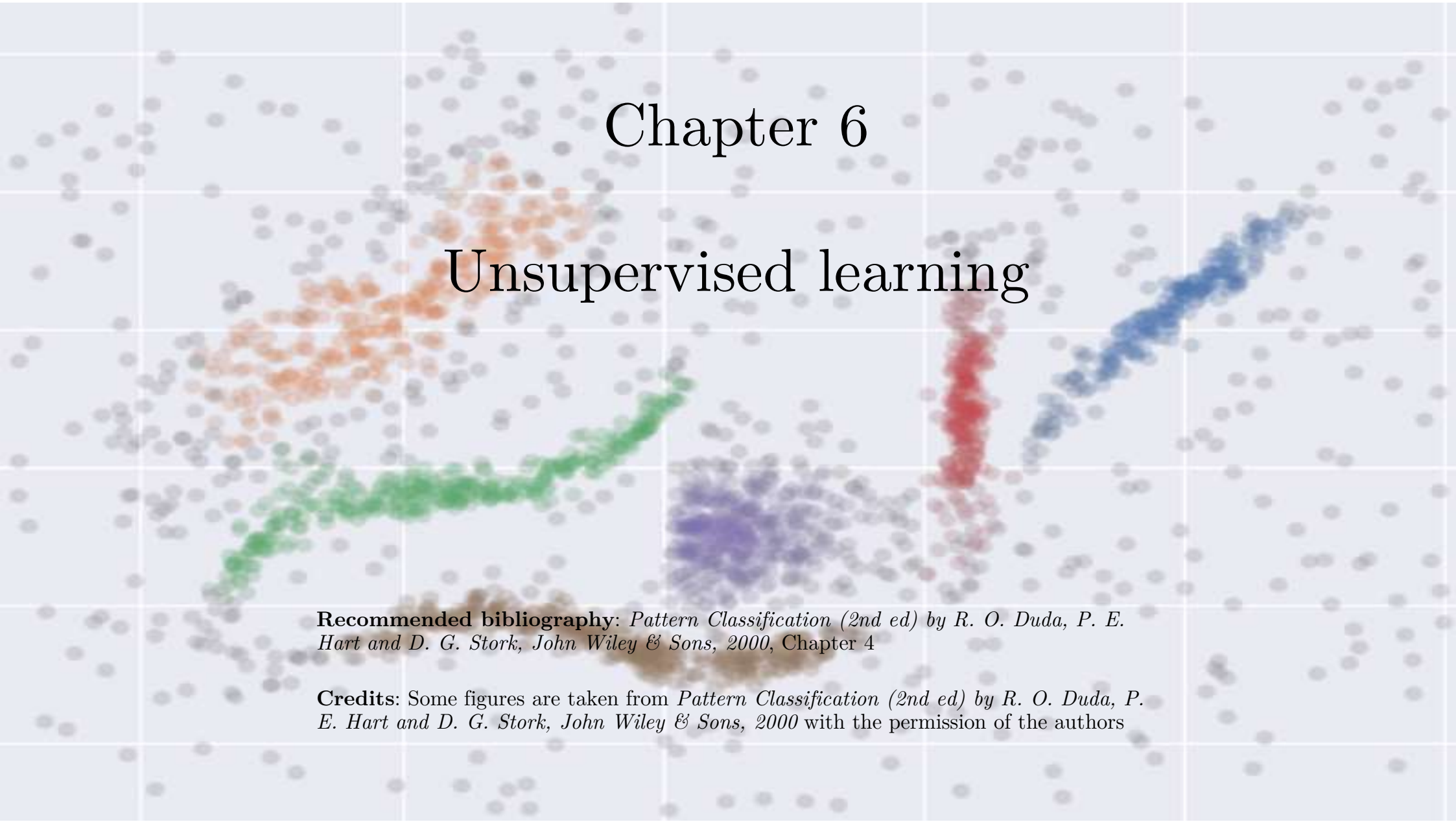


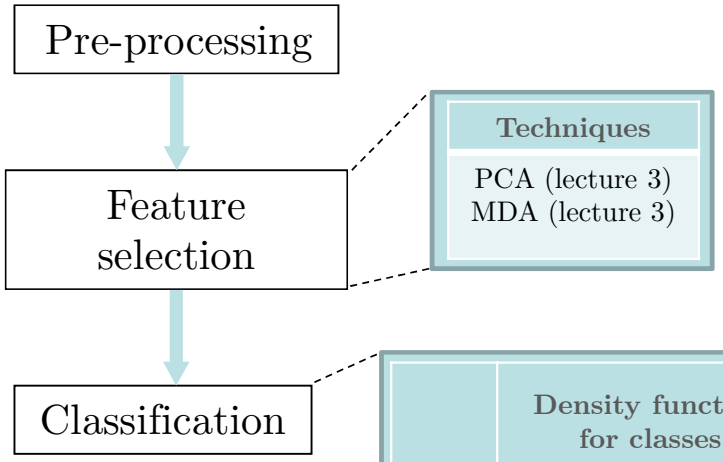
Chapter 6

Unsupervised learning

Recommended bibliography: *Pattern Classification (2nd ed) by R. O. Duda, P. E. Hart and D. G. Stork, John Wiley & Sons, 2000*, Chapter 4

Credits: Some figures are taken from *Pattern Classification (2nd ed) by R. O. Duda, P. E. Hart and D. G. Stork, John Wiley & Sons, 2000* with the permission of the authors

Course overview

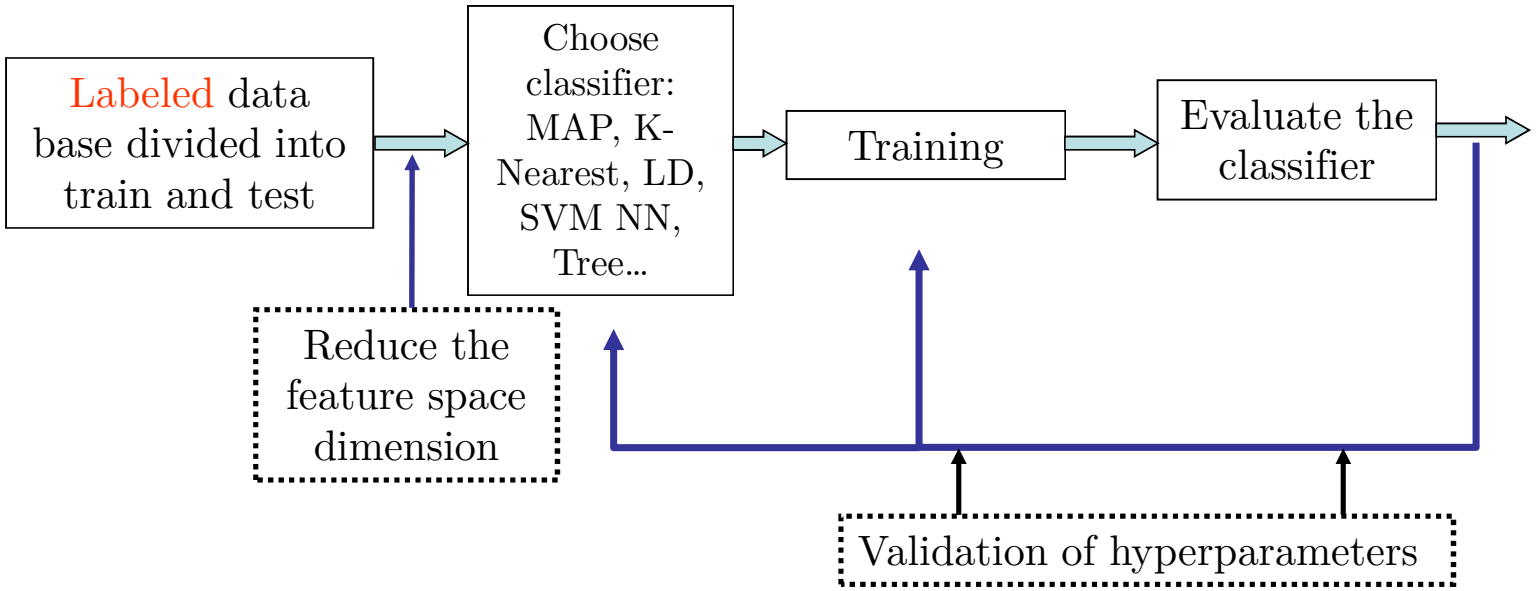


	Density function for classes	Classifier
Supervised	Parametrized Gaussian	Linear discriminant Quadratic discriminant[1] (lecture 2)
	Non-parametric	K-nearest neighbours[2] Fuzzy logic (lecture 4)
	Unknown	SVM[3] Neural networks[3] Decision trees[4] (lecture 4)
Unsupervised	Parametrized Gaussian	K-means (lecture 6)
	Unknown	Clustering (lecture 6)

1. Useful only if covariance matrices are not rank deficient.
2. Useful with the number of features is very large, even larger that the number of training vectors.
3. Imposes a structure to the classifier irrespective of the training data base.
4. Useful when non-numeric features are present.

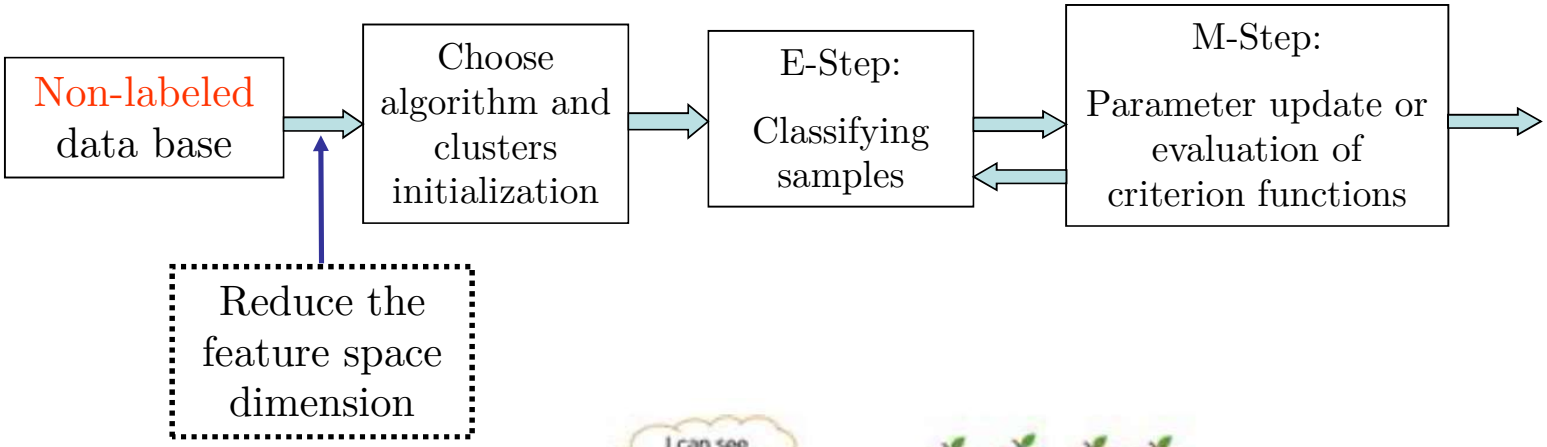
SUPERVISED METHODS

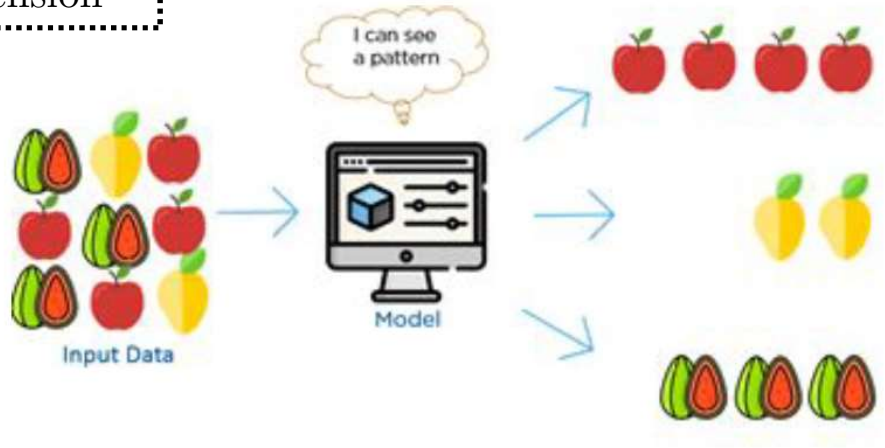
Labeled database



UNSUPERVISED METHODS

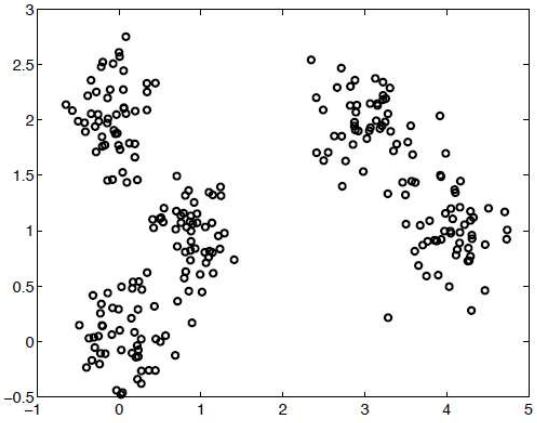
Non-labeled database



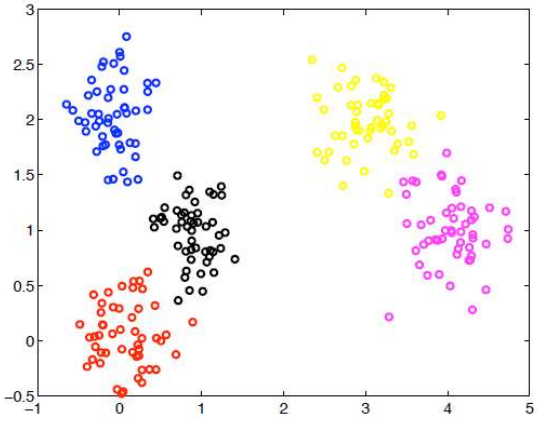


Goal: Find structure in the data by isolating groups of vectors that are similar in some well-defined sense...

