualisation techniques like scatter plots or 2/3-dimensional PCA are methods for finding patterns by visual inspection.
- Correlation analysis can find patterns in the form of dependencies of pairs of variables.

Methods for finding patterns

Methods that will be discussed:

Cluster analysis. Identifying groups of "similar" data objects.

Association rules. Finding associations between attributes or typical combinations of values like

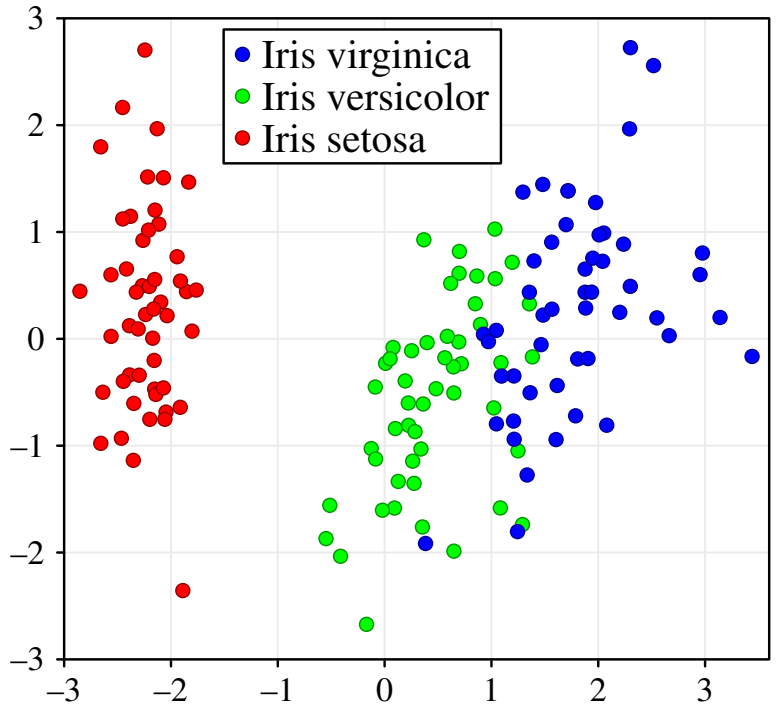
If *Demand=high* and *Supply=low* then *Price=high*.

Deviation analysis. Finding groups that deviate from the rest of the data.

Section 2

Hierarchical Clustering

Hierarchical clustering



In the two-dimensional MDS representation of the Iris data set, two clusters can be identified. (The colours, indicating the species of the flowers, are ignored here.)

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.

Hierarchical clustering

- In order to decide which data objects should belong to the same cluster, a (dis-)similarity measure is needed.
- All that is needed for hierarchical clustering is an $n \times n$-matrix $[d_{i,j}]$, where $d_{i,j}$ is the dissimilarity of data objects i and j. (n is the number of data objects.)

Hierarchical clustering: Dissimilarity matrix

The dissimilarity matrix $[d_{i,j}]$ should at least satisfy the following conditions.

- $d_{i,j} \geq 0$, i.e. dissimilarity cannot be negative.
- $d_{i,i} = 0$, i.e. each data object is completely similar to itself.
- $d_{i,j} = d_{j,i}$, i.e. data object i is (dis-)similar to data object j to the same degree as data object j is (dis-)similar to data object i.

It is often useful if the dissimilarity is a (pseudo-)metric, satisfying also the

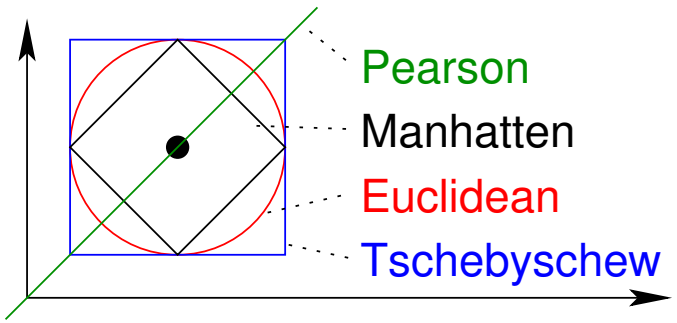
- triangle inequality $d_{i,k} \leq d_{i,j} + d_{j,k}$.

Section 3

Similarity Metrics

Notion of (dis-)similarity: Numerical attributes

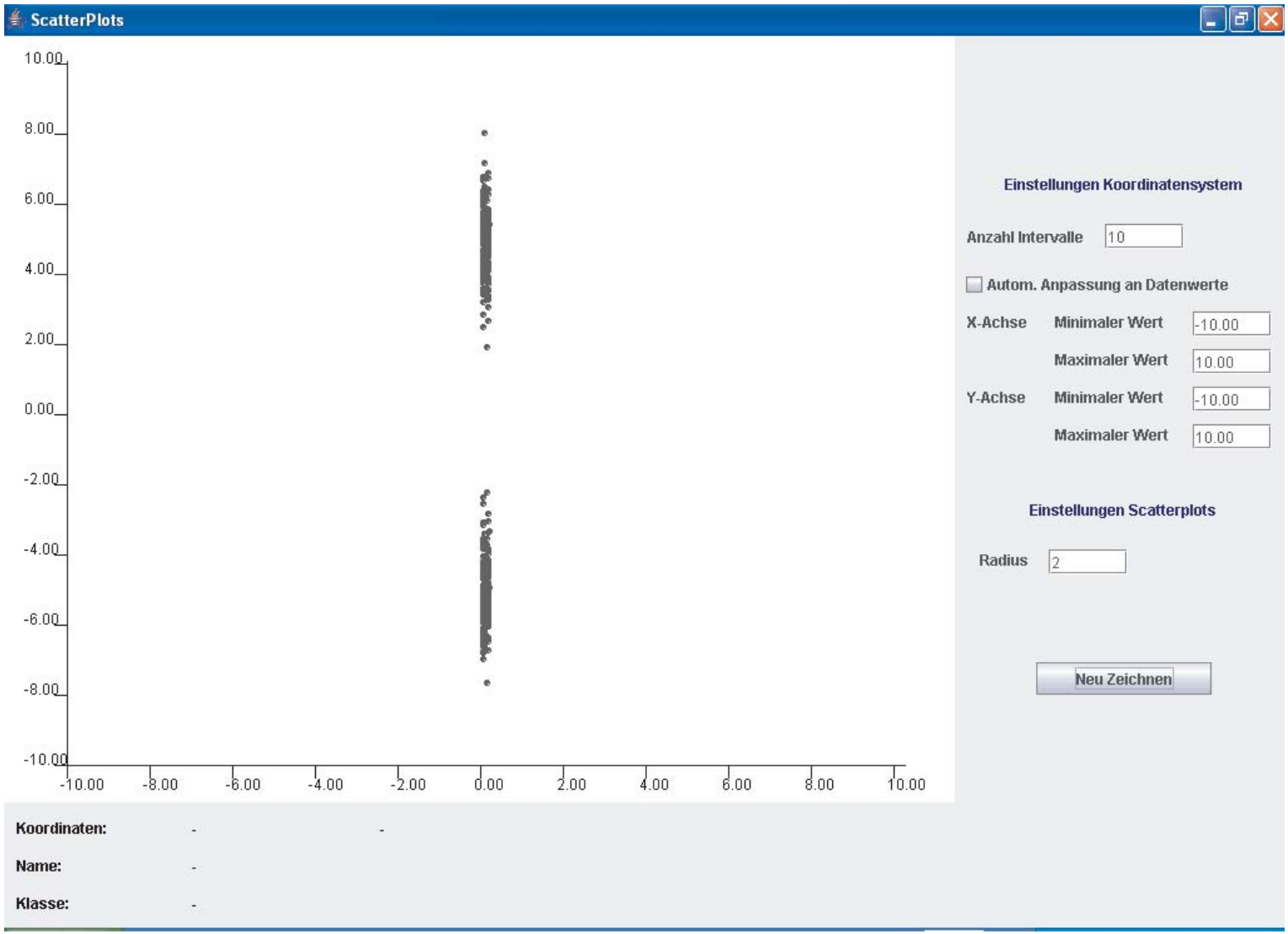
Euclidean	L_2	$d_E(x,y) = \sqrt{(x_1 - y_1)^2 + \ldots + (x_n - y_n)^2}$
Manhattan	L_1	$d_M(x,y) = \lvert x_1 - y_1 \rvert + \ldots + \lvert x_n - y_n \rvert$
Tschebyschew	L_∞	$d_\infty(x,y) = \max\{\lvert x_1 - y_1 \rvert, \ldots, \lvert x_n - y_n \rvert\}$
Minkowski	L_p	$d_p(x,y) = \left(\sqrt{\sum_{i=1}^{n} \lvert x_i - y_i \rvert^p}\right)^{\frac{1}{p}}$
Cosine		$d_C(x,y) = 1 - \frac{x^\top y}{\lVert x \rVert \lVert y \rVert}$
Tanimoto		$d_T(x,y) = \frac{x^\top y}{\lVert x \rVert^2 + \lVert y \rVert^2 - x^\top y}$
Pearson		Euclidean of z-score transformed x, y



