Computational Intelligence Laboratory

Lecture 6

Data Clustering and Mixture Models

Thomas Hofmann

ETH Zurich – cil.inf.ethz.ch

27 March 2020

Section 1

Motivation

Motivation: Data Clustering

- Given: set of data points $\mathbf{x}_1, \ldots, \mathbf{x}_N \in \mathbb{R}^D$
- Goal: find a meaningful partition of the data
  - i.e. an assignment of each data point to a cluster

$$\pi : \{1, \ldots, N\} \to \{1, \ldots, K\} \quad \text{or}$$

$$\pi : \mathbb{R}^D \to \{1, \ldots, K\}$$

  - note: numbering of clusters is arbitrary
  - j-th cluster recovered by

$$\pi^{-1}(j) \subseteq \{1, \ldots, N\} \quad \text{or} \quad \subseteq \mathbb{R}^D$$

Motivation: Data Clustering

- Clustering via similarity:
  - group together similar data points

    avoid grouping together dissimilar ones
  - uncover hidden group structure of data
  - learn a data density model
  - may give rise to data compression schemes

Clustering Example

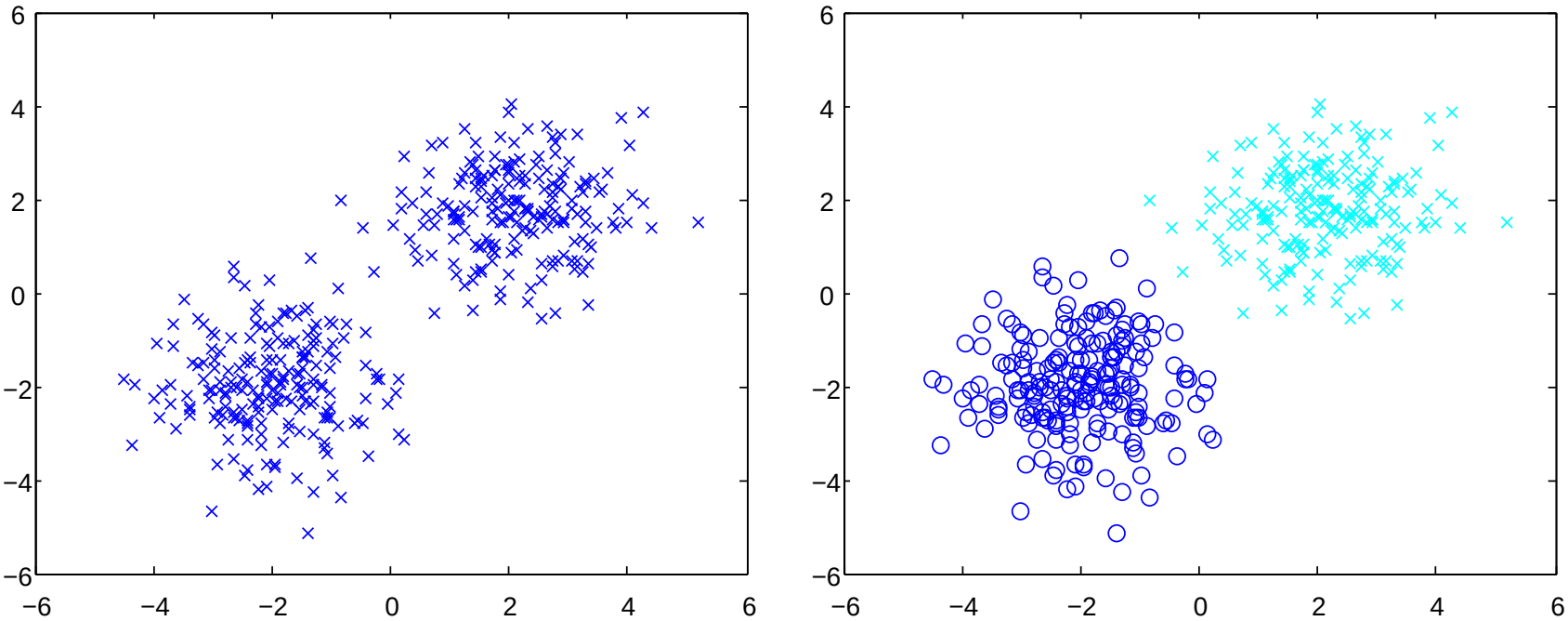


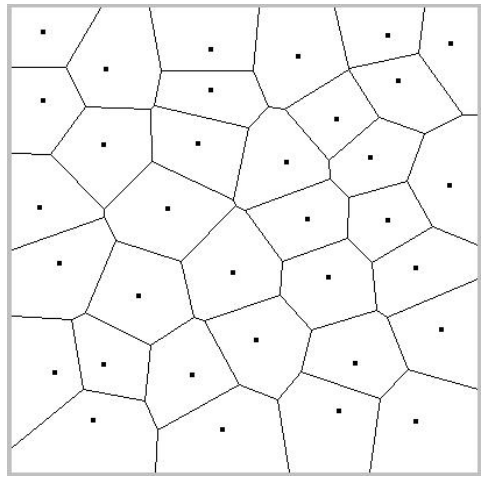
Figure: A simple clustering example. Left: 1 cluster, right: 2 clusters.

