Chapter 5-2: **Clustering**

Jilles Vreeken

Revision 1, November 20th
typo's fixed: dend**r**ogram

Revision 2, December 10th
clarified: we do consider a point x as a member of its own ϵ-neighborhood

IRDM '15/16

12 Nov 2015

UNIVERSITÄT DES SAARLANDES

CLUSTER OF EXCELLENCE

max planck institut informatik

The First Midterm Test

November 19th 2015

Where: Günter-Hotz Hörsaal (E2.2)

Material: the first four lectures, the first two homeworks

You are allowed to bring one (1) sheet of A4 paper with handwritten or printed notes on both sides .

No other material (notes, books, course materials) or devices (calculator, notebook, cell phone, toothbrush, etc) allowed.

Bring an ID; either your UdS card, or passport.

The Final Exam

Preliminary dates: February 15th and 16th 2016

Oral exam.

Can only be taken when you passed two out of three mid-term tests.

More details later.

IRDM Chapter 5, overview

1. Basic idea
2. Representative-based clustering
3. Probabilistic clustering
4. Validation
5. Hierarchical clustering
6. Density-based clustering
7. Clustering high-dimensional data

You'll find this covered in
Aggarwal Ch. 6, 7
Zaki & Meira, Ch. 13—15

IRDM Chapter 5, today

1. Basic idea
2. Representative-based clustering
3. Probabilistic clustering
4. Validation
5. Hierarchical clustering
6. Density-based clustering
7. Clustering high-dimensional data

You'll find this covered in
Aggarwal Ch. 6, 7
Zaki & Meira, Ch. 13—15

Chapter 5.5:

Hierarchical Clustering

Aggarwal Ch. 6.4

The basic idea

Create clustering for each number of clusters $k = 1,2,\ldots,n$

The clusterings must be **hierarchical**

- every cluster of k-clustering is a union of some clusters in an l-clustering for all $k < l$
- i.e. for all l, and for all $k > l$, every cluster in an l-clustering is a subset of some cluster in the k-clustering

Example:

$k = 6$

The basic idea

Create clustering for each number of clusters $k = 1, 2, \ldots, n$

The clusterings must be **hierarchical**

- every cluster of k-clustering is a union of some clusters in an l-clustering for all $k < l$
- i.e. for all l, and for all $k > l$, every cluster in an l-clustering is a subset of some cluster in the k-clustering

Example:

$\mathrm{k} = 5$

The basic idea

Create clustering for each number of clusters $k = 1,2,\ldots,n$

The clusterings must be **hierarchical**

- every cluster of k-clustering is a union of some clusters in an l-clustering for all $k < l$
- i.e. for all l, and for all $k > l$, every cluster in an l-clustering is a subset of some cluster in the k-clustering

Example:

$\mathrm{k} = 4$

The basic idea

Create clustering for each number of clusters $k = 1,2,\ldots,n$

The clusterings must be **hierarchical**

- every cluster of k-clustering is a union of some clusters in an l-clustering for all $k < l$
- i.e. for all l, and for all $k > l$, every cluster in an l-clustering is a subset of some cluster in the k-clustering

Example:

$\mathrm{k} = 3$

The basic idea

Create clustering for each number of clusters $k = 1,2,\dots,n$

The clusterings must be **hierarchical**

- every cluster of k-clustering is a union of some clusters in an l-clustering for all $k < l$
- i.e. for all l, and for all $k > l$, every cluster in an l-clustering is a subset of some cluster in the k-clustering

Example:

$\mathrm{k} = 2$

The basic idea

Create clustering for each number of clusters $k = 1,2,\dots,n$

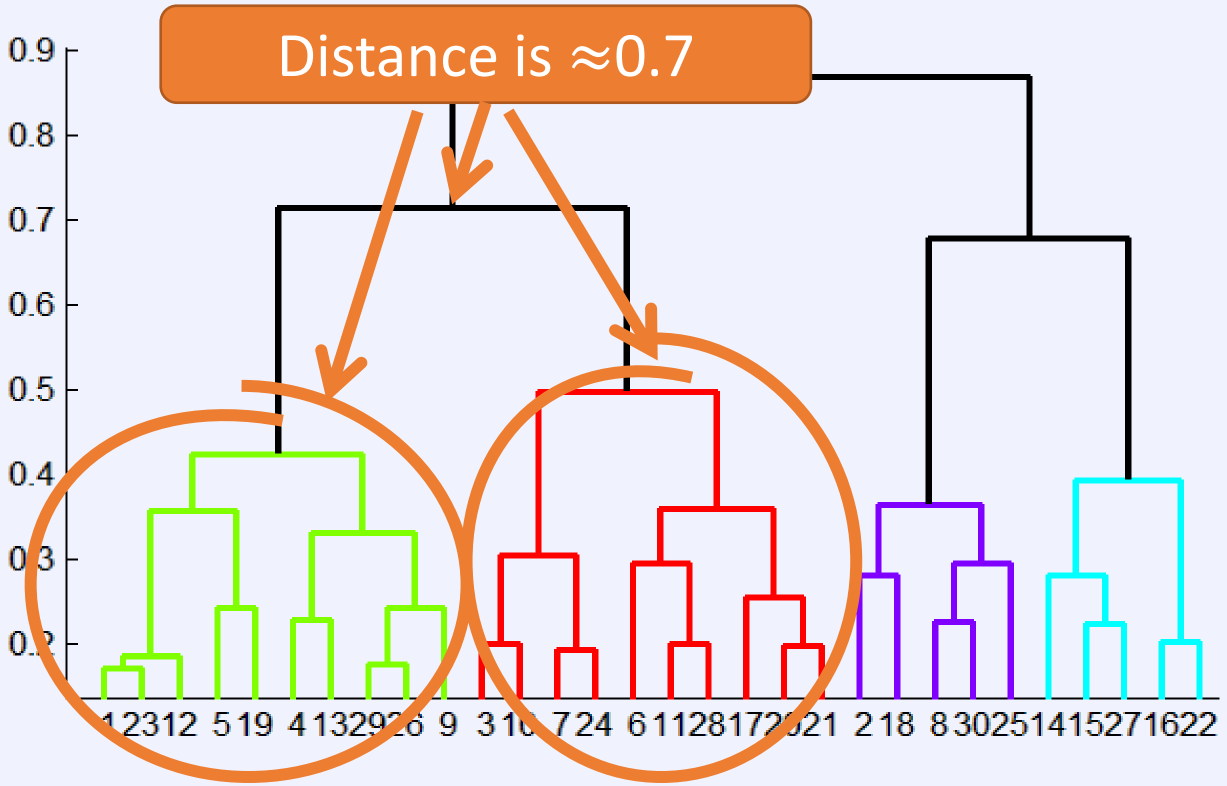
The clusterings must be **hierarchical**

- every cluster of k-clustering is a union of some clusters in an l-clustering for all $k < l$
- i.e. for all l, and for all $k > l$, every cluster in an l-clustering is a subset of some cluster in the k-clustering

Example:

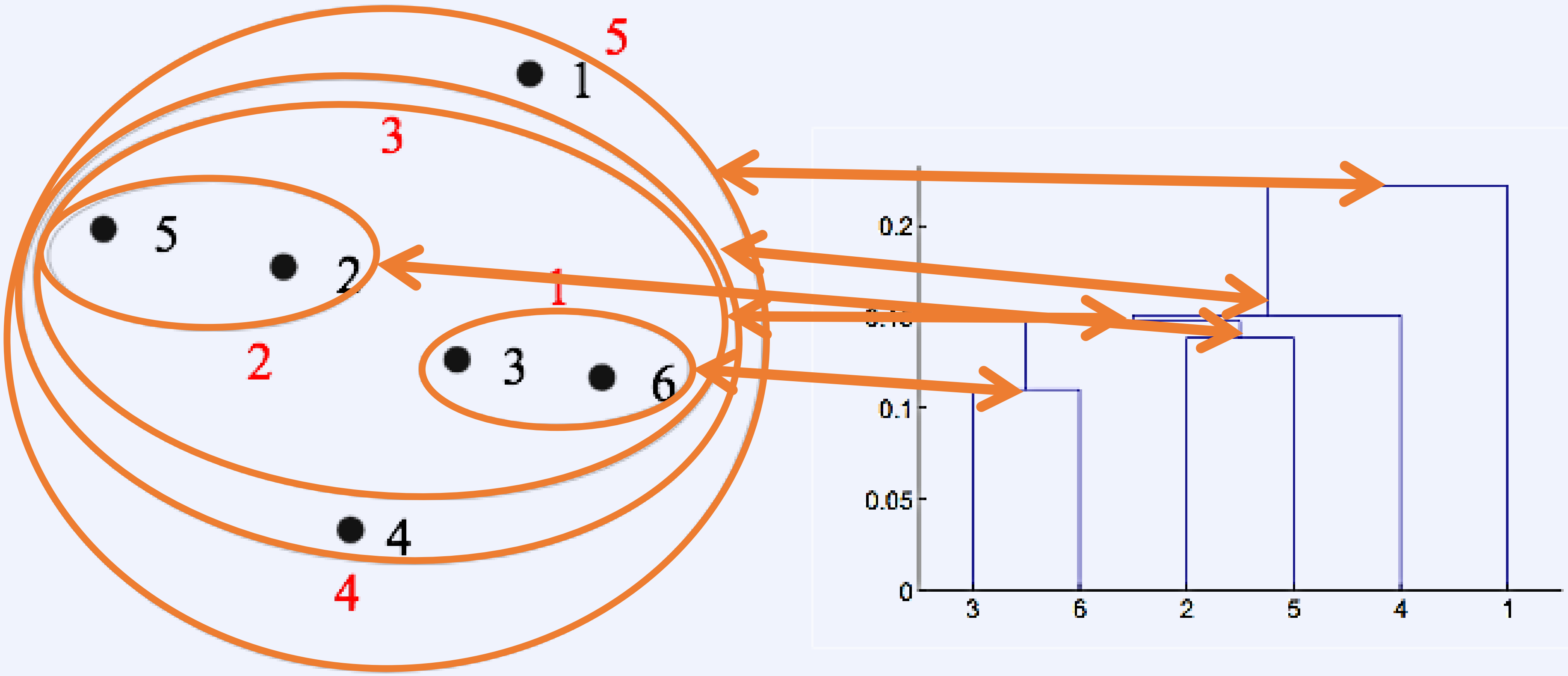
$k = 1$

Dendrograms



The difference in height between the tree and its subtrees shows the distance between the two branches