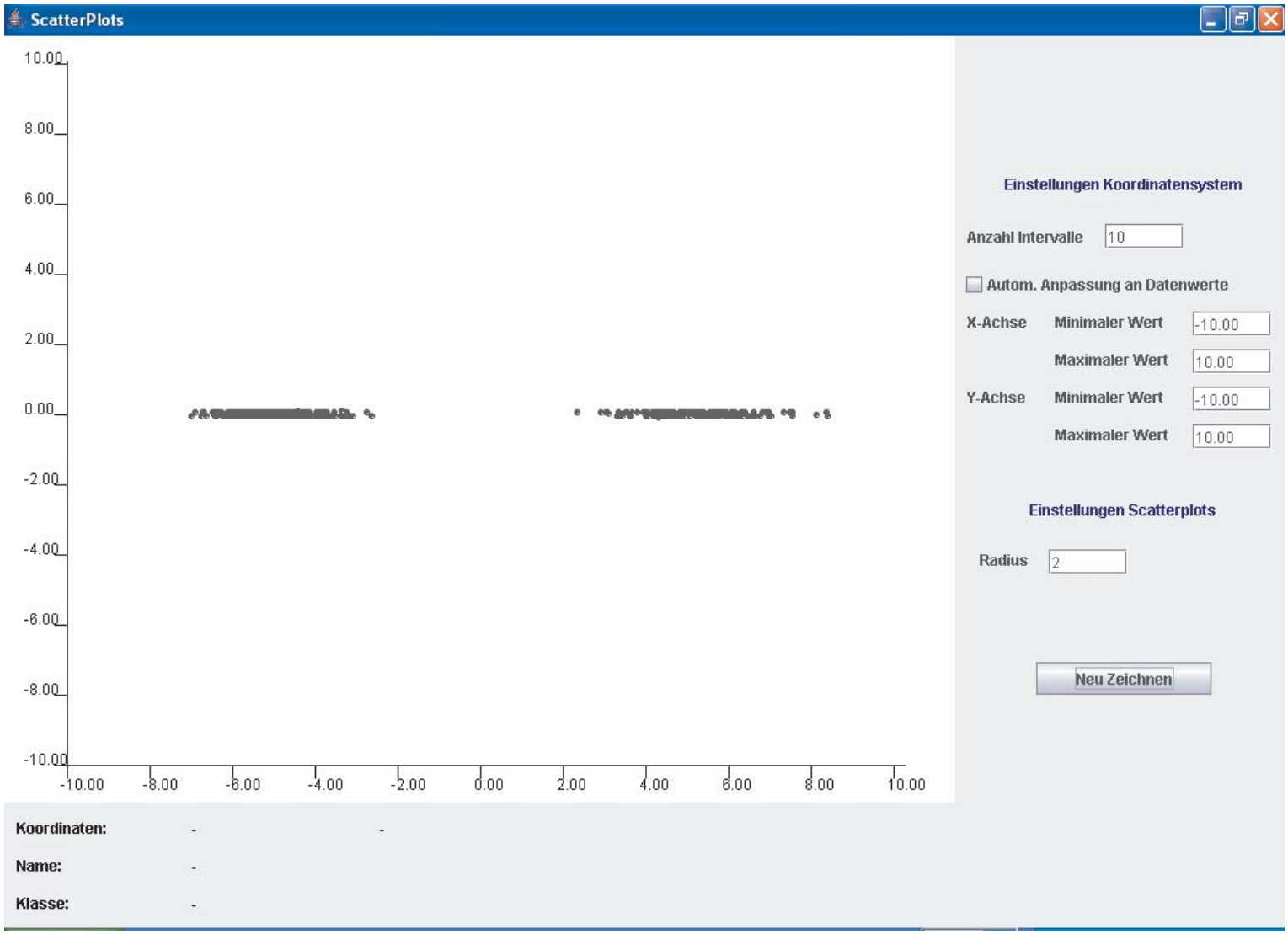
All 2-D points that have a distance of 1.0 from the small centre bullet

Clustering example: original data

Clustering example: x unit change



Clustering example: y unit change



Scaling

The previous three slides show the same data set.

- In the second slide, the unit on the x-axis was changed to centi-units.
- In the third slide, the unit on the y-axis was changed to centi-units.

Clusters should not depend on the measurement unit!

Therefore, some kind of normalisation (see lectures on data preparation) should be carried out before clustering or use appropriate nonisotrophic distance measure.

Nonisotrophic distances

- **Isotropic** means that the distance grows in all directions equally fast (e.g. Euclidean distance).
- **In nonisotrophic** distances tries to capture inherent variations in variable values.
- One of the most common nonistrophic distances is Mahalanobis distance defined as

$$d(x, y) = \sqrt{(x - y)^T \Sigma^{-1} (x - y)}$$

considers the variances across all variables. Here Σ is covariance matrix.

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

Dissimilarity measures for two vectors of binary attributes.

	binary attributes	sets of properties
simple match	$d_S = 1 - \frac{b+n}{b+n+x}$	
Russel & Rao	$d_R = 1 - \frac{b}{b+n+x}$	$1 - \frac{\lvert X \cap Y \rvert}{\lvert \Omega \rvert}$
Jaccard	$d_J = 1 - \frac{b}{b+x}$	$1 - \frac{\lvert X \cap Y \rvert}{\lvert X \cup Y \rvert}$
Dice	$d_D = 1 - \frac{2b}{2b+x}$	$1 - \frac{2\lvert X \cap Y \rvert}{\lvert X \rvert + \lvert Y \rvert}$

	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

Notion of (dis-)similarity: Nominal attributes

- Nominal attributes may be transformed into a set of binary attributes, each of them indicating one particular feature of the attribute.
- Only one of the introduced binary attributes may be active at a time.

  Example. . Attribute *Manufacturer* with the values *BMW*, *Chrysler*, *Dacia*, *Ford*, *Volkswagen*.

manufacturer	...
Volkswagen	...
Dacia	...
Ford	...

$\rightarrow$

binary vector
00001
01000
00100

  Then one of the dissimilarity measures for binary attributes can be applied.

Section 4

Agglomerative Hierarchical Clustering

Agglomerative hierarchical clustering: Algorithm

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

1. Start with n clusters, each data object 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

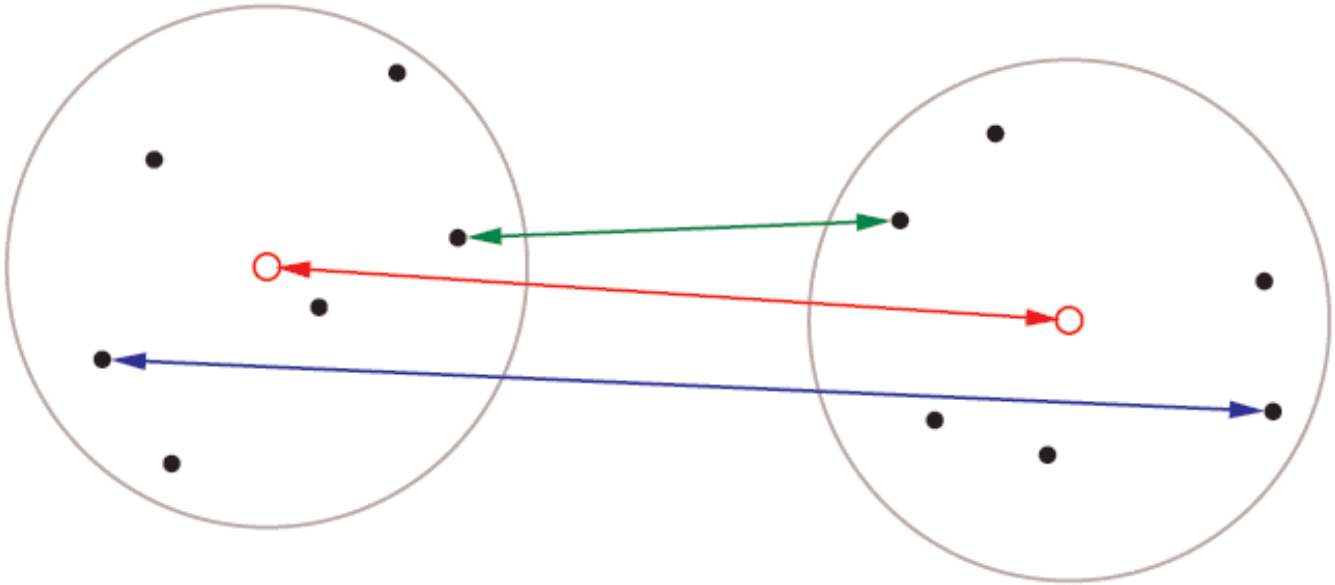
- The dissimilarity between two clusters containing only one data objects each is simply the dissimilarity of the two data objects specified in the dissimilarity matrix $[d_{i,j}]$.
- How do we compute the dissimilarity between clusters that contain more than one data object?

Measuring dissimilarity between clusters

- **Centroid** (red)
  Distance between the centroids (mean value vectors) of the two clusters.
- **Average Linkage**
  Average dissimilarity between two points of the two clusters.
- **Single Linkage** (green)
  Dissimilarity between the two most similar data objects of the two clusters.
- **Complete Linkage** (blue)
  Dissimilarity between the two most dissimilar data objects of the two clusters.

