CS 2750 Machine Learning
Lecture 19

Clustering

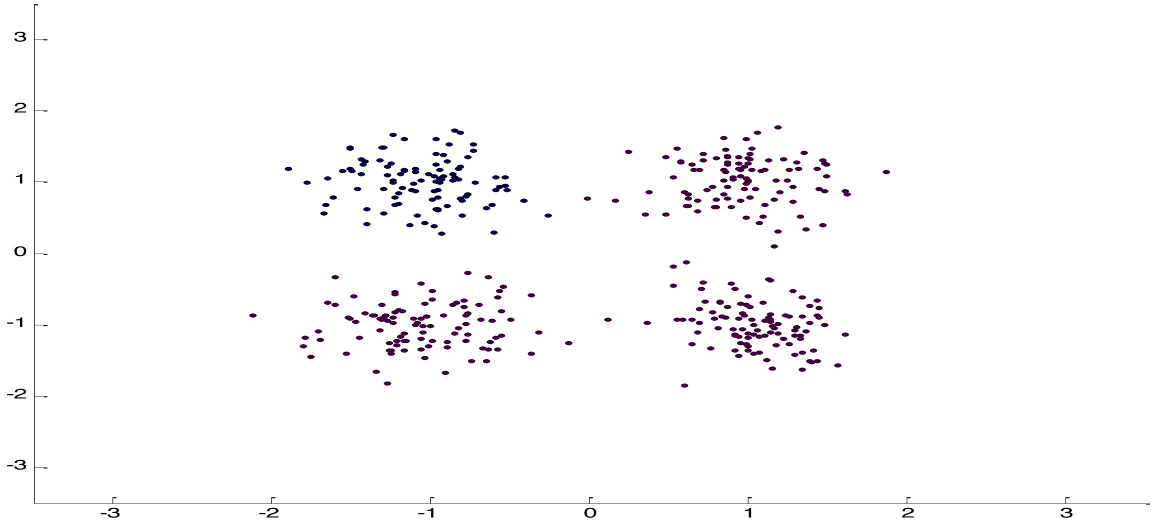
Milos Hauskrecht
milos@pitt.edu
5329 Sennott Square

Clustering

Groups together "similar" instances in the dataset

Basic clustering problem:

- distribute data into *k* different groups such that data points **similar** to each other are in the same group
- **Similarity** between data points is typically defined in terms of some similarity measure or a distance metric

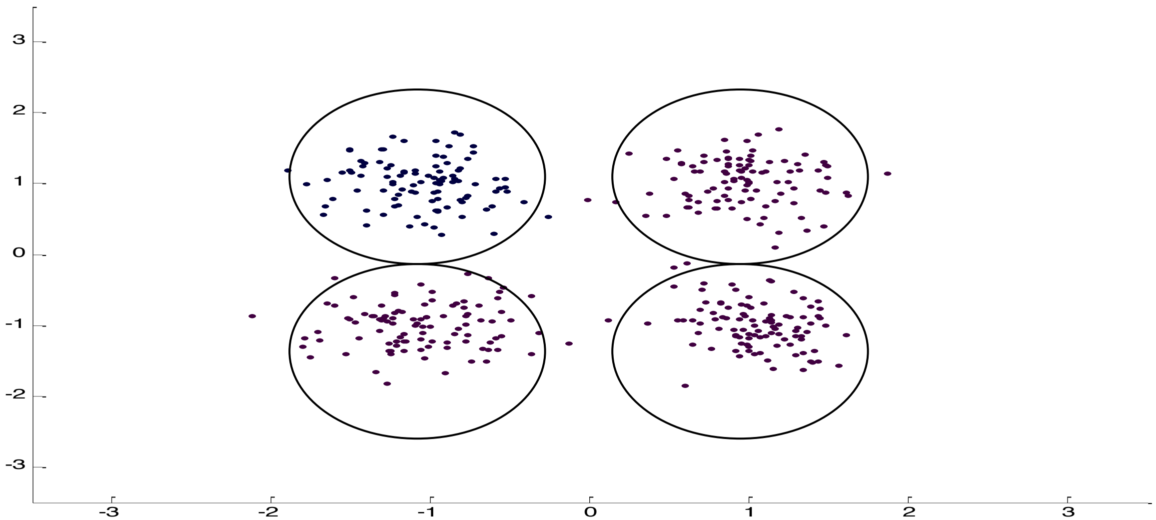


Clustering

Groups together "similar" instances in the dataset

Basic clustering problem:

- distribute data into *k* different groups such that data points **similar** to each other are in the same group
- **Similarity** between data points is typically defined in terms of some distance metric (can be chosen)



Clustering example

- **Clustering.** Group together similar examples in the dataset
- Clustering could be applied to different types of data instances

Clustering example

- **Clustering.** Group together similar examples in the dataset
- Clustering could be applied to different types of data instances

Patient #	**Age**	**Sex**	**Heart Rate**	**Blood pressure …**
Patient 1	55	M	85	125/80
Patient 2	62	M	87	130/85
Patient 3	67	F	80	126/86
Patient 4	65	F	90	130/90
Patient 5	70	M	84	135/85

Clustering example

- **Clustering.** Group together similar examples in the dataset
- Clustering could be applied to different types of data instances

Patient #	Age	Sex	Heart Rate	Blood pressure …
Patient 1	55	M	85	125/80
Patient 2	62	M	87	130/85
Patient 3	67	F	80	126/86
Patient 4	65	F	90	130/90
Patient 5	70	M	84	135/85

Key question: How to define similarity between instances?

Similarity and dissimilarity measures

- **Dissimilarity measure**
  - Numerical measure of how different two data objects are
  - Often expressed in terms **of a distance metric**
  - **Example:** Euclidean:

$$d(a,b)=\sqrt{\sum_{i=1}^{k}(a_i-b_i)^2}$$

- **Similarity measure**
  - Numerical measure of how alike two data objects are
  - **Examples:**
    - Gaussian kernel:

$$K(a,b)=\frac{1}{\left(2\pi h^2\right)^{d/2}}\exp\left[-\frac{\|a-b\|_2^2}{2h^2}\right]$$

    - Cosine similarity: $K(a,b)=a^T b$

Distance metrics

Dissimilarity is often measured with the help of a distance metrics.

Properties of distance metrics:

Assume 2 data entries *a, b*

Positiveness: $d(a,b) \geq 0$

Symmetry: $d(a,b) = d(b,a)$

Identity: $d(a,a) = 0$

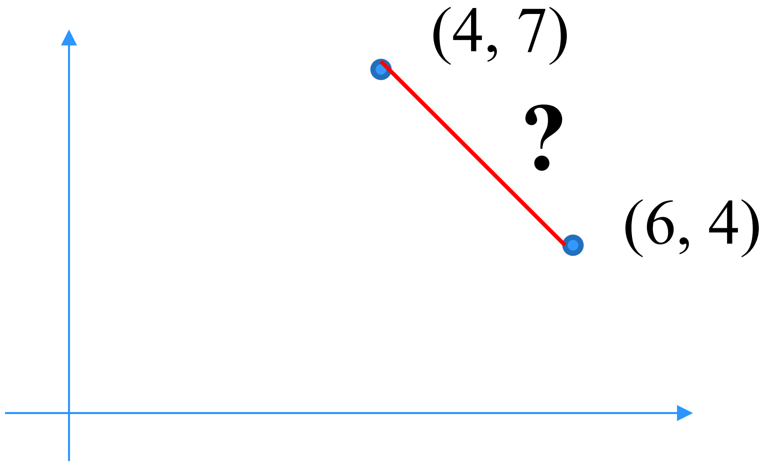
Triangle inequality: $d(a,c) \leq d(a,b) + d(b,c)$

Distance metrics

Assume 2 real-valued data-points:

a=(6, 4)

b=(4, 7)



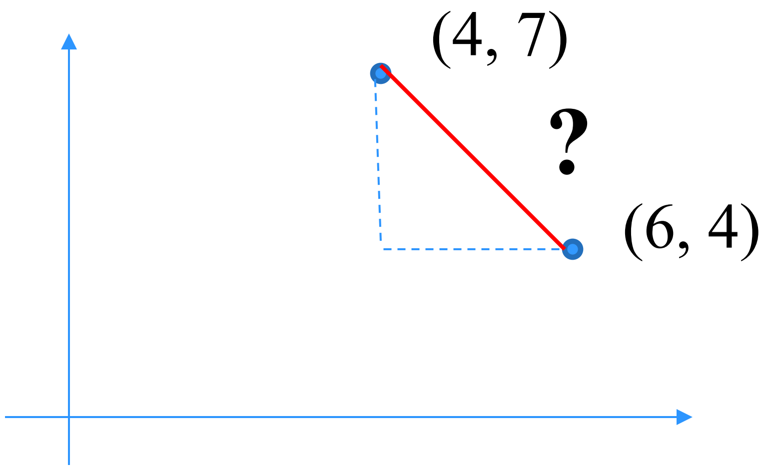
What distance metric to use?

Distance metrics

Assume 2 real-valued data-points:

a=(6, 4)

b=(4, 7)



What distance metric to use?

Euclidian:

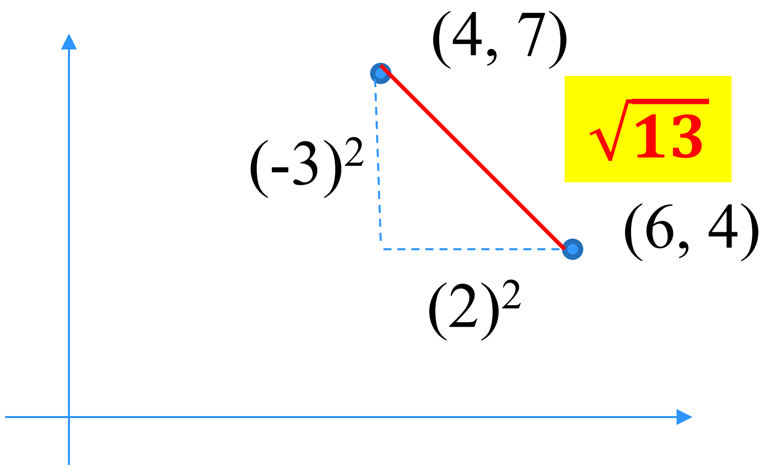
$$d(a,b) = \sqrt{\sum_{i=1}^{k} (a_i - b_i)^2}$$

Distance metrics

Assume 2 real-valued data-points:

a=(6, 4)

b=(4, 7)



What distance metric to use?