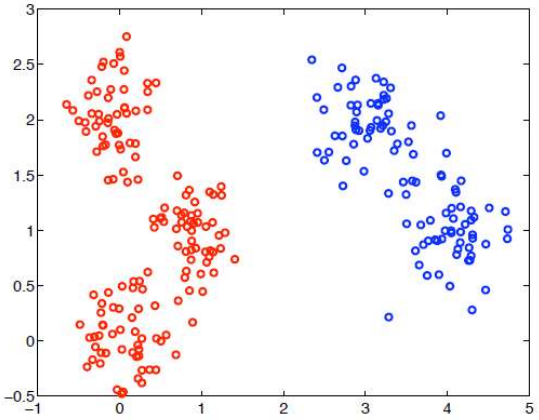
Original data set



Possible clustering

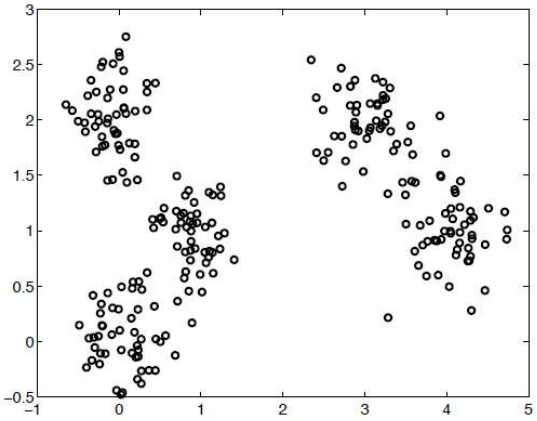


Possible clustering

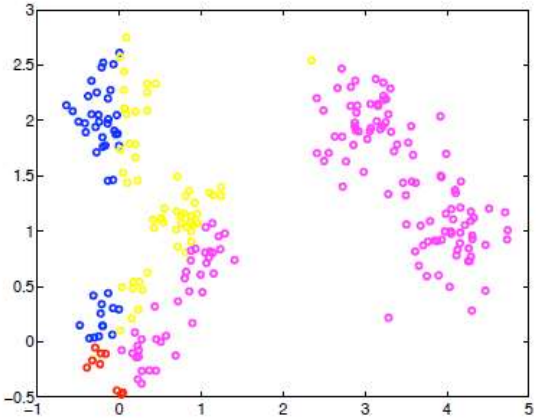


Goal: Find structure in the data by isolating groups of vectors that are similar in some well-defined sense...

Original data set



Non-desired clustering



Some good reasons to learn from unlabeled data...

- Collecting and labeling a large data base can be very costly
- In a supervised problem, model each class as a mixture of clusters $f(\mathbf{x}|\omega)$, and find posterior probabilities than can be used for categorisation by another classifier (e.g. MAP, MBR, NN,...)
- Gain some insight on the structure of the data beyond what can be obtained from scatter plots
- Dimensionality reduction
- Anomaly detection (detection of outlier vectors)
- Tracking the characteristics of the patterns in time: some patterns may jump from one class to another

Two approaches...

- **Parametric methods** assume a certain density function for the clusters
- **Non-parametric methods**, based on some similarity measure of the vectors to be clustered

CONTENTS

1.1 MIXTURE DENSITIES AND IDENTIFIABILITY

Assumptions:

1. The number c of clusters

2. Prior probabilities for each cluster are unknown (mixing parameters). $\Pr(\omega_j) \quad j = 1,\ldots,c$

3. The form of the class-conditional probabilities densities are known $f_{\mathbf{x}|\omega_j,\theta_j}\left(\mathbf{x}\middle|\omega_j,\boldsymbol{\theta}_j\right)$

4. The values for parameters are unknown $\boldsymbol{\theta} = [\theta_1,\ldots,\theta_c]$

5. The category labels are unknown: unsupervised

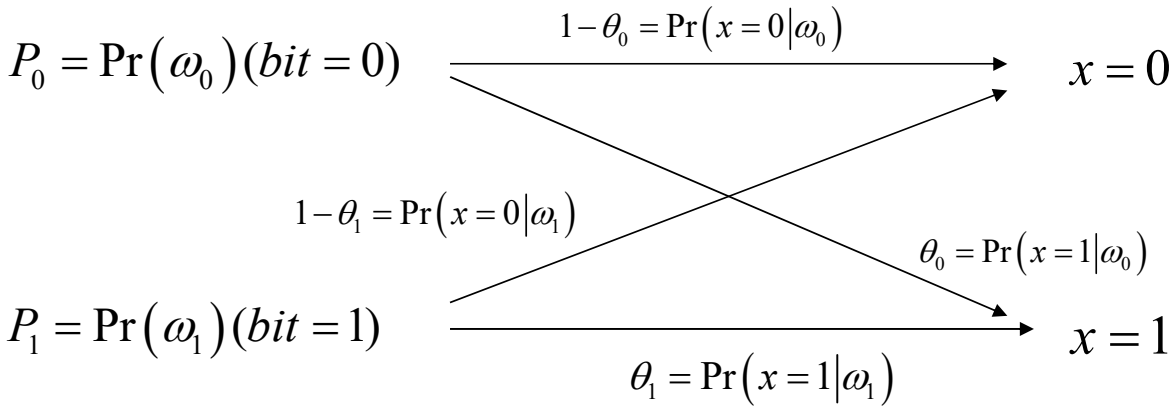
Mixture density:

$$f_{\mathbf{x}|\boldsymbol{\theta}}\left(\mathbf{x}\middle|\boldsymbol{\theta}\right)=\sum_{j=1}^{c} f_{\mathbf{x}|\omega_j,\boldsymbol{\theta}_j}\left(\mathbf{x}\middle|\omega_j,\boldsymbol{\theta}_j\right)\Pr\left(\omega_j\right)$$

1. For the moment, assumed that only the parameter vector $\boldsymbol{\theta}$ is unknown (priors are known).
2. Necessary conditions for identifiability:

$$f_{\mathbf{x}|\theta}\left(\mathbf{x}\middle|\boldsymbol{\theta}\right)\neq f_{\mathbf{x}|\theta}\left(\mathbf{x}\middle|\boldsymbol{\theta}'\right) \qquad \boldsymbol{\theta}=\left[\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_c\right]$$

Example 1: Identifiability in the binary symmetric channel



Vector of parameters: $\boldsymbol{\theta} = \begin{pmatrix} \theta_0 \\ \theta_1 \end{pmatrix}$

Mixture density (probability):

$$\Pr(x|\boldsymbol{\theta}) = P_0 \theta_0^{\,x} (1-\theta_0)^{1-x} + P_1 \theta_1^{\,x} (1-\theta_1)^{1-x}$$

1.2 ML ESTIMATES AND THE EM-ALGORITHM

Likelihood of the statistical independent observed samples:

$$D = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$$

$$f_D(D|\boldsymbol{\theta}) = \prod_{k=1}^{n} f_{\mathbf{x}}(\mathbf{x}_k|\boldsymbol{\theta}) \qquad l = \sum_{k=1}^{n} \ln f_{\mathbf{x}}(\mathbf{x}_k|\boldsymbol{\theta})$$

$$f_{\mathbf{x}}(\mathbf{x}_k|\boldsymbol{\theta}) = \sum_{j=1}^{c} f_{\mathbf{x}}(\mathbf{x}_k|\omega_j, \boldsymbol{\theta}_j)\Pr(\omega_j)$$

Assume the unknowns $\boldsymbol{\theta}_i$, $\boldsymbol{\theta}_j$ are independent variables, then the ML criterion provides c equations:

$$\nabla_{\boldsymbol{\theta}_i} l = \sum_{k=1}^{n} \Pr(\omega_i|\mathbf{x}_k, \hat{\boldsymbol{\theta}}) \nabla_{\boldsymbol{\theta}_i} \ln f_{\mathbf{x}}(\mathbf{x}_k|\omega_i, \hat{\boldsymbol{\theta}}_i) = \mathbf{0} \qquad i = 1, ..., c$$

Proof:

$$\nabla_{\boldsymbol{\theta}_i} l = \sum_{k=1}^{n} \frac{1}{f_{\mathbf{x}}\left(\mathbf{x}_k \middle| \boldsymbol{\theta}\right)} \nabla_{\boldsymbol{\theta}_i} f_{\mathbf{x}}\left(\mathbf{x}_k \middle| \boldsymbol{\theta}\right)$$

$$= \sum_{k=1}^{n} \frac{1}{f_{\mathbf{x}}\left(\mathbf{x}_k \middle| \boldsymbol{\theta}\right)} \nabla_{\boldsymbol{\theta}_i} \left(\sum_{j=1}^{c} f_{\mathbf{x}}\left(\mathbf{x}_k \middle| \omega_j, \boldsymbol{\theta}_j\right) \Pr(\omega_j) \right)$$

$$= \sum_{k=1}^{n} \frac{1}{f_{\mathbf{x}}\left(\mathbf{x}_k \middle| \boldsymbol{\theta}\right)} \nabla_{\boldsymbol{\theta}_i} \left(f_{\mathbf{x}}\left(\mathbf{x}_k \middle| \omega_i, \boldsymbol{\theta}_i\right) \Pr(\omega_i) \right)$$

$$= \sum_{k=1}^{n} \frac{\Pr(\omega_i)}{f_{\mathbf{x}}\left(\mathbf{x}_k \middle| \boldsymbol{\theta}\right)} \nabla_{\boldsymbol{\theta}_i} f_{\mathbf{x}}\left(\mathbf{x}_k \middle| \omega_i, \boldsymbol{\theta}_i\right) =$$

Bayes theorem

$$= \sum_{k=1}^{n} \Pr\left(\omega_i \middle| \mathbf{x}_k, \boldsymbol{\theta}\right) \nabla_{\boldsymbol{\theta}_i} \ln f_{\mathbf{x}}\left(\mathbf{x}_k \middle| \omega_i, \boldsymbol{\theta}_i\right) = \mathbf{0} \qquad i = 1, \ldots, c$$

The EM algorithm, including estimation of the unknown prior probabilities: (convergence guaranteed, without proof)

1. Compute conditioned probability for classes (Bayes rule)

$$\hat{\Pr}\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\theta}} \right.\right) = \frac{f_{\mathbf{x}}\left(\mathbf{x}_k \left| \omega_i, \hat{\boldsymbol{\theta}}_i \right.\right)\hat{\Pr}\left(\omega_i\right)}{\sum_{j=1}^{c} f_{\mathbf{x}}\left(\mathbf{x}_k \left| \omega_j, \hat{\boldsymbol{\theta}}_j \right.\right)\hat{\Pr}\left(\omega_j\right)}$$

Only depends on parameters of class i

2. Estimate vector parameter estimates (gradient equal to zero)

$$\sum_{k=1}^{n} \hat{\Pr}\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\theta}} \right.\right) \nabla_{\boldsymbol{\theta}_i} \ln f_{\mathbf{x}}\left(\mathbf{x}_k \left| \omega_i, \hat{\boldsymbol{\theta}}_i \right.\right) = \mathbf{0} \quad i = 1, \ldots, c$$

ML criterion, averaged with estimated a posteriori probabilities

3. Estimate prior probabilities

$$\hat{\Pr}\left(\omega_i\right) = \frac{1}{n}\sum_{k=1}^{n} \hat{\Pr}\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\theta}} \right.\right)$$

This is a chicken-and-egg problem, solved in an iterative way...