Measuring dissimilarity between clusters

- Red: Centroid
- Green: Single Linkage
- Blue: Complete Linkage

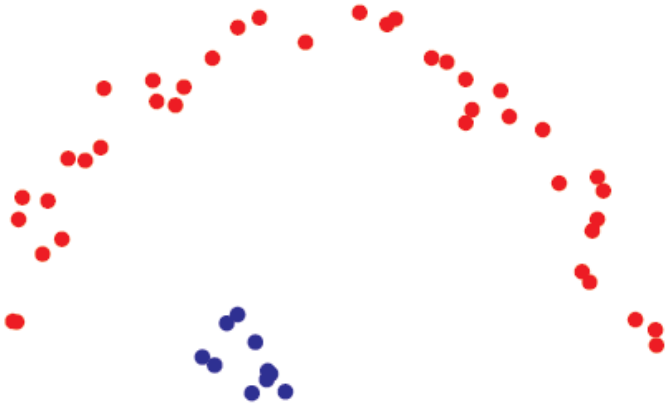


Measuring dissimilarity between clusters

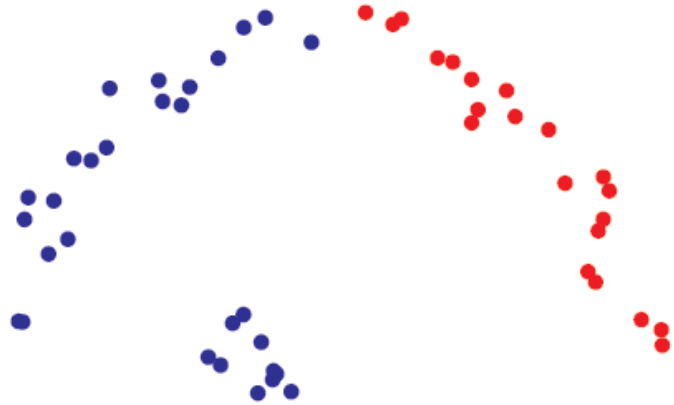
Some typical observations about dissimilarity measures:

- 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

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

Here $d'(\mathcal{C}, \mathcal{C}') = \min\{d(x, y) | x \in \mathcal{C}, y \in \mathcal{C}'\}$, i.e., the distance to a cluster is equivalent to the distance of the nearest point of the cluster.

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

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

Section 5

Dendrograms

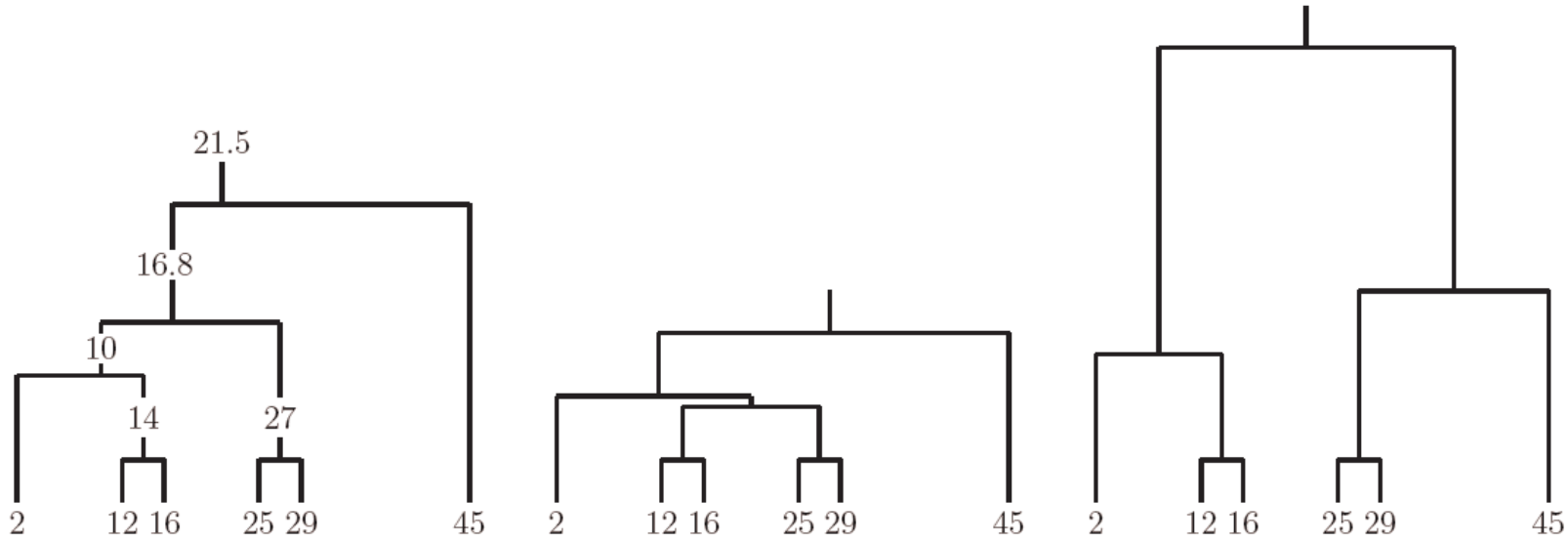
Dendrograms

- 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

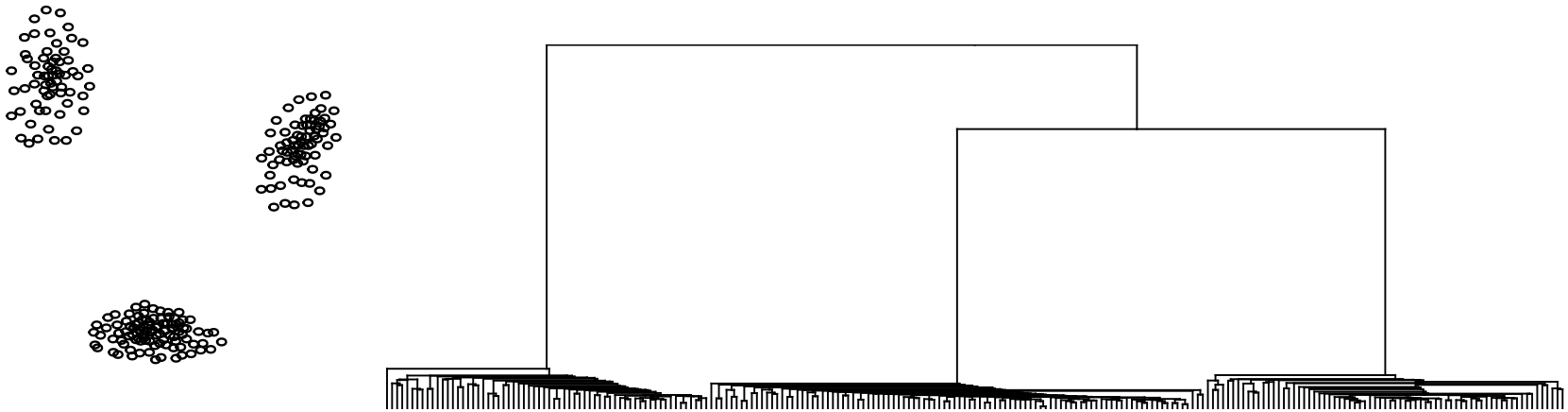
- Example: Clustering of the 1-dimensional data set $\{2, 12, 16, 25, 29, 45\}$.
- All three approaches to measure the distance between clusters lead to different dendrograms.

Hierarchical clustering

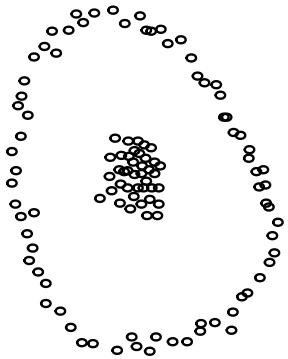


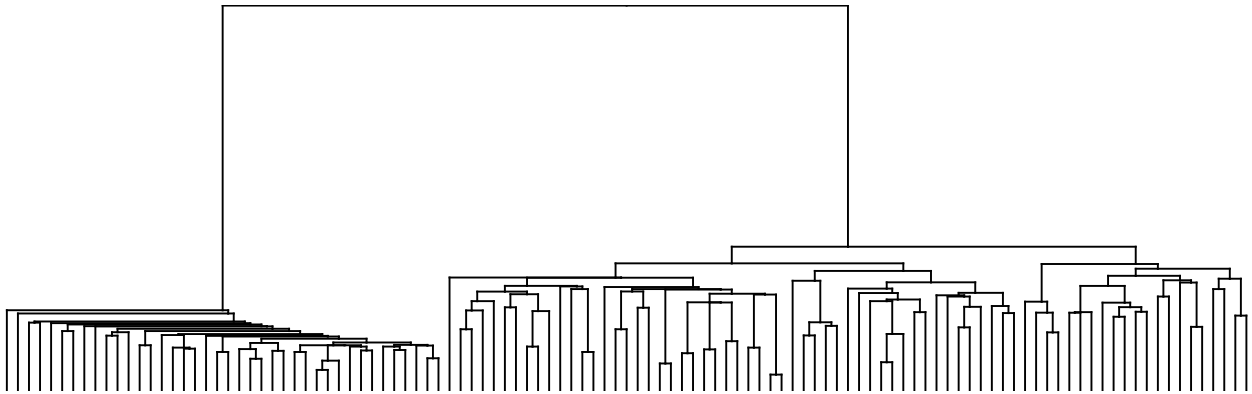
Centroid Single linkage Complete linkage

Dendrograms

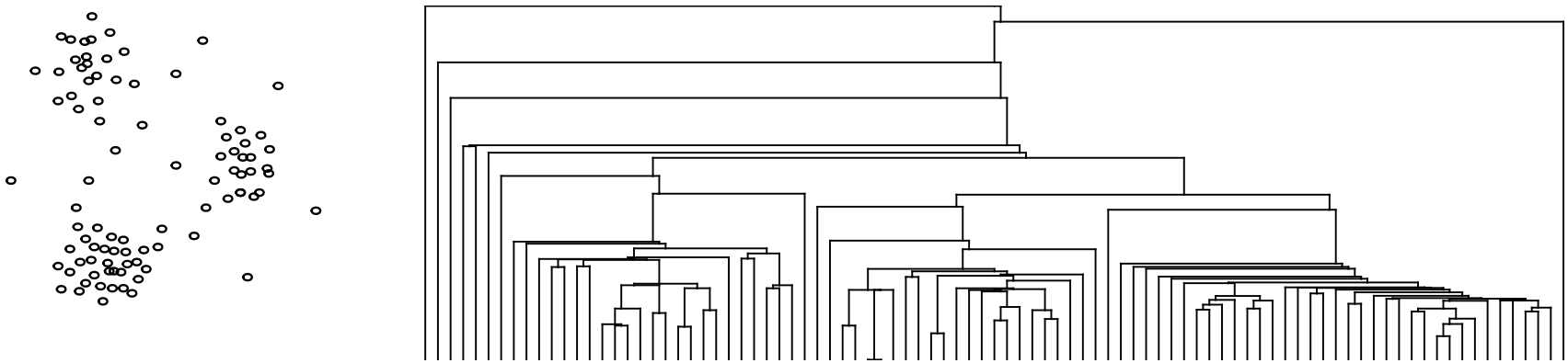


Dendrograms





Dendrograms



Choosing the 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 need not be strictly horizontal.