Dendrograms and clusters



Dendrograms, revisited

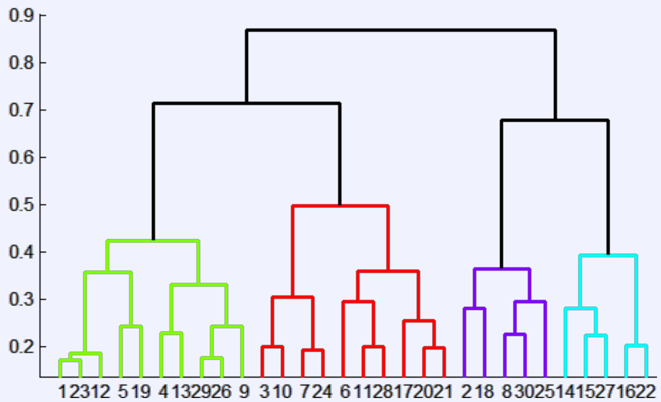
Dendrograms show the hierarchy of the clustering

Number of clusters can be deduced from a dendrogram

- higher branches

Outliers can be detected from a dendrogram

- single points that are far from others



Agglomerative and Divisive

Agglomerative: bottom-up

- start with n clusters
- combine two closest clusters into a cluster of one bigger cluster

Divisive: top-down

- start with 1 cluster
- divide the cluster into two
  - divide the largest (per diameter) cluster into smaller clusters

Cluster distances

The distance between two points x and y is $d(x, y)$

What is the distance between two clusters?

Many intuitive definitions – no universal truth

- different cluster distances yield different clusterings
- the selection of cluster distance depends on application

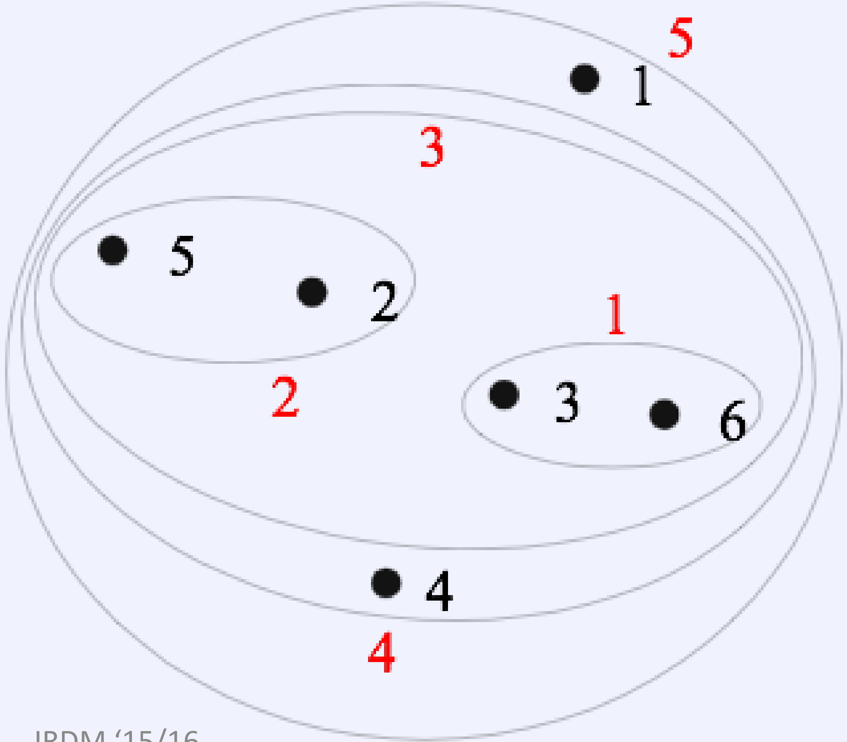
Some distances between clusters B and C:

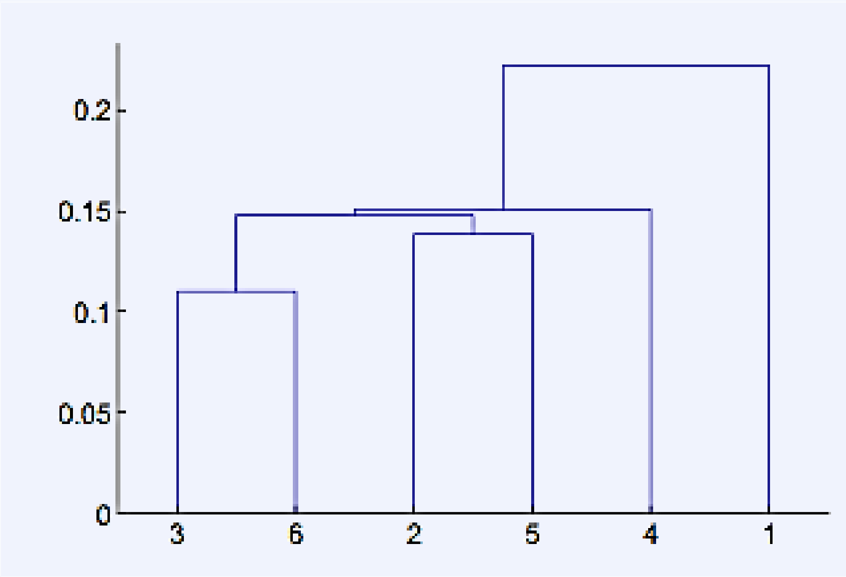
- minimum distance $d(B, C) = \min\{d(x, y) : x \in B \textit{ and } y \in C\}$
- maximum distance $d(B, C) = \max\{d(x, y) : x \in B \textit{ and } y \in C\}$
- average distance $d(B, C) = avg\{d(x, y) : x \in B \textit{ and } y \in C\}$
- distance of centroids $d(B, C) = d(\mu_B, \mu_C)$,
  where μ_B is the centroid of B and μ_C is the centroid of C

Single link

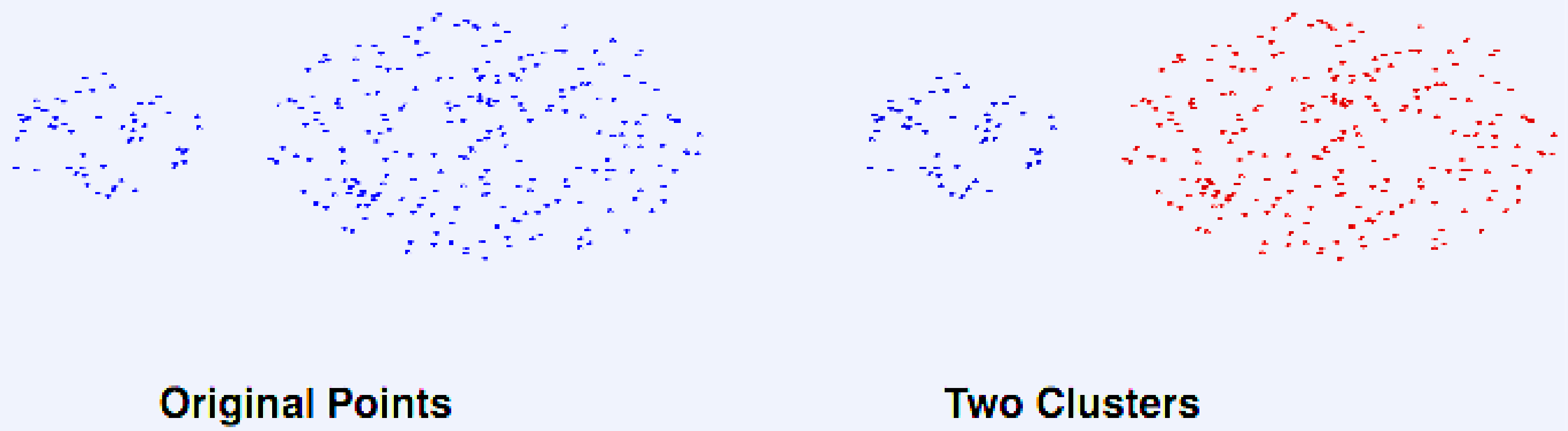
The distance between two clusters is the distance between the closest points