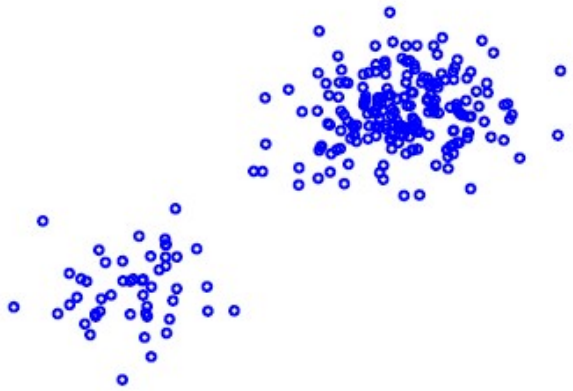
Application to Gaussian distributions:

$$\ln f_{\mathbf{x}}\left(\mathbf{x}_k \left| \omega_i, \boldsymbol{\theta}_i \right.\right) = \frac{1}{2}\ln\left|\mathbf{C}_i^{-1}\right| - \frac{d}{2}\ln(2\pi) - \frac{1}{2}\left(\mathbf{x}_k - \boldsymbol{\mu}_i\right)^T \mathbf{C}_i^{-1}\left(\mathbf{x}_k - \boldsymbol{\mu}_i\right)$$

Parameters to estimate: $\boldsymbol{\theta} = \left(\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_c\right)$ $\qquad \boldsymbol{\theta}_i = \left(\boldsymbol{\mu}_i, \mathbf{C}_i\right)$

and the prior probabilities $\Pr\left\{\omega_i\right\} \quad i = 1, ..., c$

An example for $c = 2$



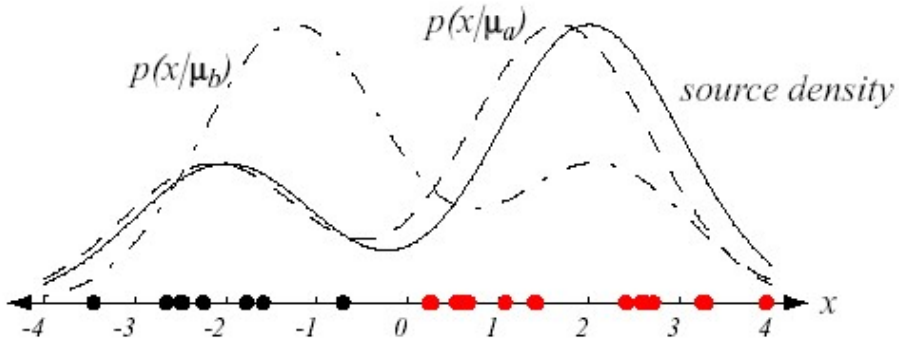
Case 1: Unknown mean vectors and prior probabilities, covariance known

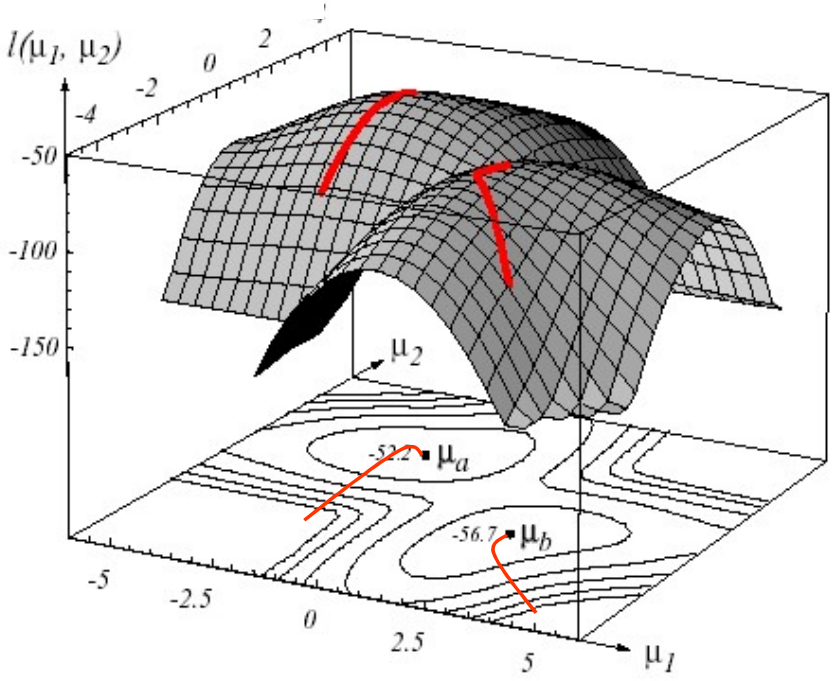
$$\sum_{k=1}^{n} \Pr\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\theta}} \right.\right) \nabla_{\boldsymbol{\mu}_i} \ln f_{\mathbf{x}}\left(\mathbf{x}_k \left| \omega_i, \boldsymbol{\mu}_i \right.\right) = \sum_{k=1}^{n} \Pr\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\theta}} \right.\right) \mathbf{C}_i^{-1}\left(\mathbf{x}_k - \boldsymbol{\mu}_i\right) = 0 \quad i = 1, \ldots, c$$

After multiplying by $\mathbf{C}_i$ and rearranging...

$$\hat{\boldsymbol{\mu}}_i = \frac{\sum_{k=1}^{n} \Pr\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\mu}} \right.\right) \mathbf{x}_k}{\sum_{k=1}^{n} \Pr\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\mu}} \right.\right)} \longrightarrow \hat{\boldsymbol{\mu}}_i\left(j+1\right) = \frac{\sum_{k=1}^{n} \Pr\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\mu}}\left(j\right) \right.\right) \mathbf{x}_k}{\sum_{k=1}^{n} \Pr\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\mu}}\left(j\right) \right.\right)}$$

This is a hill-climbing procedure that may converge to local maxima, also called Expectation-Maximization (EM) algorithm.





On each iteration, the EM-algorithm monotonically increases the (log-) likelihood of the n samples:

$$\ln f_{\mathbf{x}}\left(D|\boldsymbol{\theta}\right)$$

Case 2: Unknown mean vectors, covariance matrices and prior probabilities

1. **Expectation (E-Step): estimate the probability of belonging to each class**

$$\hat{\Pr}\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\theta}} \right.\right) = \frac{\left|\hat{\mathbf{C}}_i^{-1}\right|^{\frac{1}{2}} \exp\left(-\frac{1}{2}\left(\mathbf{x}_k - \hat{\boldsymbol{\mu}}_i\right)^T \hat{\mathbf{C}}_i^{-1}\left(\mathbf{x}_k - \hat{\boldsymbol{\mu}}_i\right)\right)\hat{\Pr}\left(\omega_i\right)}{\sum_{j=1}^{c}\left|\hat{\mathbf{C}}_j^{-1}\right|^{\frac{1}{2}} \exp\left(-\frac{1}{2}\left(\mathbf{x}_k - \hat{\boldsymbol{\mu}}_j\right)^T \hat{\mathbf{C}}_j^{-1}\left(\mathbf{x}_k - \hat{\boldsymbol{\mu}}_j\right)\right)\hat{\Pr}\left(\omega_j\right)}$$

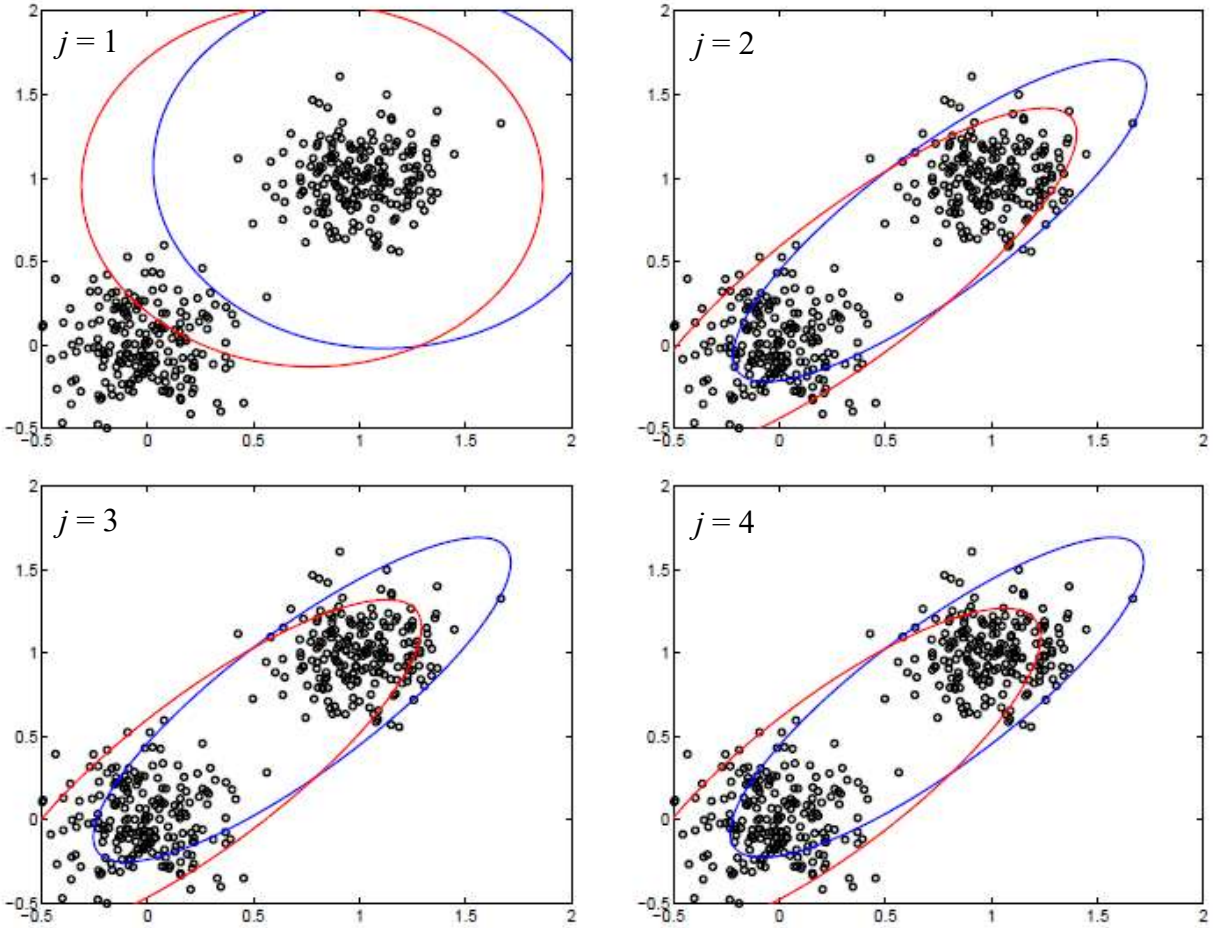
2. **Maximization (M-Step): estimate the parameters of each class via ML**

$$\hat{\boldsymbol{\mu}}_i = \frac{\sum_{k=1}^{n} \hat{\Pr}\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\theta}} \right.\right)\mathbf{x}_k}{\sum_{k=1}^{n} \hat{\Pr}\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\theta}} \right.\right)} \qquad \hat{\mathbf{C}}_i = \frac{\sum_{k=1}^{n} \hat{\Pr}\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\theta}} \right.\right)\left(\mathbf{x}_k - \hat{\boldsymbol{\mu}}_i\right)\left(\mathbf{x}_k - \hat{\boldsymbol{\mu}}_i\right)^T}{\sum_{k=1}^{n} \hat{\Pr}\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\theta}} \right.\right)}$$

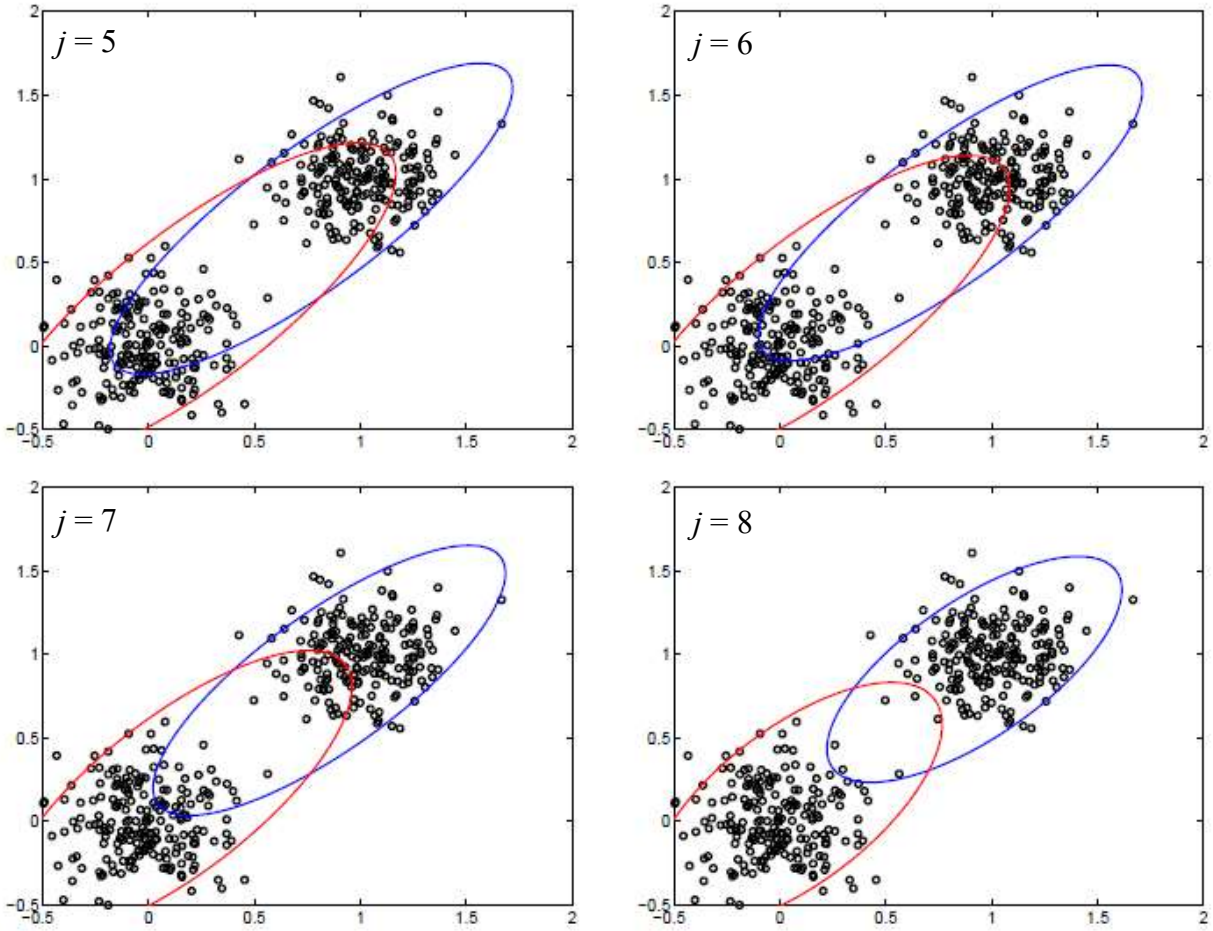
$$\hat{\Pr}\{\omega_i\} = \frac{1}{n}\sum_{k=1}^{n} \hat{\Pr}\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\theta}} \right.\right)$$

Iterations between E and M steps stop when estimated parameters do not change.