Choosing the clusters

- **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.

Section 6

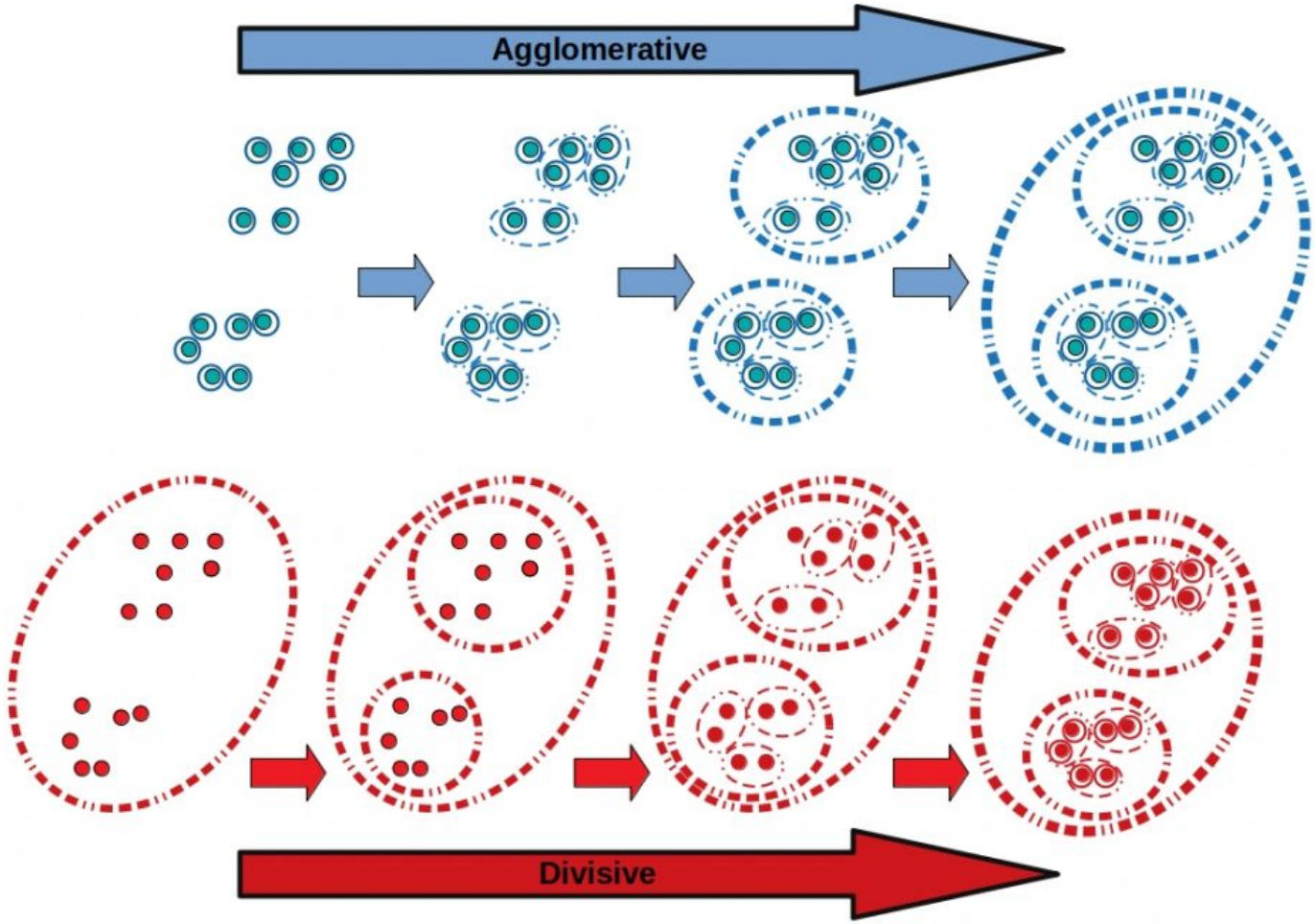
Divisive Hierarchical Clustering

Divisive Hierarchical Clustering

The top-down approach of divisive hierarchical clustering is seldom used.

- In agglomerative clustering the minimum of the pairwise dissimilarities has to be determined, leading to a quadratic complexity in each step (quadratic in the number of clusters still present in the corresponding step).
- In divisive clustering for each cluster all possible splits would have to be considered.
- In the first step, there are $2^{n-1} - 1$ possible splits, where n is the number of data objects.

Divisive Hierarchical Clustering



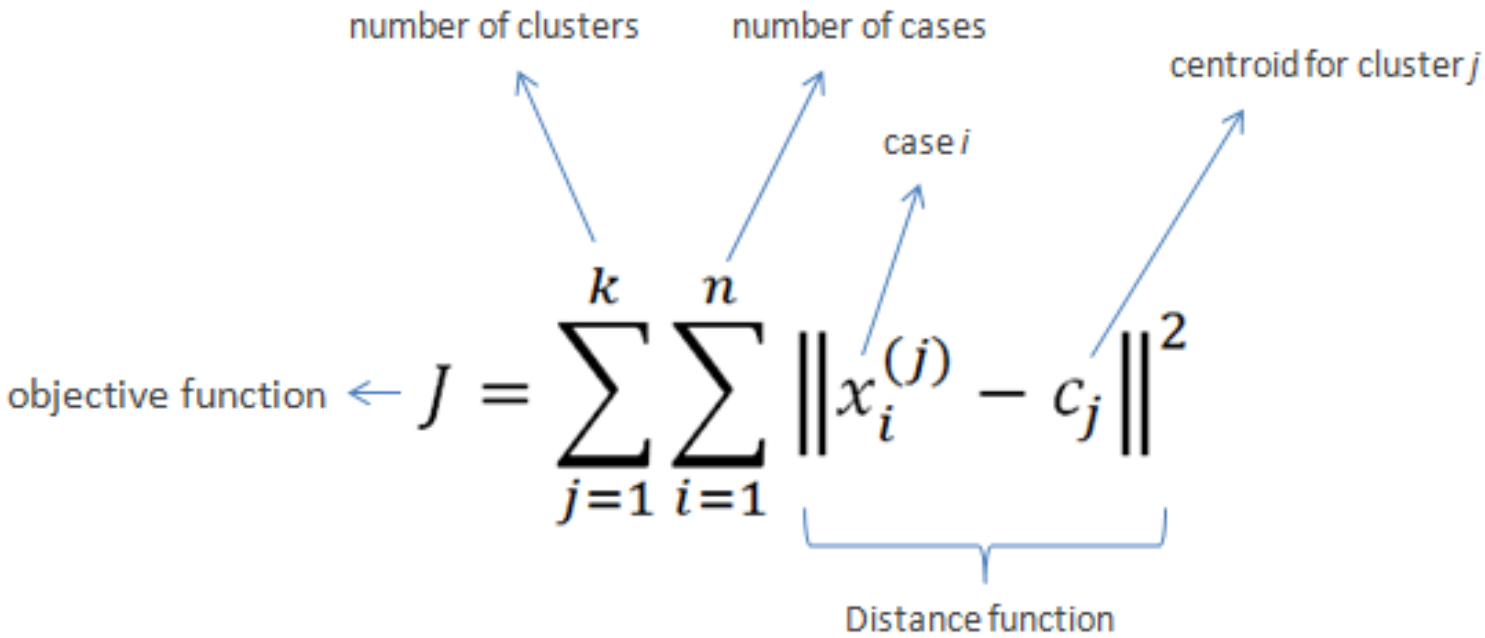
Source: `https://starship-knowledge.com/tag/agglomerative-vs-divisive`

Section 7

k-Means Clustering

k-Means Clustering

- k-means partitions data points into exactly k clusters.
- k must be chosen in advance.
- The objective of K-Means clustering is to minimize the total intra-cluster variance (the squared error function)



Source: `https://www.saedsayad.com/clustering_kmeans.htm`

Alternating Optimization

- 1) Assuming the cluster centres to be fixed, an instance should be assigned to the cluster with the closest centre in order to minimize the objective function.
- 2) Assuming the assignments to the clusters to be fixed, each cluster centre should be chosen as the mean vector of the instances assigned to the cluster in order to minimize the objective function.
- Repeat steps 1 and 2.
- This is a greedy algorithm (may get stuck in local optima).