- $d(B,C) = \min\{d(x,y) : x \in B \text{ and } y \in C\}$



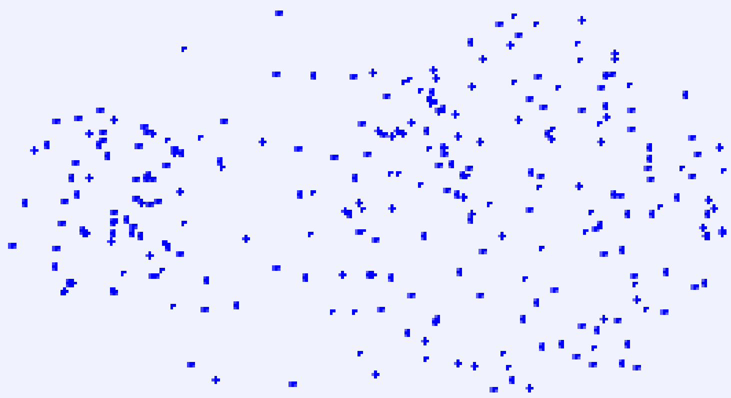


Strength of single-link

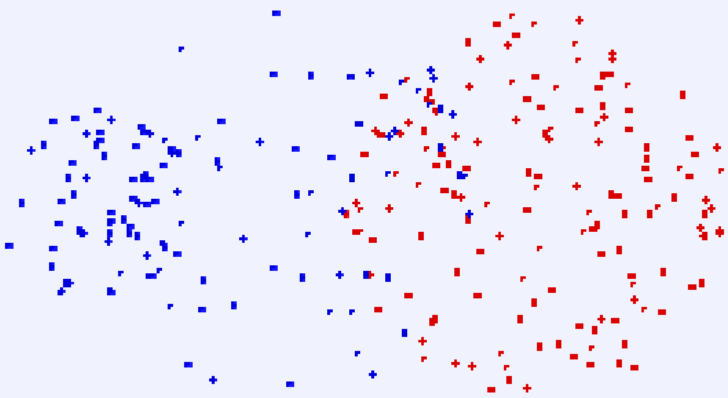


Can handle non-spherical clusters of unequal size

Weaknesses of single-link



Original Points



Two Clusters

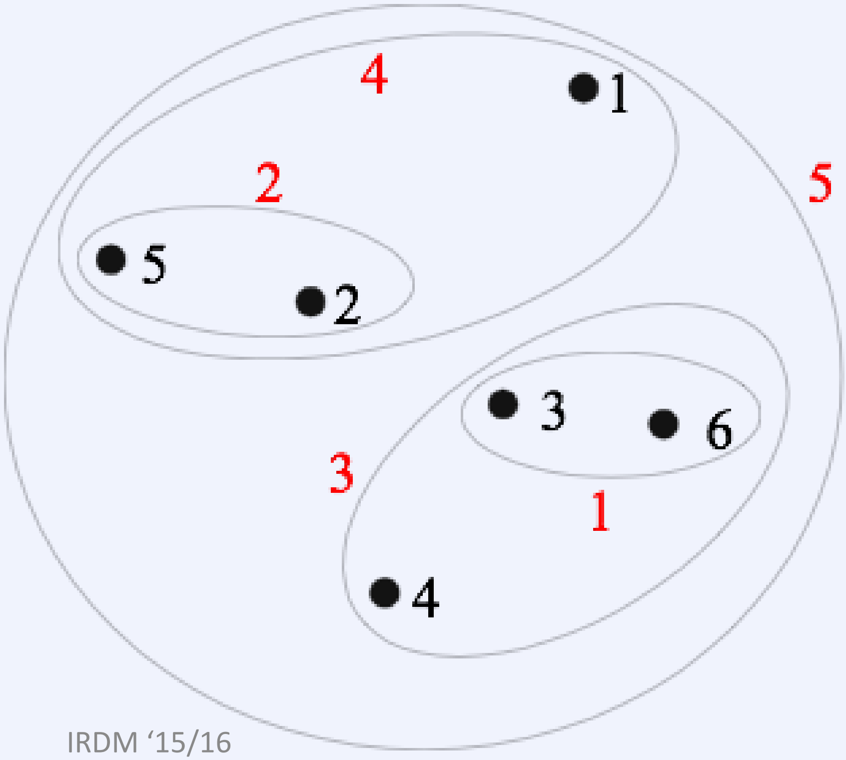
Sensitive to noise and outliers

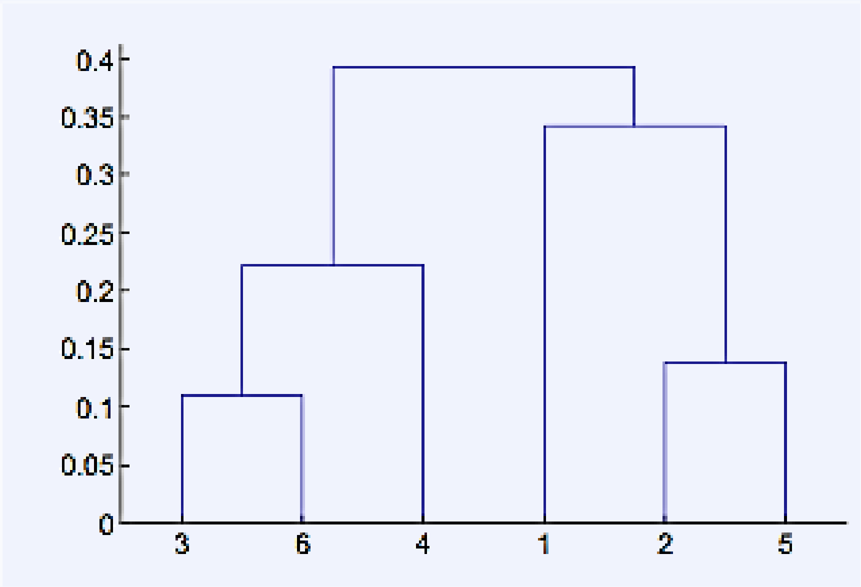
Produces elongated clusters

Complete link

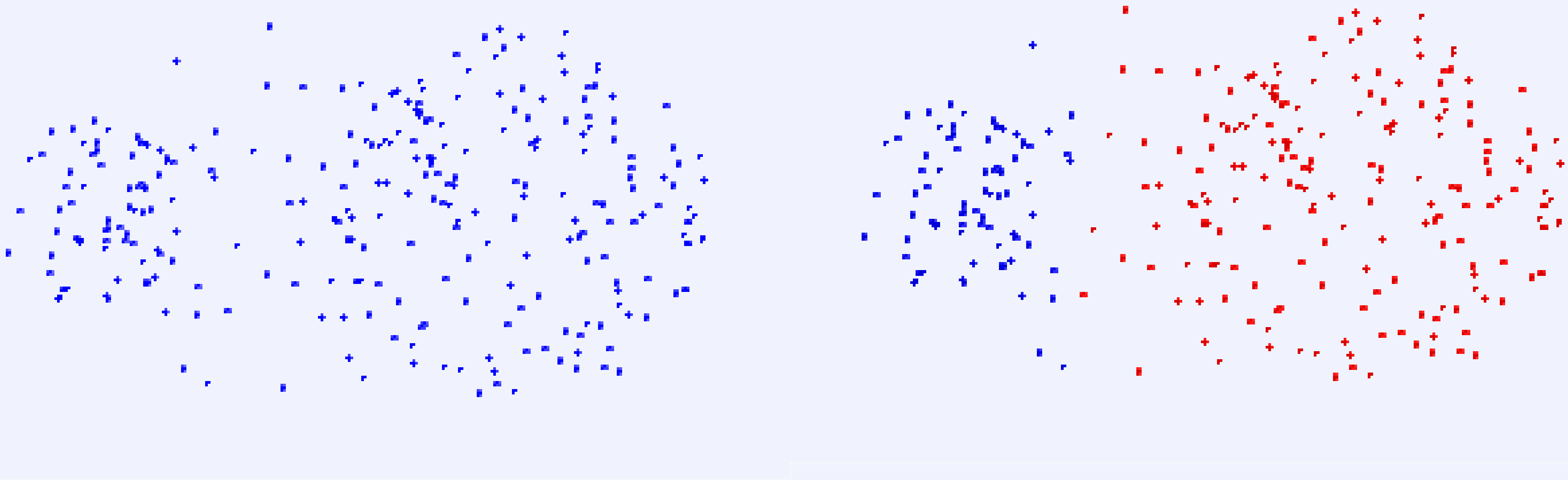
The distance between two clusters is the distance between the furthest points

- $d(B,C) = \max\{d(x,y) : x \in B \text{ and } y \in C\}$





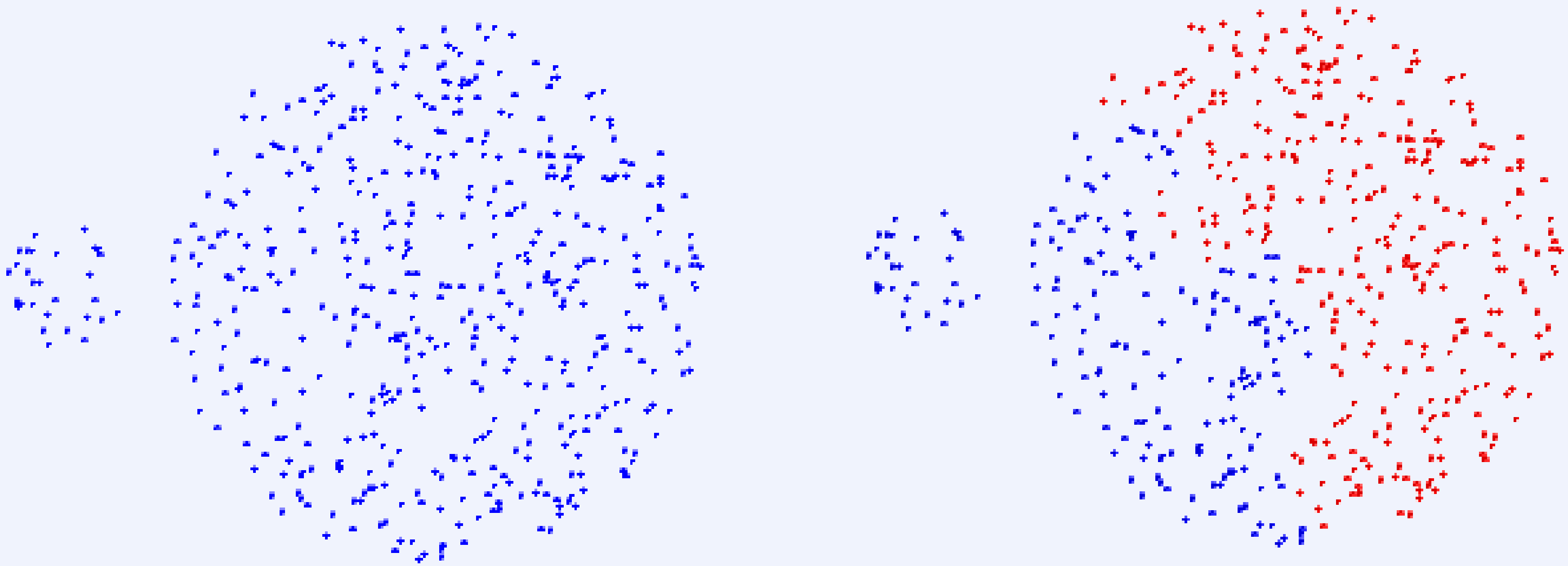
Strengths of complete link



Original Points Two Clusters

Less susceptible to noise and outliers

Weaknesses of complete-link



Breaks largest clusters

Biased towards spherical clusters

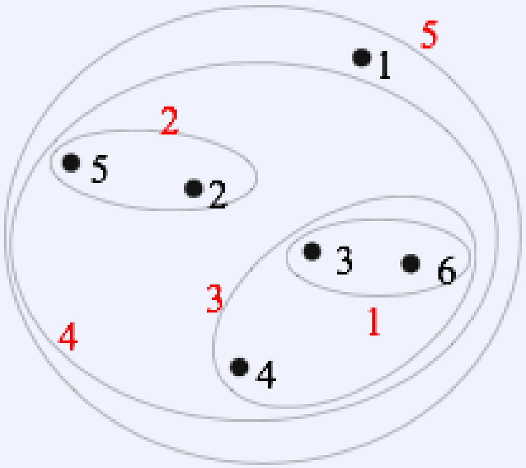
Group average and Mean distance

Group average is the average of pairwise distances

- $d(B,C) = \text{avg}\{d(x,y): x \in B \text{ and } y \in C\} = \sum_{x\in B, y\in C} \frac{d(x,y)}{|B||C|}$