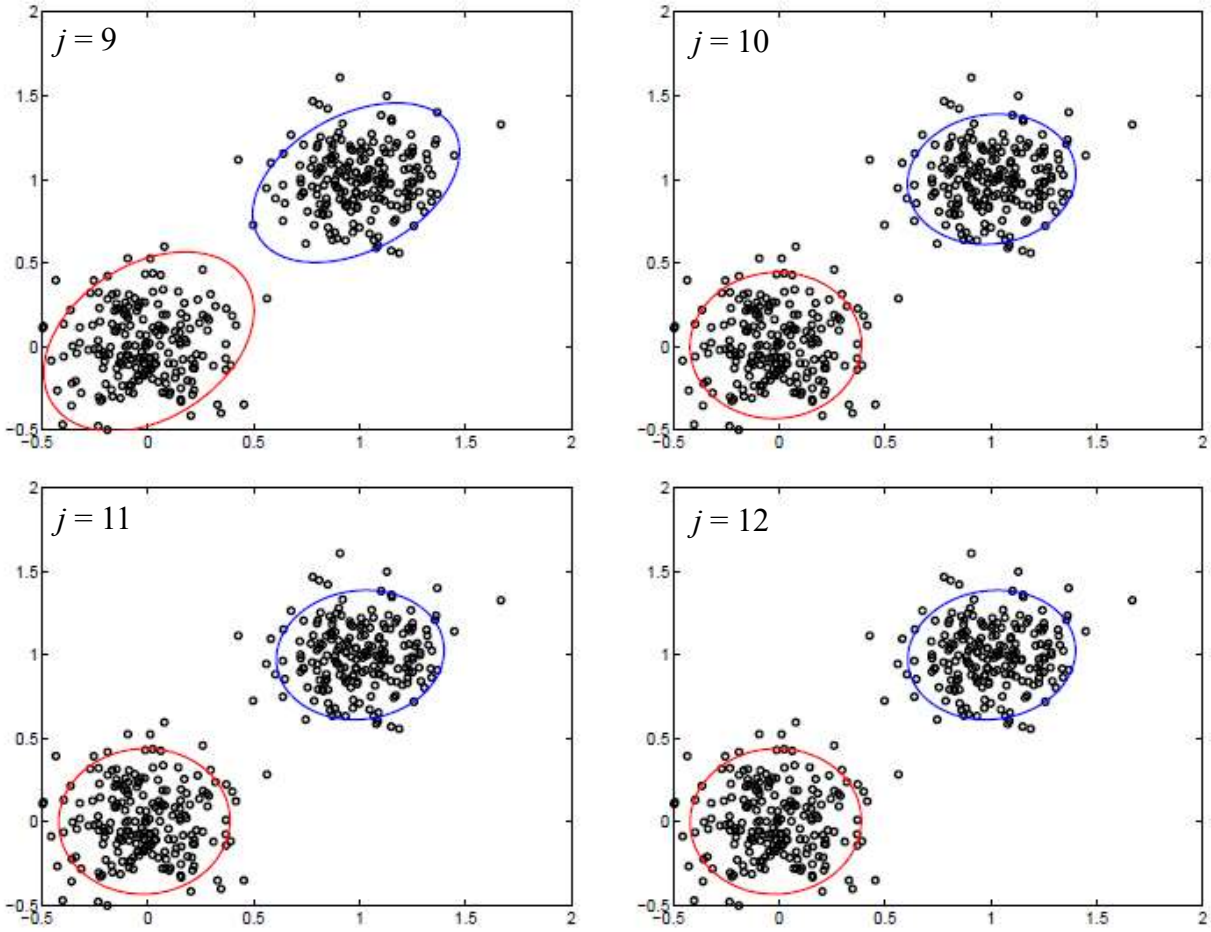
Example 2: sequence of 12 EM iterations for two Gaussian classes



From Tommi Jaakkola, MIT AI Lab



From Tommi Jaakkola, MIT AI Lab



From Tommi Jaakkola, MIT AI Lab

How to select the appropriate number of clusters c?

We can find m that minimize the following asymptotic approximation to the description length criterion:

$$c = \arg\min_{m} DL(m)$$

$$DL(m) \simeq -\ln f_{\mathbf{x}}\left(D\middle|\hat{\boldsymbol{\theta}}_{m,ML}\right) + \frac{d_m}{2}\ln(n)$$

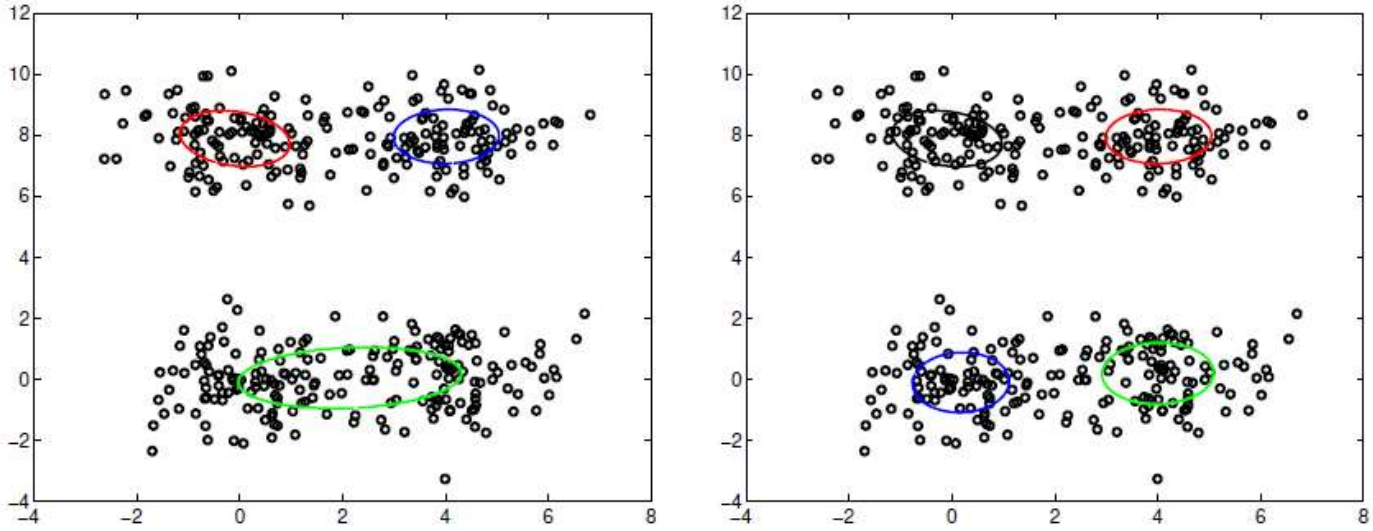
Number of parameters in the m-clusters mixture

Number of vectors in D

ML estimates of the m-clusters mixture

This is called the Bayesian Information Criterion (BIC).

Example 3: number of mixture components




```
m=1, -logP(data)=2017.38, penalty=14.98, DL=2032.36
m=2, -logP(data)=1712.69, penalty=32.95, DL=1745.65
m=3, -logP(data)=1678.56, penalty=50.93, DL=1729.49
m=4, -logP(data)=1649.08, penalty=68.90, DL=1717.98
```

$\frac{dm}{2} \ln(n)$

From Tommi Jaakkola, MIT AI Lab

1.3 K-MEANS CLUSTERING

We can reduce convergence time by doing a hard classification of samples at the E-step. In a simple case of equal diagonal covariance matrices…

$$\boldsymbol{\theta}_i = \boldsymbol{\mu}_i \qquad \mathbf{C}_i = \sigma^2 \mathbf{I}$$

Euclidean distance of $\mathbf{x}_k$ to each estimated $\boldsymbol{\mu}_j$, obtained from posterior probability in slide 15 (if ignoring $\hat{\Pr}(\omega_i)$)

$$\hat{\Pr}\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\mu}}_i \right.\right) = \begin{cases} 1 & d_e\left(\mathbf{x}_k, \hat{\boldsymbol{\mu}}_i\right) < d_e\left(\mathbf{x}_k, \hat{\boldsymbol{\mu}}_j\right) \quad \forall j \neq i \\ 0 & \text{otherwise} \end{cases}$$

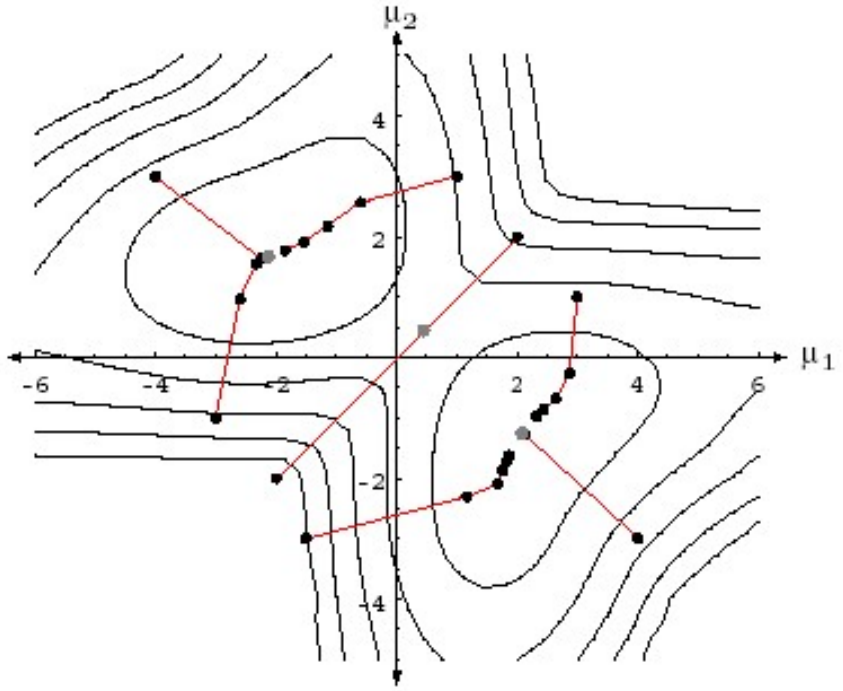
$$\hat{\Pr}\{\omega_i\} = \frac{1}{n}\sum_{k=1}^{n} \hat{\Pr}\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\mu}}_i \right.\right) = \frac{n_i}{n}$$

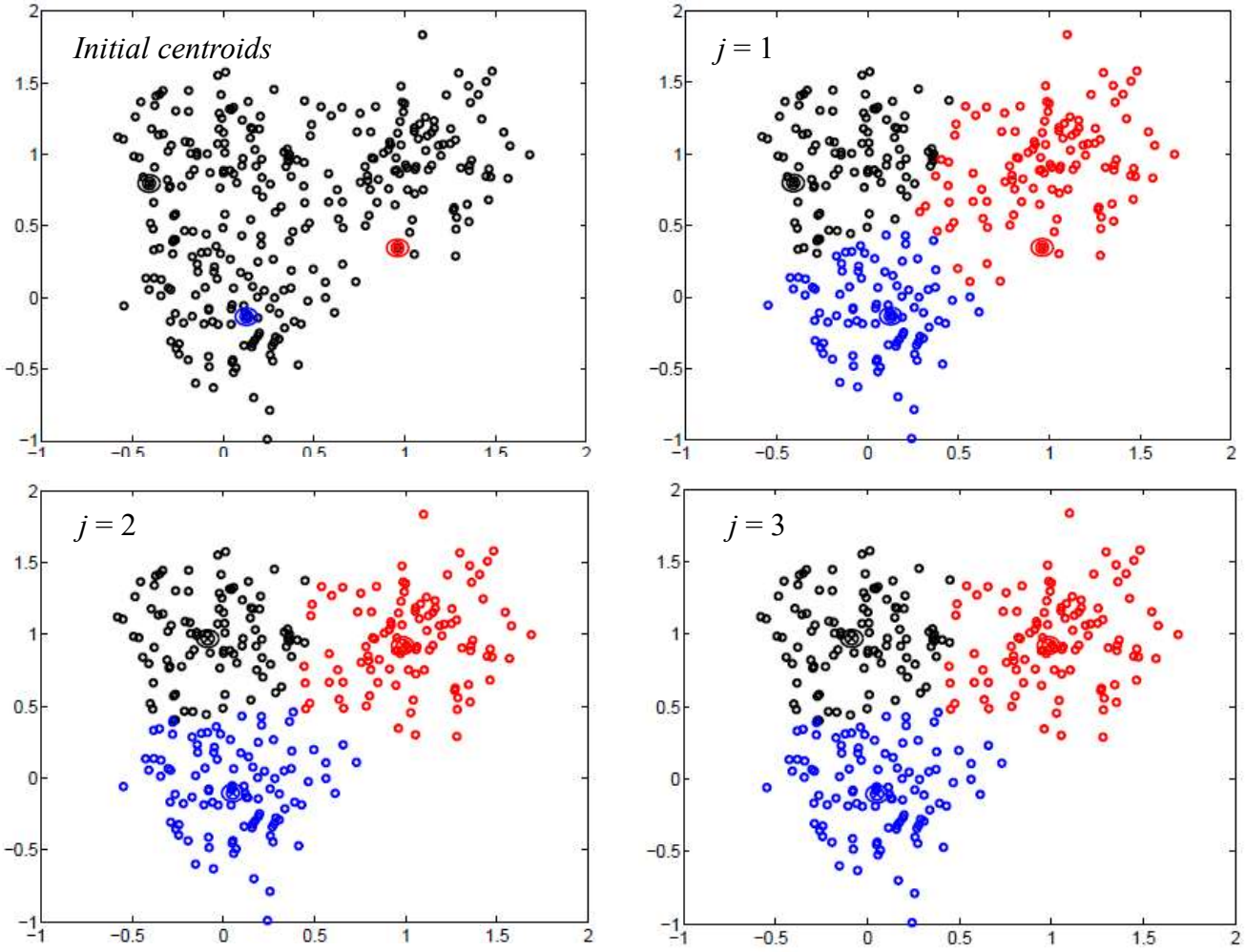
$$\hat{\boldsymbol{\mu}}_i = \frac{1}{n_i}\sum_{k=1}^{n} \hat{\Pr}\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\mu}}_i \right.\right) \mathbf{x}_k$$

- **K-Means clustering algorithm**

```
1 begin initialize n, c, μ1, μ2, ..., μc
2         do classify n samples according to nearest μi      % E-step
3            recompute μi                                    % M-step
4         until no change in μi
5   return μ1, μ2, ..., μc
6 end
```

Convergence is achieved in a few iterations




Initial centroids
j = 1
j = 2
j = 3

1.4 FUZZY K-MEANS CLUSTERING

We can relax the condition of every sample belonging to just one cluster, and assume some fuzzy membership, related to the a posteriori probabilities $\hat{\Pr}\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\theta}} \right.\right)$

We seek to minimize a heuristic function:

integer value larger than 1 to adjust blending of clusters

$$J_{Fuzzy} = \sum_{i=1}^{c} \sum_{k=1}^{n} \hat{\Pr}\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\theta}} \right.\right)^b d_e\left(\mathbf{x}_j, \hat{\boldsymbol{\mu}}_i\right)$$

leading to the following optimum (mean vector estimated only):

$$\hat{\boldsymbol{\mu}}_i = \frac{\sum_{k=1}^{n} \hat{\Pr}\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\theta}} \right.\right)^b \mathbf{x}_k}{\sum_{k=1}^{n} \hat{\Pr}\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\theta}} \right.\right)^b}$$

...for other values of b

For $b = 1$ we get the k-means, what happens for other values of b?