Euclidian:

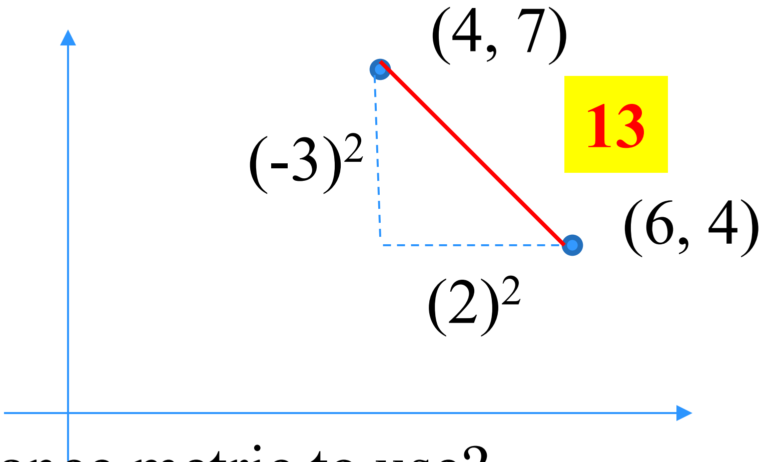
$$d(a,b) = \sqrt{\sum_{i=1}^{k} (a_i - b_i)^2}$$

Distance metrics

Assume 2 real-valued data-points:

a=(6, 4)

b=(4, 7)



What distance metric to use?

Squared Euclidian: works for an arbitrary k-dimensional space

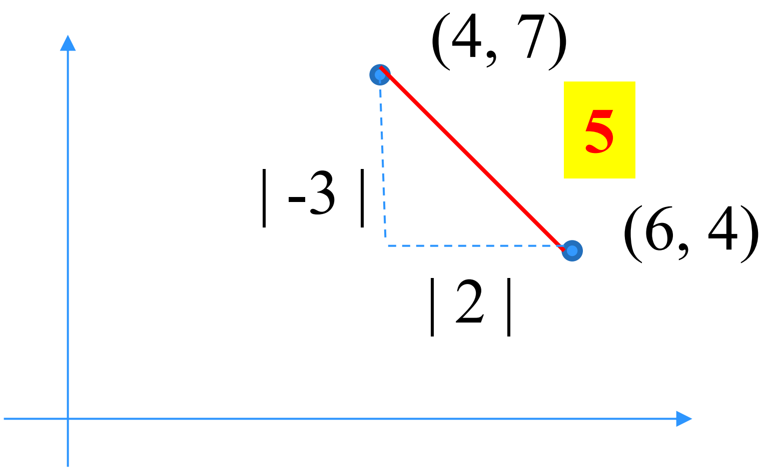
$$d^2(a,b) = \sum_{i=1}^{k} (a_i - b_i)^2$$

Distance metrics

Assume 2 real-valued data-points:

a=(6, 4)

b=(4, 7)



Manhattan distance:

works for an arbitrary k-dimensional space

$$d(a,b) = \sum_{i=1}^{k} | a_i - b_i |$$

Γ^{-1}

Distance measures

Generalized distance metric:

$$d^2(\mathbf{a},\mathbf{b}) = (\mathbf{a}-\mathbf{b})^T \boldsymbol{\Gamma}^{-1}(\mathbf{a}-\mathbf{b})$$

Γ semi-definite positive matrix

Γ^{-1} is a matrix that weights attributes proportionally to their importance. Different weights lead to a different distance metric.

If $\Gamma = I$ we get **squared Euclidean**

$\Gamma = \Sigma$ (covariance matrix) – we get the **Mahalanobis distance** that takes into account correlations among attributes

Γ^{-1}

Distance measures

Generalized distance metric:

$$d^2(\mathbf{a},\mathbf{b}) = (\mathbf{a}-\mathbf{b})^T \boldsymbol{\Gamma}^{-1}(\mathbf{a}-\mathbf{b})$$

Special case: $\Gamma = I$ we get **squared Euclidean**

Example:

$$\mathbf{a} = \begin{bmatrix} 6 \\ 4 \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} 4 \\ 7 \end{bmatrix} \qquad \Gamma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \Gamma^{-1}$$

$$d^2(\mathbf{a},\mathbf{b}) = \begin{bmatrix} 2 & -3 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ -3 \end{bmatrix} = 2^2 + (-3)^2 = 13$$

Γ^{-1}

Distance measures

Generalized distance metric:

$$d^2(\mathbf{a},\mathbf{b}) = (\mathbf{a}-\mathbf{b})^T \mathbf{\Gamma}^{-1} (\mathbf{a}-\mathbf{b})$$

Special case: $\Gamma = \Sigma$ defines **Mahalanobis distance**

Example: Assume dimensions are independent in data

Covariance matrix **Inverse covariance**

$$\Sigma = \begin{pmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{pmatrix} \qquad \Sigma^{-1} = \begin{pmatrix} \frac{1}{\sigma_1^2} & 0 \\ 0 & \frac{1}{\sigma_2^2} \end{pmatrix}$$

$$d^2(\mathbf{a},\mathbf{b}) = \begin{bmatrix} 2 & -3 \end{bmatrix} \begin{bmatrix} \frac{1}{\sigma_1^2} & 0 \\ 0 & \frac{1}{\sigma_2^2} \end{bmatrix} \begin{bmatrix} 2 \\ -3 \end{bmatrix} = \frac{2^2}{\sigma_1^2} + \frac{(-3)^2}{\sigma_2^2}$$

Contribution of each dimension to the squared Euclidean is normalized (rescalled) by the variance of that dimension

Distance measures

Assume categorical data where integers represent the different categories:

0	1	1	0	0
1	0	3	0	1
2	1	1	0	2
1	1	1	1	2
…				

What distance metric to use?

Distance measures

Assume categorical data where integers represent the different categories:

0	1	1	0	0
1	0	3	0	1
2	1	1	0	2
1	1	1	1	2
…				

What distance metric to use?

Hamming distance: The number of values that need to be changed to make them the same

Distance measures.

Assume pure binary values data:

0	1	1	0	1
1	0	1	0	1
0	1	1	0	1
1	1	1	1	1
…				

One metric is the **Hamming distance:** The number of bits that need to be changed to make the entries the same

How about squared Euclidean?

$$d^2(a,b) = \sum_{i=1}^{k} (a_i - b_i)^2$$

Distance measures.

Assume pure binary values data:

0	1	1	0	1
1	0	1	0	1
0	1	1	0	1
1	1	1	1	1

…

One metric is the **Hamming distance:** The number of bits that need to be changed to make the entries the same

How about the squared Euclidean?

$$d^2(a,b) = \sum_{i=1}^{k} (a_i - b_i)^2$$

The same as Hamming distance

Distance measures

Combination of real-valued and categorical attributes

Patient #	Age	Sex	Heart Rate	Blood pressure …
Patient 1	55	M	85	125/80
Patient 2	62	M	87	130/85
Patient 3	67	F	80	126/86
Patient 4	65	F	90	130/90
Patient 5	70	M	84	135/85

What distance metric to use?

Distance measures

Combination of real-valued and categorical attributes

Patient #	Age	Sex	Heart Rate	Blood pressure …
Patient 1	55	M	85	125/80
Patient 2	62	M	87	130/85
Patient 3	67	F	80	126/86
Patient 4	65	F	90	130/90
Patient 5	70	M	84	135/85

What distance metric to use? **Solutions:**

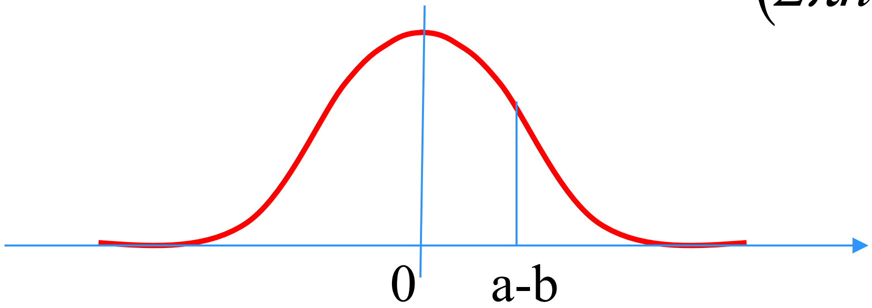
- **A weighted sum approach: e.g. a mix of Euclidian and Hamming distances for subsets of attributes**
- **Generalized distance metric (a weighted combination, use one-hot representation of categories)**

More complex solutions: tensors and decompositions

Distance metrics and similarity

- Dissimilarity/distance measure
- **Similarity measure**
  - Numerical measure of how alike two data objects are
  - Do not have to satisfy the properties like the ones for the distance metric
  - **Examples:**
    - Cosine similarity: $K(a,b) = a^T b$
    - Gaussian kernel:

$$K(a,b) = \frac{1}{\left(2\pi h^2\right)^{d/2}} \exp\left[-\frac{\| a-b \|_2^2}{2h^2}\right]$$



Clustering

Clustering is useful for:

- **Similarity/dissimilarity analysis**

  Analyze what data points in the data are close to each other
- **Dimensionality reduction**

  High dimensional data replaced with a group (cluster) label
- **Data reduction:** Replaces many data-points with a point representing the group mean

Challenges:

- How to measure similarity (problem/data specific)?
- How to choose the number of groups?
  - Many clustering algorithms require us to provide the number of groups ahead of time

Clustering algorithms

- **K-means algorithm**
- **Probabilistic (soft) clustering methods (with EM) = soft clustering**
  - **Latent variable models**: class (cluster) is represented by a latent (hidden) variable value
  - Every point goes to the class with the highest posterior
  - **Examples:** mixture of Gaussians, Naïve Bayes with a hidden class
- **Hierarchical methods**
  - **Agglomerative**
  - **Divisive**

K-means clustering algorithm

- an iterative clustering algorithm
- works in the d-dimensional *R* space representing **x**

K-Means clusterting algorithm:

Initialize randomly *k* values of means (centers)