Mean distance is the distance of the cluster centroids

- $d(B,C) = d(\mu_B, \mu_C)$



Properties of group average

A compromise between single and complete link

Less susceptible to noise and outliers

- similar to complete link

Biased towards spherical clusters

- similar to complete link

Ward's method

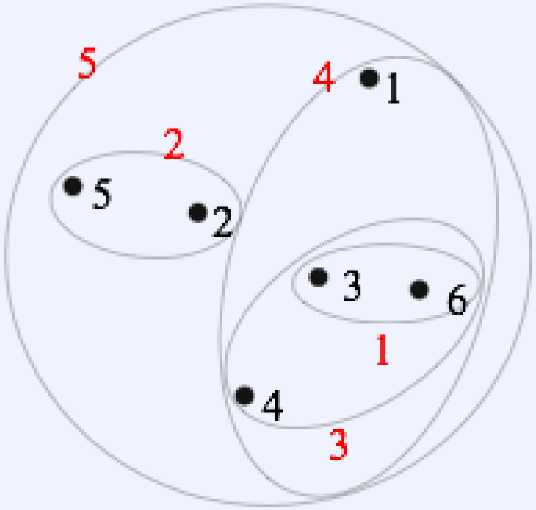
Ward's distance between clusters A and B is the increase in sum of squared errors (SSE) when the two clusters are merged

- SSE for cluster A is $SSE_A = \sum_{x \in A} \|x - \mu_A\|^2$
- difference for merging clusters A and B into cluster C is then

$$d(A,B) = \Delta SSE_C = SSE_C - SSE_A - SSE_B$$

- or, equivalently, weighted mean distance

$$d(A,B) = \frac{|A||B|}{|A|+|B|} \|\mu_{\mathrm{A}} - \mu_B\|^2$$



Discussion on Ward's method

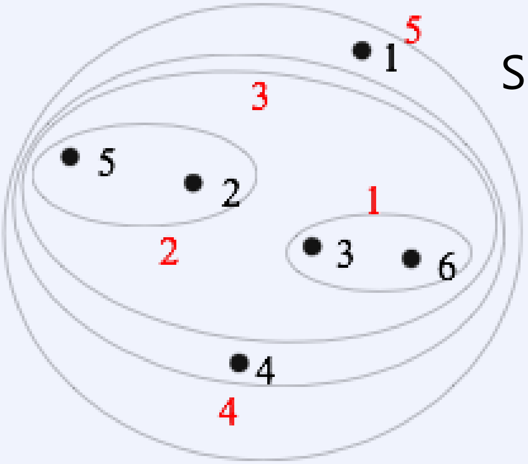
Less susceptible to noise and outliers

Biases towards spherical clusters

Hierarchical analogue of k-means

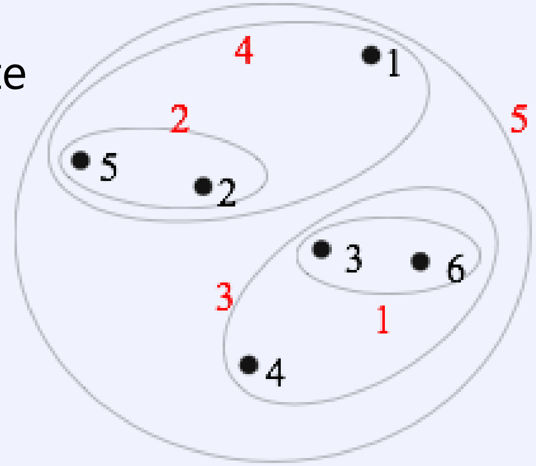
- hence many shared pro's and con's
- can be used to initialise k-means