Vector Quantization

- Partitioning of the space $\mathbb{R}^D$
- Clusters represented by centroids $\mathbf{u}_j \in \mathbb{R}^D$.
- Mapping induced via nearest centroid rule

$$\pi(\mathbf{x}) = \underset{j=1,\ldots,K}{\arg\min} \|\mathbf{u}_j - \mathbf{x}\|$$

- Voronoi (or Dirichlet) tesselation of $\mathbb{R}^D$



Color Reduction by Vector Quantization

Figure: Top: original images, Bottom: image represented with 10 colors, selected by clustering color vectors in RGB space.

Section 2

K-Means

Encoding via Indicators

- Formalize clustering problem as optimization problem
  - find centroids $\mathbf{u}_j \in \mathbb{R}^D$ and assignment π minimizing ...
  - loss function or distortion, e.g. squared Euclidean norm
- Encode π via indicator matrix $\mathbf{Z} \in \{0,1\}^{N \times K}$

$$z_{ij} := \begin{cases} 1 & \text{if } \pi(\mathbf{x}_i) = j \\ 0 & \text{otherwise} \end{cases}$$

  - note that

$$\sum_{j=1}^{K} z_{ij} = 1 \quad (\forall i)$$

Objective Function

- K-means objective function

$$J(\mathbf{U}, \mathbf{Z}) = \sum_{i=1}^{N} \sum_{j=1}^{K} z_{ij} \|\mathbf{x}_i - \mathbf{u}_j\|^2$$

$$= \|\mathbf{X} - \mathbf{U}\mathbf{Z}^\top\|_F^2$$

  - where

$$\mathbf{X} = [\mathbf{x}_1 \cdots \mathbf{x}_N] \in \mathbb{R}^{D \times N}, \quad \text{data matrix}$$

$$\mathbf{U} = [\mathbf{u}_1 \cdots \mathbf{u}_K] \in \mathbb{R}^{D \times K}, \quad \text{centroid matrix}.$$

K-means Algorithm: Idea

- How do we minimize the K-means objective?
- Simple observation:
  - determining optimal centroids given assignments is easy (continuous variables)
  - determining optimal assignments given centroids is easy (integer variables)
- Computational strategy: alternating minimization

K-means Algorithm: Optimal Assignment

- Compute optimal assignment $\mathbf{Z}$, given centroids $\mathbf{U}$
  - each data point contributes to exactly one term in outer sum
  - minimize assignment of each data point separately

$$z_{ij}^* = \begin{cases} 1 & \text{if } j = \arg\min_k \|\mathbf{x}_i - \mathbf{u}_k\|^2 \\ 0 & \text{otherwise} \end{cases}$$

  - map each data point to the closest centroid

K-means Algorithm: Optimal Centroids

- Compute optimal choice of $\mathbf{U}$, given assignments $\mathbf{Z}$
  - continuous variables: compute gradient and set to zero (1st order optimality condition)
  - look at (partial) gradient for every centroid $\mathbf{u}_j$

$$\nabla_{\mathbf{u}_j} J(\mathbf{U}, \mathbf{Z}) = \sum_{i=1}^{N} z_{ij} \underbrace{\tfrac{1}{2} \nabla_{\mathbf{u}_j} \|\mathbf{x}_i - \mathbf{u}_j\|^2}_{=\mathbf{u}_j - \mathbf{x}_i}$$

  - setting gradient to zero

$$\nabla_{\mathbf{U}} J(\mathbf{U}, \mathbf{Z}) \stackrel{!}{=} 0 \quad \Longrightarrow \quad \mathbf{u}_j^* = \frac{\sum_{i=1}^{N} z_{ij}\, \mathbf{x}_i}{\sum_{i=1}^{N} z_{ij}}, \quad \text{if } \sum_{i=1}^{N} z_{ij} \geq 1$$

  - centroid condition (center of mass of assigned data points)

K-means Algorithm: Summary

initialize $\mathbf{U}$ on K distinct random data points

initialize $\mathbf{Z} \leftarrow \mathbf{Z}^*(\mathbf{U})$

repeat

$\mathbf{U} \leftarrow \mathbf{U}^*(\mathbf{Z})$ (see above)

$\mathbf{Z}^{\text{new}} \leftarrow \mathbf{Z}^*(\mathbf{U})$ (see above)

same $= (\mathbf{Z}^{\text{new}} == \mathbf{Z})$

$\mathbf{Z} \leftarrow \mathbf{Z}^{\text{new}}$

until (same)

- different initialization strategies, here: random points
- better handling of empty clusters: random re-initialization

K-means Algorithm:

- Computational cost of each iteration is $O(knd)$
- K-means convergence is guaranteed
  - non-increasing objective, bounded from below by 0
- K-means optimizes a non-convex objective
  - we are not guaranteed to find the global optimum

