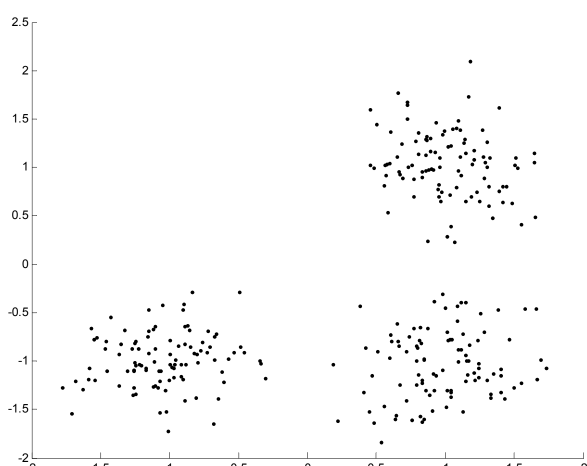
CS 2750 Machine Learning
Lecture 15

Clustering

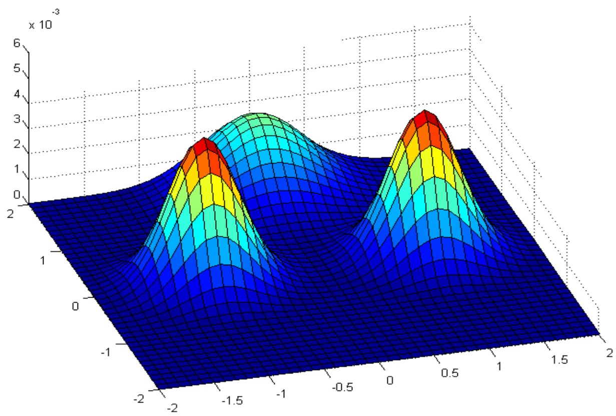
Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

Gaussian mixture model



Mixture of Gaussians

- Density function for the Mixture of Gaussians model



Gaussian mixture model

Probability of occurrence of a data example $\boldsymbol{x}$ is modeled as

$$p(\mathbf{x}) = \sum_{i=1}^{m} p(C = i)\, p(\mathbf{x} \mid C = i)$$

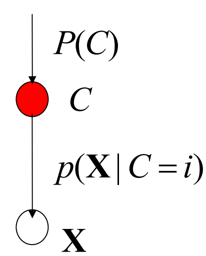
where

$p(C = i)$

= probability of a data point coming from class $C=i$

$p(\mathbf{x} \mid C = i) \approx N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$

= class conditional density (modeled as a Gaussian) for class i



<u>Remember:</u> *C* is hidden !!!!

Generative classifier model

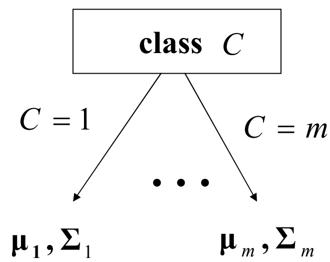
- Generative classifier model with Gaussian densities
- Assume the class labels are known. The ML estimate is

$$N_i = \sum_{j:C_l = i} 1$$

$$\tilde{\pi}_i = \frac{N_i}{N}$$

$$\tilde{\boldsymbol{\mu}}_i = \frac{1}{N_i} \sum_{j:C_l = i} \mathbf{x}_j$$

$$\tilde{\boldsymbol{\Sigma}}_i = \frac{1}{N_i} \sum_{j:C_l = i} (\mathbf{x}_j - \boldsymbol{\mu}_i)(\mathbf{x}_j - \boldsymbol{\mu}_i)^T$$



Gaussian mixture model

- In the Gaussian mixture Gaussians are not labeled
- We can apply **EM algorithm**:
  - re-estimation based on the class posterior

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

$$N_i = \sum_l h_{il}$$

Count replaced with the expected count

$$\tilde{\pi}_i = \frac{N_i}{N}$$

$$\tilde{\boldsymbol{\mu}}_i = \frac{1}{N_i} \sum_l h_{il} \mathbf{x}_j$$

$$\tilde{\boldsymbol{\Sigma}}_i = \frac{1}{N_i} \sum_l h_{il} (\mathbf{x}_j - \boldsymbol{\mu}_i)(\mathbf{x}_j - \boldsymbol{\mu}_i)^T$$