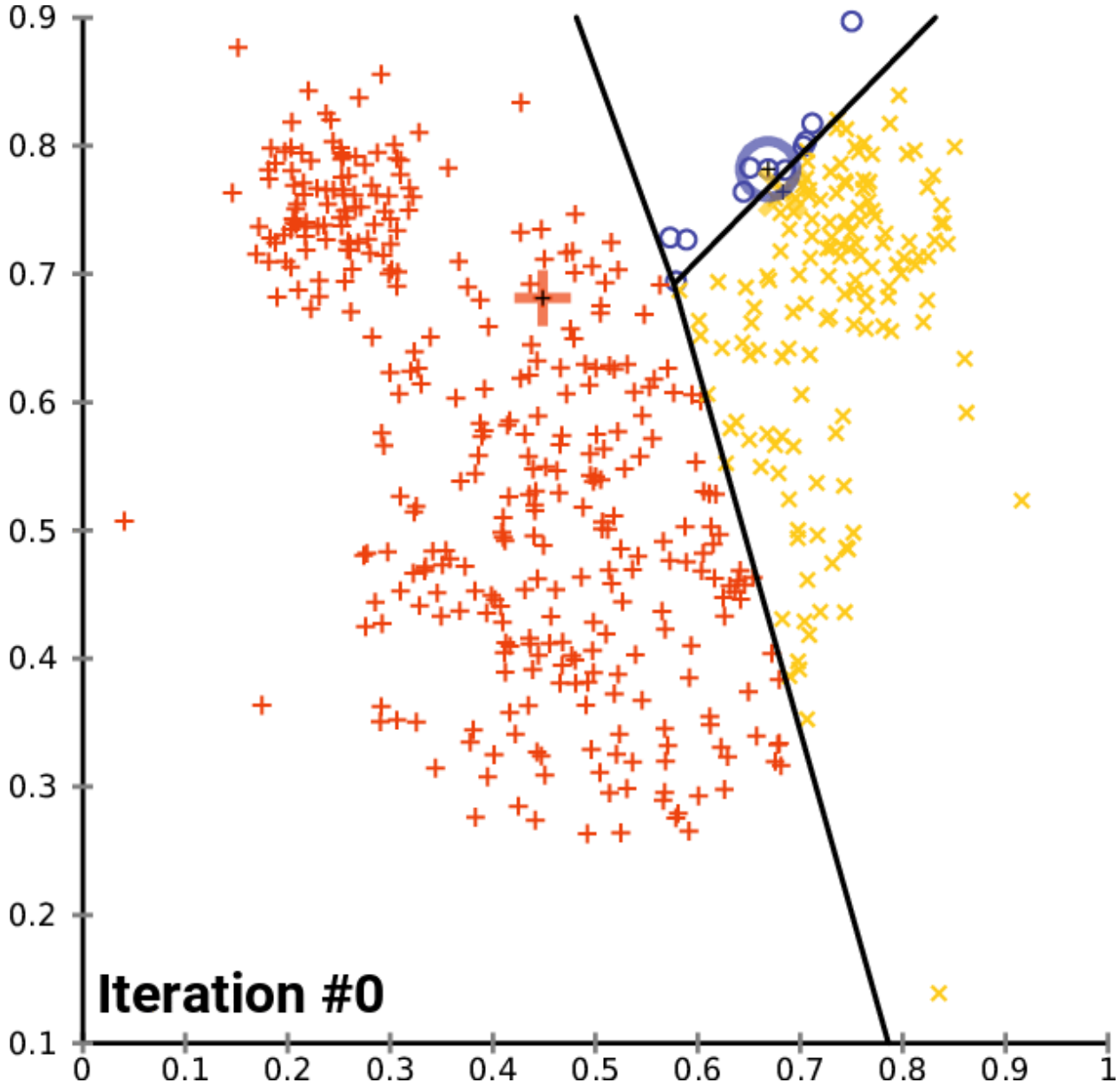
k-Means clustering

- Choose a number k of clusters to be found (user input).
- Initialize the cluster centres randomly
  (for instance, by randomly selecting k data points).
- **Data point assignment**:
  Assign each data point to the cluster centre that is closest to it (i.e. closer than any other cluster centre).
- **Cluster centre update**:
  Compute new cluster centres as the mean vectors of the assigned data points. (Intuitively: centre of gravity if each data point has unit weight.)

k-Means clustering

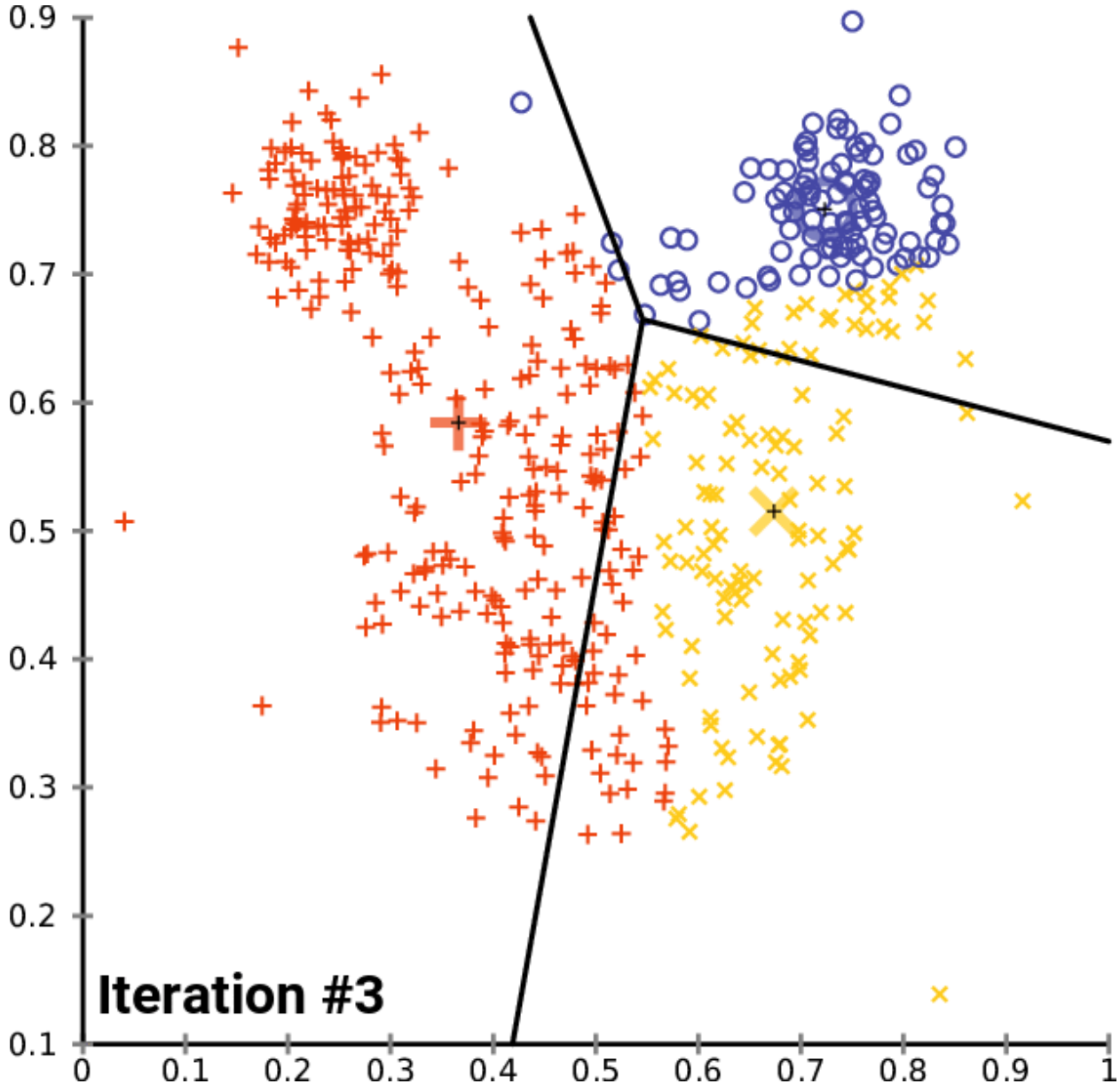
- Repeat these two steps (data point assignment and cluster centre update) until the clusters centres do not change anymore.
- It can be shown that this scheme must converge, i.e., the update of the cluster centres cannot go on forever.