Illustration of the K-means Algorithm

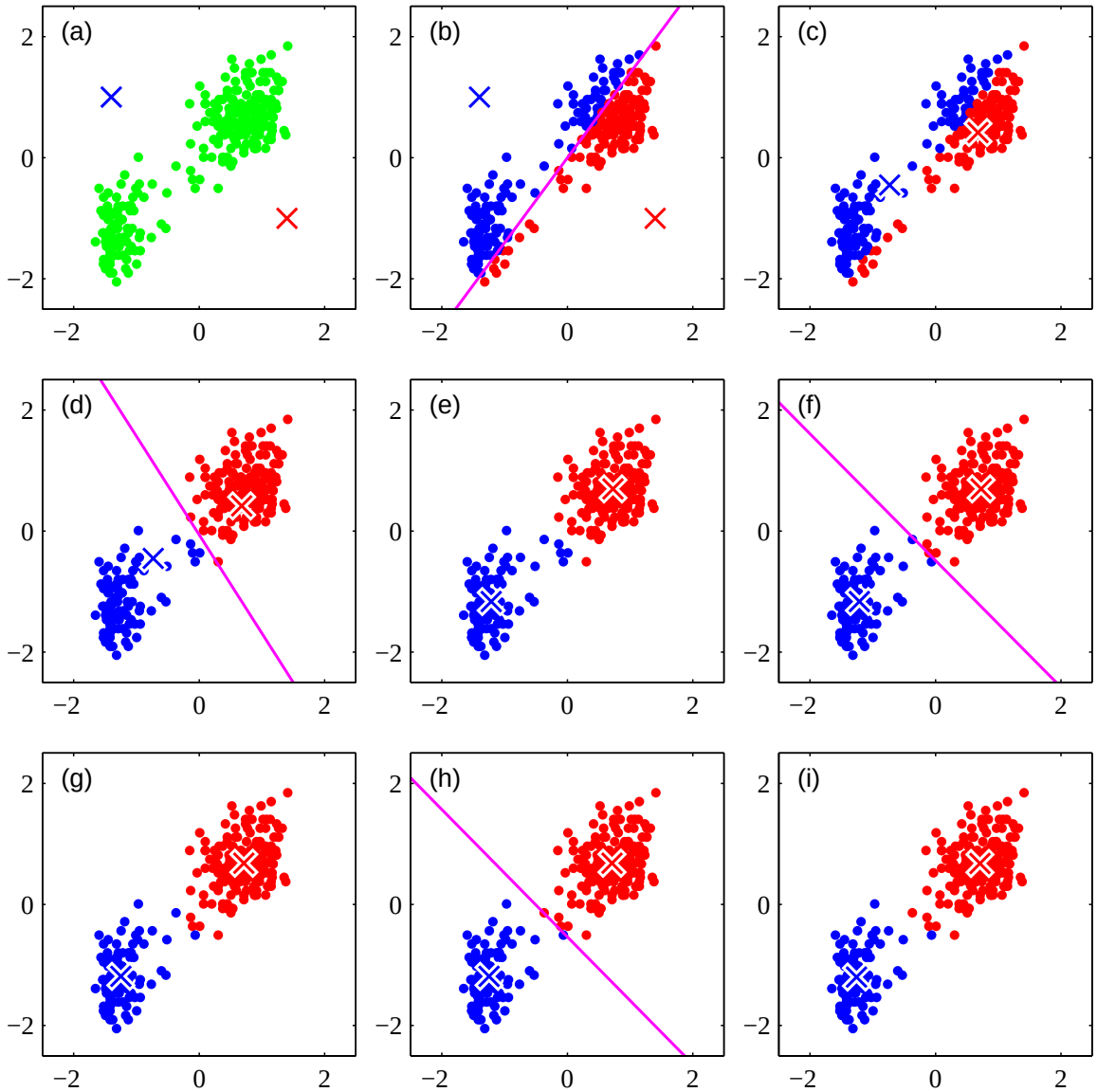


Figure: Bishop, *Pattern Recognition & Machine Learning*, Springer (2006).

K-means++

- More sophisticated seeding: Arthur & Vassilvitskii, 2007
- Incremental D^2 sampling
  - Initial centroid set $\mathcal{U}_1 = \{\mathbf{x}_I\}$, where $I \sim \text{Uniform}[1 : N]$
  - For $k = 1 \dots K - 1$

$$D_i := \min_{\mathbf{u} \in \mathcal{U}_k} \|\mathbf{x}_i - \mathbf{u}\|, \quad \mathcal{U}_{k+1} := \mathcal{U}_k \cup \{\mathbf{x}_I\}, \quad \text{where}$$

$$I \sim \text{Categorical}(\mathbf{p}), \quad p_i := \frac{D_i^2}{\sum_{j=1}^{N} D_j^2}$$

  - more expensive (though: parallelization), but consistently better experimental results
  - theoretical guarantee: $\mathbf{O}(\log K)$-competitiveness in expectation