Gaussian mixture algorithm

- **Special case:** fixed covariance matrix for all hidden groups (classes) and a uniform prior on classes
- **Algorithm**:

  Initialize means $\boldsymbol{\mu}_i$ for all classes i

  Repeat two steps until no change in the means:

  1. Compute the class posterior for each Gaussian and each point (a kind of responsibility for a Gaussian for a point)

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

  2. Move the means of the Gaussians to the center of the data, weighted by the responsibilities

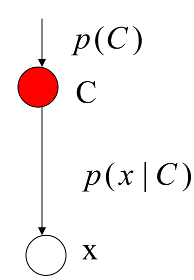
     New mean: $$\boldsymbol{\mu}_i = \frac{\sum_{l=1}^{N} h_{il}\,\mathbf{x}_l}{\sum_{l=1}^{N} h_{il}}$$

Gaussian mixture model. Gradient ascent.

- A set of parameters

  $$\Theta = \{\pi_1, \pi_2, ..\pi_m, \mu_1, \mu_2, ...\mu_m\}$$

  Assume unit variance terms and fixed priors



$$P(\mathbf{x} \mid C = i) = (2\pi)^{-1/2} \exp\left\{-\frac{1}{2}\|x - \mu_i\|^2\right\}$$

$$P(D \mid \Theta) = \prod_{l=1}^{N} \sum_{i=1}^{m} \pi_i (2\pi)^{-1/2} \exp\left\{-\frac{1}{2}\|x_l - \mu_i\|^2\right\}$$

$$l(\Theta) = \sum_{l=1}^{N} \log \sum_{i=1}^{m} \pi_i (2\pi)^{-1/2} \exp\left\{-\frac{1}{2}\|x_l - \mu_i\|^2\right\}$$

$$\frac{\partial l(\Theta)}{\partial \mu_i} = \sum_{l=1}^{N} h_{il}(x_l - \mu_i)$$

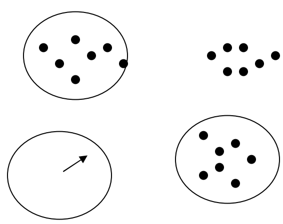
- very easy on-line update

EM versus gradient ascent

Gradient ascent

$$\mu_i \leftarrow \mu_i + \alpha \sum_{l=1}^{N} h_{il}(x_l - \mu_i)$$

Learning rate

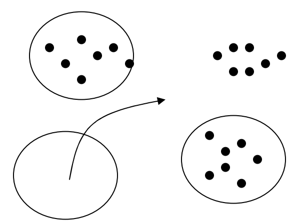


Small pull towards distant uncovered data

EM

$$\mu_i \leftarrow \frac{\sum_{l=1}^{N} h_{il} \mathbf{x}_l}{\sum_{l=1}^{N} h_{il}}$$

No learning rate



Renormalized – big jump in the first step

K-means approximation to EM

Mixture of Gaussians with the fixed covariance matrix:

- posterior measures the responsibility of a Gaussian for every point

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

- **Re-estimation of means:** $\boldsymbol{\mu}_i = \frac{\sum_{l=1}^{N} h_{il} \mathbf{x}_l}{\sum_{l=1}^{N} h_{il}}$
- **K- Means approximations**
- Only the closest Gaussian is made responsible for a point

$h_{il} = 1$ If i is the closest Gaussian

$h_{il} = 0$ Otherwise

- Results in moving the means of Gaussians to the center of the data points it covered in the previous step

K-means algorithm

K-Means algorithm:

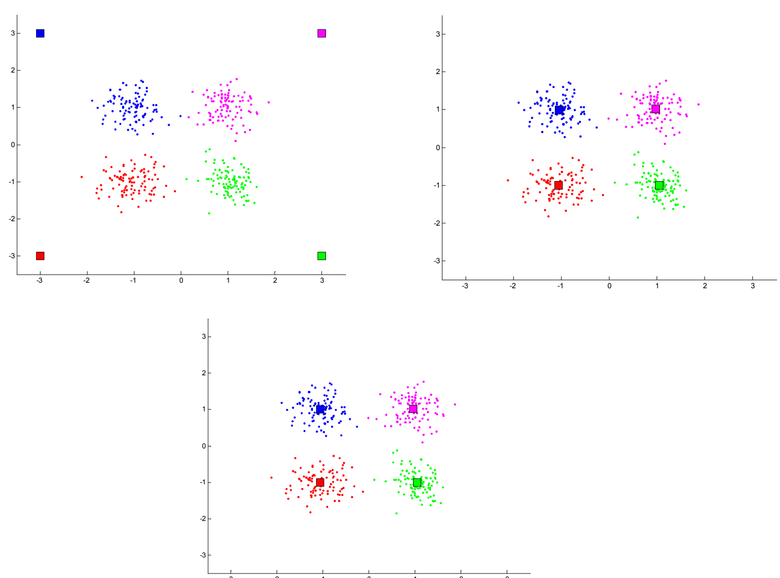
Initialize k values of means (centers)

Repeat two steps until no change in the means:

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

- **Used frequently for clustering data**

K-Means example



Clustering

Groups together "similar" instances in the data sample

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 defined in terms of some distance metric (can be chosen)

Clustering is useful for:

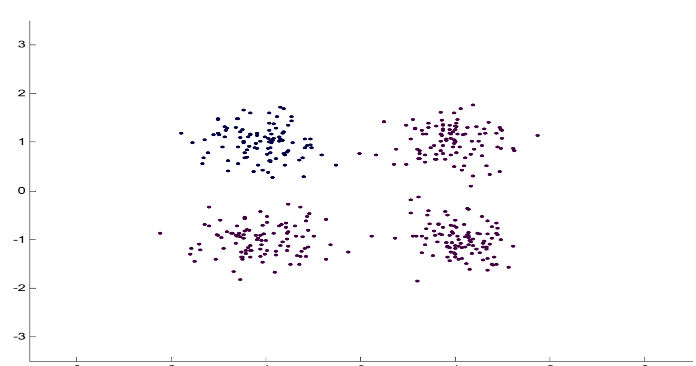
- **Similarity/Dissimilarity analysis**

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

  High dimensional data replaced with a group (cluster) label

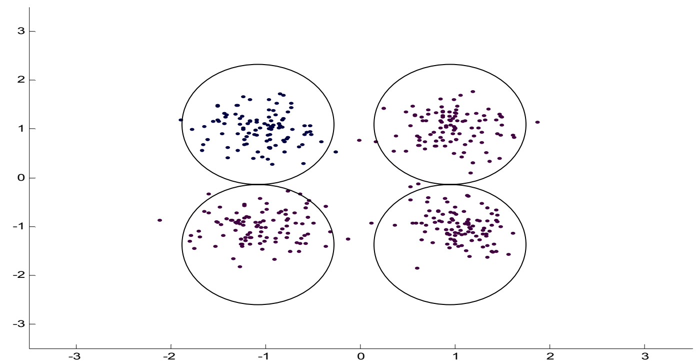
Clustering example

- We see data points and want to partition them into groups
- Which data points belong together?



Clustering example

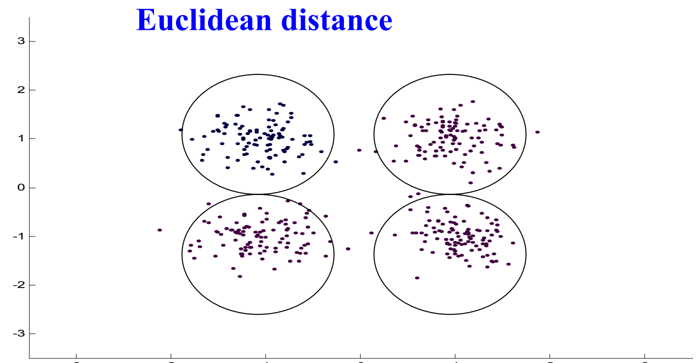
- We see data points and want to partition them into the groups
- Which data points belong together?



Clustering example

- We see data points and want to partition them into the groups
- Requires a distance measure to tell us what points are close to each other and are in the same group

Euclidean distance



Clustering example

- A set of patient cases
- We want to partition them into groups based on similarities

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

- A set of patient cases
- We want to partition them into the groups based on similarities

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

How to design the distance metric to quantify similarities?

Clustering example. Distance measures.

