Machine Learning (IN2064)

Lecture 12: Clustering

Prof. Dr. Stephan Günnemann
Aleksandar Bojchevski

Data Analytics and Machine Learning
Technical University of Munich

Winter term 2020/2021

Unsupervised learning

Main idea

- Given some unlabeled data $\{\boldsymbol{x}_i\}$ we want to discover latent structure in it.

Last week: Dimensionality reduction

- Given high-dimensional data $\boldsymbol{x}_i \in \mathbb{R}^D$, find a continuous representation $\boldsymbol{z}_i \in \mathbb{R}^K$ with $K \ll D$, such that the structure is preserved.

This week: Clustering

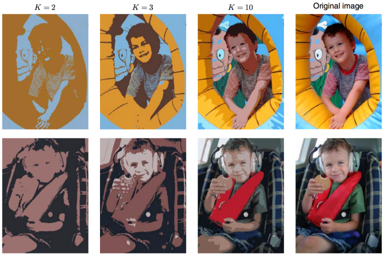
- Group objects $\boldsymbol{x}_i$ into K groups (clusters) based on their similarity.

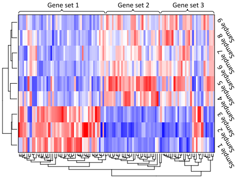
discrete rep

Clustering: Examples

K increase

- Image segmentation
- User profiling
- Gene expression analysis
- Data compression
- Visualization
- ...





Sources: https://blogs.sas.com/content/subconsciousmusings/2016/05/26/data-mining-clustering/, https://en.wikipedia.org/wiki/Transcriptomics_technologies#/media/File:Transcriptomes_heatmap_example.svg, and C. Bishop - "Pattern Recognition and Machine Learning"

Problem definition

Given

- Collection of data points $\boldsymbol{X} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_N\} \subset \mathbb{X}$.

Goal

- For each object $\boldsymbol{x}_i$ find the corresponding assignment $z_i \in \{1, \ldots, K\}$ to one of the K groups (clusters), such that:
  - objects within the same group are similar to each other;
  - objects between different groups are dissimilar from each other;

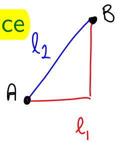
Overlapping : one instance might belong to multiple clusters

Section 1

K-means Algorithm

Distance-based clustering

- If we want to group similar objects together, we need some notion of similarity to begin with.
- Typically defined as similarity function $s(\cdot,\cdot) : \mathbb{X} \times \mathbb{X} \to \mathbb{R}$.
- Alternatively: minimize dissimilarity, usually provided by a distance function $d(\cdot,\cdot)$.



- Typical distance functions include
  - Manhattan distance: $d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sum_d |\boldsymbol{x}_{id} - \boldsymbol{x}_{jd}| = ||\boldsymbol{x}_i - \boldsymbol{x}_j||_1$
  - Euclidean distance: $d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{\sum_d (\boldsymbol{x}_{id} - \boldsymbol{x}_{jd})^2} = ||\boldsymbol{x}_i - \boldsymbol{x}_j||_2$
  - Mahalanobis distance: $d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x}_i - \boldsymbol{x}_j)}$

generalization of

K-means

Model

- Consider the standard $\mathbb{R}^D$ space with Euclidean distance as metric.
- Each of K clusters is defined by its centroid $\boldsymbol{\mu}_k \in \mathbb{R}^D$.
- Cluster indicator $\boldsymbol{z}_i \in \{0,1\}^K$ with $\boldsymbol{z}_{ik} = 1 \iff \boldsymbol{x}_i$ belongs to cluster i (i.e. one-hot encoding). $z_i = [0\ 0\ 1\ 0\ 0\ 0]$
- The K-means objective (also called distortion measure) is defined as

$$J(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{\mu}) = \sum_{i=1}^{N} \sum_{k=1}^{K} \boldsymbol{z}_{ik} ||\boldsymbol{x}_i - \boldsymbol{\mu}_k||_2^2$$