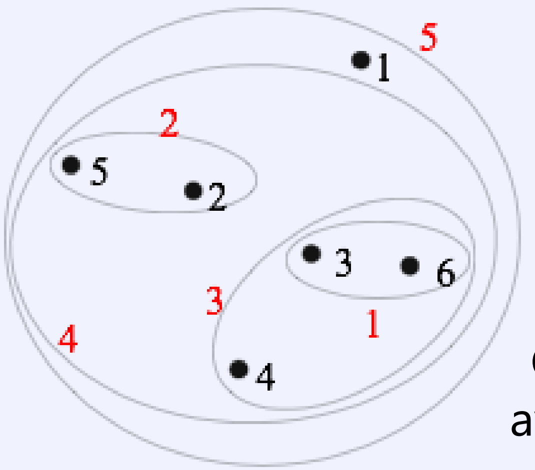
Comparison



Single link

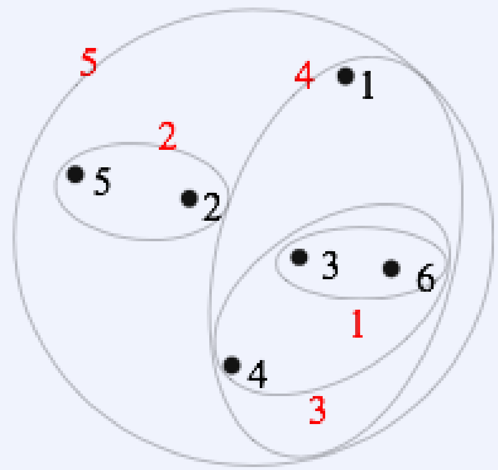
Complete link



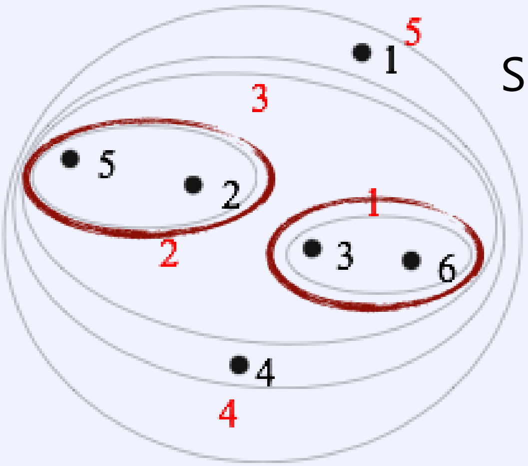


Group average

Ward's method

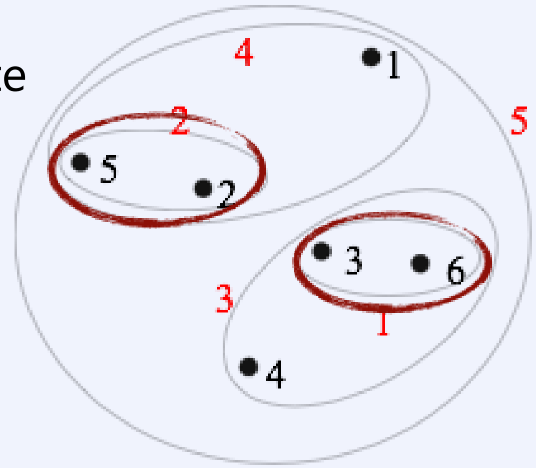


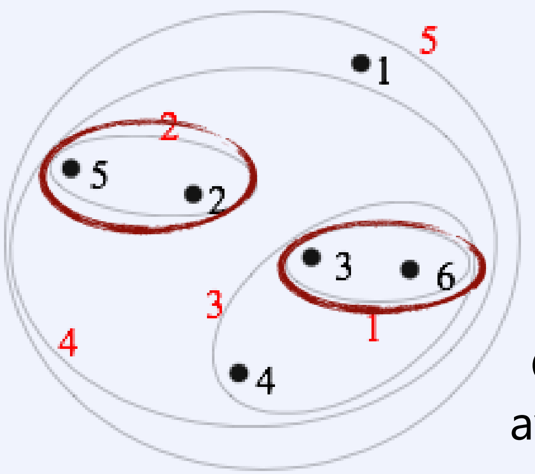
Comparison



Single link

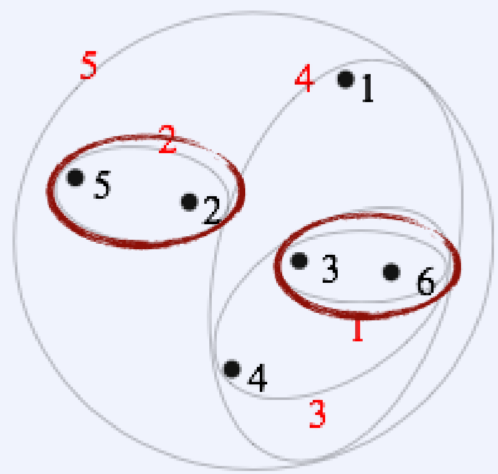
Complete link



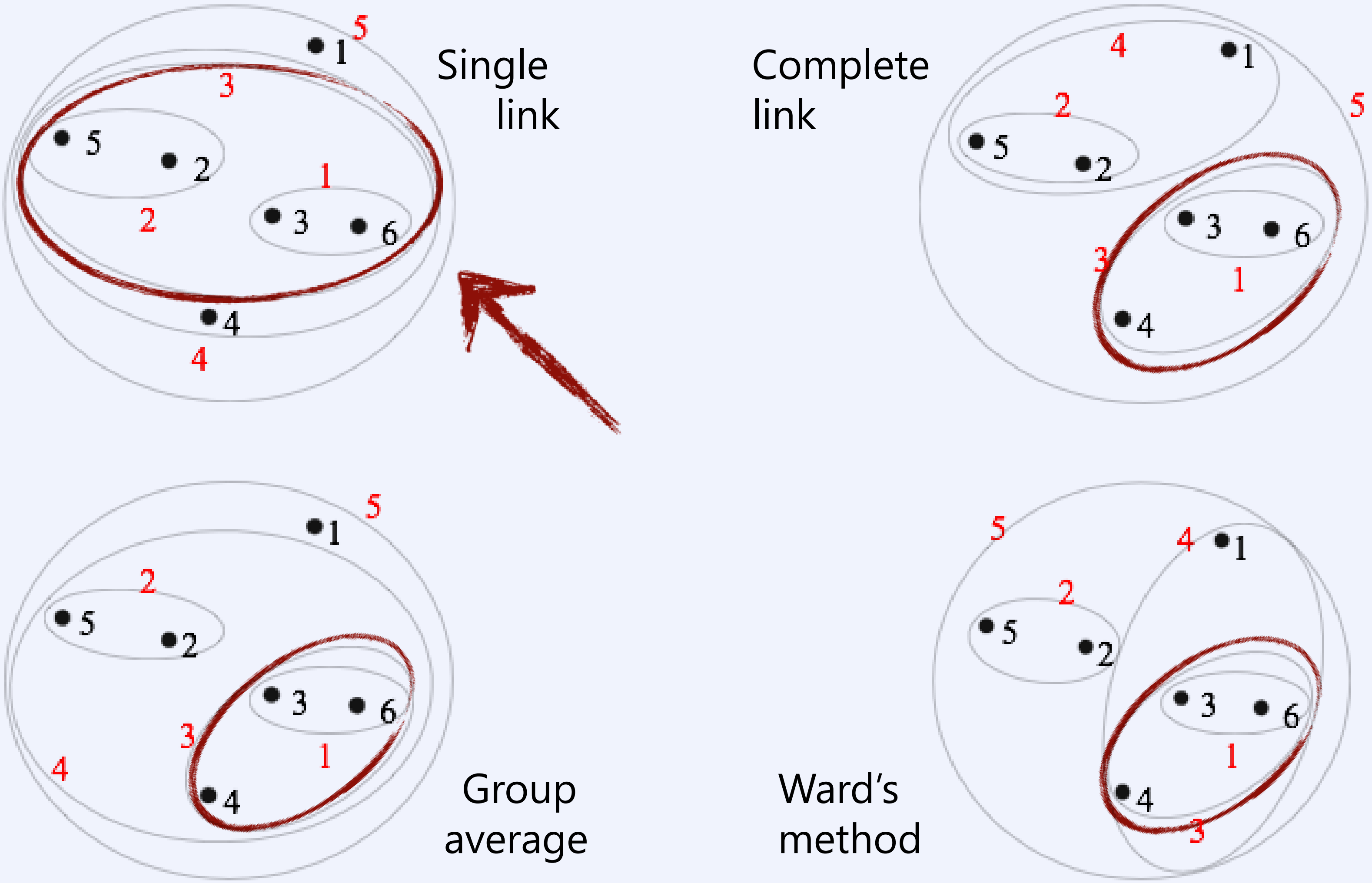


Group average

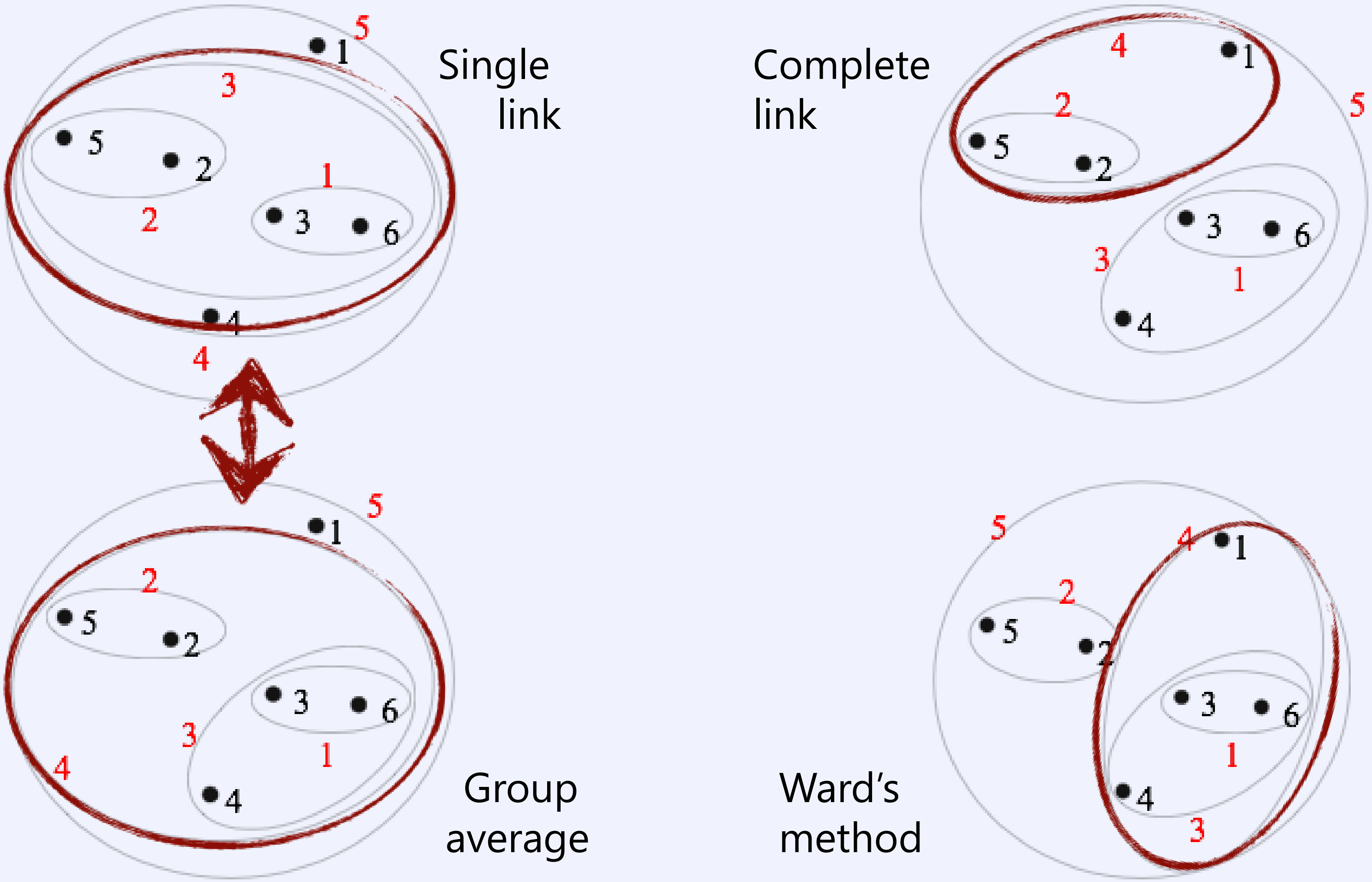
Ward's method



Comparison



Comparison



Lance-Williams formula

After merging clusters A and B into cluster C we need to compute C's distance to another cluster Z. The Lance-Williams formula provides a general equation for this:

$$d(C,Z) = \alpha_A d(A,Z) + \alpha_B d(B,Z) + \beta d(A,B) + \gamma|d(A,Z) - d(B,Z)|$$

	α_A	α_B	β	γ
Single link	$1/2$	$1/2$	0	$-1/2$
Complete link	$1/2$	$1/2$	0	$1/2$
Group average	$\|A\|/(\|A\| + \|B\|)$	$\|B\|/(\|A\| + \|B\|)$	0	0
Mean distance	$\|A\|/(\|A\| + \|B\|)$	$\|B\|/(\|A\| + \|B\|)$	$-\|A\|\|B\|/(\|A\| + \|B\|)2$	0
Ward's method	$(\|A\| + \|Z\|)/(\|A\| + \|B\| + \|Z\|)$	$(\|B\| + \|Z\|)/(\|A\| + \|B\| + \|Z\|)$	$-\|Z\|/(\|A\| + \|B\| + \|Z\|)$	0

Computational complexity

Takes $O(n^3)$ time in most cases

- n steps
- in each step, n^2 distance matrix must be updated and searched

$O(n^2 \log(n))$ time for some approaches that use appropriate data structures

- e.g. keep distances in a heap
- each step takes $O(n \log n)$ time

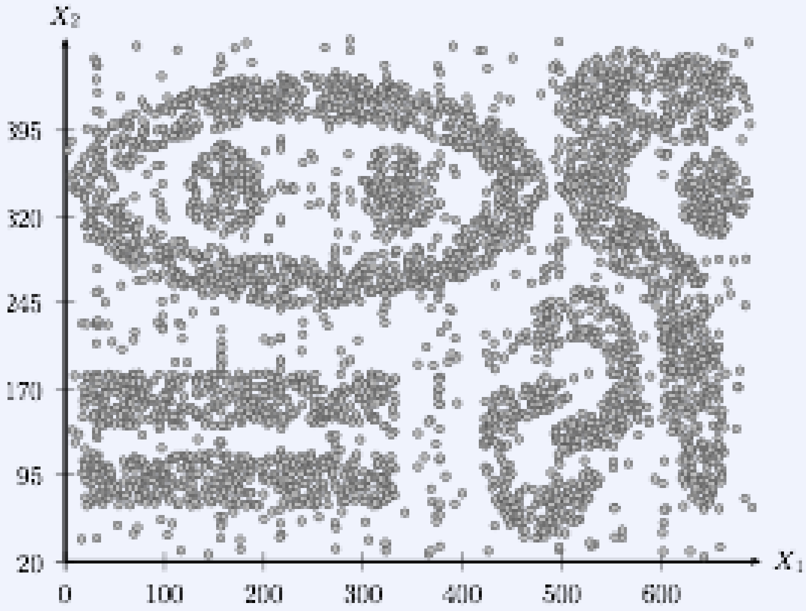
$O(n^2)$ space complexity

- have to store the distance matrix

Chapter 5.6:
Grid and Density-based

Aggarwal Ch. 6.6

The idea



Representation-based clustering can find only convex clusters

- data may contain interesting non-convex clusters

In **density-based clustering** a cluster is a 'dense area of points'

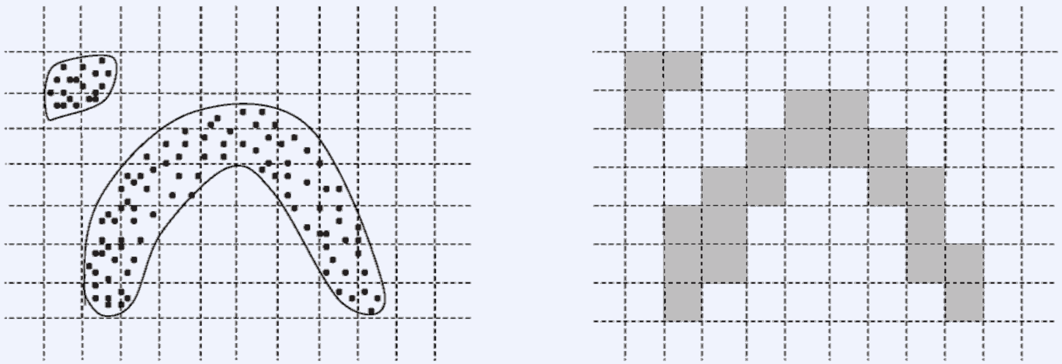
- how to define 'dense area'?