In general, one can choose an arbitrary distance measure.

Properties of distance metrics:

Assume 2 data entries *a, b*

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

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

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

Distance measures.

Assume pure real-valued data-points:

12	34.5	78.5	89.2	19.2
23.5	41.4	66.3	78.8	8.9
33.6	36.7	78.3	90.3	21.4
17.2	30.1	71.6	88.5	12.5

…

What distance metric to use?

Distance measures

Assume pure real-valued data-points:

12	34.5	78.5	89.2	19.2
23.5	41.4	66.3	78.8	8.9
33.6	36.7	78.3	90.3	21.4
17.2	30.1	71.6	88.5	12.5

…

What distance metric to use?

Euclidian: works for an arbitrary k-dimensional space

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

Distance measures

Assume pure real-valued data-points:

12	34.5	78.5	89.2	19.2
23.5	41.4	66.3	78.8	8.9
33.6	36.7	78.3	90.3	21.4
17.2	30.1	71.6	88.5	12.5

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 measures.

Assume pure real-valued data-points:

12	34.5	78.5	89.2	19.2
23.5	41.4	66.3	78.8	8.9
33.6	36.7	78.3	90.3	21.4
17.2	30.1	71.6	88.5	12.5

Manhattan distance:

works for an arbitrary k-dimensional space

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

Etc. ..

Γ^{-1}

Distance measures

Generalized distance metric:

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

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

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
…

What distance metric to use?

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
…

What distance metric to use?

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

The same metric can be used for pure categorical values:

- number of values that need to be changed to make them the same

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?

A weighted sum approach: e.g. a mix of Euclidian and Hamming distances for subsets of attributes

Clustering

Clustering is useful for:

- **Similarity/Dissimilarity analysis**
  Analyze what data points in the sample are close to each other
- **Dimensionality reduction**
  High dimensional data replaced with a group (cluster) label
- **Data reduction:** Replaces many datapoints with the point representing the group mean

Problems:

- Pick the correct similarity measure (problem specific)
- Choose the correct number of groups
  - Many clustering algorithms require us to provide the number of groups ahead of time

Clustering algorithms

- **K-means algorithm**
  - **suitable** only when data points have continuous values; groups are defined in terms of cluster centers (also called **means**). Refinement of the method to categorical values: **K-medoids**
- **Probabilistic methods (with EM)**
  - **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

K-Means algorithm:

Initialize randomly *k* values of means (centers)

Repeat two steps until no change in the means:

- 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

Stop when no change in the means

Properties:

- Minimizes the sum of **squared center-point distances** for all clusters
- The algorithm always converges (to the local optima).

K-means 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 (EM-based) algorithms

- **Latent variable models**

  Examples: Naïve Bayes with hidden class

  Mixture of Gaussians
- **Partitioning:**
  - the data point belongs to the class with the highest posterior
- **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.

Uses an arbitrary similarity/dissimilarity measure.

Typical similarity measures *d(a,b)* :

Pure real-valued data-points:
- Euclidean, Manhattan, Minkowski distances

Pure binary values data:
- Hamming distance - Number of matching values

Pure categorical data:
- Number of matching values

Combination of real-valued and categorical attributes
- Weighted approaches

Hierarchical 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
  - **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

Cluster merging

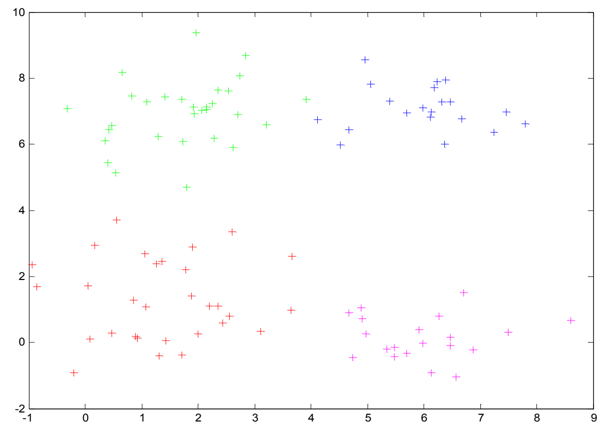
- **Construction of clusters through greedy agglomerative approach**
  - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
  - Merge clusters based on **cluster (or linkage) distances**. Defined in terms of point distances. **Examples:**