- $b = 2$ Higher weight when computing $\hat{\boldsymbol{\mu}}_i$ for those $\mathbf{x}_i$ with higher probability
- $b \to \infty$ $$\hat{\boldsymbol{\mu}}_i = \arg\max_{\mathbf{x}_k} \frac{\sum_{k=1}^{n} \hat{\Pr}\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\theta}}\right.\right)^b \mathbf{x}_k}{\sum_{k=1}^{n} \hat{\Pr}\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\theta}}\right.\right)^b}$$

Each mean is computed using a single vector $\mathbf{x}_k$

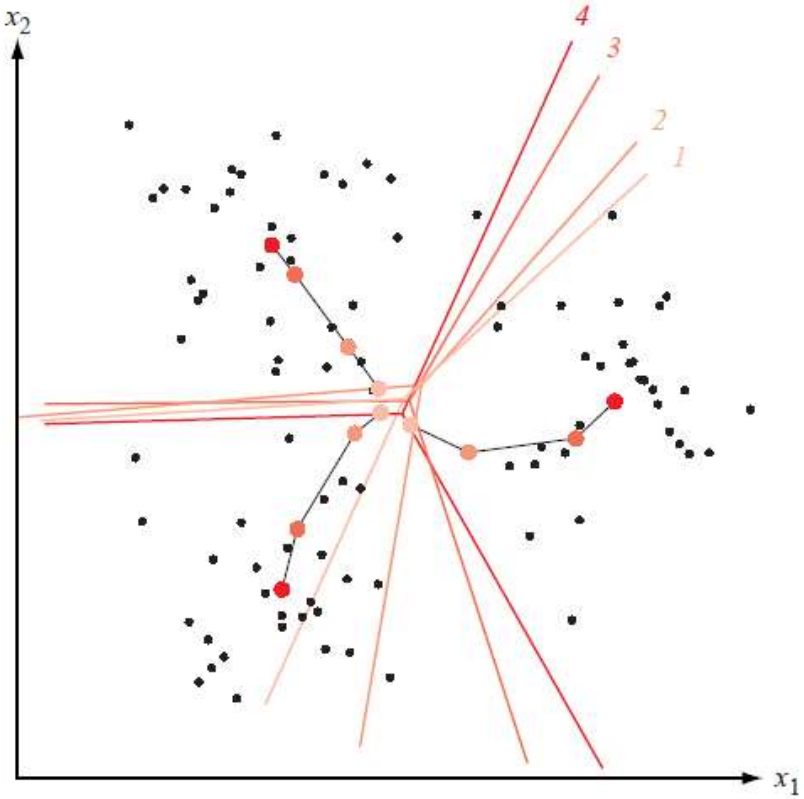
The value of b can be used to robustify the determination of centroids in the presence of outliers.

Exercise: Derive the estimate of $\hat{\Pr}\left(\omega_i \left| \mathbf{x}_k, \hat{\boldsymbol{\theta}}\right.\right)$, have in mind the sum from $i = 1$ to c must be 1.

```
1 begin initialize n, μ_1, ..., μ_c, P(ω_i | x_j), i = 1..., c; j = 1, ..., n
2         normalize proabilities of cluster memberships
3         do classify n samples according to nearest μ_i
4             recompute μ_i
5             recompute P(ω_i | x_j)
6         until no change in μ_i and P(ω_i | x_j)
7   return μ_1, μ_2, ..., μ_c
8 end
```

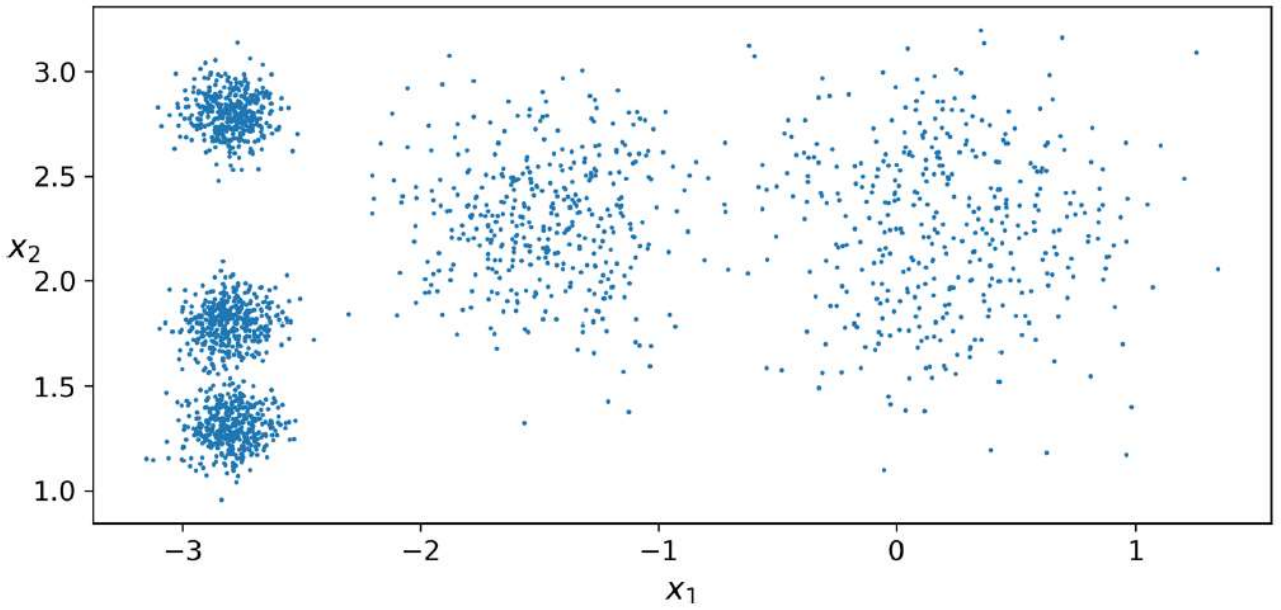
Warning! If c is incorrectly specified, serious problems may arise

For an application in vector quantization of a n-dimensional real valued vector: *J. Proakis: "Digital Communications", Chapter 3: Source Coding.*



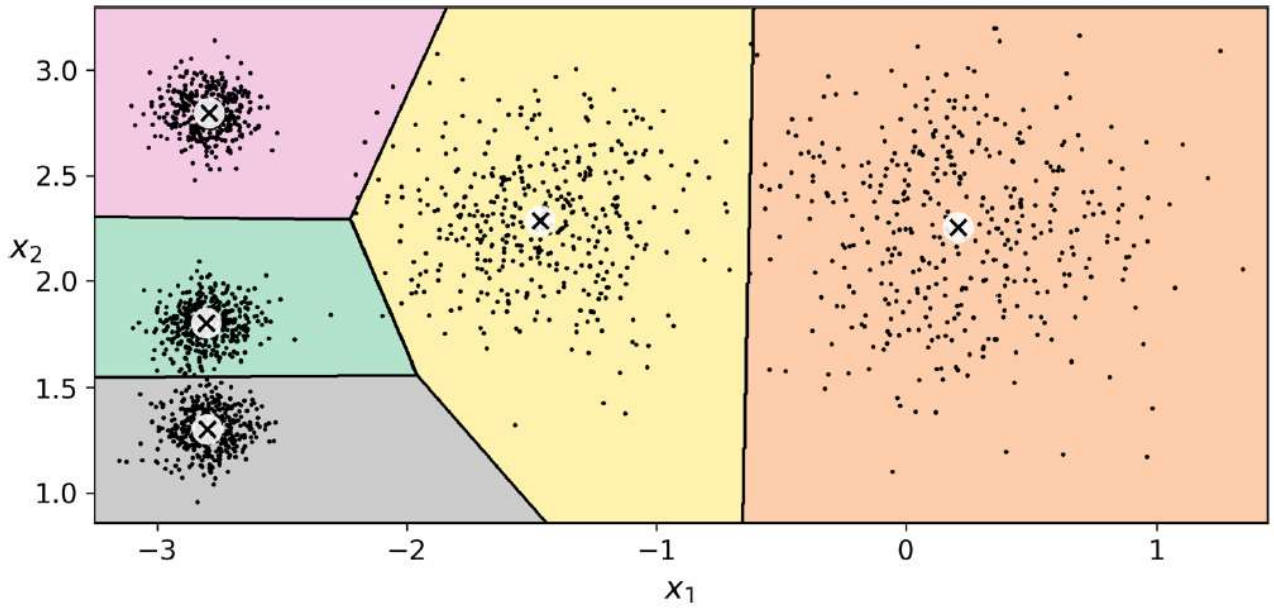
Evolution of the regions after 4 iterations of the fuzzy k-means (b=2).

K-means limitations



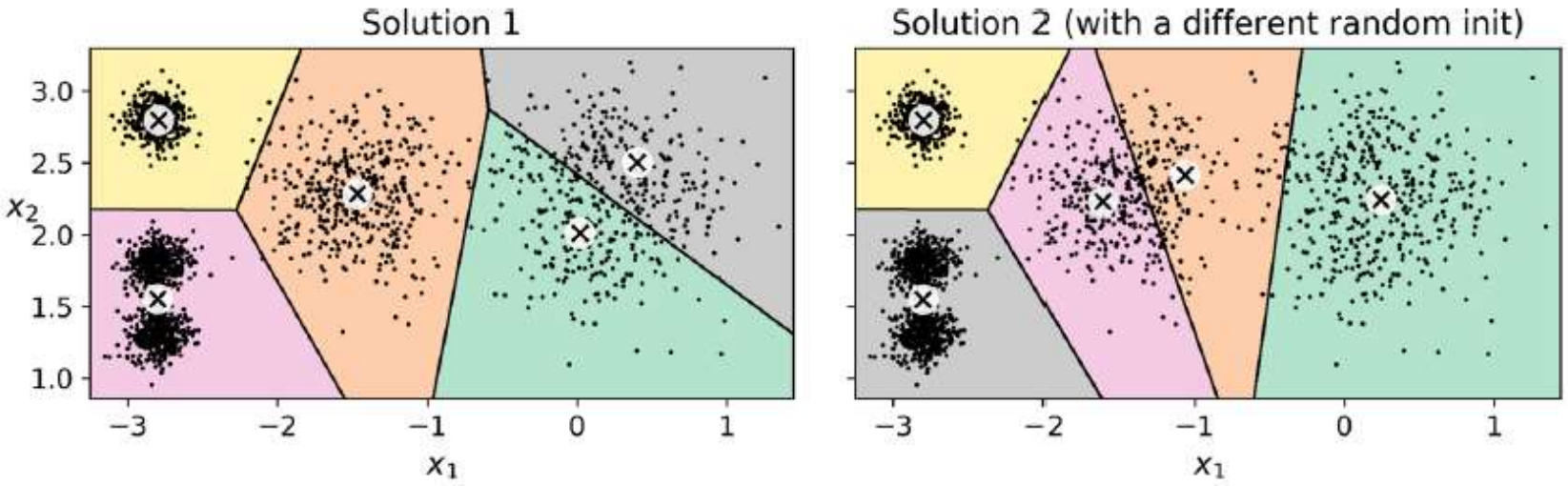
K-means limitations

Cluster size: difficulties if clusters of different sizes and densities.



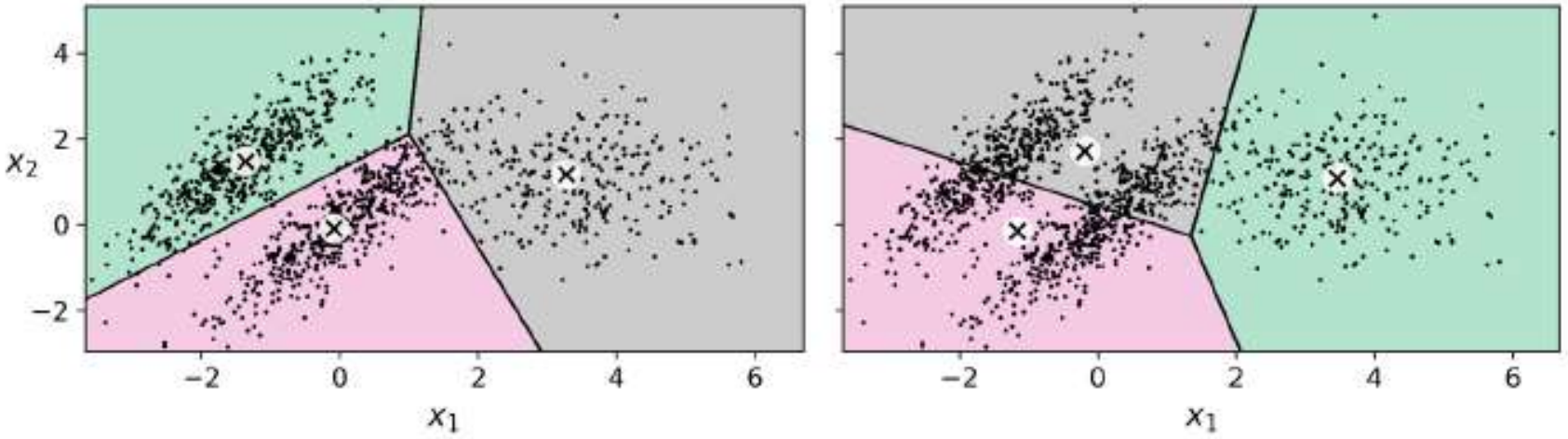
K-means limitations

Convergence is guaranteed, but possibly to local minima, depending on the initialization.