k-Means clustering: Example



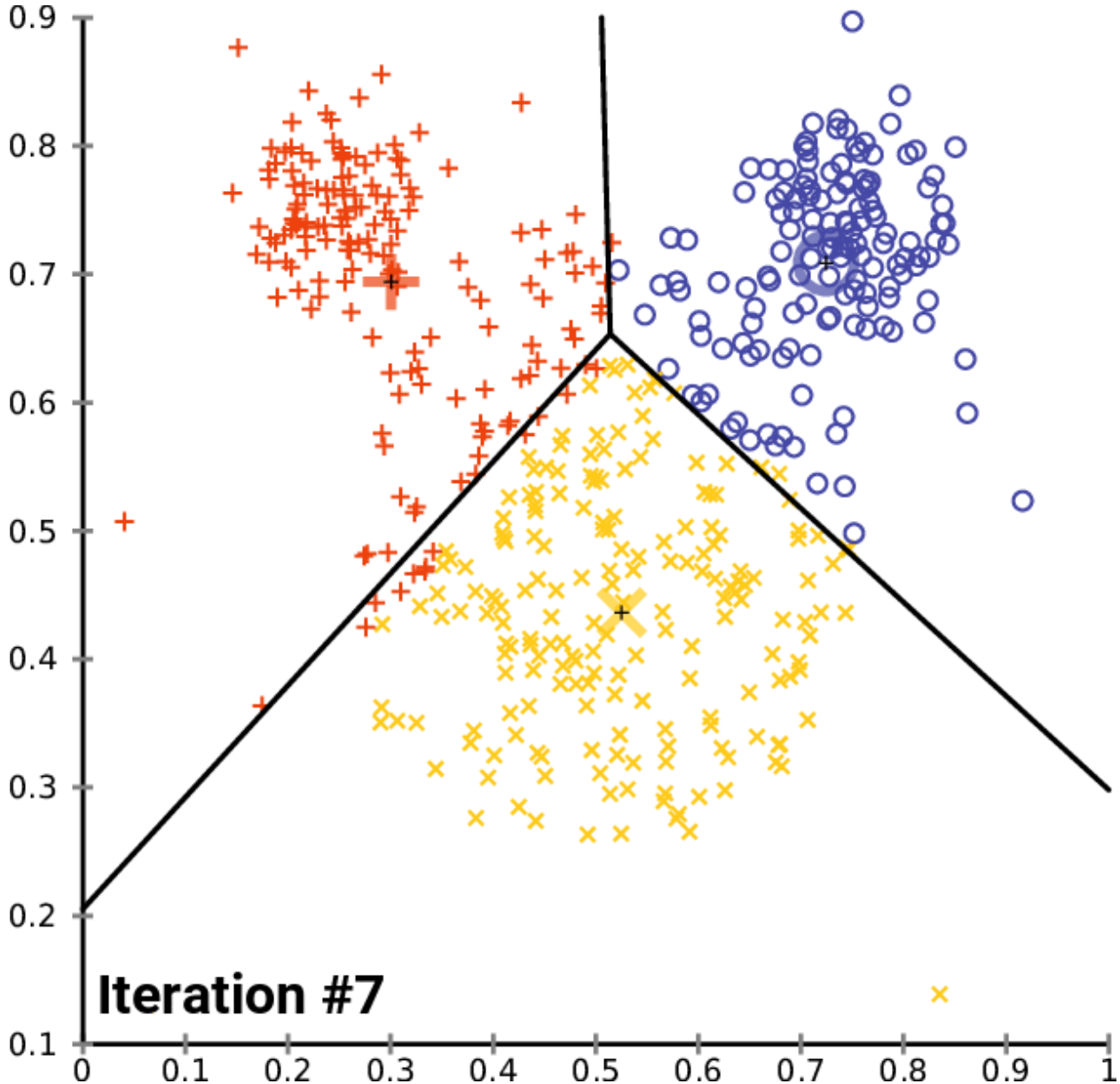
Source: Wikimedia Commons, Chire

k-Means clustering: Example



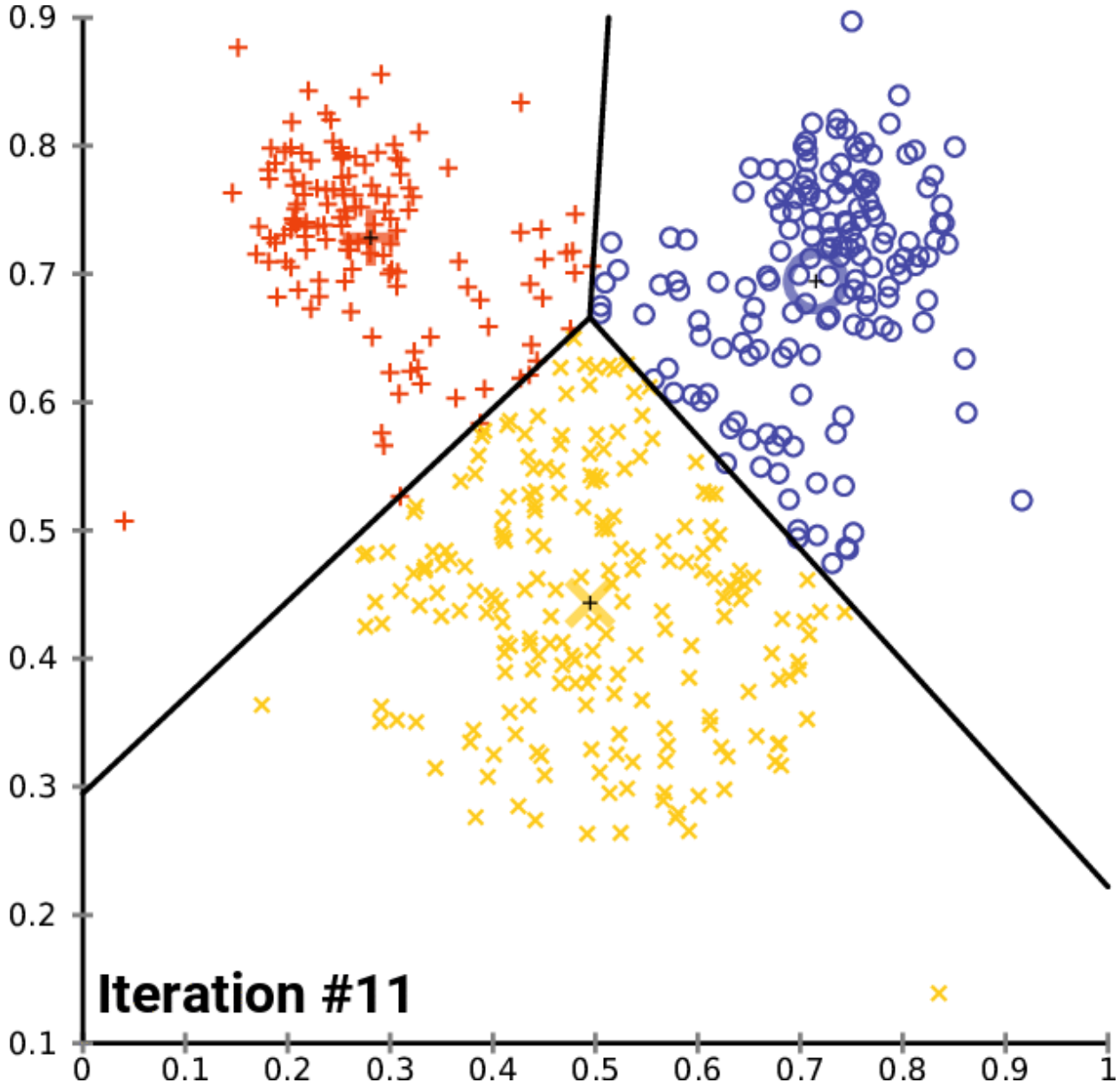
Source: Wikimedia Commons, Chire

k-Means clustering: Example



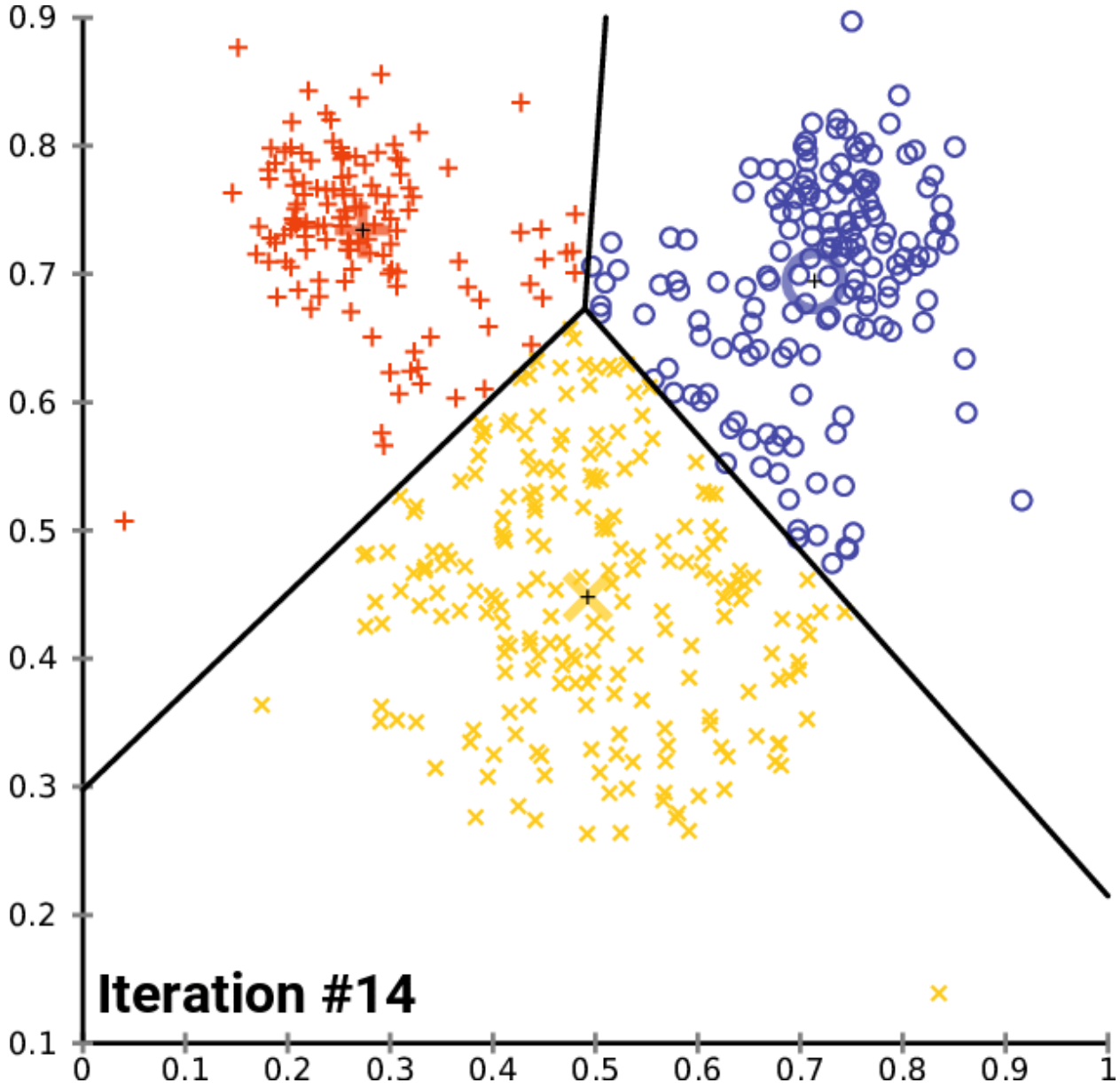
Source: Wikimedia Commons, Chire

k-Means clustering: Example



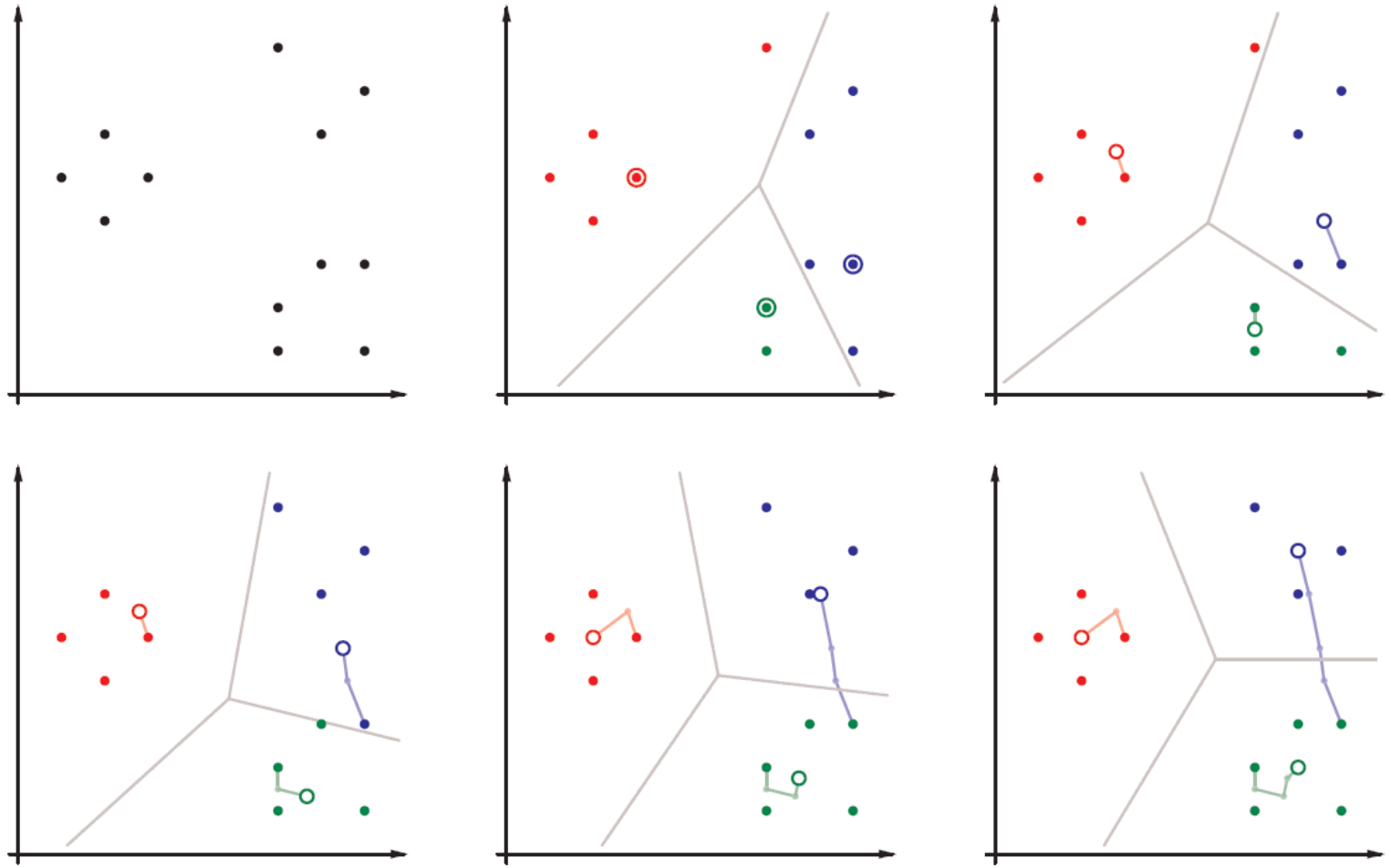
Source: Wikimedia Commons, Chire

k-Means clustering: Example



Source: Wikimedia Commons, Chire

k-Means clustering: Convergence

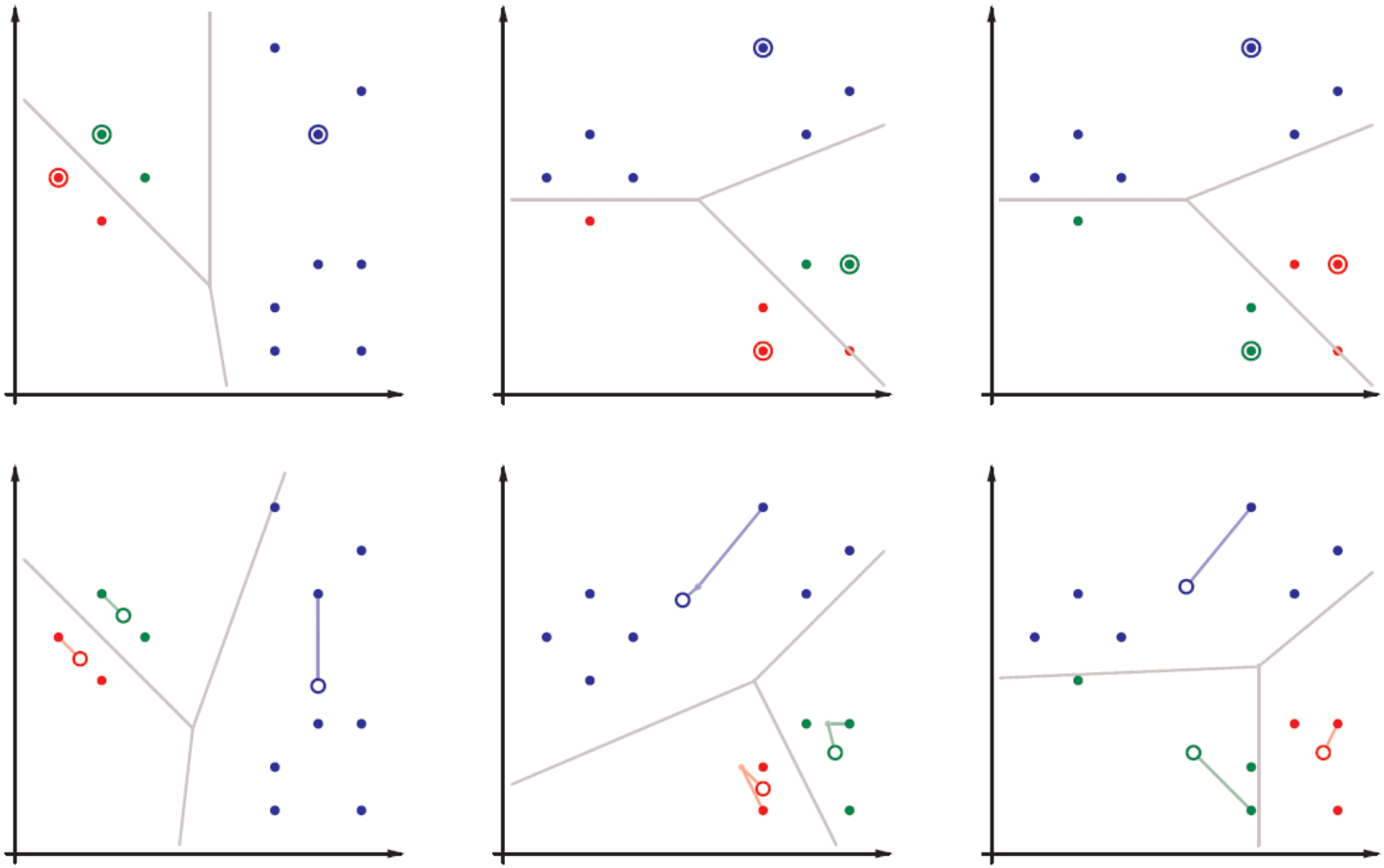


k-Means clustering: Local minima and k

- Clustering is successful in the previous example:
  The clusters found are those that would have been formed intuitively.
- Convergence is achieved after only 5 steps.
  (This is typical: convergence is usually very fast.)
- However: The clustering result is fairly **sensitive to the initial positions** of the cluster centres.
- With a bad initialisation clustering may fail
  (the alternating update process gets stuck in a local minimum).
- Optimal k can be estimated by methods shown in the book, but many times is selected manually by checking the meanings of clusters.