Repeat

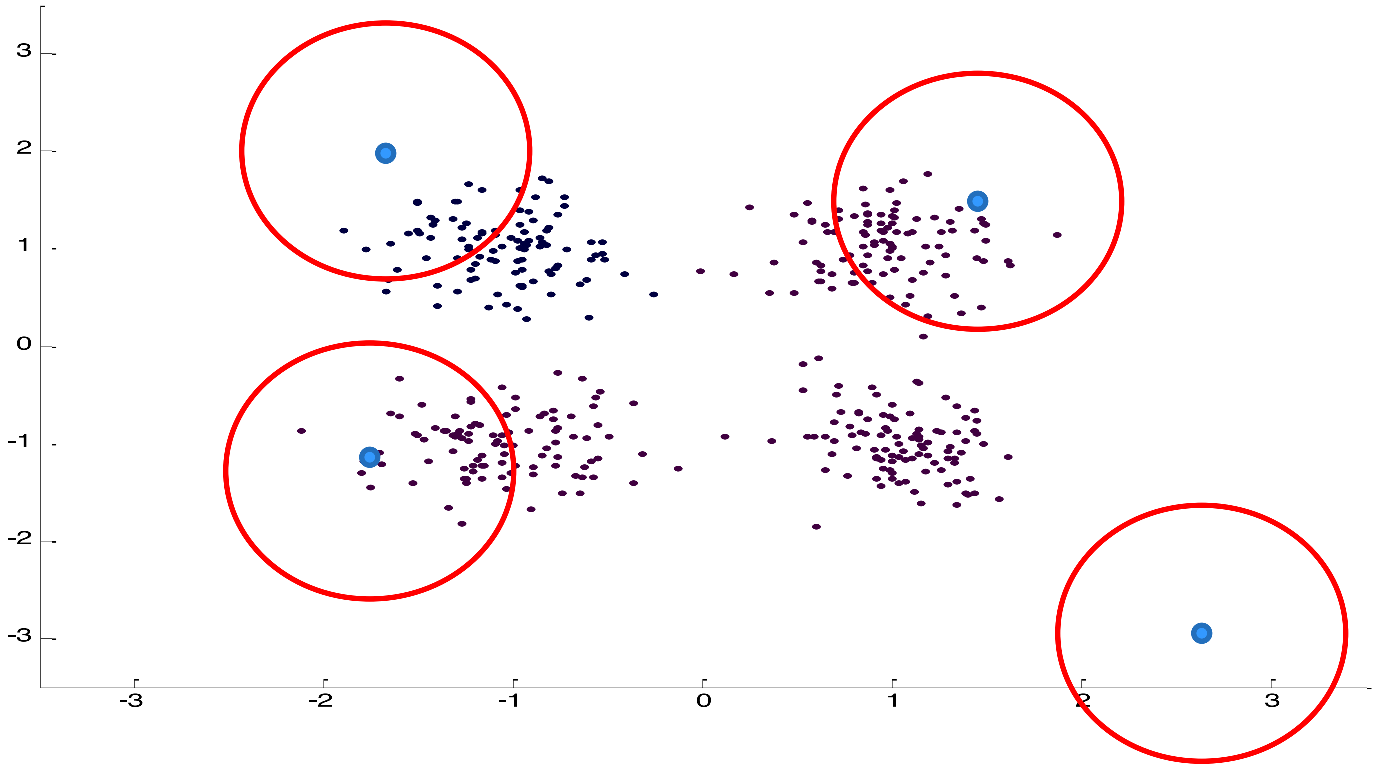
– Partition the data according to the current set of means (using the similarity measure)