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

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

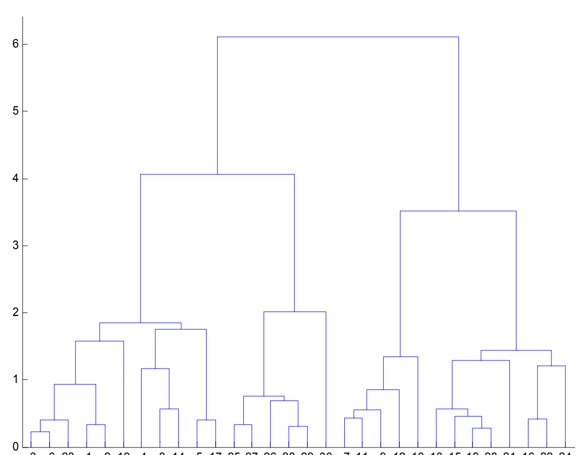
Mean distance $$d_{mean}(C_i, C_j) = \left| \frac{1}{|C_i|} \sum_i p_i - \frac{1}{|C_j|} \sum_j q_j \right|$$

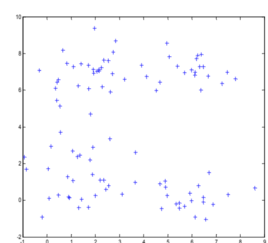
Hierarchical clustering example



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

Clustering

Groups together "similar" instances in the data sample

Basic clustering problem:

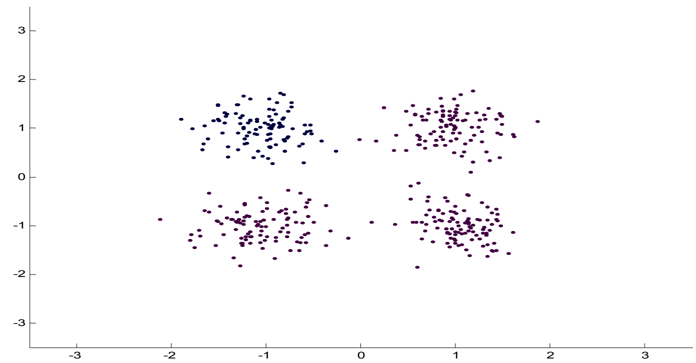
- distribute data into *k* different groups such that data points similar to each other are in the same group
- Similarity between data points is defined in terms of some distance metric (can be chosen)

Clustering is useful for:

- **Similarity/Dissimilarity analysis**
  Analyze what data points in the sample are close to each other
- **Dimensionality reduction**
  High dimensional data replaced with a group (cluster) label

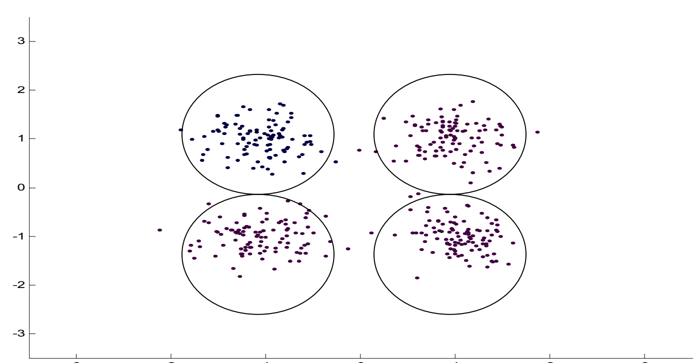
Clustering example

- We see data points and want to partition them into groups
- Which data points belong together?



Clustering example

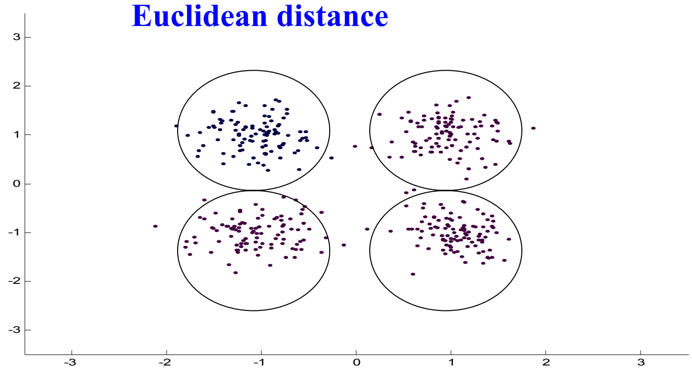
- We see data points and want to partition them into the groups
- Which data points belong together?