Goal

- We need to find the "best" centroids $\boldsymbol{\mu}$ and cluster assignments $\boldsymbol{Z}$

min the distance bet point & centriod, whenever the indicator is triggered

$$\boldsymbol{Z}^*, \boldsymbol{\mu}^* = \arg\min_{\boldsymbol{Z}, \boldsymbol{\mu}} J(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{\mu})$$

Minimizing the K-means objective

- The joint optimization problem is difficult to solve (discrete optim.; even if $\boldsymbol{Z}$ is assumed to be continuous it is non-convex)

NP - Hard

$$\min_{\boldsymbol{Z},\boldsymbol{\mu}} J(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{\mu}) = \min_{\boldsymbol{Z},\boldsymbol{\mu}} \sum_{i=1}^{N} \sum_{k=1}^{K} \boldsymbol{z}_{ik} ||\boldsymbol{x}_i - \boldsymbol{\mu}_k||_2^2$$

- However, the two subproblems

$$\min_{\boldsymbol{Z}} J(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{\mu}) \qquad \text{and} \qquad \min_{\boldsymbol{\mu}} J(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{\mu})$$

  have simple closed form solutions!

- **Idea:** alternating optimization - update $\boldsymbol{Z}$ and $\boldsymbol{\mu}$ in turns!

fixed one, optimize for the other

- This method for solving K-means is called Lloyd's algorithm.

Lloyd's algorithm for K-means

1. Initialize the centroids $\boldsymbol{\mu} = \{\boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_K\}$.
2. Update cluster indicators (solve $\min_{\boldsymbol{Z}} J(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{\mu})$)

$$\boldsymbol{z}_{ik} = \begin{cases} 1 & \text{if } k = \arg\min_j ||\boldsymbol{x}_i - \boldsymbol{\mu}_j||^2, \\ 0 & \text{else.} \end{cases}$$

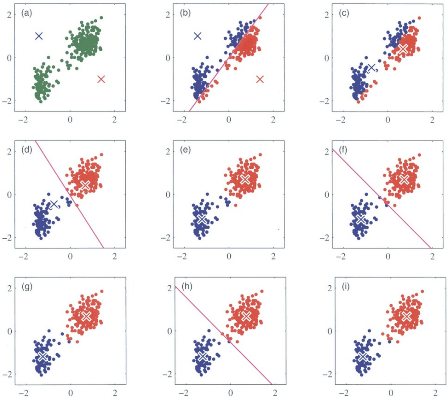
3. Update centroids (solve $\min_{\boldsymbol{\mu}} J(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{\mu})$)

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{i=1}^{N} \boldsymbol{z}_{ik} \boldsymbol{x}_i \qquad \text{where } N_k = \sum_{i=1}^{N} \boldsymbol{z}_{ik}.$$

4. If objective $J(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{\mu})$ has not converged, return to step 2.

K-means in action

better soln is obtained
when start from
a more sparse centroids



An interactive demo is available at

- `http://stanford.edu/class/ee103/visualizations/kmeans/kmeans.html`

Source: Bishop, Figure. 9.1

Initialization of the centroids

We saw in the interactive demo that the initial locations of the centroids $\boldsymbol{\mu}_k$ greatly affect the performance.

So, how do we initialize the centroids in a good way?

K-means++ algorithm

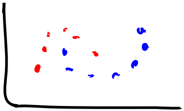
1. Choose the first centroid $\boldsymbol{\mu}_1$ uniformly at random among the data points.
2. For each point $\boldsymbol{x}_i$ compute the distance $D_i^2 = ||\boldsymbol{x}_i - \boldsymbol{\mu}_1||_2^2$.
3. Sample the next centroid $\boldsymbol{\mu}_k$ from $\{\boldsymbol{x}_i\}$ with probability proportional to D_i^2. i.e those farther apart
4. Recompute the distances $D_i^2 = \min\{||\boldsymbol{x}_i - \boldsymbol{\mu}_1||_2^2, ..., ||\boldsymbol{x}_i - \boldsymbol{\mu}_k||_2^2\}$.
5. Continue steps 3 and 4 until K initial centroids have been chosen.