Grid-based Clustering

Algorithm GENERICGRID(data $\boldsymbol{D}$, num-ranges p, min-density τ) :

- discretise each dimension of $\boldsymbol{D}$ into p ranges
- determine those cells with density $\geq \tau$
- create a graph G with a node per dense cell,
  add an edge if the two cells are adjacent
- determine the connected components

return points in each component as a cluster



(a) Data points and grid

(b) Agglomerating adjacent grids

Discussing Grid-based clustering

The Good

- we don't have to specify k
- we can find arbitrarily shaped clusters

The Bad

- we have to specify a global minimal density τ
- only points in dense cells are part of clusters, all points in neighbouring sparse cells are ignored

The Ugly

- we consider only a single, global, rectangular-shaped grid
- number of grid cells increases exponentially with dimensionality

Some definitions

An **ϵ-neighbourhood** of point $\boldsymbol{x}$ of data $\boldsymbol{D}$ is the set of points of $\boldsymbol{D}$ that are within ϵ distance from $\boldsymbol{x}$

- $N_\epsilon(\boldsymbol{x}) = \{\boldsymbol{y} \in \boldsymbol{D}: d(\boldsymbol{x}, \boldsymbol{y}) \leq \epsilon\}$ -- note, we count x aswell!
- parameter ϵ is set by the user

Point $\boldsymbol{x} \in \boldsymbol{D}$ is a **core point** if $|N_\epsilon(\boldsymbol{x})| \geq minpts$

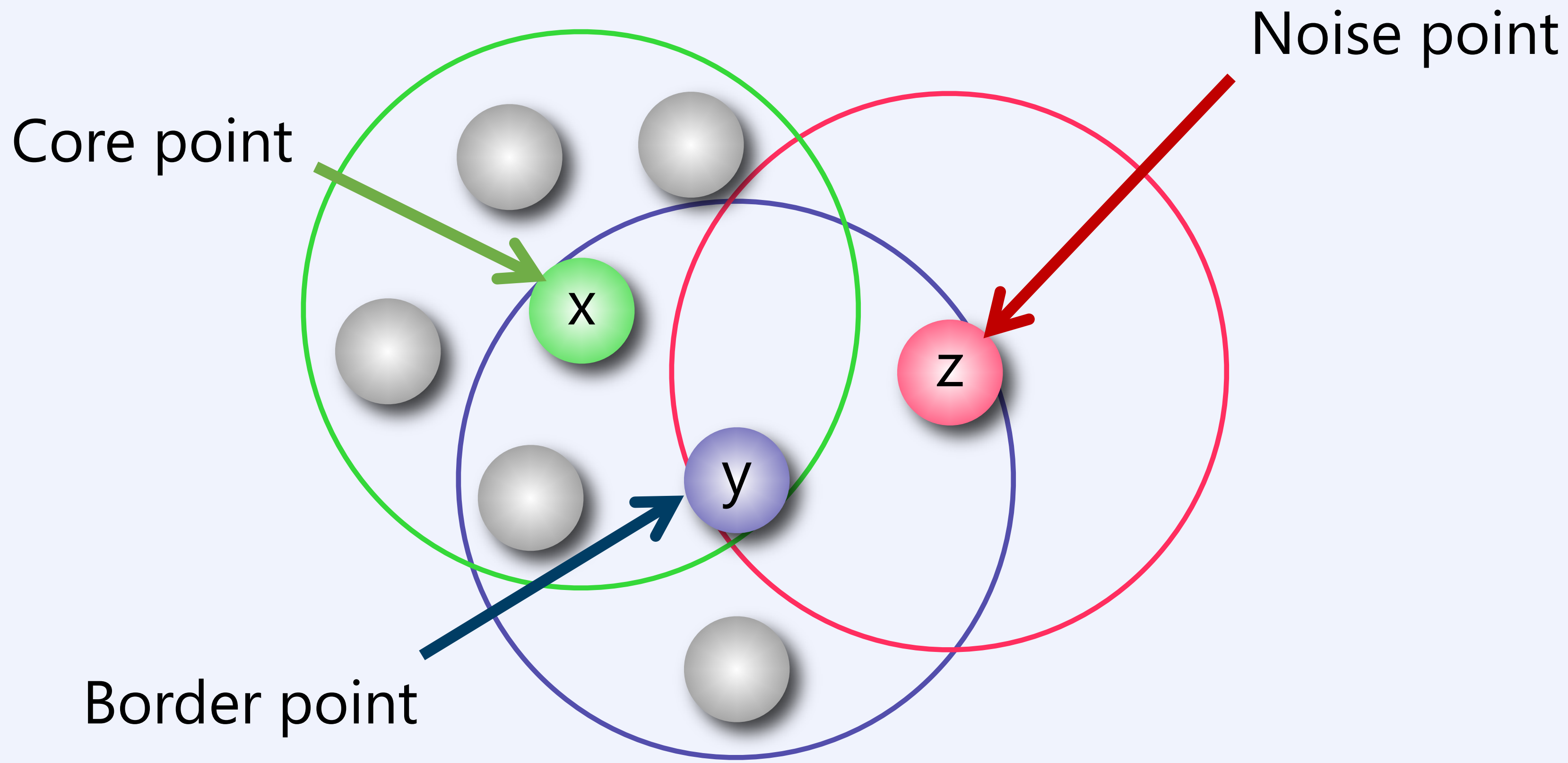
- **minpts (aka** τ) is a user supplied parameter

Point $\boldsymbol{x} \in \boldsymbol{D}$ is a **border point** if it is not a core point, but $\boldsymbol{x} \in N_\epsilon(\boldsymbol{z})$ for some core point $\boldsymbol{z}$

A point $\boldsymbol{x} \in \boldsymbol{D}$ that is neither a core point nor a border point is called a **noise point**

(be aware: some definitions do count a point as a member of its own ϵ-neighborhood, some do not. Here we do.)

Example



minpts = 6

(minpts was 5, now 6 to make clear we count x as an epsilon-neighbor of itself)

Density reachability

Point $\boldsymbol{x} \in \boldsymbol{D}$ is **directly density reachable** from point $\boldsymbol{y} \in \boldsymbol{D}$ if

- $\boldsymbol{y}$ is a core point
- $\boldsymbol{x} \in N_\epsilon(\boldsymbol{y})$

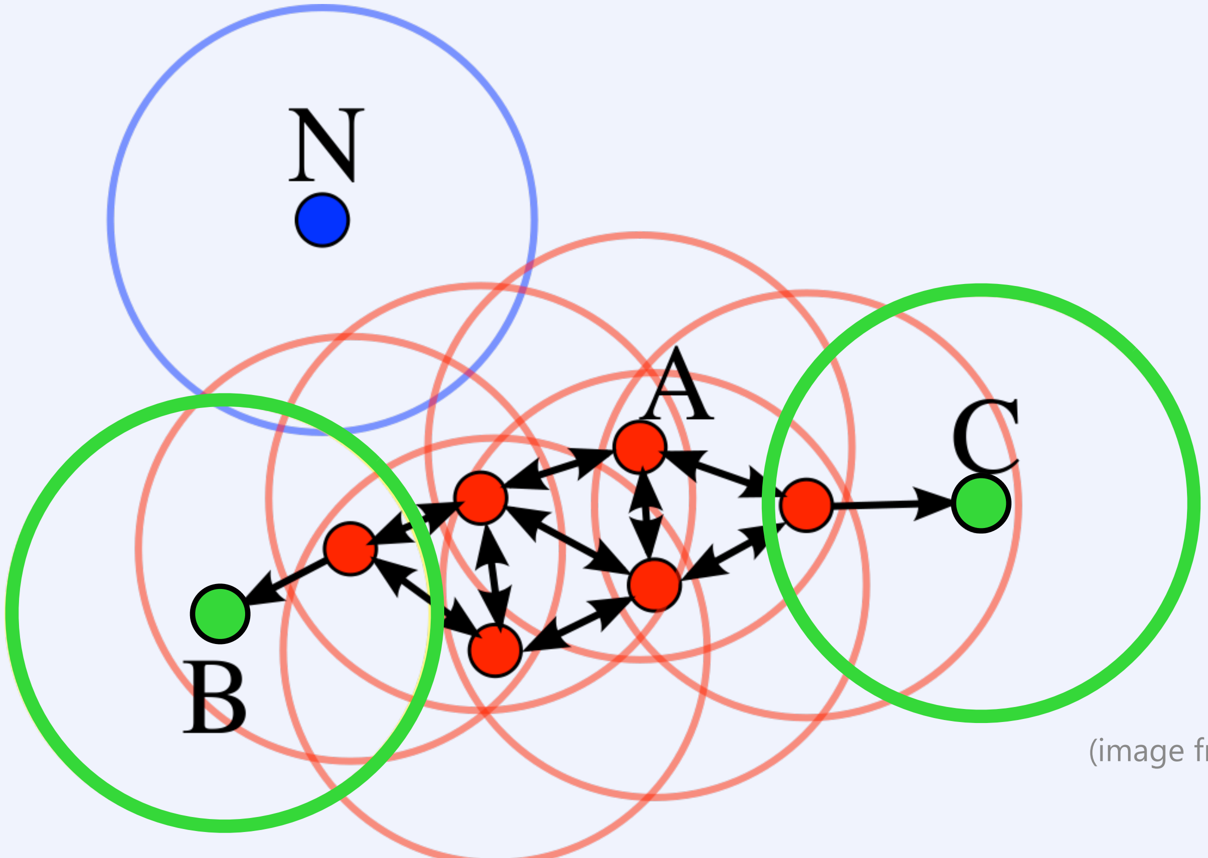
Point $\boldsymbol{x} \in \boldsymbol{D}$ is **density reachable** from point $\boldsymbol{y} \in \boldsymbol{D}$ if there is a chain of points $\boldsymbol{x}_0, \boldsymbol{x}_1, \dots, \boldsymbol{x}_l$ s.t. $\boldsymbol{x} = \boldsymbol{x}_0, \boldsymbol{y} = \boldsymbol{x}_l$, and $\boldsymbol{x}_{\boldsymbol{i-1}}$ is directly density reachable from $\boldsymbol{x}_i$ for all $i = 1, \dots, l$

- not a symmetric relationship (!)