Clustering example

- We see data points and want to partition them into the groups
- Requires a distance measure to tell us what points are close to each other and are in the same group

Euclidean distance



Clustering example

- A set of patient cases
- We want to partition them into the groups based on similarities

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

- A set of patient cases
- We want to partition them into the groups based on similarities

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

How to design the distance metric to quantify similarities?

Clustering example. Distance measures.

In general, one can choose an arbitrary distance measure.

Properties of the distance measures:

Assume 2 data entries *a,b*

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

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

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

Distance measures.

Assume pure real-valued data-points:

12	34.5	78.5	89.2	19.2
23.5	41.4	66.3	78.8	8.9
33.6	36.7	78.3	90.3	21.4
17.2	30.1	71.6	88.5	12.5

What distance metric to use?

Distance measures.

Assume pure real-valued data-points:

12	34.5	78.5	89.2	19.2
23.5	41.4	66.3	78.8	8.9
33.6	36.7	78.3	90.3	21.4
17.2	30.1	71.6	88.5	12.5

What distance metric to use?

Euclidian: works for an arbitrary k-dimensional space

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

Distance measures

Assume pure real-valued data-points:

12	34.5	78.5	89.2	19.2
23.5	41.4	66.3	78.8	8.9
33.6	36.7	78.3	90.3	21.4
17.2	30.1	71.6	88.5	12.5

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 measures.

Assume pure real-valued data-points:

12	34.5	78.5	89.2	19.2
23.5	41.4	66.3	78.8	8.9
33.6	36.7	78.3	90.3	21.4
17.2	30.1	71.6	88.5	12.5

Assume that two variables are highly correlated in k-dimensional space

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

Etc. ..

Γ^{-1}

Distance measures.

Generalized distance metric:

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

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

$\Gamma = \Sigma$ Mahalanobis distance takes into account correlations among attributes

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
…

What distance metric to use?

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

…

What distance metric to use?

Edit distance: The number of attributes that need to be changed to make the entries the same

The same metric can be used for pure categorical values

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?

A weighted sum approach: e.g. a mix of Euclidian and Edit distances for subsets of attributes

Clustering

Clustering is useful for:

- **Similarity/Dissimilarity analysis**
  Analyze what data points in the sample are close to each other
- **Dimensionality reduction**
  High dimensional data replaced with a group (cluster) label

Problems:

- Pick the correct similarity measure (problem specific)
- Choose the correct number of groups
  - Many clustering algorithms require us to provide the number of groups ahead of time

Clustering algorithms

Partitioning algorithms:

- **K-means algorithm**
  - **suitable** only when data points have continuous values; groups are defined in terms of cluster centers (also called **means**).
  - refinement of the method to categorical values: **K-medoids**
- **Probabilistic methods (with EM)**
  - **Latent variable models**: class (cluster) is represented by a latent (hidden) variable value.
  - **Examples:** mixture of Gaussians, Naïve Bayes with a hidden class
- **Hierarchical methods**
  - **Agglomerative**
  - **Divisive**

K-means

K-Means algorithm:

Initialize randomly k values of means (centers)

Repeat two steps until no change in the means:

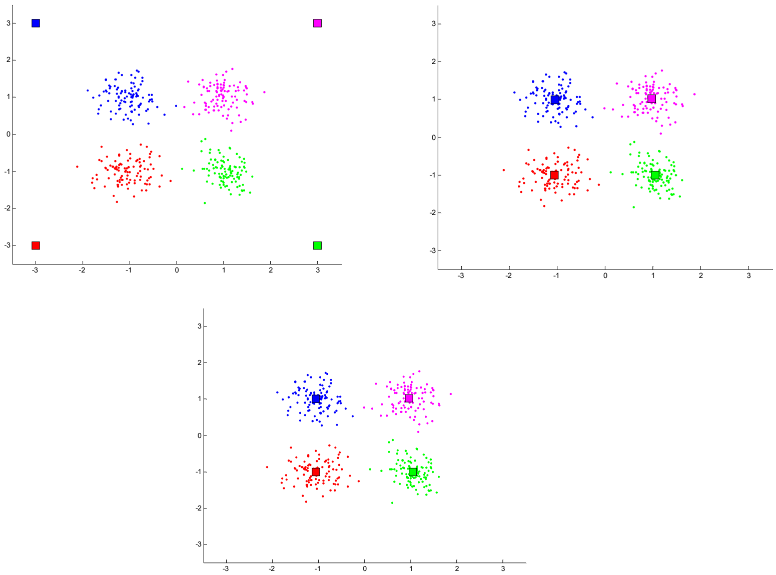
- Partition the data according to the current set of means (using the similarity measure)
- Move the means to the center of the data in the current partition