– Move the means to the center of the data in the current partition

Until no change in the means

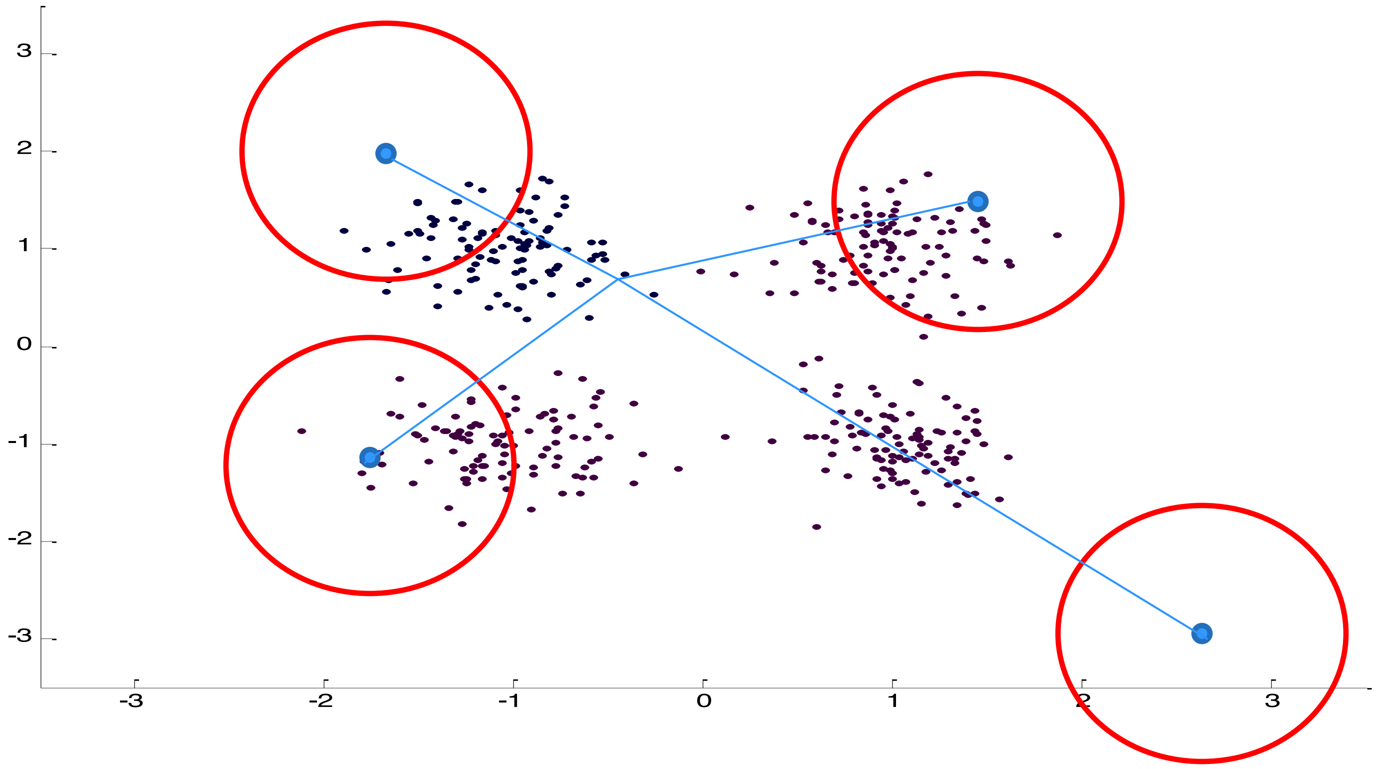
K-means: example

- Initialize the cluster centers



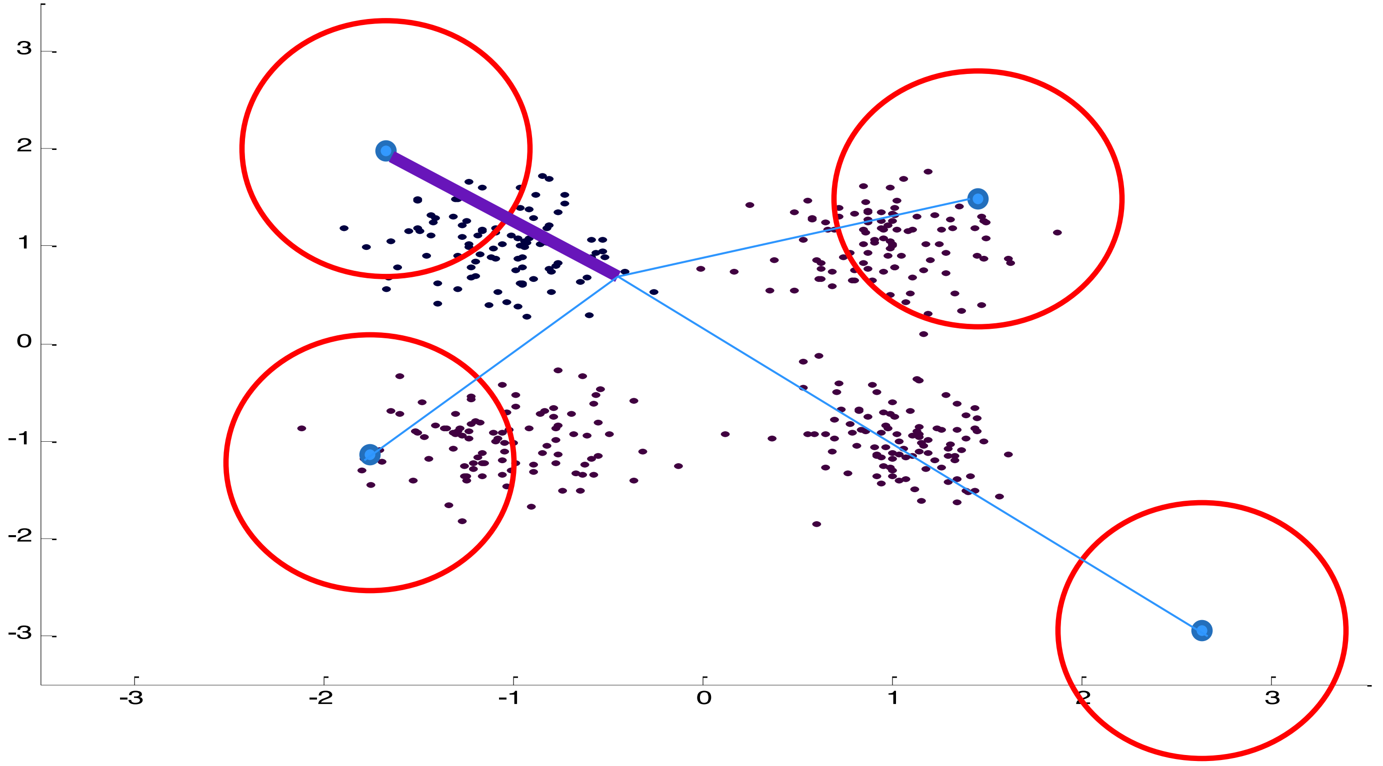
K-means: example

- Calculate the distances of each point to all centers



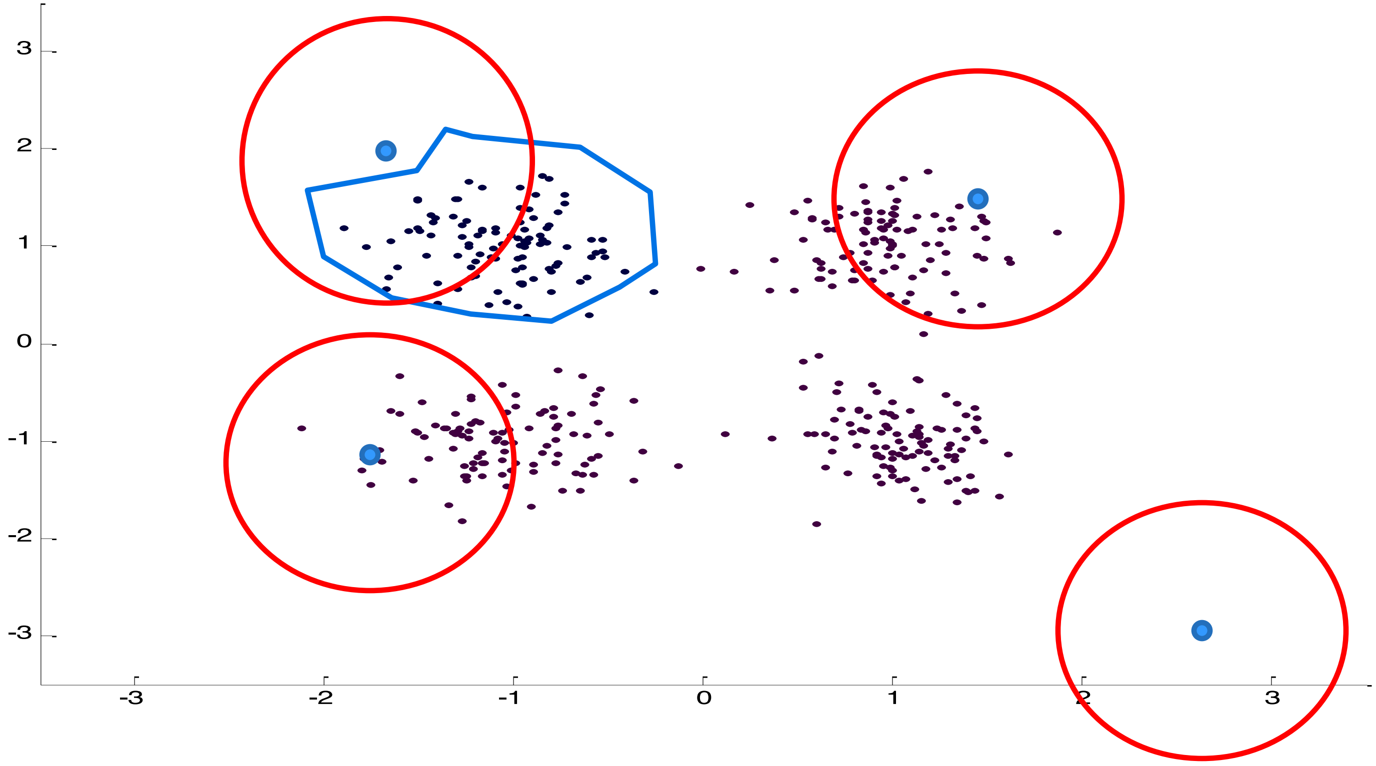
K-means: example

- **For each example pick the best (closest) center**



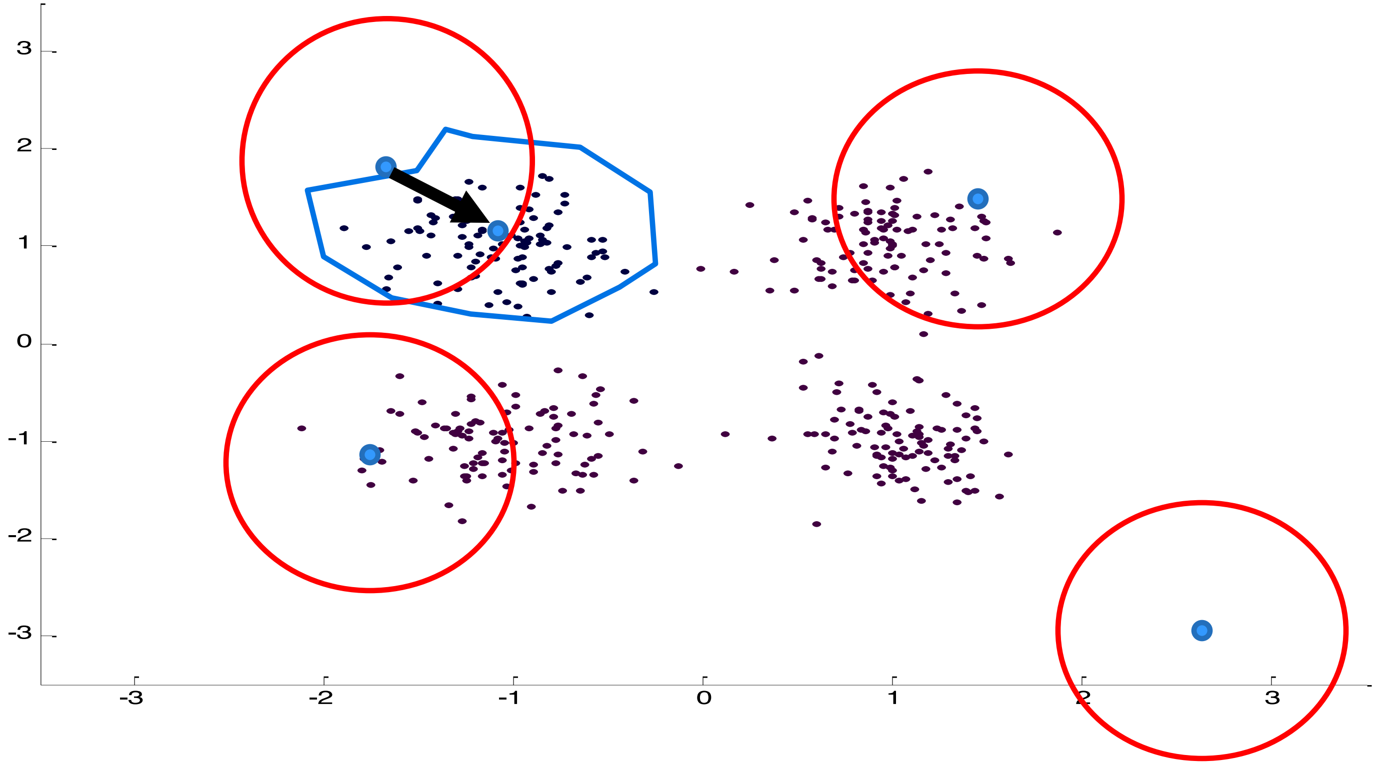
K-means: example

- **Recalculate the new mean from all data examples assigned to the same cluster center**



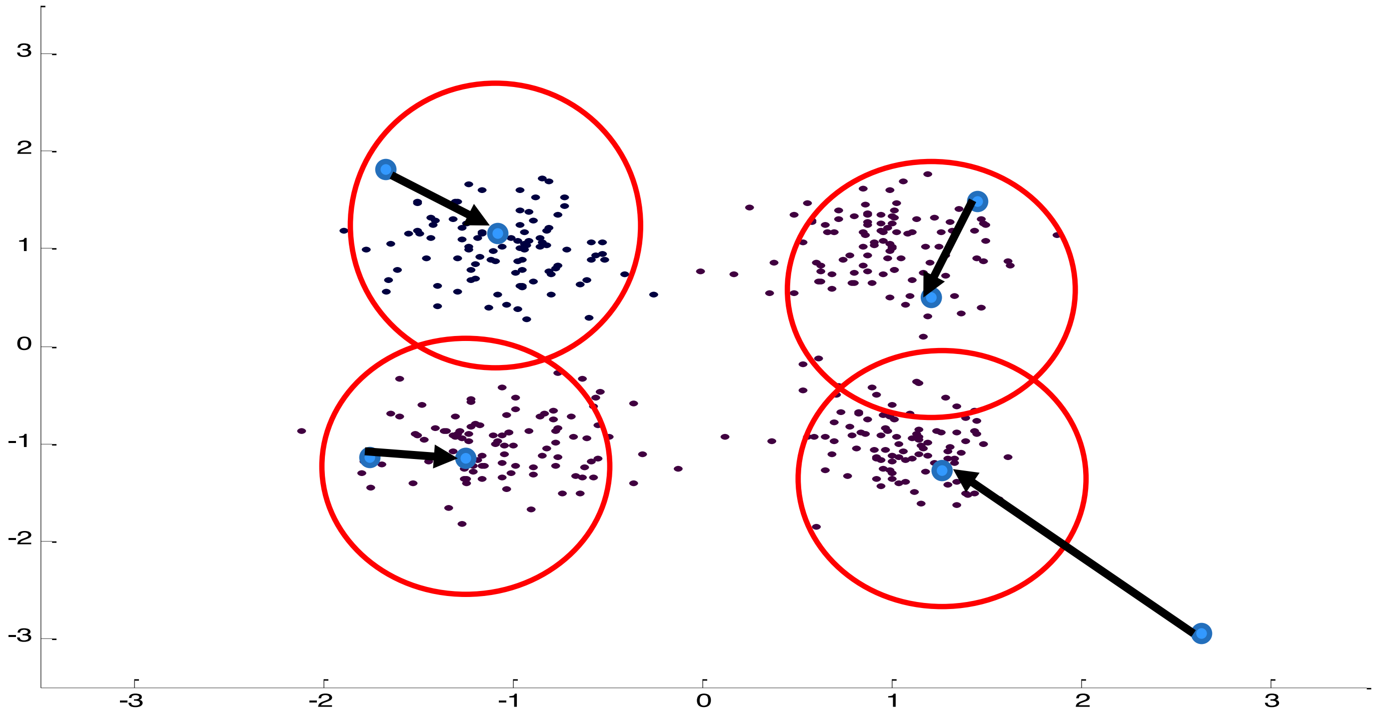
K-means: example

- Shift the cluster center to the new mean



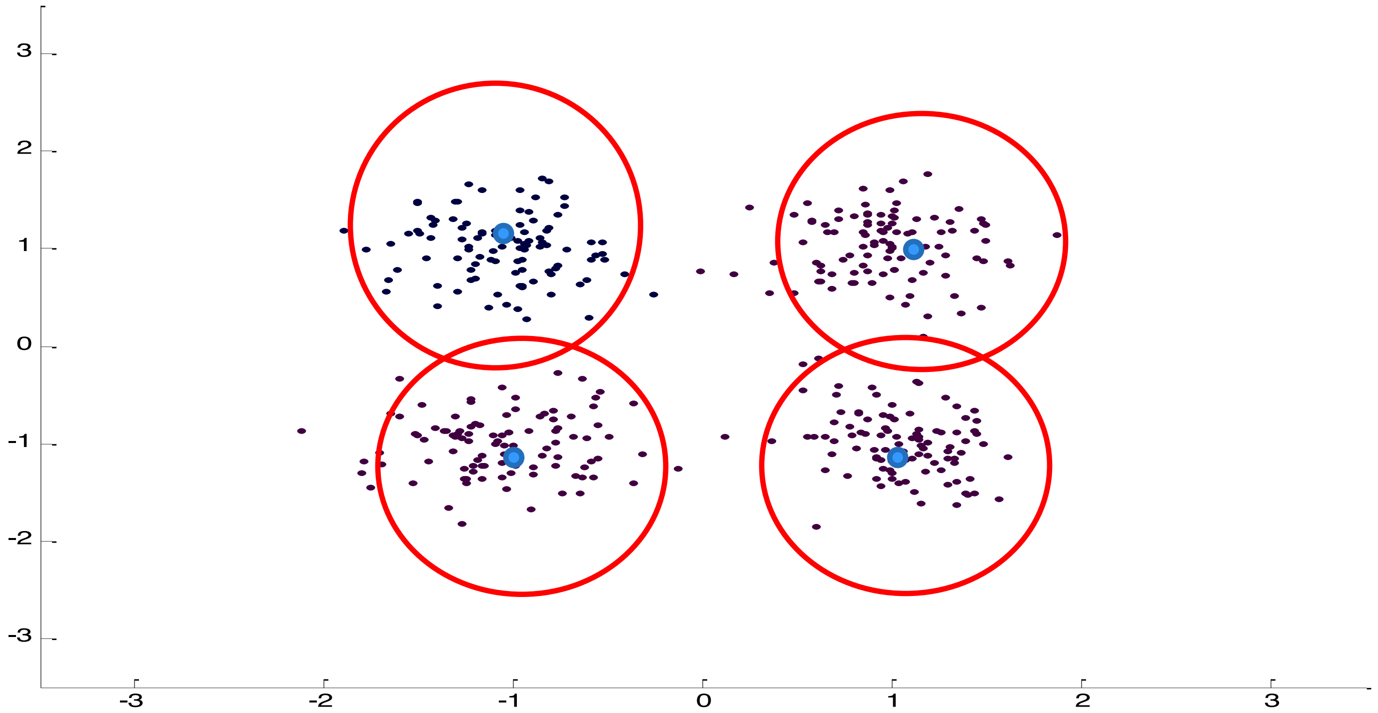
K-means: example

- Shift the cluster centers to the new calculated means



K-means: example

- And repeat the iteration …
- Till no change in the centers



K-means clustering algorithm

K-Means algorithm:

Initialize randomly k values of means (centers)

Repeat

- Partition the data according to the current set of means (using the similarity measure)
- Move the means to the center of the data in the current partition

Until no change in the means

Properties:

- Minimizes the sum of **squared center-point distances** for all clusters

$$\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{x_j \in S_i} || x_j - u_i ||^2 \qquad u_i = \text{center of cluster } S_i$$

K-means clustering algorithm

- **Properties:**
  - **converges** to centers minimizing the sum of squared center-point distances (still local optima)
  - The result is **sensitive** to the initial means' values
- **Advantages:**
  - Simplicity
  - Generality – can work for more than one distance measure
- **Drawbacks:**
  - Can perform poorly with overlapping regions
  - Lack of robustness to outliers
  - Good for attributes (features) with continuous values
    - Allows us to compute cluster means
    - k-medoid algorithm used for discrete data

Probabilistic (soft) clustering algorithms

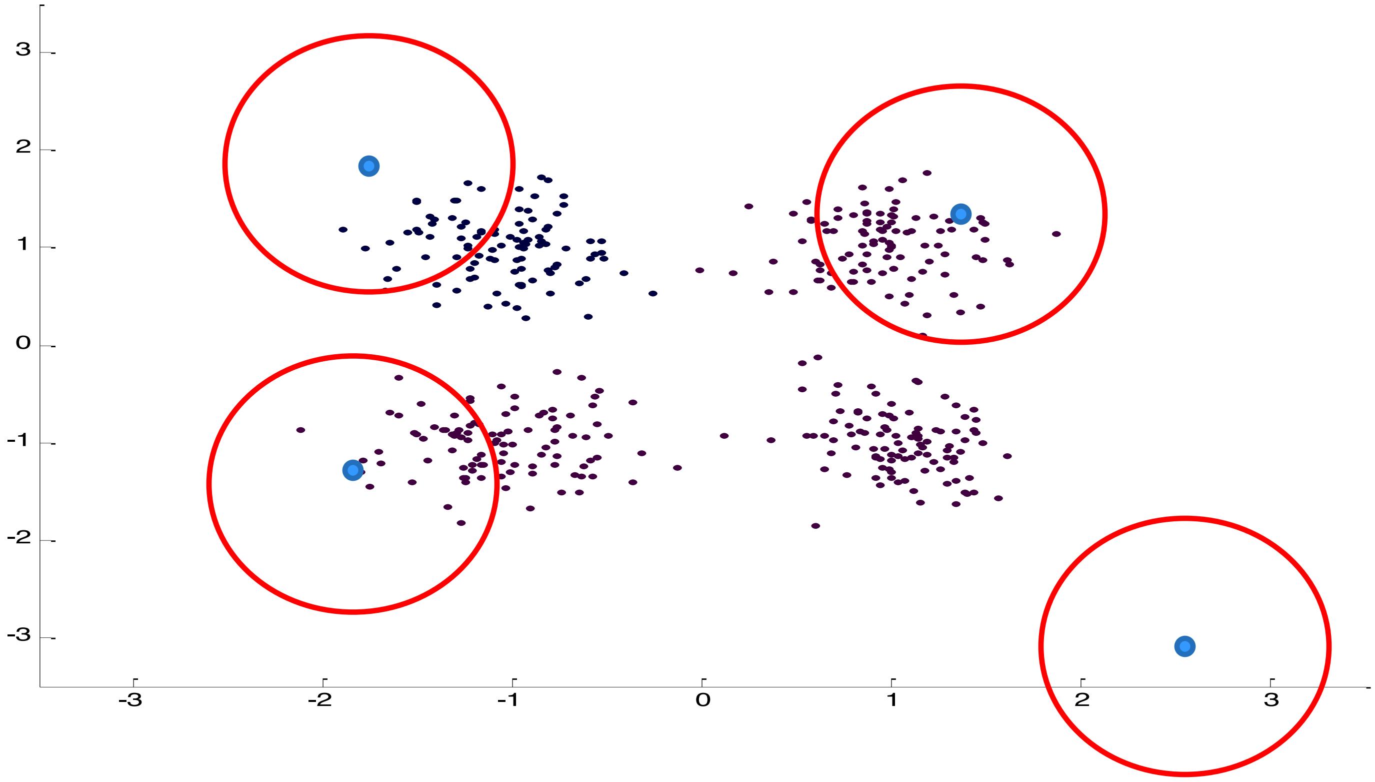
- **Latent variable models**

  Examples: Mixture of Gaussians

  Naïve Bayes with hidden class
- **Iterative algorithm:**
  - **Steps correspond to the steps of the EM algorithm**
- **Mixture of Gaussian model:**
  - Difference from k-means: each mean is responsible for every data instance, responsibilities can be different based on the distance of a Gaussian from the data instance
- **Final clusters:**
  - the data point belongs to the class with the highest posterior

Soft clustering

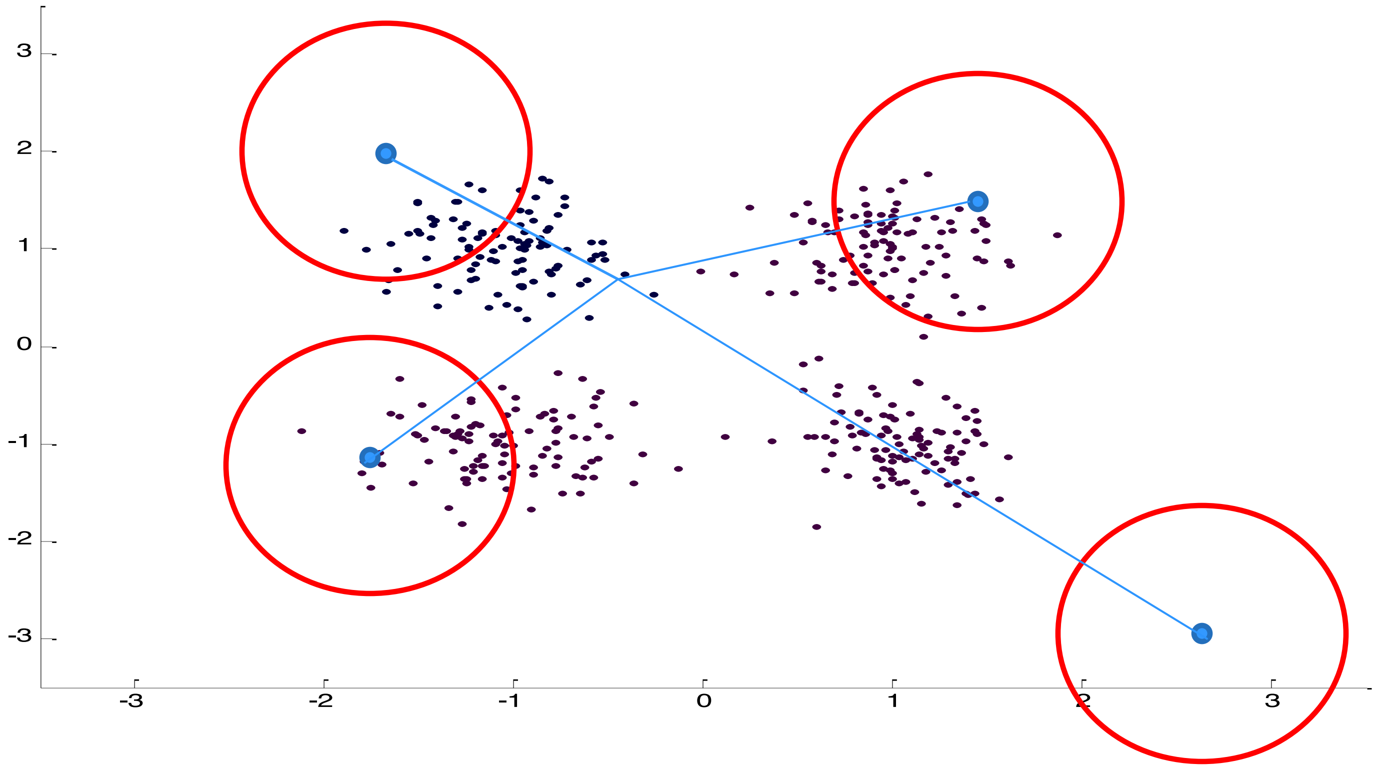
- **Gaussians centered at random mean points**



Soft clustering

- **Each Gaussian is responsible for every data instance**
  - **Responsibility**