Points $\boldsymbol{x}, \boldsymbol{y} \in \boldsymbol{D}$ are **density connected** if there exists a core point $\boldsymbol{z}$ s.t. both $\boldsymbol{x}$ and $\boldsymbol{y}$ are density reachable from $\boldsymbol{z}$

Density-based clusters

A **density-based cluster** is a maximal set of density connected points



(image from Wikipedia)

The DBSCAN algorithm

- **for each** unvisited point $\boldsymbol{x}$ in the data
  - compute $N_\epsilon(\boldsymbol{x})$
  - **if** $|N_\epsilon(\boldsymbol{x})| \geq \mathbf{minpts}$
    - EXPANDCLUSTER(x, $++\text{clusterID}$)

- EXPANDCLUSTER($\boldsymbol{x}$, ID)
  - assign $\boldsymbol{x}$ to cluster ID and set $\text{N} \leftarrow N_\epsilon(\boldsymbol{x})$
  - **for each** $\boldsymbol{y} \in N$
    - **if** $\boldsymbol{y}$ is not visited and $|N_\epsilon(\boldsymbol{y})| \geq \mathbf{minpts}$
      - $N \leftarrow N \cup N_\epsilon(\boldsymbol{y})$
    - **if** $\boldsymbol{y}$ does not belong to any cluster
      - assign $\boldsymbol{y}$ to cluster ID

More on DBSCAN

DBSCAN can return either overlapping or non-overlapping clusters

- ties are broken arbitrarily

The main time complexity comes from computing the neighborhoods

- total $O(n \log n)$ with spatial index structures
  - won't work with high dimensions, worst case is $O(n^2)$

With the neighborhoods known, DBSCAN only needs a single pass over the data

The parameters

DBSCAN requires two parameters, ϵ and $\mathbf{minpts}$

$\mathbf{minpts}$ controls the minimum size of a cluster

- $\mathbf{minpts} = 1$ allows singleton clusters
- $\mathbf{minpts} = 2$ makes DBSCAN essentially a single-link clustering
- higher values avoid the long-and-narrow clusters of single link

ϵ controls the required density

- a single ϵ is not enough if the clusters are of very different density

Chapter 5.7: More Clustering Models

Aggarwal Ch. 6.7-6.8

More clustering models

So far we've seen

- representative-based clustering
- model-based clustering
- hierarchical clustering
- density-based clustering

There are many more types of clustering, including

- co-clustering
- graph clustering (Aggarwal Ch. 6.8)
- non-negative matrix factorisation (NMF) (Aggarwal Ch. 6.9)

But we're not going to discuss these in IRDM.

- phew!

Chapter 5.8:
Clustering High-Dimensional Data

Aggarwal Ch. 7.4—7.4.2

Clustering High Dimensional Data

If we compute similarity over many dimensions,
all points will be roughly equi-distant.

There exist no clusters over many dimensions.

- or, are there?

Of course there are!

- data can have a much lower intrinsic dimensionality (SVD)
  i.e. many dimensions are noisy, irrelevant, or copies
- data can have clusters embedded in **subsets of its dimensions**

Spaces

The **full space** of data $\boldsymbol{D}$ is its set of attributes $\mathcal{A}$

A **subspace** S of $\boldsymbol{D}$ is a subset of $\mathcal{A}$, i.e. $S \subseteq \mathcal{A}$

- there exist $2^{|\mathcal{A}|} - 1$ non-empty subspaces

A **subspace cluster** is a cluster C over a subspace S

- a group of points that is highly similar over subspace S

High-dimensional Grids

In full-dimensional grid-based methods, the grid cells are determined on the intersection of the discretization ranges p across **all** dimensions.

What happens for high-dimensional data?

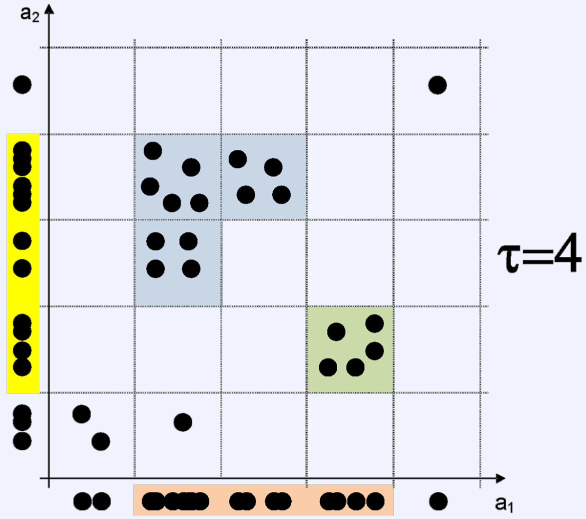
- many many grid cells will be empty

CLIQUE is a generalisation of grid-based clustering to subspaces. In CLIQUE the ranges are determined over only **a subset of dimensions** with density greater than τ.

CLustering In QUEst

CLIQUE is the first subspace clustering algorithm.

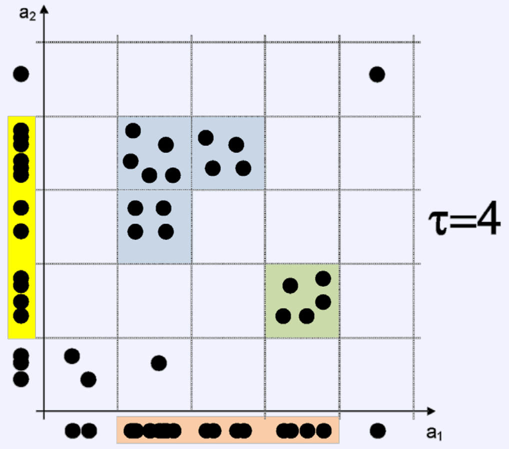
- partition each dimension into p ranges
- for each subspace we now have grid cells of the same volume
- subspace clusters are connected dense cells in the grid

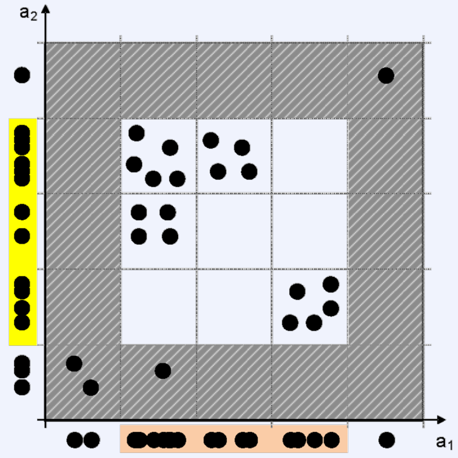


(Agrawal et al. 1998)

Finding dense cells

CLIQUE uses anti-monotonicity to find dense grid cells in subspaces: the higher the dimensionality, the sparser the cells

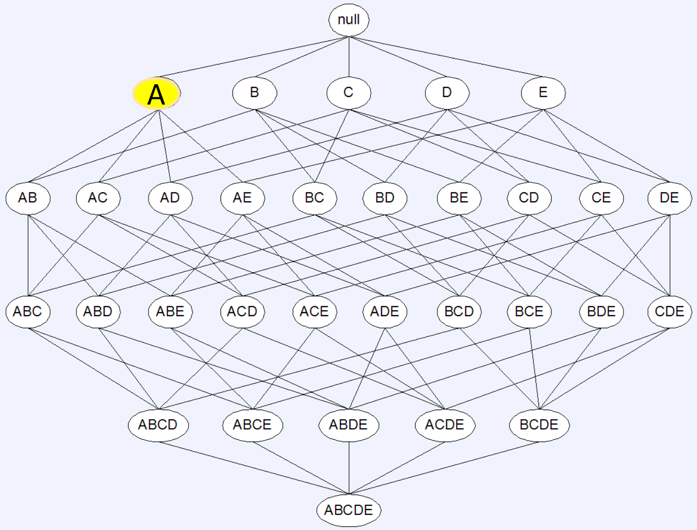




Main Idea:

- every subspace we consider is a ‘transaction database’, every cell is then a ‘transaction’. If a cell is τ-dense, the subspace ‘itemset’ has been ‘bought’.
- we now mine frequent itemsets with minsup=1

Example

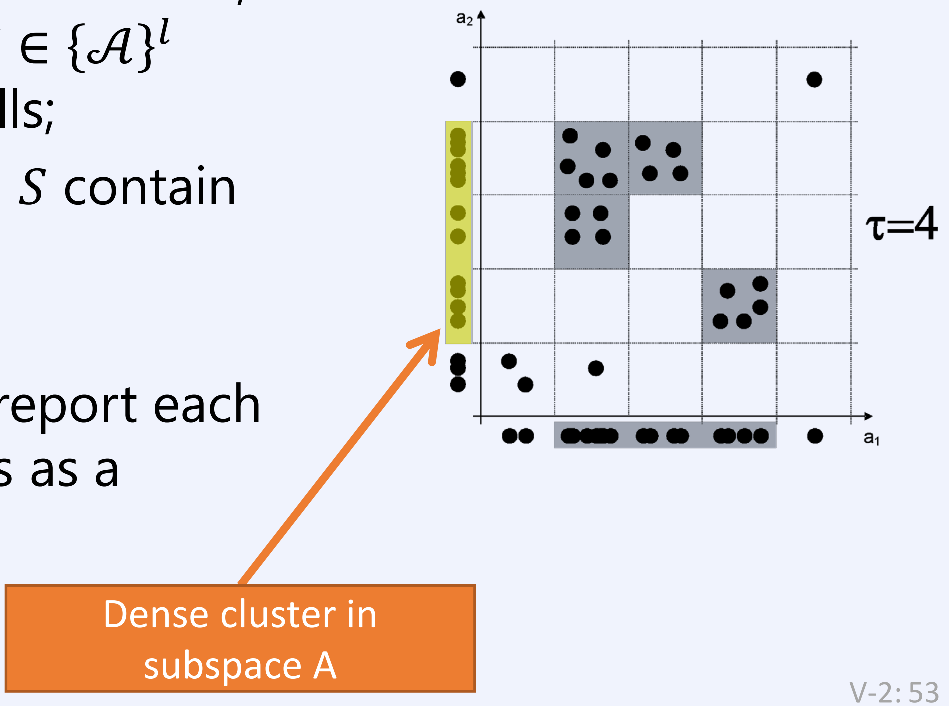


A-priori for subspace clusters:

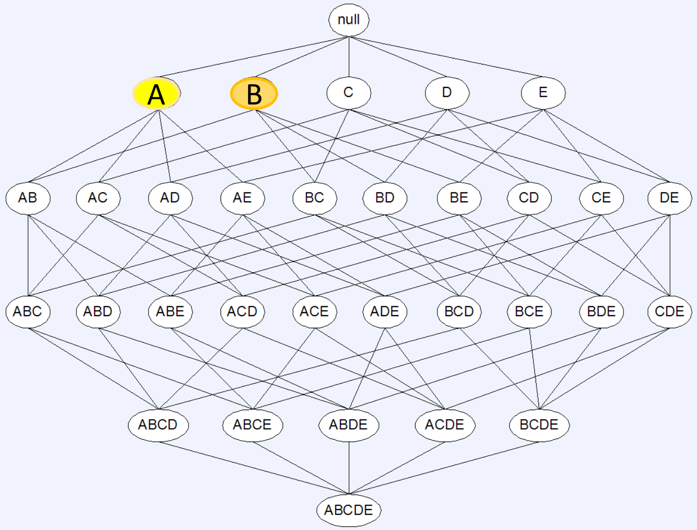
For every level l in the subspace lattice, we check, for all subspaces $S \in \{\mathcal{A}\}^l$ whether S contains dense cells;

but only if all subspaces $S' \subset S$ contain dense cells.

If S contains dense cells, we report each group of adjacent dense cells as a cluster C over subspace S



Example

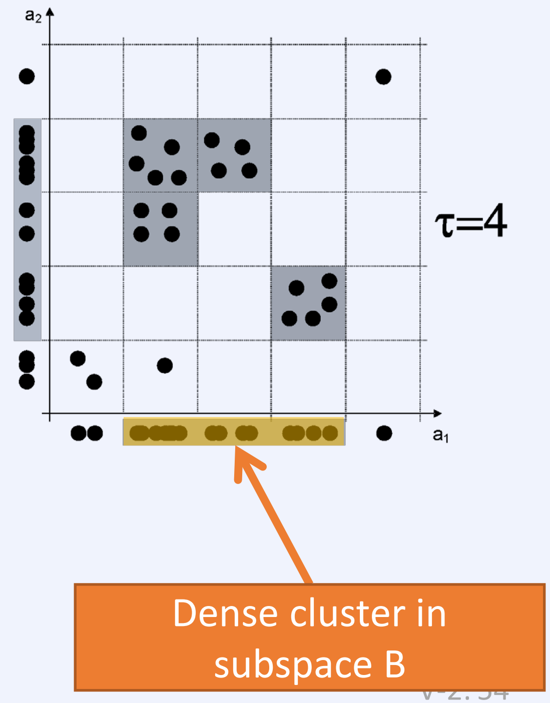


A-priori for subspace clusters:

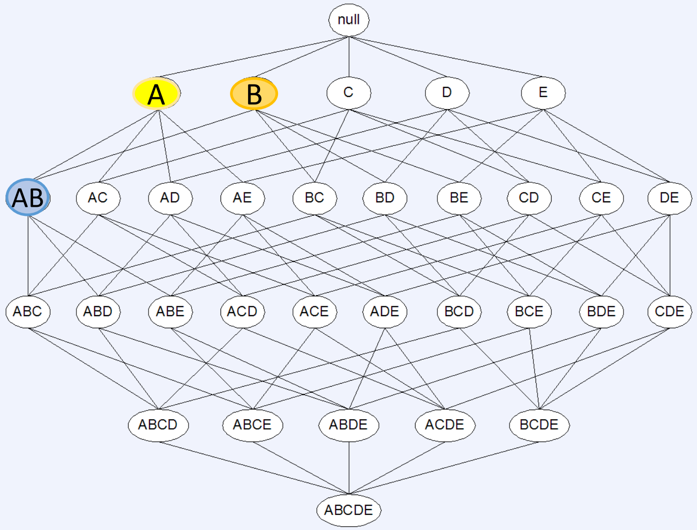
For every level l in the subspace lattice, we check, for all subspaces $S \in \{\mathcal{A}\}^l$ whether S contains dense cells;

but only if all subspaces $S' \subset S$ contain dense cells.

If S contains dense cells, we report each group of adjacent dense cells as a cluster C over subspace S



Example

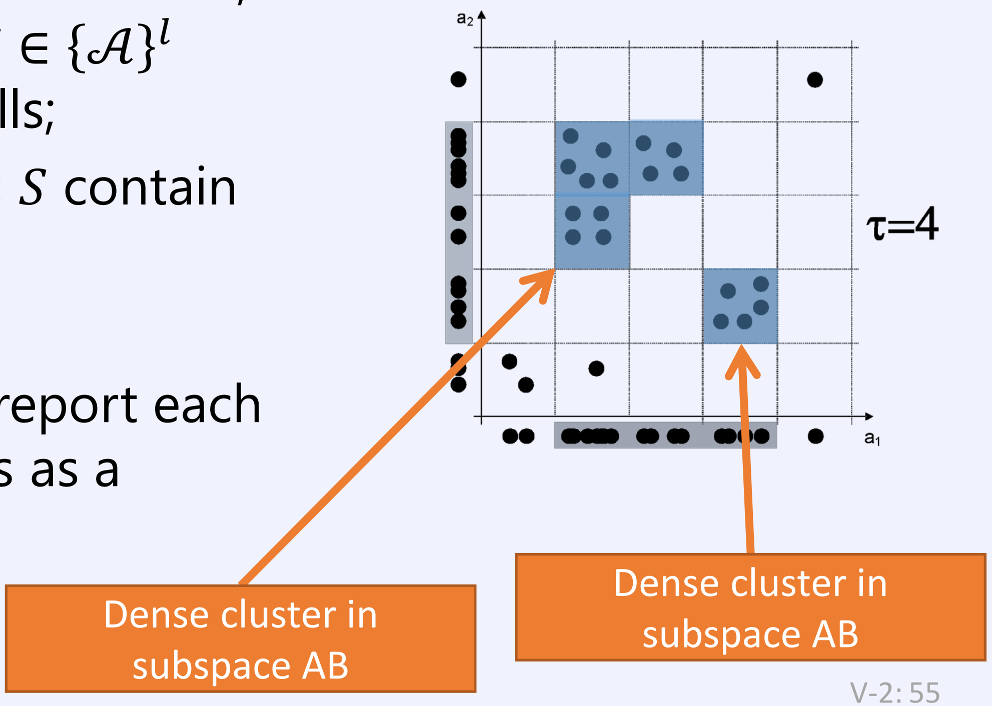


A-priori for subspace clusters:

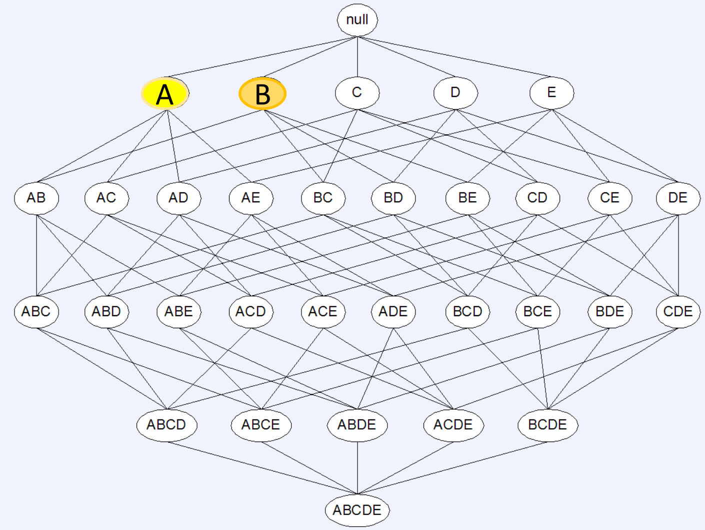
For every level l in the subspace lattice, we check, for all subspaces $S \in \{\mathcal{A}\}^l$ whether S contains dense cells;

but only if all subspaces $S' \subset S$ contain dense cells.

If S contains dense cells, we report each group of adjacent dense cells as a cluster C over subspace S



Example

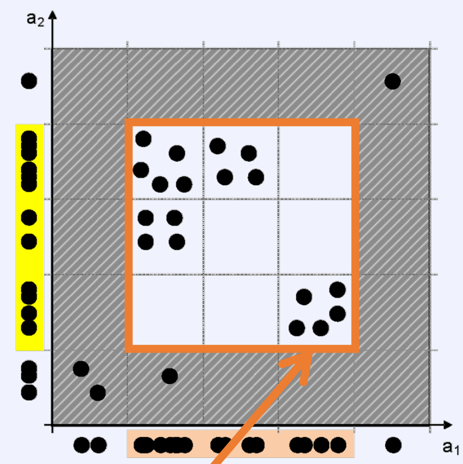


A-priori for subspace clusters:

For every level l in the subspace lattice, we check, for all subspaces $S \in \{\mathcal{A}\}^l$ whether S contains dense cells;

but only if all subspaces $S' \subset S$ contain dense cells.

If S contains dense cells, we report each group of adjacent dense cells as a cluster C over subspace S



To find dense clusters in a subspace, we **only** have to consider grid cells that are dense in all super-spaces

Discussion of CLIQUE

CLIQUE was the first subspace clustering algorithm.

- and it shows

It produces an enormous amount of clusters

- just like frequent itemset mining
- nothing like 'a summary of your data'

This, however, is general problem of subspace clustering

- there are exponentially many subspaces
- and for each subspace there are exponentially many clusters

Conclusions

Clustering is one of the most important and most used data analysis methods

There exist many different types of clustering

- we've seen representative, hierarchical, probabilistic, and density-based

Analysis of clustering methods is often difficult

Always think what you're doing if you use clustering

- in fact, just always think what you're doing

Thank you!

Clustering is one of the most important and most used data analysis methods

There exist many different types of clustering

- we've seen representative, hierarchical, probabilistic, and density-based

Analysis of clustering methods is often difficult

Always think what you're doing if you use clustering

- in fact, just always think what you're doing