K-means limitations

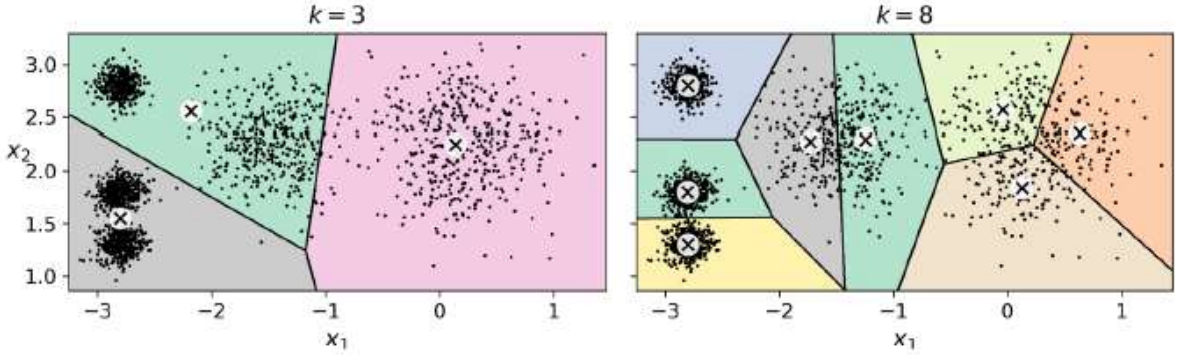
Cluster shape: difficulties if clusters are not of spherical shape.



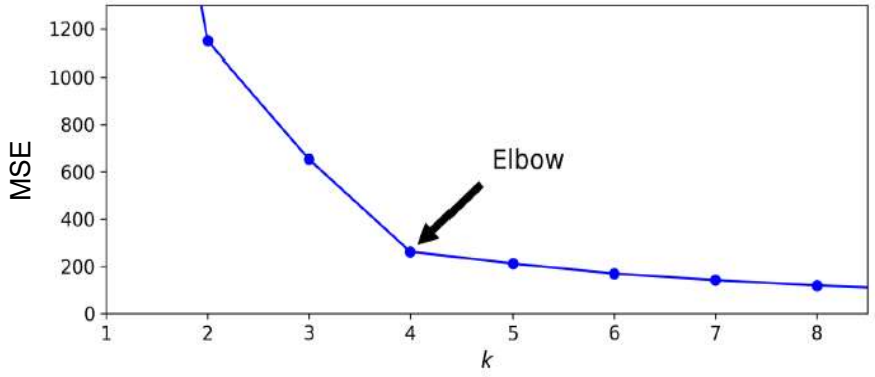
Gaussian mixture models with arbitrary $\mathbf{C}$ matrices will perform much better in this case (check the parametric models).

2.4 OPTIMUM NUMBER OF CLUSTERS

Let us revisit the estimation of the number of clusters. Examples of two bad selections:



The overall MSE as a function of the number of clusters is not a good criterion, as it is a decreasing function with k:



A more precise approach is to use the *mean silhouette score*:

1. For $h = 2$ to K
2. Compute the k-means clustering for h clusters
3. For $l = 1$ to h
4. Compute the silhouette score of each cluster as

$$a_i = \frac{1}{N_l} \sum_{\mathbf{x}_j \in \omega_l} d\left(\mathbf{x}_i, \mathbf{x}_j\right)$$

Average distance of vector $\mathbf{x}_i$ to other vectors within its own cluster

$$b_i = \frac{1}{N_m} \sum_{\mathbf{x}_j \in \omega_m} d\left(\mathbf{x}_i, \mathbf{x}_j\right)$$

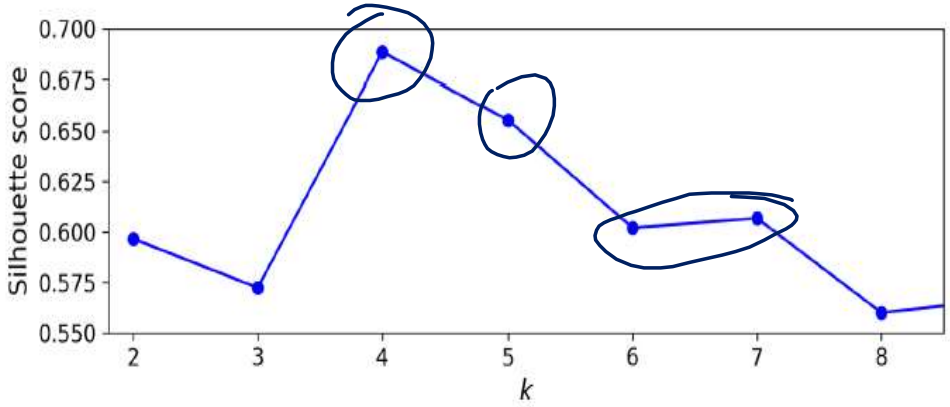
Average distance of vector $\mathbf{x}_i$ to other vectors of the nearest cluster m (the one minimizing b, which may be different for every vector $\mathbf{x}_i$)

$$S_l = \sum_{\mathbf{x}_i \in \omega_l} \left(b_i - a_i\right) / \max\left(a_i, b_i\right)$$

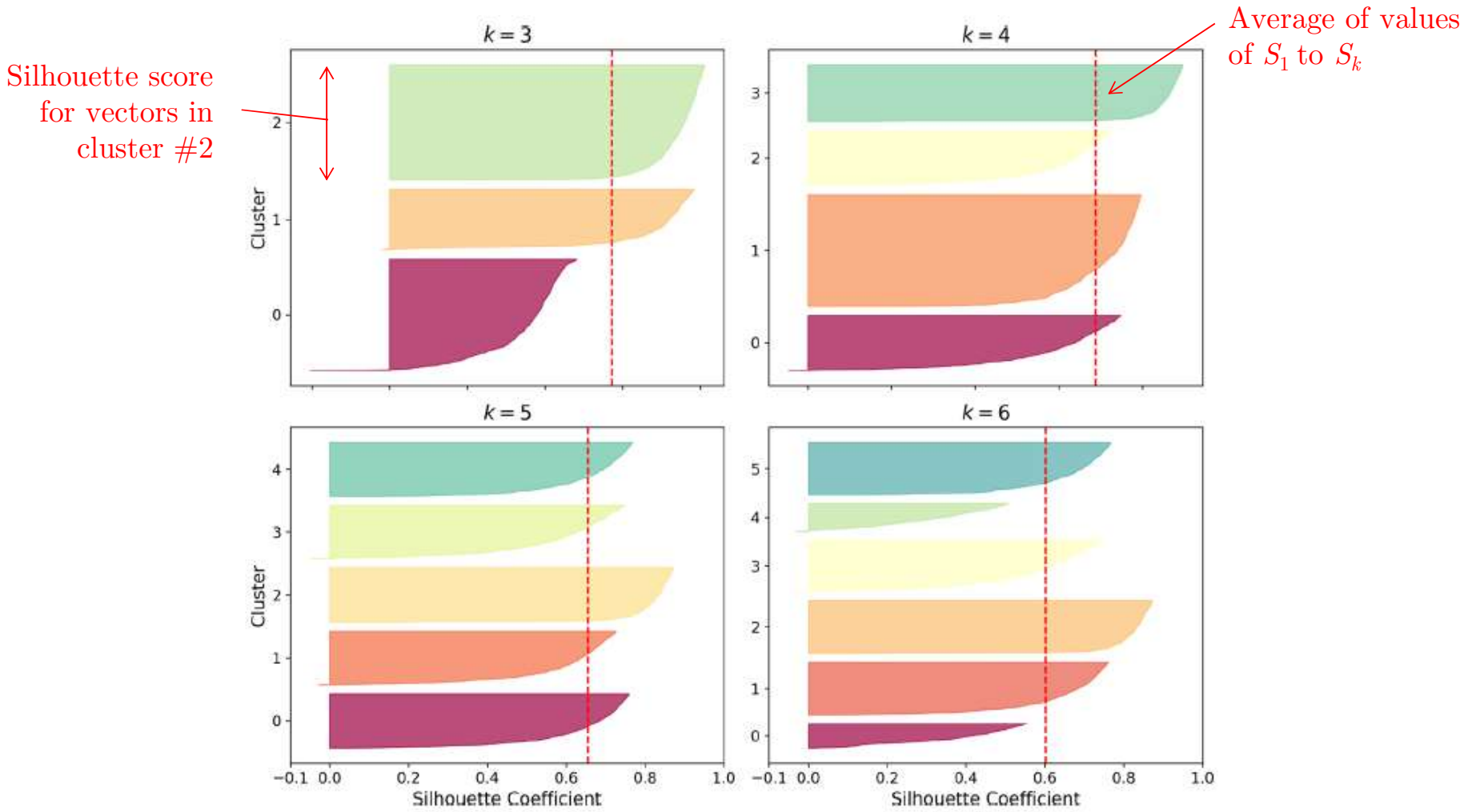
Silhouette score $\in$(-1,1) of vector $\mathbf{x}_i$

5. Average the values S_1 to S_h
6. The estimated number of clusters is the value in $\{2, K\}$ that maximizes this average

For the example above, it is confirmed that the good number of clusters is 5:



A more informative visualization is obtained when plotting every vector *silhouette coefficient sorted by the cluster*:



When many vectors in a cluster have lower silhouette coefficient than the average, the clustering is bad: those vectors are too close to other clusters.

CONTENTS

2.1 INTRODUCTION

- *Clusters* are clouds of points in a d-dimensional space.

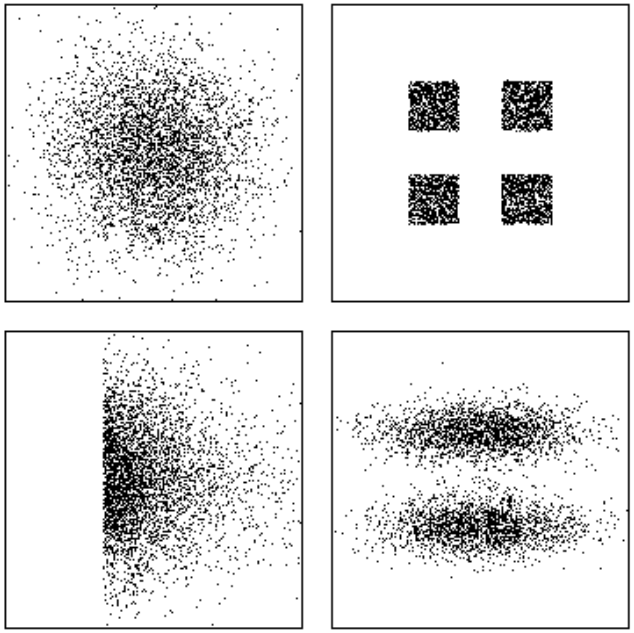
- *Normal Distribution*, sufficient statistics are the

  - *Mean sample* **m**: Locates de center of gravity of the cloud and it best represents all of the data in the sense of minimizing the sum of squared distances from **m** to the samples.

  - *Sample covariance matrix* **C**: denotes the amount the data scatters along various directions around **m**.

Sample mean vector and sample covariance matrix are not suficient statistics in a general case.

Distributions with identical mean and covariance:



Let us drop the Gaussian assumption for the clusters and adopt a non-parametic clustering approach:

Three key concepts in clustering procedures:

1. Data are grouped in clusters or groups of data points that posses strong internal **similarity**.

2. A **criterion function** is used to seek the grouping that extremizes it. To evaluate the partitioning of a set of samples into clusters, the similarity is measured between samples.

3. An **iterative algorithm** whereby vector is moved from cluster i to a cluster j only if the global criterion function decreases.

SIMILARITY MEASURES

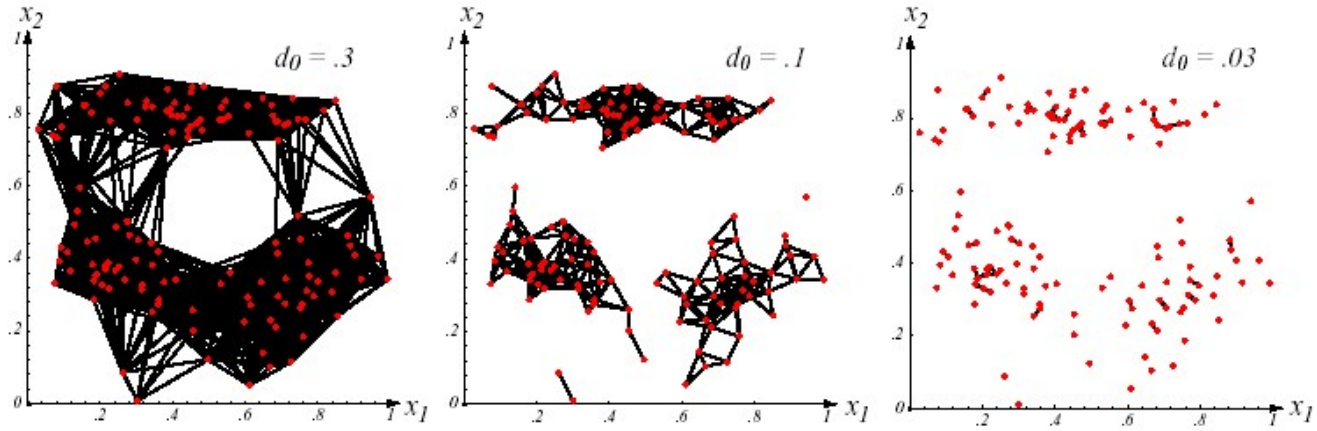
Two samples belong to the same cluster if $d(\mathbf{x}_i, \mathbf{x}_j) < d_o$

The choice of threshold d_o is critical!