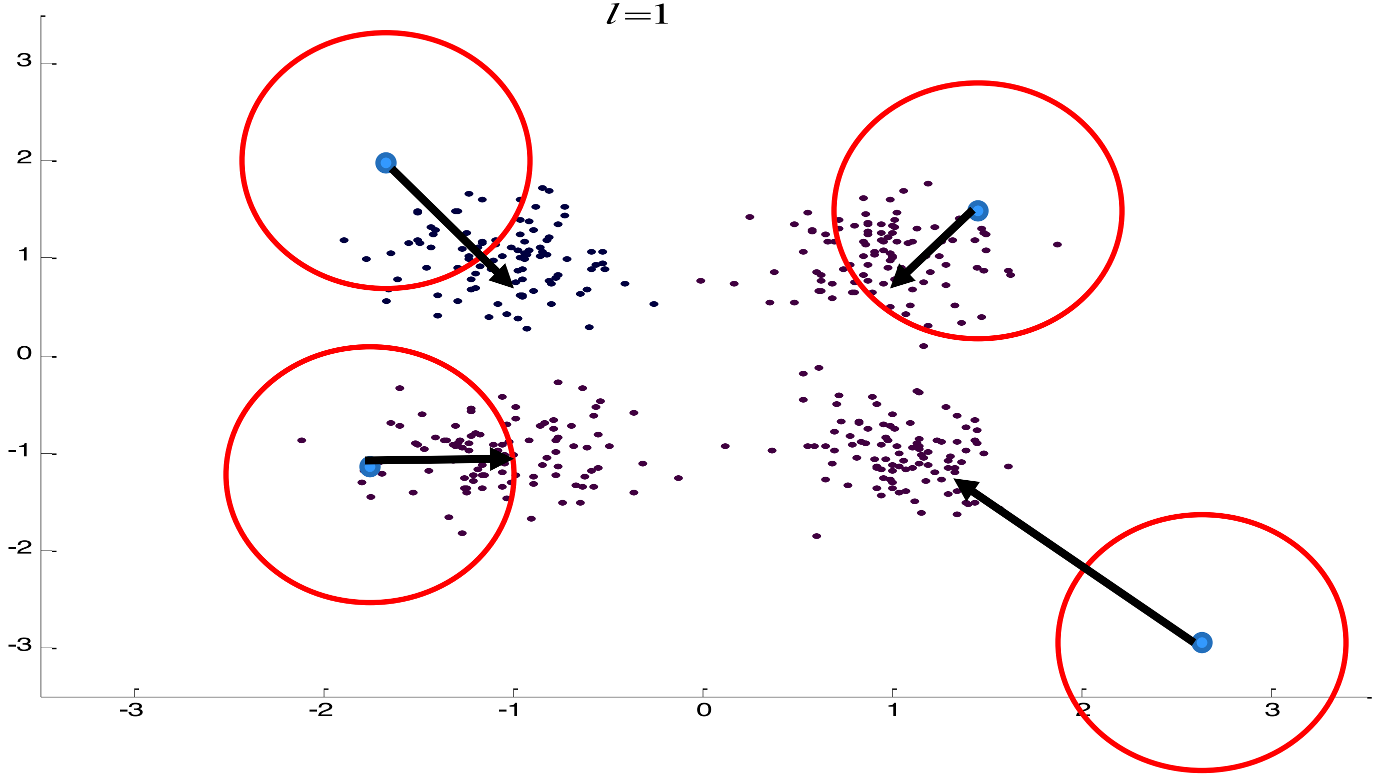
$$h_{il} = \frac{p(C_l = i \mid \Theta')\, p(x_l \mid C_l = i, \Theta')}{\sum_{u=1}^{m} p(C_l = u \mid \Theta')\, p(x_l \mid C_l = u, \Theta')}$$



Soft clustering

- Each Gaussian is repositioned by recalculating the Gaussian means:

$$\boldsymbol{\mu}_i = \frac{\sum_{l=1}^{N} h_{il}\mathbf{x}_l}{\sum_{l=1}^{N} h_{il}}$$

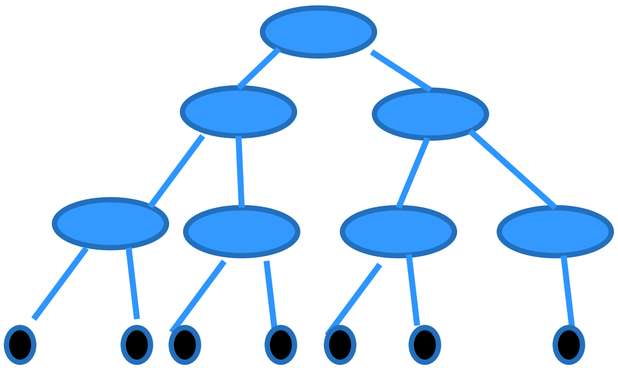


Probabilistic (soft) clustering algorithms

- **Advantages:**
  - Good performance on overlapping regions
  - Robustness to outliers
  - Data attributes can have different types of values
- **Drawbacks:**
  - EM is computationally expensive and can take time to converge
  - Density model should be given in advance

Hierarchical clustering

- **Builds a hierarchy of clusters (groups) with singleton groups at the bottom and 'all points' group on the top**



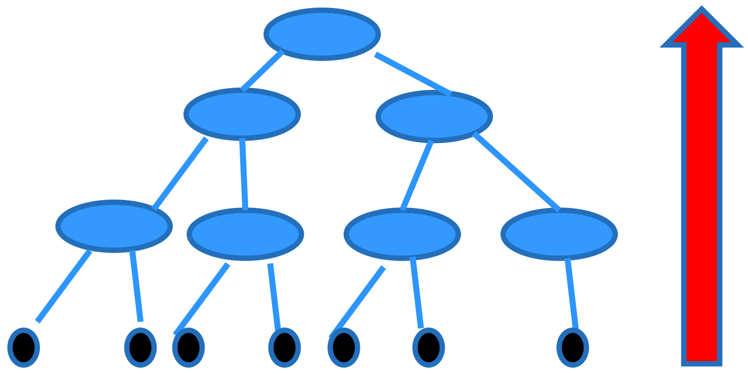
Uses many different dissimilarity measures

- **Pure real-valued data-points:**
  - Euclidean, Manhattan, Minkowski **Pure categorical data:**
  - Hamming distance,
  - **Combination of real-valued and categorical attributes**
  - Weighted, or Euclidean

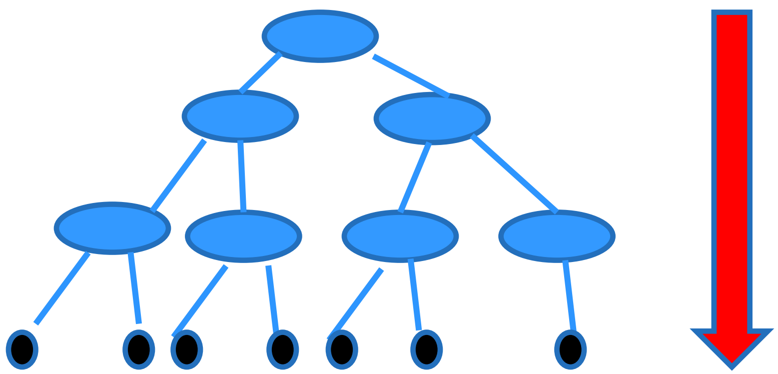
Hierarchical clustering

Two versions of the hierarchical clustering

- **Agglomerative approach**
  – Merge pair of clusters in a bottom-up fashion, starting from singleton clusters



- **Divisive approach:**
  – Splits clusters in top-down fashion, starting from one complete cluster



Hierarchical (agglomerative) clustering

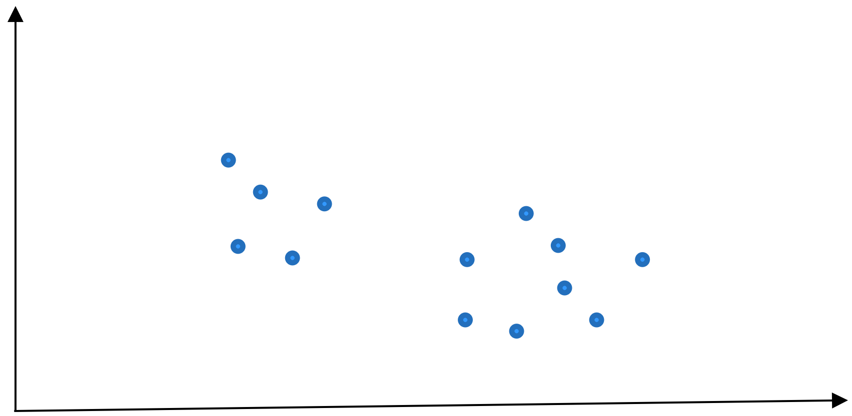
Approach:

- **Compute dissimilarity matrix for all pairs of points**
  - uses standard or other distance measures
- **Construct clusters greedily:**
  - **Agglomerative approach**
    - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
- **Stop the greedy construction** when some criterion is satisfied
  - E.g. fixed number of clusters

Hierarchical (agglomerative) clustering

Approach:

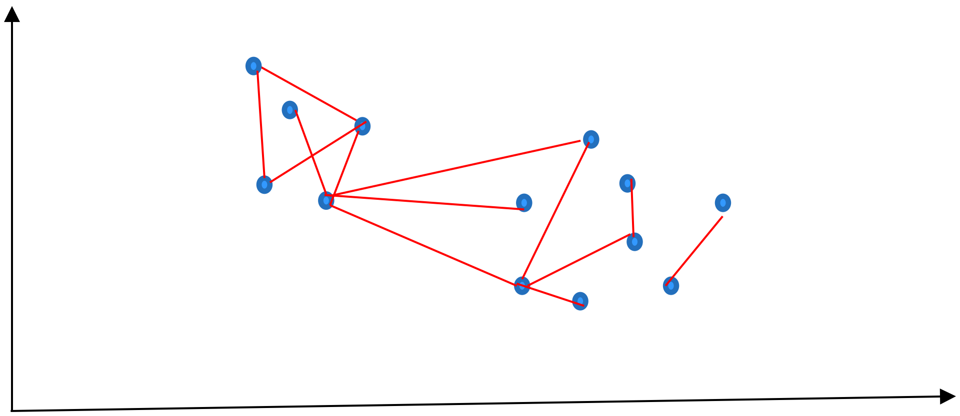
- **Compute dissimilarity matrix for all pairs of points**
  - uses standard or other distance measures



Hierarchical (agglomerative) clustering

Approach:

- **Compute dissimilarity matrix for all pairs of points**
  - uses standard or other distance measures

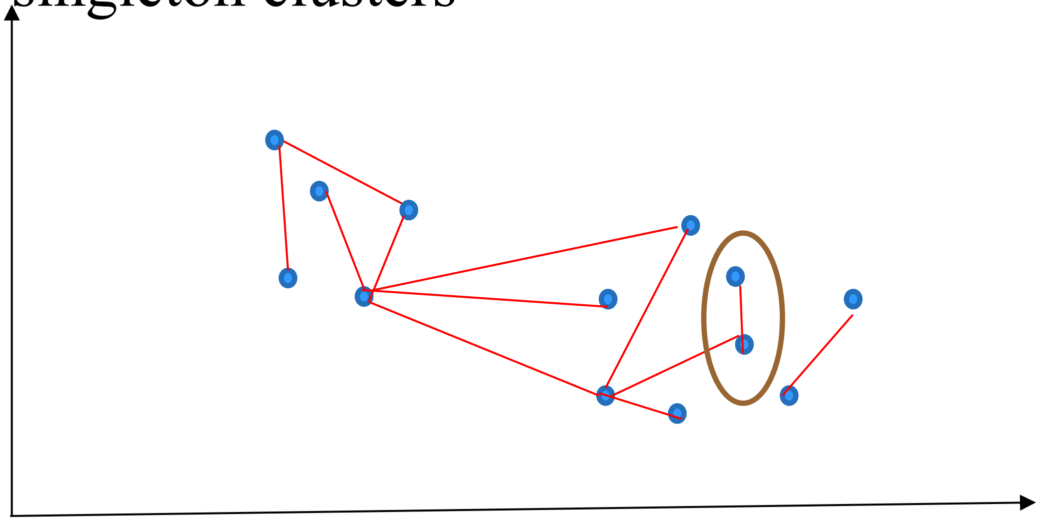


N datapoints, $O(N^2)$ pairs, $O(N^2)$ distances

Hierarchical (agglomerative) clustering

Approach:

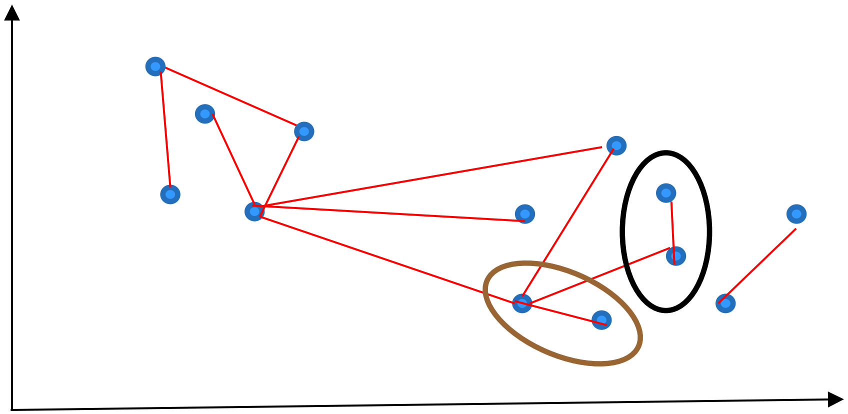
- **Compute dissimilarity matrix for all pairs of points**
  - uses standard or other distance measures
- **Construct clusters greedily:**
  - **Agglomerative approach**
    - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters



Hierarchical (agglomerative) clustering

Approach:

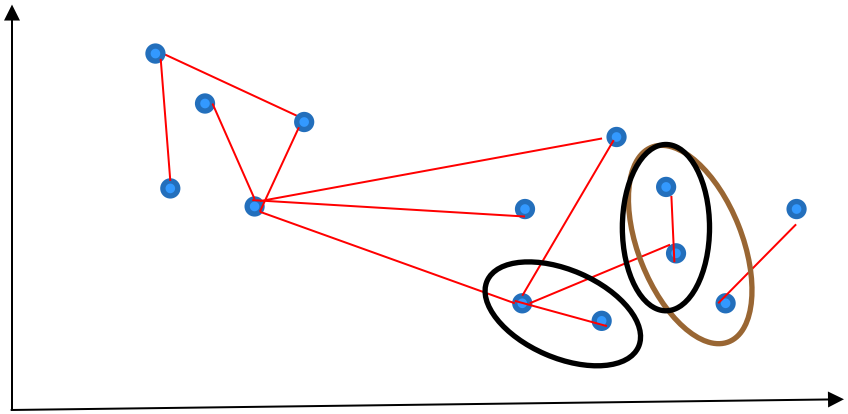
- **Compute dissimilarity matrix for all pairs of points**
  - uses standard or other distance measures
- **Construct clusters greedily:**
  - **Agglomerative approach**
    - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters



Hierarchical (agglomerative) clustering

Approach:

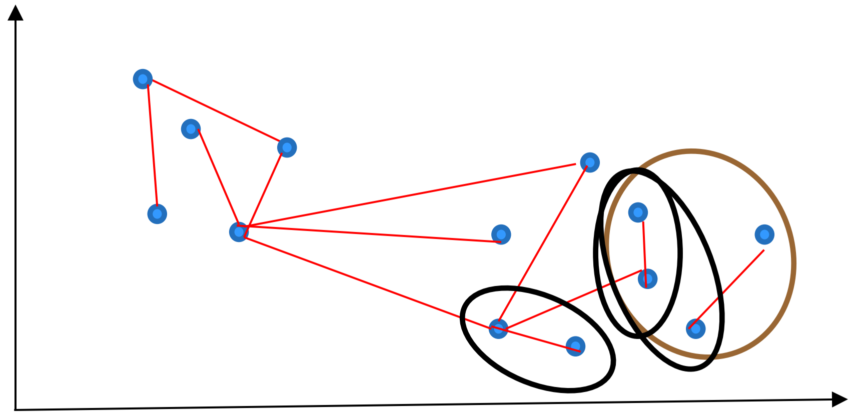
- **Compute dissimilarity matrix for all pairs of points**
  – uses standard or other distance measures
- **Construct clusters greedily:**
  – **Agglomerative approach**
    - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters



Hierarchical (agglomerative) clustering

Approach:

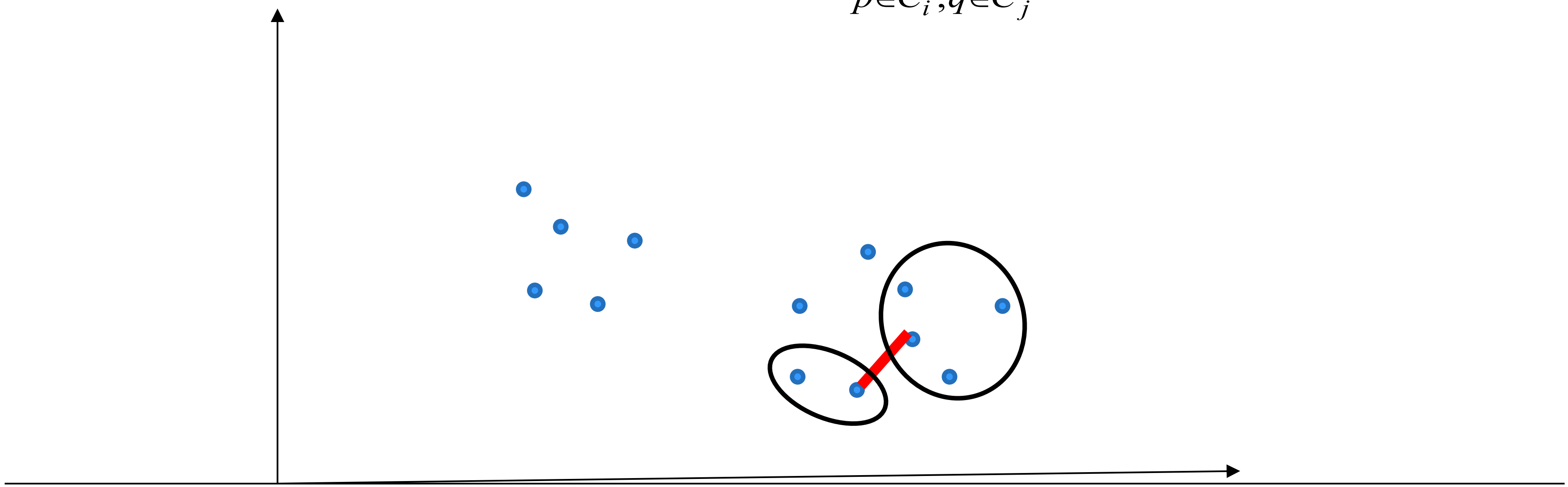
- **Compute dissimilarity matrix for all pairs of points**
  - uses standard or other distance measures
- **Construct clusters greedily:**
  - **Agglomerative approach**
    - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters



Cluster merging

- **Agglomerative approach**
  - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
  - Merge clusters based on **cluster (or linkage) distances**. Defined in terms of point distances. **Examples:**

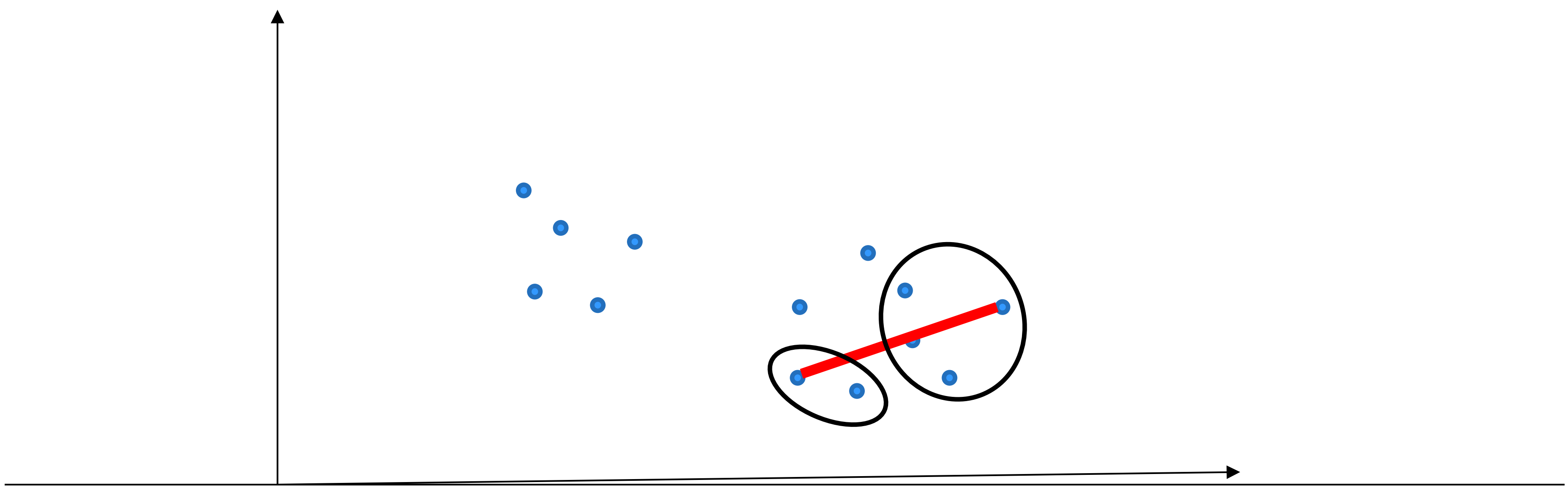
Min distance $$d_{\min}(C_i, C_j) = \min_{p \in C_i, q \in C_j} d(p, q)$$



Cluster merging

- **Agglomerative approach**
  - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
  - Merge clusters based on **cluster (or linkage) distances**. Defined in terms of point distances. **Examples:**