Core Sets for K-means

- Recent research, e.g.: Bachem, Lucic, Krause, 2018: Scalable k-Means Clustering via Lightweight Coresets
- Sample (multi-)set (**core set**) of size m

$$I \sim \text{Categ}(\mathbf{p}), \quad p_i := \frac{1}{2N} + \frac{D_i^2}{2\sum_{j=1}^N D_j^2}, \quad D_i^2 = \|\mathbf{x}_i - \mu\|^2$$

  $\mu := \frac{1}{N}\sum_i \mathbf{x}_i$. Then: give each sample a relative weight $\frac{1}{mp_i}$.
- Perform weighted **K-means on core set** (then: map all data points to closest prototype).
- ϵ-approximation guarantees (with probability δ) for

$$m \propto \frac{d\,k\log k + \log 1/\delta}{\epsilon^2}$$

Section 3

Mixture Models

Probabilistic Clustering

From hard to probabilistic assignments

- K-means: each data point assigned to exactly one cluster
- probabilistic or "soft" assignments: assign $\mathbf{x}_i$ to each cluster j with some probability z_{ij}
- generalize (relax) constraints on $\mathbf{Z}$

$$z_{ij} \in [0; 1] \ (\forall i, j), \quad \sum_{j=1}^{K} z_{ij} = 1 \ (\forall i)$$

Cluster Conditional Probability Distributions

Mocdel each cluster by a probability distribution

- Simplest choice: multivariate normal distribution
- PDF (probability density function) of univariate Gaussian with mean μ and variance σ^2:

$$p(x;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$$

- Isotropic multivariate normal distribution with mean $\boldsymbol{\mu}$, density:

$$p(\mathbf{x};\boldsymbol{\mu},\sigma) = \prod_{i=1}^{D} \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{(x_i-\mu_i)^2}{2\sigma^2}\right]$$

Cluster Conditional Probability Distributions

- Multivariate normal distribution with covariance matrix $\boldsymbol{\Sigma}$, density:

$$p(\mathbf{x}; \boldsymbol{\mu}; \boldsymbol{\Sigma}) = \frac{1}{|\boldsymbol{\Sigma}|^{\frac{1}{2}} (2\pi)^{\frac{D}{2}}} \exp\left[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu})\right]$$

  - $\boldsymbol{\Sigma}$: symmetric, positive definite
  - generally difficult to estimate for large D: $D + \frac{(D+1)D}{2}$ parameters

Probabilistic Clustering Model

- Finite **Mixture Model**

$$p(\mathbf{x};\theta) = \sum_{j=1}^{K} \pi_j \; p(\mathbf{x};\theta_j), \quad \theta = (\pi, \theta_1, \ldots, \theta_K) \in \mathbb{R}^{K+K\cdot M}$$

  - mixing proportions $\pi \geq 0$, $\sum_{j=1}^{K} \pi_j = 1$
  - component density functions $p(\mathbf{x};\theta_j)$ with $\theta_j \in \mathbb{R}^M$

- Mixture models for clustering
  - relative cluster sizes $= \pi_j$ $(j = 1, \ldots, K)$
  - location & "shape" of clusters = specific form of $p(\mathbf{x};\theta_j)$
  - special case: Gaussian densities with, $\theta_j = (\underbrace{\boldsymbol{\mu}_j}_{\text{location}}, \underbrace{\boldsymbol{\Sigma}_j}_{\text{shape}})$

Gaussian Mixture Model

- Gaussian Mixture Model (GMM):

$$p(\mathbf{x};\theta) = \sum_{j=1}^{K} \pi_j \; p(\mathbf{x};\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \quad \text{(normal densities)}$$

- Two-stage generative model: generate a data point as follows
  - sample cluster index from categorical distribution
    $j \sim \text{Categorical}(\boldsymbol{\pi})$
  - given j, sample a data point $\mathbf{x}$ from the j-th component
    $\mathbf{x} \sim \text{Normal}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$
- Cluster index j: latent variable; final outcome $\mathbf{x}$: observed
- Probabilistic clustering: compute posteriors of latent cluster memberships ...

Complete Data Distribution

- Explicitly introduce latent variables into generative model
- Assignment variable (for a generic data point)

$$\mathbf{z} \in \{0,1\}^K, \quad \sum_{j=1}^{K} z_j = 1.$$

- Categorical distribution

$$\Pr(z_j = 1) = \pi_j \quad \text{or} \quad p_\pi(\mathbf{z}) = \prod_{j=1}^{K} \pi_j^{z_j}$$

- Joint distribution over $(\mathbf{x}, \mathbf{z})$ (complete data distribution)

$$p(\mathbf{x}, \mathbf{z}; \theta) = \prod_{j=1}^{K} [\pi_j \; p(\mathbf{x}; \theta_j)]^{z_j}$$