- Minkowski Distance $d_q(\mathbf{x}_i, \mathbf{x}_j) = \left(\sum_{n=1}^{d} \left(x_i[n] - x_j[n] \right)^q \right)^{\frac{1}{q}}$
- Mahalanobis Distance $d_M^2(\mathbf{x}_i, \mathbf{x}_j) = \left(\mathbf{x}_i - \mathbf{x}_j \right)^T \boldsymbol{\Sigma}^{-1} \left(\mathbf{x}_i - \mathbf{x}_j \right)$

Advantages of Euclidean distance: clusters are invariant to rotation and translation.

Distance threshold affects the number and size of clusters:



typical intra-cluster distance < d_0 < typical inter-clusters distance

Data normalization prior to clustering:

- Each feature is translated to have zero mean
- Each feature is scaled to have unit variance
- PCA: axis coincide with the eigenvectors of the sample covariance matrix

After normalización and PCA, clusters are invariant to displacements, scale change and rotations

2.2 CRITERION FUNCTIONS

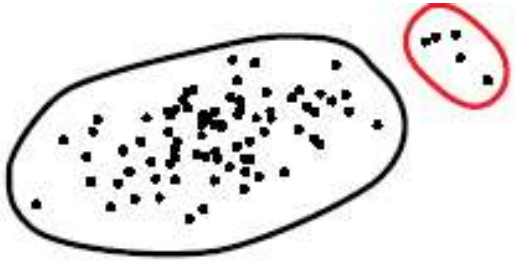
Criterion functions for clustering:

- Initial set $D = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\}$
- Partition into exactly c subsets $D_1, D_2, ..., D_c$
- Objective: To find the partition that extremizes the criterion function, e.g. sum of squared errors criterion

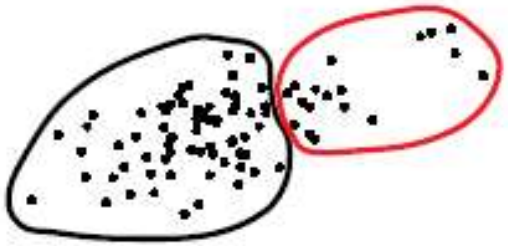
$$J_e = \sum_{i=1}^{c} \sum_{\mathbf{x} \in D_i} \|\mathbf{x} - \mathbf{m}_i\|^2$$

- $\mathbf{m}_i$ is the centroid, as the best representative of the samples in D_i.

It is appropriate when the clusters form compact clouds and uniform number of samples per cluster, otherwise it may divide natural clusters artificially:



J_e = large



J_e = small

DISTANCE-BASED CRITERIA

Other criterion function:

$$J_e = \frac{1}{2}\sum_{i=1}^{c} n_i \bar{s}_i$$

$$\bar{s}_i = \frac{1}{n_i^2}\sum_{\mathbf{x}\in D_i}\sum_{\mathbf{x}'\in D_i} \|\mathbf{x}-\mathbf{x}'\|^2 \qquad \text{Average distance in class } i$$

$$\bar{s}_i = \max_{\mathbf{x},\mathbf{x}'\in D_i} d_e(\mathbf{x},\mathbf{x}') \qquad \text{Maximum distance in class } i$$

$$\bar{s}_i = \min_{\mathbf{x},\mathbf{x}'\in D_i} d_e(\mathbf{x},\mathbf{x}') \qquad \text{Minimum distance in class } i$$

any similarity measure could be used

SCATTER-BASED CRITERIA

Sensible criterion: maximize distance among clusters and minimize in-cluster scatter

Scatter criteria:

Definitions used in clustering criteria...

- *Mean vector for the i cluster* $\mathbf{m}_i = \frac{1}{n_i}\sum_{\mathbf{x}\in D_i}\mathbf{x}$
- *Total mean vector* $\mathbf{m} = \frac{1}{n}\sum_{\mathbf{x}\in D}\mathbf{x} = \frac{1}{n}\sum_{i=1}^{c} n_i\mathbf{m}_i$
- *Scatter matrix for the i cluster* $\mathbf{S}_i = \sum_{\mathbf{x}\in D_i}(\mathbf{x}-\mathbf{m}_i)(\mathbf{x}-\mathbf{m}_i)^T$
- *Within-cluster scatter matrix* $\mathbf{S}_W = \sum_{i=1}^{c}\mathbf{S}_i$
- *Between-cluster scatter matrix* $\mathbf{S}_B = \sum_{i=1}^{c} n_i(\mathbf{m}_i-\mathbf{m})(\mathbf{m}_i-\mathbf{m})^T$
- *Total scatter matrix* $\mathbf{S}_T = \sum_{\mathbf{x}\in D}(\mathbf{x}-\mathbf{m})(\mathbf{x}-\mathbf{m})^T = \mathbf{S}_W + \mathbf{S}_B$

1. Scater criteria based on the trace...

- It measures the square of the scattering radius
- Minimize the trace of the Within-Cluster Scatter Matrix

$$Tr\left(\mathbf{S}_W\right)=\sum_{i=1}^{c}Tr\left(\mathbf{S}_i\right)=\sum_{i=1}^{c}\sum_{\mathbf{x}\in D_i}\left\|\mathbf{x}-\mathbf{m}_i\right\|^2=J_e$$

It results in function J_e!

- It is equivalent to maximize between cluster scattering matrix trace.

$$Tr\left(\mathbf{S}_W\right)=Tr\left(\mathbf{S}_T\right)-Tr\left(\mathbf{S}_B\right) \qquad Tr\left(\mathbf{S}_B\right)=\sum_{i=1}^{c}n_i\left\|\mathbf{m}_i-\mathbf{m}\right\|^2$$

2. Scatter criteria based on the determinant...

- It measures the square of the scattering volume.
- $\mathbf{S}_B$ is singular if $c \leq d$; $rank(\mathbf{S}_B) \leq min(d, c\text{-}1)$
- $\mathbf{S}_W$ is singular if $n + c < d$
- Assuming $n > d - c$

$$J_d = |\mathbf{S}_W| = \left| \sum_{i=1}^{c} \mathbf{S}_i \right|$$

- It does not change if the axes are scaled

3. Invariant scatter criteria...

Eigenvalues of $\mathbf{S}_W^{-1}\mathbf{S}_B$ are invariant to nonsigular linear transformations of the data (see chapter 3). We can invent criteria based on the eigenvalues:

$$Tr\left(\mathbf{S}_W^{-1}\mathbf{S}_B\right)=\sum_{i=1}^{d}\lambda_i \qquad \frac{|\mathbf{S}_W|}{|\mathbf{S}_T|}=\prod_{i=1}^{d}\frac{1}{1+\lambda_i}$$

$$J_f = trace\left(\mathbf{S}_T{}^{-1}\mathbf{S}_W\right)=\sum_{i=1}^{d}\frac{1}{1+\lambda_i}$$

It is the sum of squared errors after normalization with matrix $\mathbf{S}_T$

relying on the fact that good partitions are those for which the non-zero eigenvalues are large.

- Other proposed criteria (equivalent for $c = 2$): $\min Tr\left(\mathbf{S}_T{}^{-1}\mathbf{S}_W\right)$ $\max Tr\left(\mathbf{S}_W{}^{-1}\mathbf{S}_B\right)$

Property...

$$\lambda_1,..,\lambda_i,..,\lambda_d = eigenvalues(\mathbf{S}_W^{-1}\mathbf{S}_B) \Rightarrow$$
$$\frac{1}{1+\lambda_1},..,\frac{1}{1+\lambda_i},..,\frac{1}{1+\lambda_d} = eigenvalues(\mathbf{S}_T^{-1}\mathbf{S}_W)$$

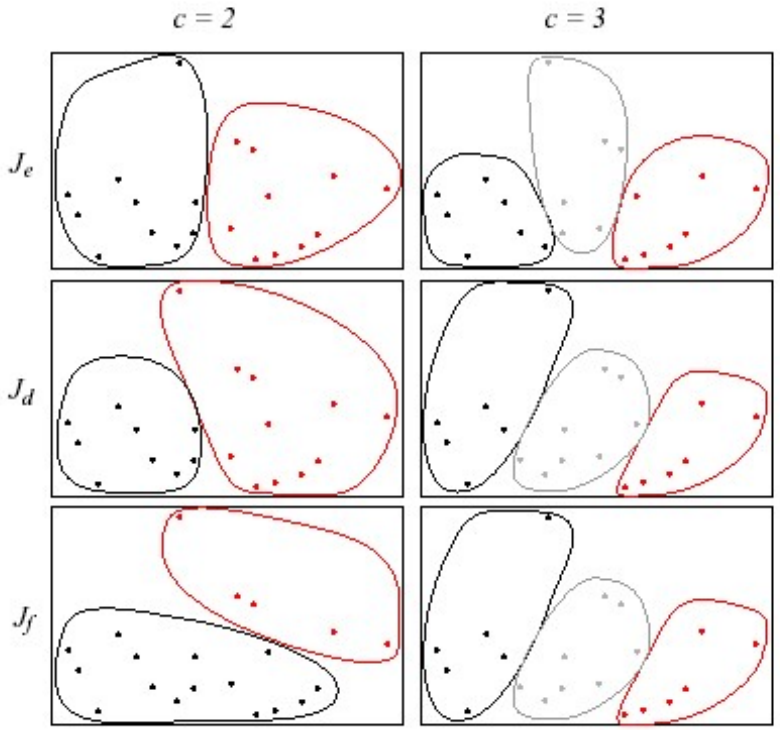
- Proof:

$$\mathbf{S}_B\mathbf{v}_i = \lambda_i\mathbf{S}_W\mathbf{v}_i \Rightarrow$$
$$\mathbf{S}_T\mathbf{v}_i = \mathbf{S}_B\mathbf{v}_i + \mathbf{S}_W\mathbf{v}_i = \lambda_i\mathbf{S}_W\mathbf{v}_i + \mathbf{S}_W\mathbf{v}_i = (1+\lambda_i)\mathbf{S}_W\mathbf{v}_i \Rightarrow$$
$$\mathbf{v}_i = (1+\lambda_i)\mathbf{S}_T^{-1}\mathbf{S}_W\mathbf{v}_i \Rightarrow$$
$$\mathbf{S}_T^{-1}\mathbf{S}_W\mathbf{v}_i = \frac{1}{1+\lambda_i}\mathbf{v}_i$$

Example:

- Trace criteria
- Determinant criteria
- Invariant criteria

It turns out that all criteria are equivalent as c increases



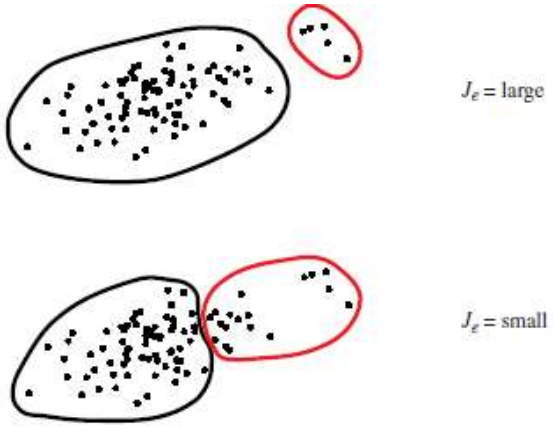
Conclusions on clustering procedures...

- Underlying model: assumes that samples form c fairly well separated clouds of points.
- $\mathbf{S}_W$ measures the compactness of these clouds, so it may not work in certain situations:
  - If there is a dense cluster inside a diffuse cluster
  - Separate interwined line-like clusters

- **Problem**: Computational complexity to evaluate the overall number of possibilities in partitioning is impractical. We need some iterative procedure...

2.3 ITERATIVE OPTIMIZATION

- We adopt the euclidean distance criterion in the sequel.
- Direct partitioning by full enumeration of clusters has large complexity: $c^n/c!$, which is completely infeasible
- Practical solution: Initiate with some reasonable partition and move samples from one group to another if such move improves the value of the criterion function.
- It guarantees local but not global optimization...