Limitations of K-means

It is important to distinguish between the model (distortion $J(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{\mu})$ in our case) and the algorithm used to fit it (here Lloyd's algorithm).

Modeling issues

- Underlying assumptions are not explicit
- Can't detect clusters with overlapping convex hulls
- Sensitivity to outliers
- No uncertainty measure



Algorithmic issues

- Extreme sensitivity to initialization

How can we address (some of) these issues?

Switch to the more principled probabilistic framework!

Section 2

Gaussian Mixture Model

Probabilistic unsupervised learning

- Goal: Design a probabilistic model for the data, $\mathrm{p}(\boldsymbol{x} \mid \boldsymbol{\theta})$, parametrized by $\boldsymbol{\theta}$ which we have to estimate.
- It's difficult to come up with a useful model $\mathrm{p}(\boldsymbol{x} \mid \boldsymbol{\theta})$ without making any assumptions about the data.
- When doing clustering, we implicitly assume that each point $\boldsymbol{x}$ belongs to some cluster (indicated by $\boldsymbol{z}$).
  $\Rightarrow$ Let's model $\mathrm{p}(\boldsymbol{x}, \boldsymbol{z} \mid \boldsymbol{\theta})$ instead!
- Let's recall what tools we developed during our discussion of supervised learning, and see if any of them can help us here.
- For this, assume for now, that we know the true cluster indicators $\boldsymbol{z}$, and are only interested in estimating $\boldsymbol{\theta}$.

Parameter estimation

How can we estimate the parameters $\boldsymbol{\theta}$ of a given model $\mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\theta})$?

Supervised learning

- Both $\boldsymbol{X} \in \mathbb{R}^{N \times D}$ and $\boldsymbol{Z} \in \{0,1\}^{N \times K}$ are known.
- Simple solution - maximum likelihood estimation

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} | \boldsymbol{\theta})$$

Unsupervised learning

- Only the data $\boldsymbol{X} \in \mathbb{R}^{N \times D}$ is given.
- Is there any way to estimate $\boldsymbol{\theta}$ in this case?

Latent variables

How can we estimate $\boldsymbol{\theta}$ when $\boldsymbol{Z}$ is unknown?

Since we chose to model $\mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\theta})$, we can compute the joint probability and obtain $\mathrm{p}(\boldsymbol{X} \mid \boldsymbol{\theta})$ by marginalization

cluster prior

$$\mathrm{p}(\boldsymbol{X} \mid \boldsymbol{\theta}) = \sum_{\boldsymbol{Z}} \mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\theta}) = \sum_{\boldsymbol{Z}} \mathrm{p}(\boldsymbol{X} \mid \boldsymbol{Z}, \boldsymbol{\theta}) \cdot \mathrm{p}(\boldsymbol{Z} \mid \boldsymbol{\theta})$$

and optimize w.r.t $\boldsymbol{\theta}$

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \mathrm{p}(\boldsymbol{X} \mid \boldsymbol{\theta})$$

Since the true $\boldsymbol{Z}$ are never observed and are only our abstraction used to simplify the model, they are called latent variables.[1]

[1] Sometimes also called hidden or explanatory variables.