Posterior Assignments

- Generation: given $\mathbf{z}$, generate $\mathbf{x}$; Inference: given $\mathbf{x}$, infer $\mathbf{z}$
- Bayes rule
  - reminder, posterior $p(A|B) = \frac{p(B|A)p(A)}{p(B)}$
  - here: $p(A)$ prior, $p(B|A)$ likelihood and $p(B)$ evidence
- Posterior probabilities for assignments

$$\Pr(z_j = 1 \mid \mathbf{x}) = \frac{\Pr(z_j = 1)p(\mathbf{x} \mid z_j = 1)}{\sum_{l=1}^{K} \Pr(z_l = 1)p(\mathbf{x} \mid z_l = 1)} = \frac{\pi_j \; p(\mathbf{x}; \theta_j)}{\sum_{l=1}^{K} \pi_l \; p(\mathbf{x}; \theta_l)}$$

  - assumes access to parameters π, $\{\theta_j = (\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)\}$

Maximum Likelihood: Mixture Model

- MLE requires to optimize

$$\hat{\theta} = \arg\max_{\theta} \sum_{i=1}^{N} \log \left[\sum_{j=1}^{K} \pi_j \; p(\mathbf{x}_i; \theta_j) \right]$$

- Challenge: summation over j inside the logarithm

  $\Rightarrow$ MLE has no closed-form solution

Lower Bounding the Log-Likelihood

- Expectation Maximization
  - maximize a lower bound on the log-likelihood
  - based on complete data distribution

- Specifically:

$$\log p(\mathbf{x};\theta) = \log\left[\sum_{j=1}^{K} \pi_j \; p(\mathbf{x};\theta_j)\right] = \log\left[\sum_{j=1}^{K} q_j \frac{\pi_j \; p(\mathbf{x};\theta_j)}{q_j}\right]$$

$$\geq \sum_{j=1}^{K} q_j \left[\log p(\mathbf{x};\theta_j) + \log \pi_j - \log q_j\right]$$

  - follows from Jensen's inequality (concavity of logarithm)
  - can be done for the contribution of each data point (additive)

Mixture Model: Expectation Step

- Optimize bound with regard to the distribution q
  - formulate Lagrangian (decoupled for each data point)

$$\max_q \left\{ \sum_{j=1}^{K} q_j \left[\log p(\mathbf{x}; \theta_j) + \log \pi_j - \log q_j \right] + \lambda \left(\sum_{j=1}^{K} q_j - 1 \right) \right\}$$

  - first order optimality condition (setting gradient to zero):

$$\log p(\mathbf{x}; \theta_j) + \log \pi_j - \log q_j - 1 + \lambda \stackrel{!}{=} 0 \iff$$

$$q_j^* = \frac{\pi_j \; p(\mathbf{x}; \theta_j)}{\sum_{l=1}^{K} \pi_l \; p(\mathbf{x}; \theta_l)} \stackrel{\text{Bayes rule}}{=} \Pr(z_j = 1 \mid \mathbf{x})$$

  - optimal q–distribution equals posterior (given the parameters)
  - E–step selects the best lower bound on the log-likelihood

Mixture Model: Maximization Step

- Optimize expected complete data log-likelihood with regard to the model parameters
  - problem decouples for each cluster and with regard to π
  - solution for mixing proportions π

$$\pi_j^* = \frac{1}{N} \sum_{i=1}^{N} q_{ij}$$

  - solution for $\theta_j = (\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$

$$\boldsymbol{\mu}_j^* = \frac{\sum_{i=1}^{N} q_{ij} \mathbf{x}_i}{\sum_{i=1}^{N} q_{ij}}, \quad \boldsymbol{\Sigma}_j^* = \frac{\sum_{i=1}^{N} q_{ij} \ (\mathbf{x}_i - \boldsymbol{\mu}_j)(\mathbf{x}_i - \boldsymbol{\mu}_j)^\top}{\sum_{i=1}^{N} q_{ij}}$$

Expectation Maximization Algorithm

- Alternate E-step and M-step
  - both E- and M-step maximize the same (bounded) objective
  - guaranteed convergence towards a point θ^*
  - like in K-means: θ^* may not be the global maximizer
  - convergence criterion (e.g. change in objective)
- E-step: compute probabilistic assignments of points to clusters (keeping their location and shape fixed)
- M-step: recompute optimal cluster locations and shapes, given probabilistic assignments

Example of EM for Gaussian Mixtures

Illustration of the EM algorithm using the Old Faithful data set.