Stop when no change in the means

Properties:

- Minimizes the sum of **squared center-point distances** for all clusters
- The algorithm always converges (local optima).

K-Means example



K-means 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 (EM-based) algorithms

- **Latent variable models**

  Examples: Naïve Bayes with hidden class

  Mixture of Gaussians
- **Partitioning:**
  – the data point belongs to the class with the highest posterior
- **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.

Uses an arbitrary similarity/dissimilarity measure.

Typical similarity measures *d(a,b)* :

Pure real-valued data-points:
– Euclidean, Manhattan, Minkowski distances

Pure binary values data:
– Number of matching values

Pure categorical data:
– Number of matching values

Combination of real-valued and categorical attributes
– Weighted approaches

Hierarchical 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
  - **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

Cluster merging

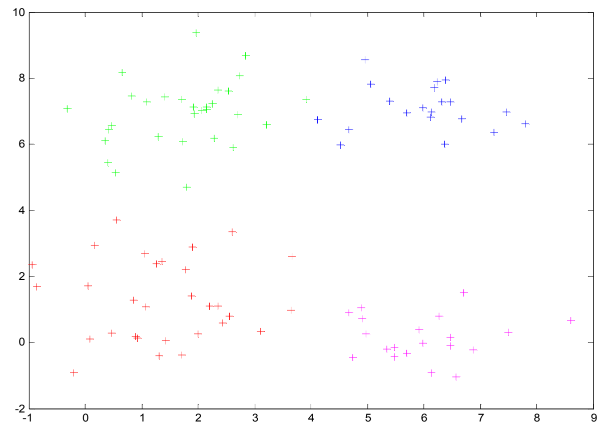
- **Construction of clusters through greedy agglomerative approach**
  - Merge pair of clusters in a bottom-up fashion, starting from singleton clusters
  - Merge clusters based on cluster (or linkage) distances. Defined in terms of point distances. **Examples:**

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

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

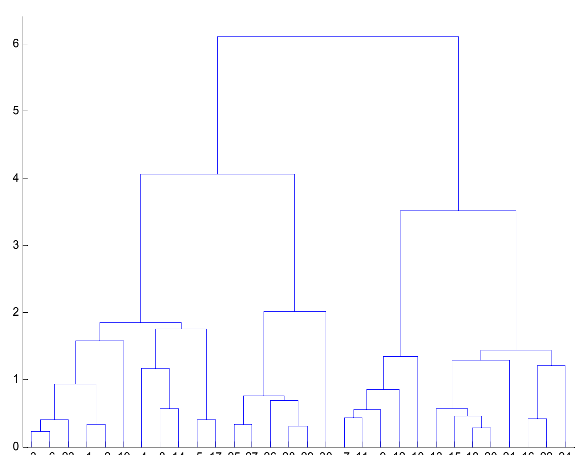
Mean distance $$d_{mean}(C_i, C_j) = \left| \frac{1}{|C_i|} \sum_i p_i - \frac{1}{|C_j|} \sum_j q_j \right|$$

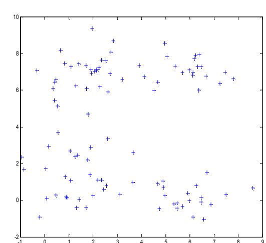
Hierarchical clustering example



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

Other clustering methods

- **Spectral clustering**
  - Uses similarity matrix
- **Multidimensional scaling**
  - techniques often used in data visualization for exploring similarities or dissimilarities in data.