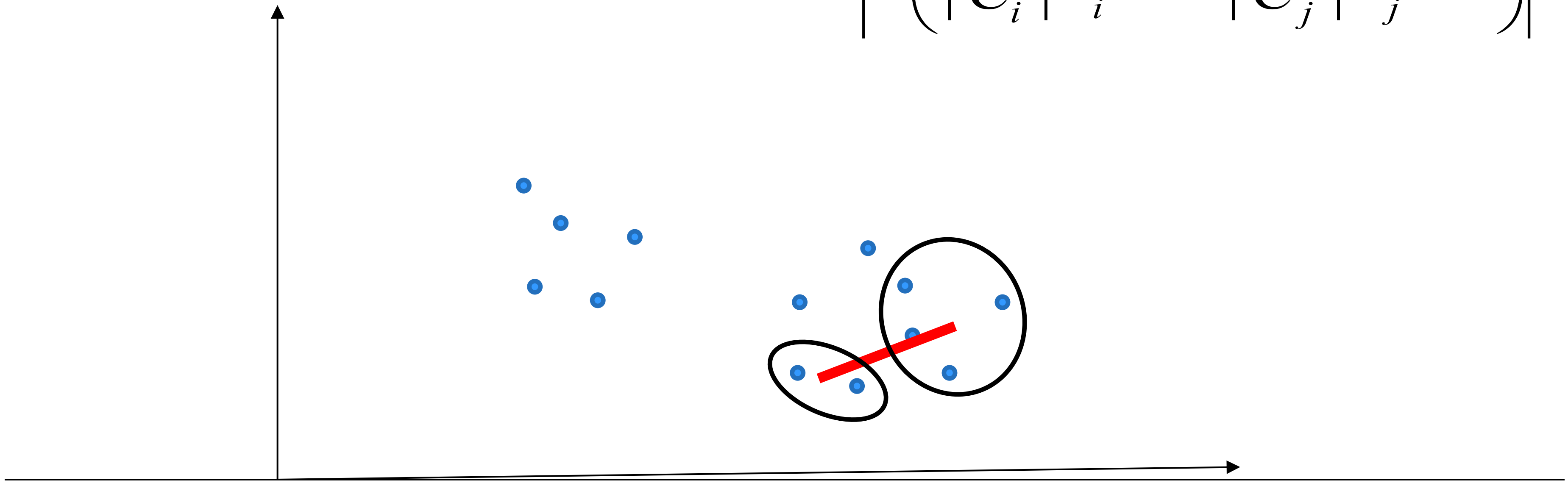
Max distance $$d_{\max}(C_i, C_j) = \max_{p \in C_i, q \in C_j} d(p, q)$$



Cluster merging

- **Agglomerative approach**
  - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
  - Merge clusters based on **cluster (or linkage) distances**. Defined in terms of point distances. **Examples:**

Mean distance $$d_{mean}(C_i, C_j) = \left| d\left(\frac{1}{|C_i|} \sum_i p_i ; \frac{1}{|C_j|} \sum_j q_j \right) \right|$$



Hierarchical (agglomerative) clustering

Approach:

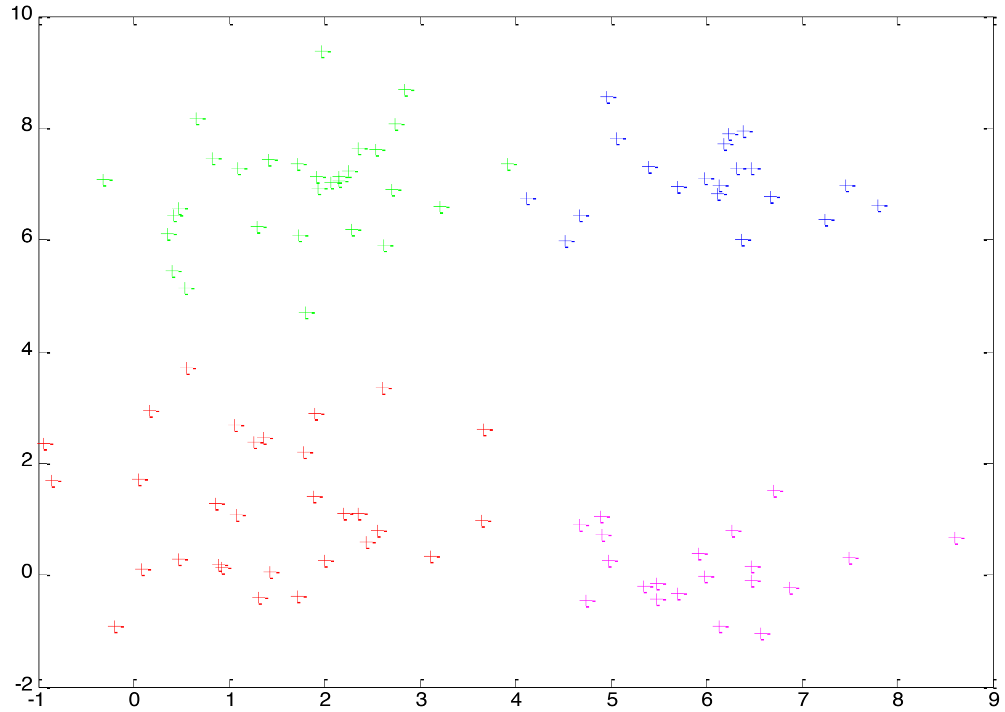
- **Compute dissimilarity matrix for all pairs of points**
  - uses standard or other distance measures
- **Construct clusters greedily:**
  - **Agglomerative approach**
    - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
- **Stop the greedy construction** when some criterion is satisfied
  - E.g. fixed number of clusters

Hierarchical (divisive) clustering

Approach:

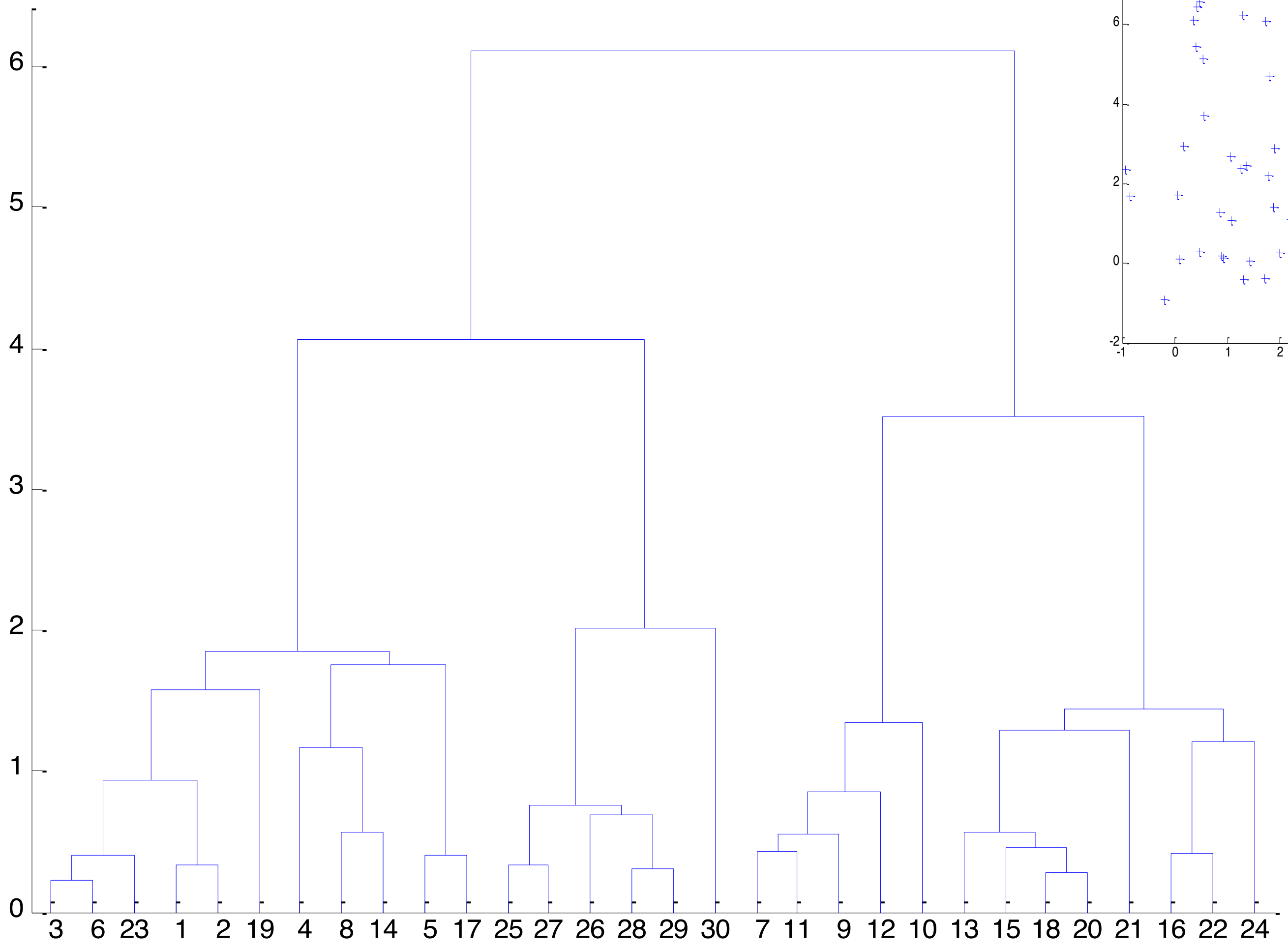
- **Compute dissimilarity matrix for all pairs of points**
  - uses standard distance or other dissimilarity measures
- **Construct clusters greedily:**
  - **Agglomerative approach**
    - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
  - **Divisive approach:**
    - Splits clusters in top-down fashion, starting from one complete cluster
- **Stop the greedy construction** when some criterion is satisfied
  - E.g. fixed number of clusters

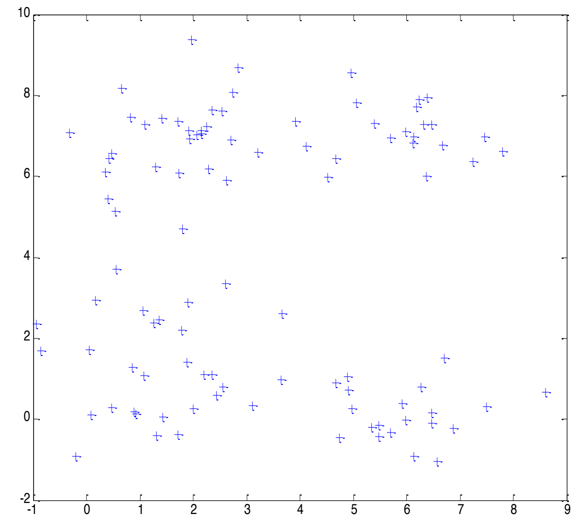
Hierarchical clustering example



Hierarchical clustering example

- **Dendogram**





Hierarchical clustering

- **Advantage:**
  - Smaller computational cost; avoids scanning all possible clusterings
- **Disadvantage:**
  - Greedy choice fixes the order in which clusters are merged; cannot be repaired
- **Partial solution:**
  - combine hierarchical clustering with iterative algorithms like k-means algorithm

Other clustering methods

- **Spectral clustering**
  - Relies on similarity matrix and its spectral decomposition (eigenvalues and eigenvectors)