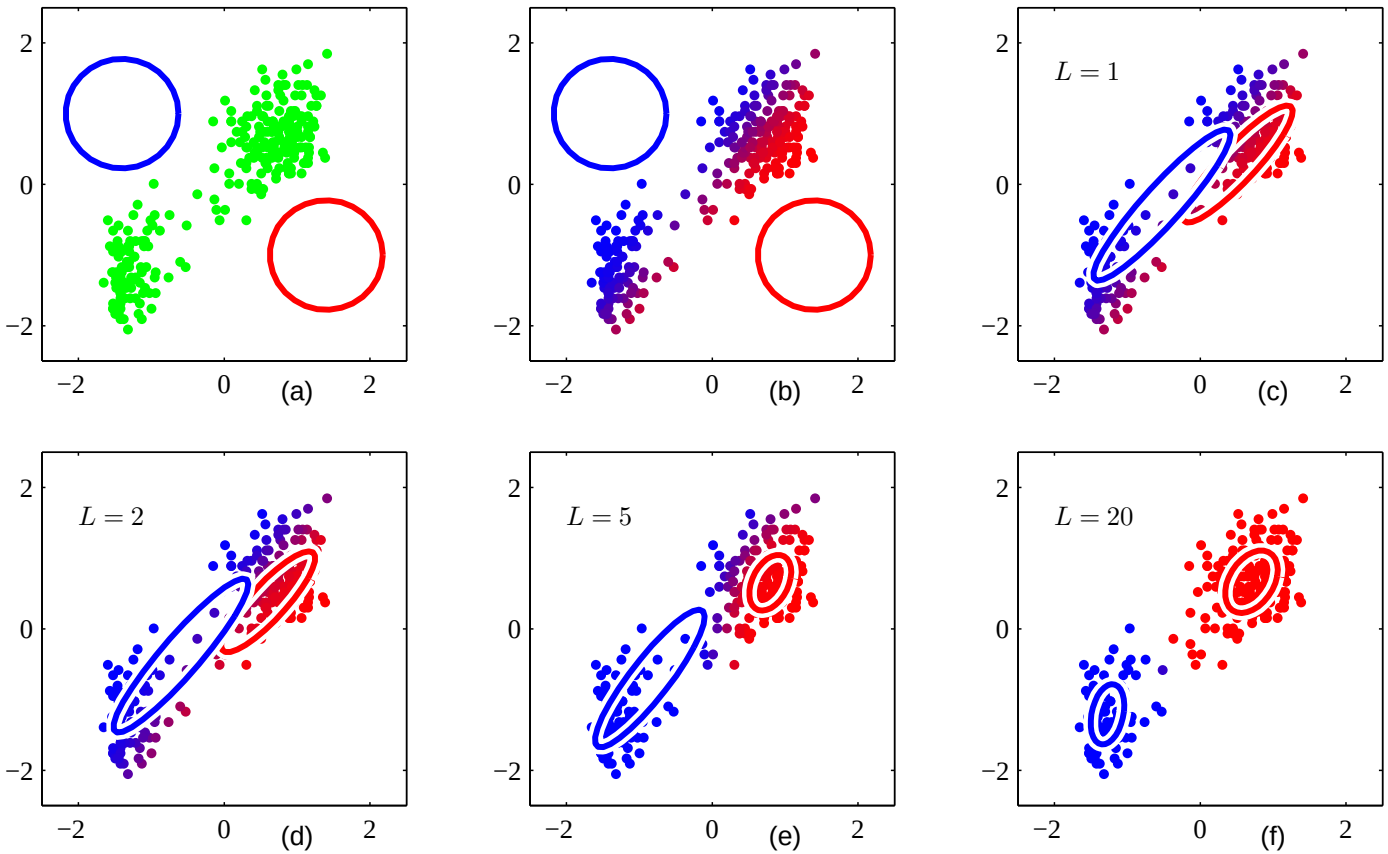


Figure: Gaussian mixture model fitting via EM for two clusters. Remark: here the covariance is also estimated (illustrated by the two ellipsoids).

Comparison with K-means

- Assignments
  - K-means algorithm: hard assignment points to clusters
  - EM algorithm: soft assignment based on posteriors
- Shapes
  - K-means: spherical cluster shapes, uniform spread
  - EM algorithm: can learn covariance matrix
- K-means as a special case
  - Gaussian mixture model with (fixed) covariances $\mathbf{\Sigma}_j = \sigma^2 \mathbf{I}$
  - in the limit of $\sigma \to 0$, recover K-means (hard assignments)
  - can be more formally derived (EM objective $\to$ K-means objective)

Practical Points about K-means and EM

- EM algorithm
  - takes many more iterations to reach convergence
  - each cycle requires significantly more computation.
- K-means algorithm can be used to find a good initialization
  - covariance matrices can be initialized to the sample covariances of the clusters found by the K-means algorithm.
  - mixing coefficients can be set to the fractions of data points assigned to the respective clusters

Section 4

Model Selection

Occam's Razor

William Occam:

Entities must not be multiplied beyond necessity.

Model order selection: General principle

Trade-off between two conflicting goals:

Data fit: We want to predict the data well, e.g., maximize the likelihood. The likelihood usually increases with increasing number of clusters.

Complexity: Choose a model that is not very complex which is often measured by the number of free parameters.