- Iterative improvement to minimize the sum of squared error criterion J_e.
- Effective error per cluster J_i.

$$J_e = \sum_{i=1}^{c} J_i; \qquad J_i = \sum_{\mathbf{x} \in D_i} \left\| \mathbf{x} - \mathbf{m}_i \right\|^2$$

- A sample is moved from cluster i to cluster j.

$$\hat{\mathbf{x}} \in D_i \Rightarrow \hat{\mathbf{x}} \in D_j$$

$$\mathbf{m}_j^* = \mathbf{m}_j + \frac{\hat{\mathbf{x}} - \mathbf{m}_j}{n_j + 1}; \qquad \mathbf{m}_i^* = \mathbf{m}_i - \frac{\hat{\mathbf{x}} - \mathbf{m}_i}{n_i - 1}$$

$$n_j = n_j + 1; \qquad n_i = n_i - 1$$

- Increasing / Decreasing "Effective error per cluster" (**Prove it as exercise**)

Contribution to error criterion due to cluster j

$$J_j^* = \sum_{\mathbf{x}\in D_j} \left\|\mathbf{x}-\mathbf{m}_j^*\right\|^2 + \left\|\hat{\mathbf{x}}-\mathbf{m}_j^*\right\|^2 =$$

$$= \left(\sum_{\mathbf{x}\in D_j} \left\|\mathbf{x}-\mathbf{m}_j - \frac{\hat{\mathbf{x}}-\mathbf{m}_j}{n_j+1}\right\|^2 \right) + \left\|\tfrac{n_j}{n_j+1}\left(\hat{\mathbf{x}}-\mathbf{m}_j\right)\right\|^2 = J_j + \tfrac{n_j}{n_j+1}\left\|\hat{\mathbf{x}}-\mathbf{m}_j\right\|^2$$

Contribution to error criterion due to cluster i

$$J_i^* = \sum_{\mathbf{x}\in D_i} \left\|\mathbf{x}-\mathbf{m}_i^*\right\|^2 - \left\|\hat{\mathbf{x}}-\mathbf{m}_i^*\right\|^2 =$$

$$= \left(\sum_{\mathbf{x}\in D_i} \left\|\mathbf{x}-\mathbf{m}_i + \frac{\hat{\mathbf{x}}-\mathbf{m}_i}{n_i-1}\right\|^2 \right) - \left\|\tfrac{n_i}{n_i-1}\left(\hat{\mathbf{x}}-\mathbf{m}_i\right)\right\|^2 = J_i - \tfrac{n_i}{n_i-1}\left\|\hat{\mathbf{x}}-\mathbf{m}_i\right\|^2$$

- The sample move from cluster i to cluster j is advantageous if

$$\frac{n_i}{n_i - 1}\left\|\hat{\mathbf{x}} - \mathbf{m}_i\right\|^2 > \frac{n_j}{n_j + 1}\left\|\hat{\mathbf{x}} - \mathbf{m}_j\right\|^2$$

BASIC ITERATIVE MSE CLUSTERING

1 **begin initialize** $n, c, \mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_c$

2 **do** randomly select a sample $\hat{\mathbf{x}}$;

3 $i \leftarrow \arg\min_{i'} \|\mathbf{m}_{i'} - \hat{\mathbf{x}}\|$ (classify $\hat{\mathbf{x}}$)

update $\mathbf{m}_i$

4 **if** $n_i \neq 1$ **then** compute

5 $$\rho_j = \begin{cases} \frac{n_j}{n_j+1}\|\hat{\mathbf{x}} - \mathbf{m}_j\|^2 & j \neq i \\ \frac{n_j}{n_j-1}\|\hat{\mathbf{x}} - \mathbf{m}_i\|^2 & j = i \end{cases}$$

6 **if** $\rho_k \leq \rho_j$ for all j **then** transfer $\hat{\mathbf{x}}$ to $\mathcal{D}_k$

7 recompute $J_e, \mathbf{m}_i, \mathbf{m}_k$

8 **until** no change in J_e in n attempts

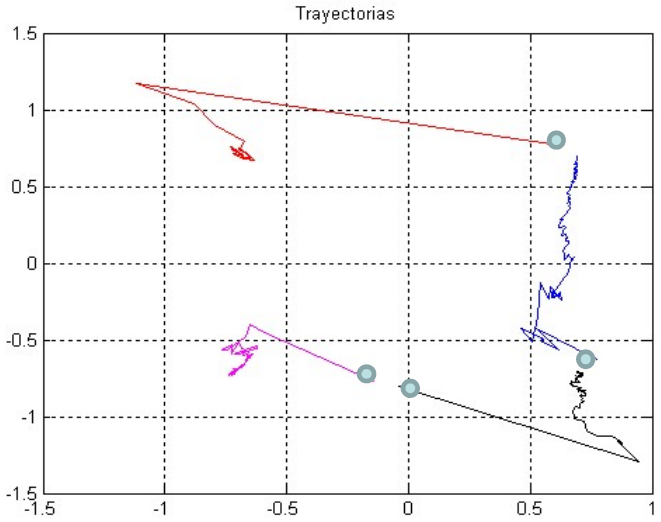
9 **return** $\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_c$

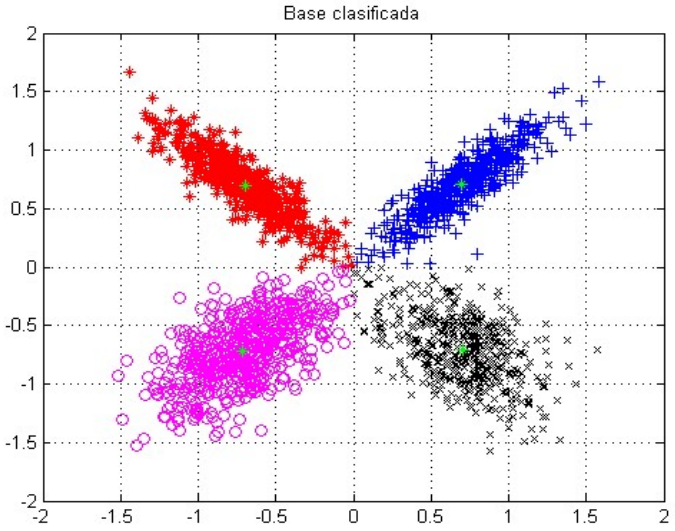
10 **end**

If the number of samples in the cluster i is >1

Features of the algorithm...

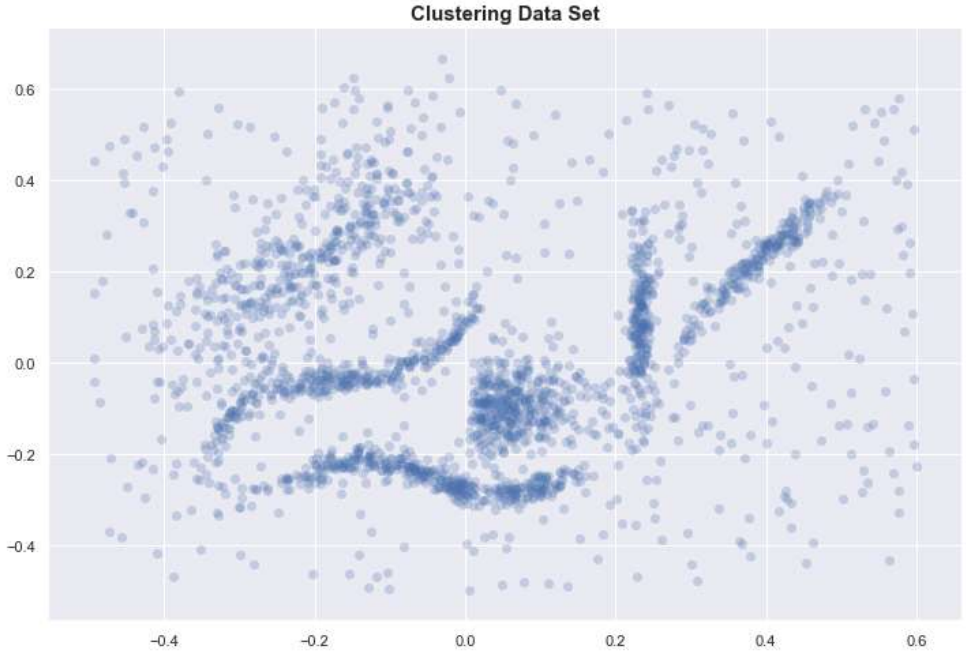
- It is a sequential version of the k-means procedure (k-means waits until all n samples are re-classified before updating means)
- This method is more susceptible to being trapped in local minima.
- Initial starting points can be obtained from the centroids obtained when seeking for c - 1 clusters
- Suitable for problems where samples are acquired on-line.

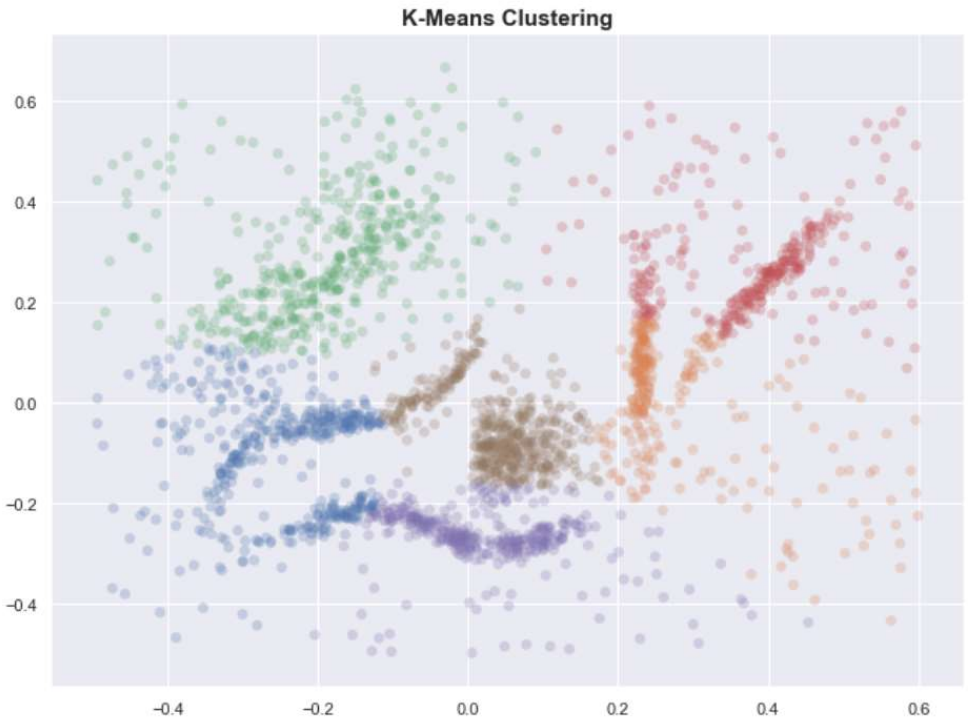


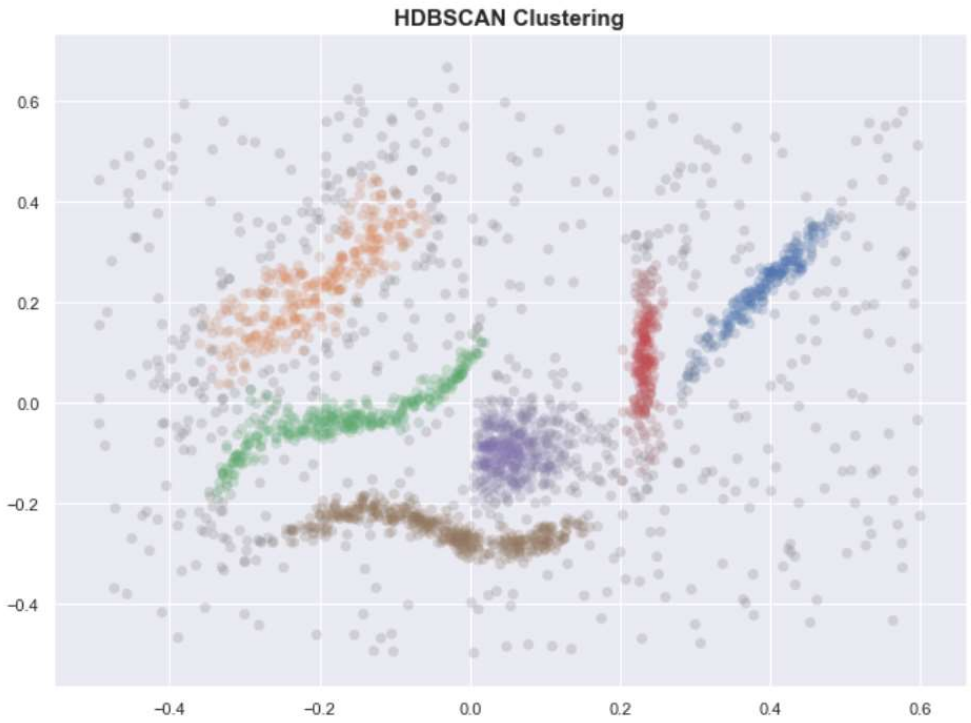


2.4 DBSCAN

The *Density-Based Spatial Clustering of Applications with Noise*[1] is a clustering algorithm that defines clusters as continuous regions of high density. An example of a data base where k-means fails:







K-means fail because clusters are not spherical, not equally sized, not equally dense and contaminated by noise.

DBSCAN

The principles of the algorithm:

1. For each vector $\mathbf{x}_i$, count how many samples are within a small distance ε.

2. If a vector has at least `min_samp` other vectors in its ε-neighborhood, it is considered a *core instance*.

3. All vectors in the neighborhood belong to the same cluster. This include other possible *core instances*.

4. Any vector that is not a core instance and do not have one in its neighborhood is considered an anomaly.

Further reading

The DBSCAN algorithm

Campello, Ricardo JGB, Davoud Moulavi, and Jörg Sander. "Density-based clustering based on hierarchical density estimates." *Pacific-Asia conference on knowledge discovery and data mining*. Springer, Berlin, Heidelberg, 2013.

https://towardsdatascience.com/understanding-hdbscan-and-density-based-clustering-121dbee1320e

The hierharchical DBSCAN

John Healy. HDBSCAN, Fast Density Based Clustering, the How and the Why. PyData NYC. 2018

2.5 CONCLUSIONS

- When underlying distribution comes from a mixture of component densities described by a set of unknown parameters, these parameters can be estimated by Bayesian or ML (EM algorithm) methods.

- Clustering is more general approach that does not rely on models. K-means and DSCAN are popular approaches.

- Selecting the number of clusters is crucial to have good performance.


North Pole
MAIL SERVICE
25
DECEMBER
APPROVED BY
Santa