k-Means clustering: Local minima

Three examples of bad initialization



k-Means clustering: choosing k

- How to choose the number of clusters?
- Objective function of k-means: not useful, since it improves as k grows
- Prior constraints: "customers have to be assigned to k account managers..."
- Exploring the data: "with this number of clusters the results make sense..."
- Validity measures: assign a numeric value to how good the clustering is $\rightarrow$ may not work well unless the structure of the data matches the assumptions of the validity measure

Example: Silhouette coefficient

- The silhouette value is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation).
- The silhouette ranges from -1 to +1
- A high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters.
- If most objects have a high value, then the clustering configuration is appropriate.
- If many points have a low or negative value, then the clustering configuration may have too many or too few clusters.

$$s(i) = \frac{b(i) - a(i)}{max\{a(i), b(i)\}}$$

$a(i)$: average distance between i:th instance and instances in same cluster
$b(i)$: average distance between i:th instance and instances in other clusters

Section 8

Density-based Clustering

Density-based Clustering

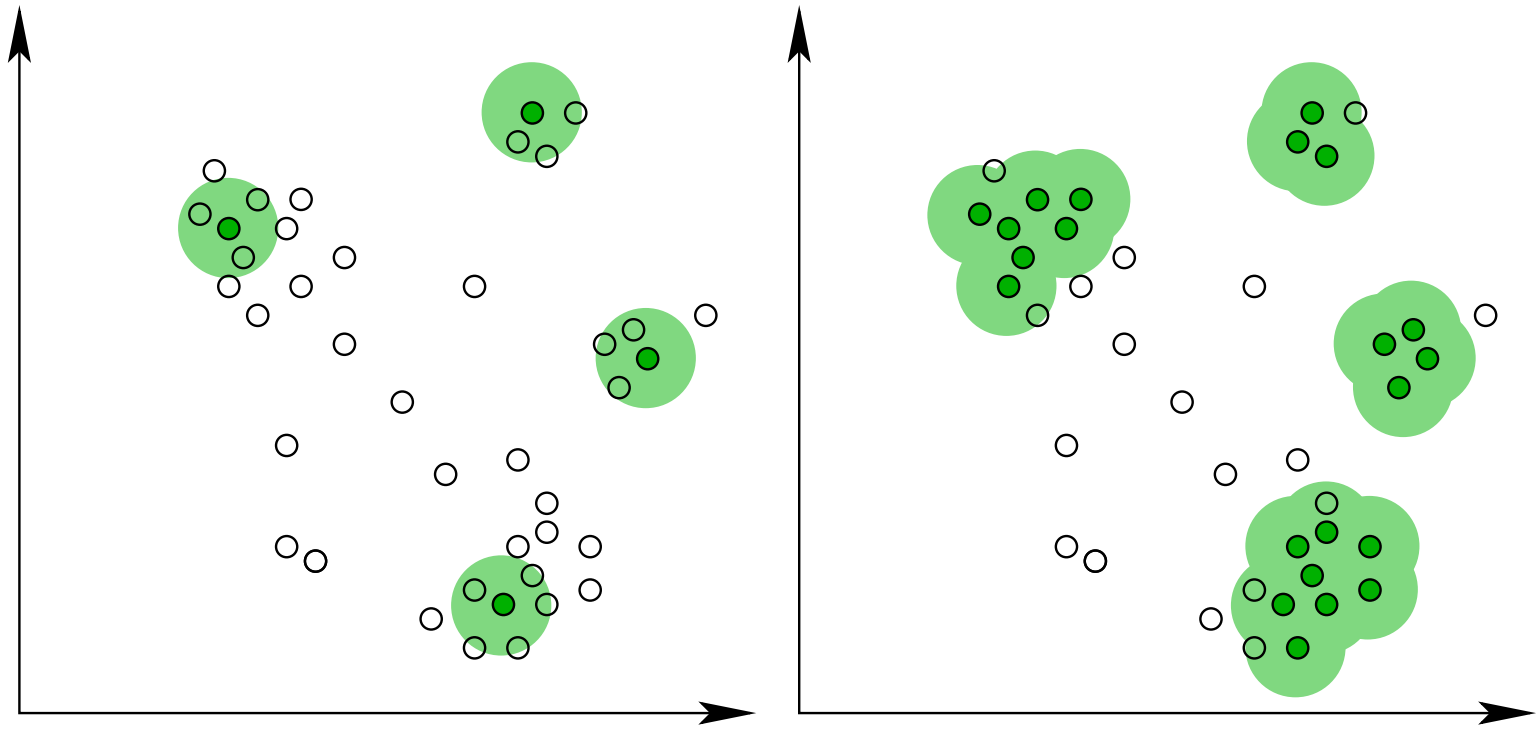
- For numerical data, the density-based clustering algorithm can be also considered.
- Principle: A connected region with high data density corresponds to one cluster.
- DBScan is one of the density-based clustering algorithms.

Density-based Clustering: DBScan

The principle idea of DBScan:

1. Find a data point where the data density is high, i.e. in whose ε-neighbourhood are at least ℓ other points. (ε and ℓ are parameters of the algorithm to be chosen by the user.)
2. All the points in the ε-neighbourhood are considered to belong to one cluster.
3. Expand this ε-neighbourhood (the cluster) as long as the high density criterion is satisfied.
4. Remove the cluster (all data points assigned to the cluster) from the data set and continue with 1. as long as data points with a high data density around them can be found.

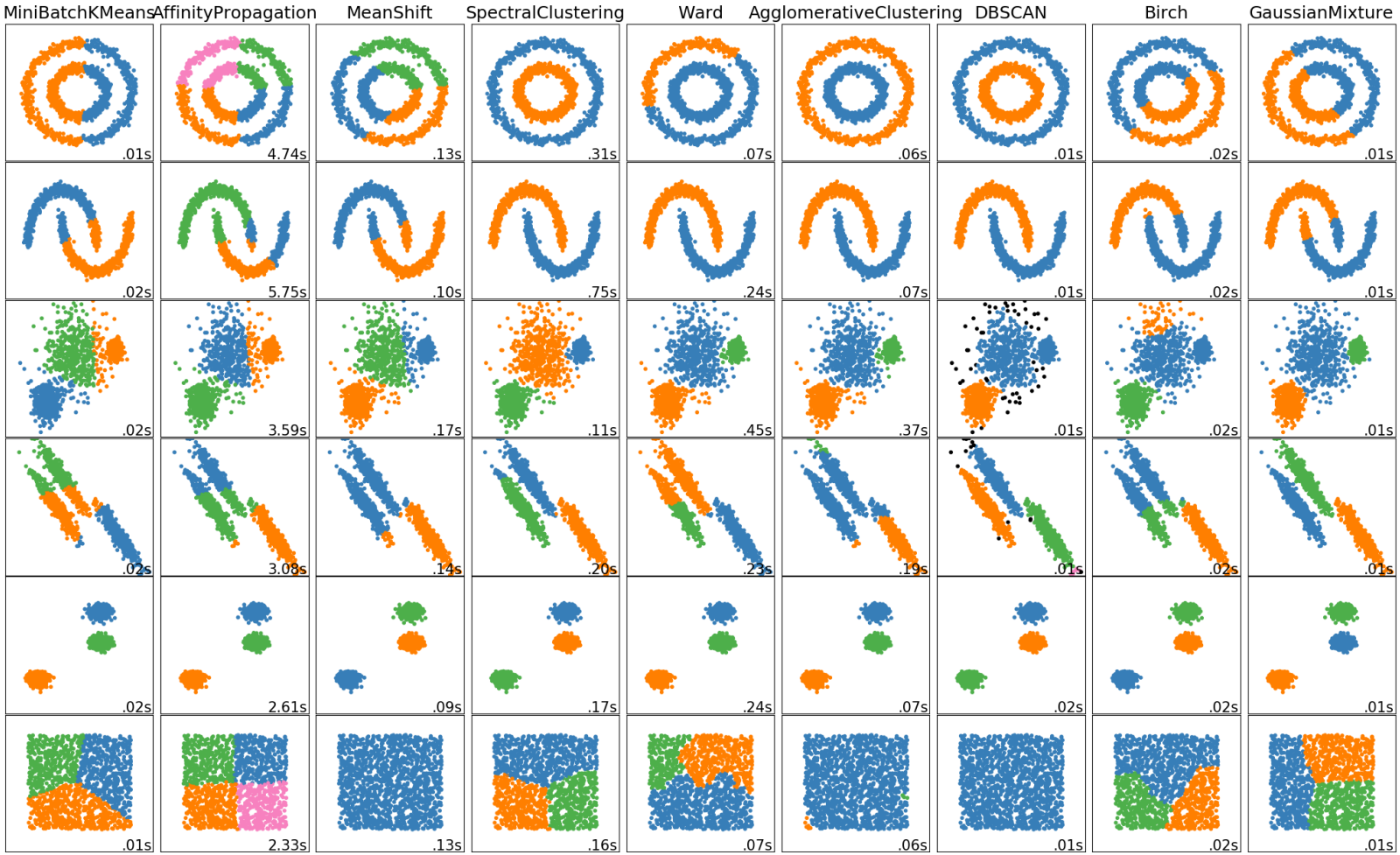
Density-based Clustering: DBScan



neighbourhood cell with at least 3 hits

neighbourhood cell with at least 3 hits

Comparing Clustering Methods



Example from scikit-learn documentation