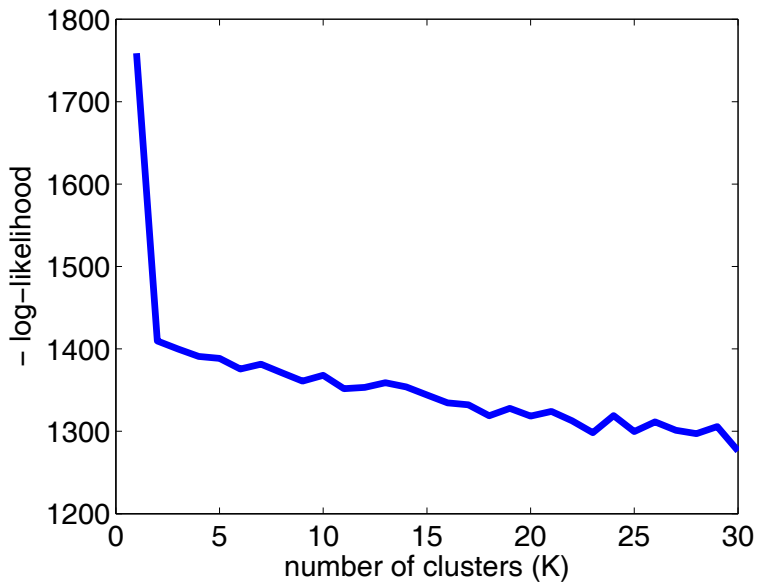
Find a compromise between these two goals!

Better fit with increasing K

Negative Log-Likelihood of data for K mixture Gaussians:

$$-\log p(\mathbf{X};\theta) = -\sum_{i=1}^{N} \log \left[\sum_{j=1}^{K} \pi_j \; p(\mathbf{x}_i;\theta_j) \right] .$$

- smaller negative log-likelihood = better fit
- decreasing with K (some noise due to local minima)



AIC and BIC

- Model complexity: can be measured by the number of free parameters $\kappa(\cdot)$.
- Different Heuristics for choosing K
  - Akaike Information Criterion (AIC)

$$AIC(\theta|\mathbf{X}) = -\log p(\mathbf{X};\theta) + \kappa(\theta)$$

  - Bayesian Information Criterion (BIC)

$$BIC(\theta|\mathbf{X}) = -\log p(\mathbf{X};\theta) + \frac{1}{2}\kappa(\theta)\log N$$

- Generally speaking, the BIC criterion penalizes complexity more than the AIC criterion.

AIC and BIC: Remarks and Example

Analysis

A single AIC (BIC) result is meaningless. One has to repeat the analysis for different Ks and compare the differences: the most suitable number of clusters corresponds to the smallest AIC (BIC) value.

Example (Mixture of Gaussians)

- Number of free parameters (with fixed covariance matrices)

$$\kappa(\theta) = K \cdot D + (K - 1).$$

- Number of free parameters (with full covariance matrices)

$$\kappa(\theta) = K \cdot \left(D + \frac{D(D+1)}{2} \right) + (K - 1).$$

AIC and BIC example: 3 clusters

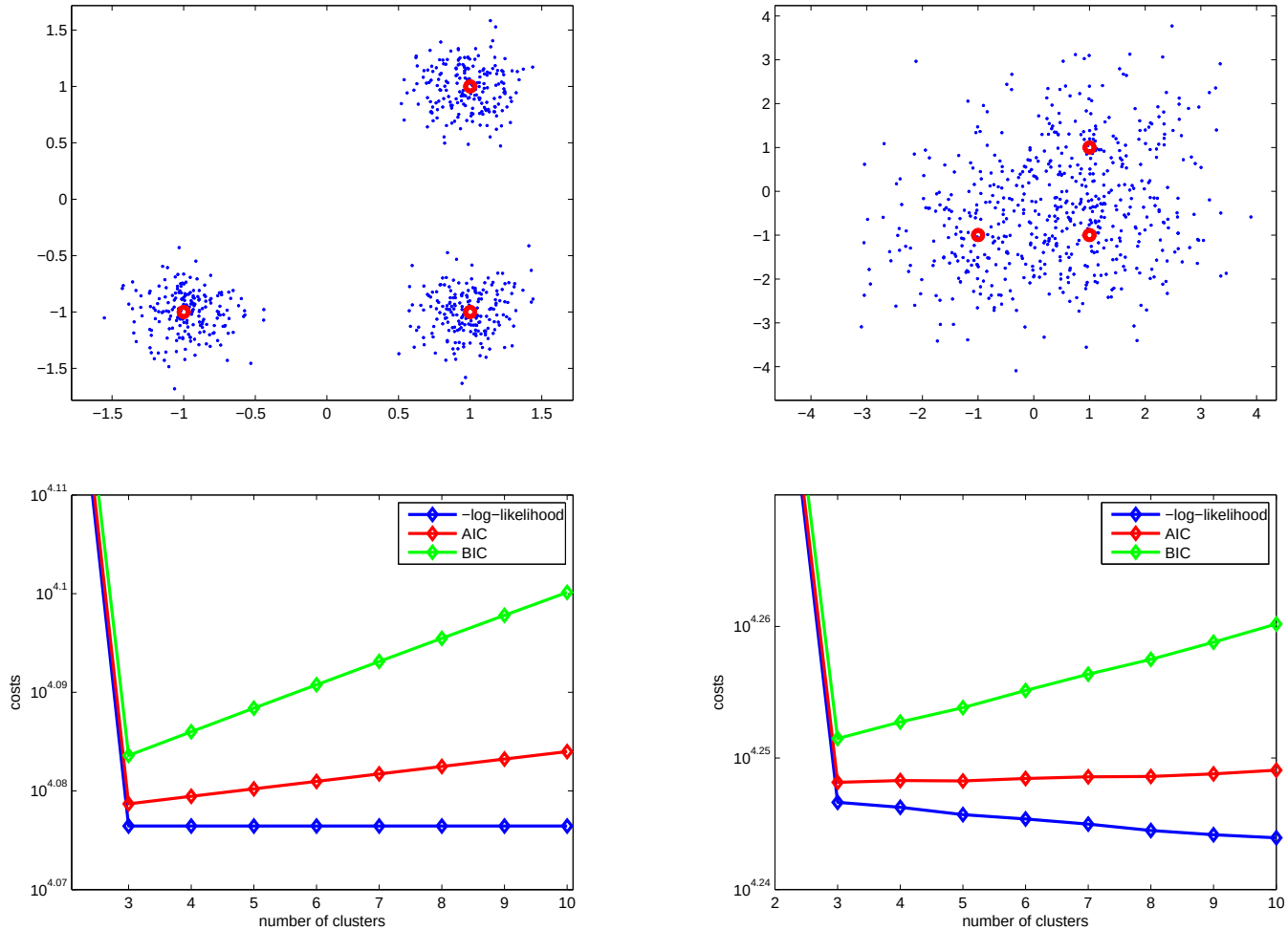


Figure: Information criteria for a synthetic dataset with 3 clusters. Synthetic data has smaller variance on the left than on the right.

AIC and BIC example: 5 clusters

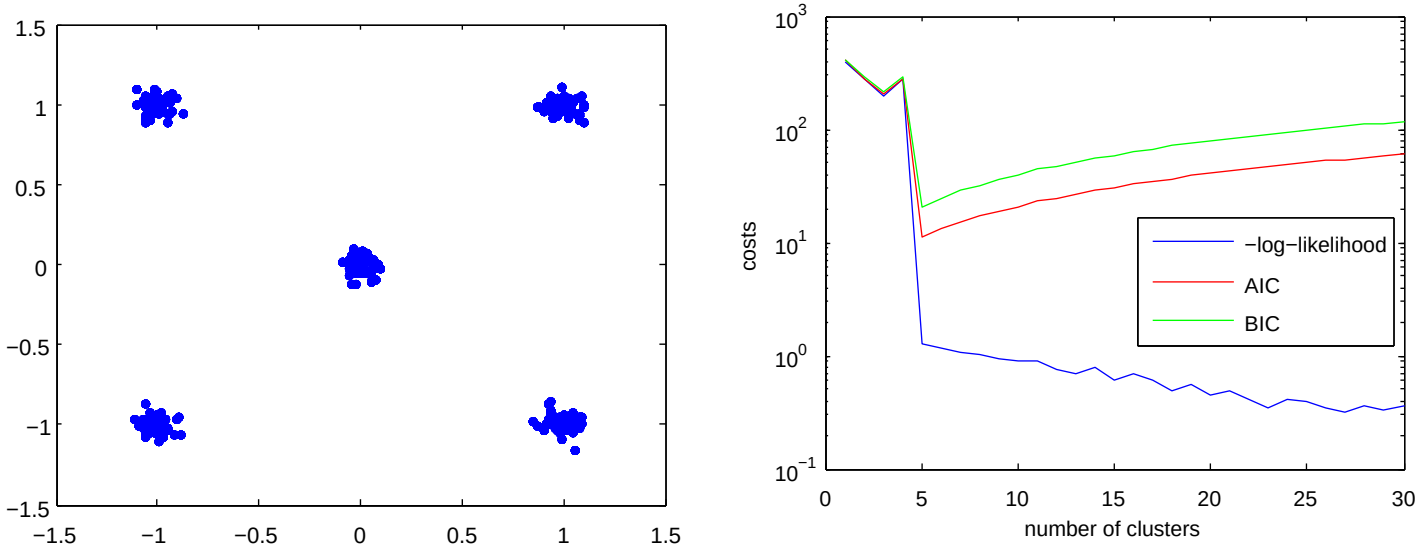


Figure: Information criteria for a synthetic dataset with 5 clusters.

Questions

1. Lecture 5, Slide 16
   Why takes the optimal Bayes discriminant this form?
   Why is this related to the pointwise mutual information?
2. Lecture 6, Slide 18:
   Why does one choose a sample weighting $\frac{1}{mp_i}$?
3. Lecture 6, Slide 29:
   What is the role of the Lagrange multiplier λ?
   Why do we not have to enforce the non-negativity of the q_j's?
4. Lecture 6, Slide 30:
   How are these paremeter estimation equations derived?