Gaussian mixture model (GMM)

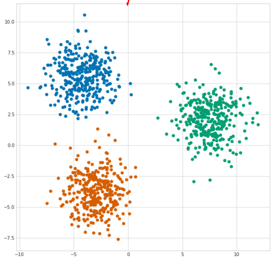
We decompose the joint probability $\mathrm{p}(\boldsymbol{x}, \boldsymbol{z} \mid \boldsymbol{\theta})$ using the product rule[2]

$$\mathrm{p}(\boldsymbol{x}, \boldsymbol{z} \mid \boldsymbol{\theta}) = \mathrm{p}(\boldsymbol{x} \mid \boldsymbol{z}, \boldsymbol{\theta}) \cdot \mathrm{p}(\boldsymbol{z} \mid \boldsymbol{\theta})$$

$\prod_i p(x_i, z_i | \theta)$

In Gaussian mixture model we assume:

μ, Σ for each



- Cluster prior is categorical

$\pi = [0.2, 0.5, 0.3]$

$$\mathrm{p}(\boldsymbol{z} \mid \boldsymbol{\theta}) = \mathrm{Cat}(\boldsymbol{\pi})$$

- Each cluster has its own multivariate normal distribution

Sample x belongs to cluster K

$$\mathrm{p}(\boldsymbol{x} \mid z_k = 1, \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

The parameters consist of the set of parameters of the prior and all conditional distributions $\boldsymbol{\theta} = \{\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}\}$.

[2] We assume that samples are drawn i.i.d., so consider only a single $(\boldsymbol{x}, \boldsymbol{z})$.

Generative process of GMM

We can draw samples from a GMM as follows.

Setup

We have K Gaussian components (clusters), each with known $\{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$, and known prior probabilities $\boldsymbol{\pi}_k$ for each cluster k.

Generative process

$\pi = [\pi_1, \dots, \pi_K]$

1. Draw a one-hot cluster indicator $\boldsymbol{z} \sim \mathrm{Cat}(\boldsymbol{\pi})$.

   $\boldsymbol{z}_k = 1 \iff$ current point belongs to cluster k.

2. Draw the sample $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ if $\boldsymbol{z}_k = 1$.

We only observe the sample $\boldsymbol{x}$, and don't know which k it came from.

GMM likelihood

Using marginalization over $\boldsymbol{z}$, the probability of a single sample $\boldsymbol{x}$ under a Gaussian mixture model can be computed as

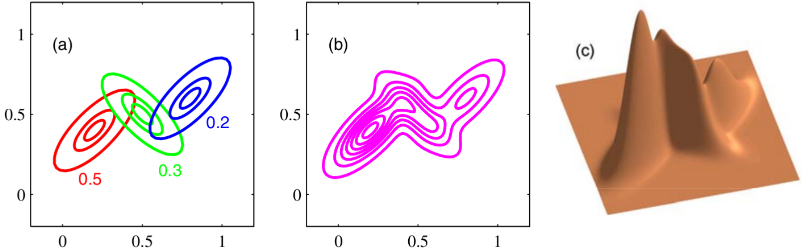
$$\begin{aligned}
\mathrm{p}(\boldsymbol{x} \mid \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) &= \sum_{k=1}^{K} \mathrm{p}(\boldsymbol{z}_k = 1 \mid \boldsymbol{\pi}) \cdot \mathrm{p}(\boldsymbol{x} \mid \boldsymbol{z}_k = 1, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \\
&= \sum_{k=1}^{K} \boldsymbol{\pi}_k \cdot \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)
\end{aligned}$$

prior of observing that particular cluster

Hence, for the entire dataset $\boldsymbol{X} = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_N\}$ the log-likelihood is

$$\log \mathrm{p}(\boldsymbol{X} | \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{i=1}^{N} \log \left(\sum_{k=1}^{K} \boldsymbol{\pi}_k \cdot \mathcal{N}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right)$$

GMM density



(a) Isocontours of each component $\mathrm{p}(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ with respective mixing coefficients $\boldsymbol{\pi}_k$.

(b) Isocontours of the GMM distribution

$$\mathrm{p}(\boldsymbol{x} \mid \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{k=1}^{K} \boldsymbol{\pi}_k \cdot \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

(c) Surface plot of $\mathrm{p}(\boldsymbol{x} \mid \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$.

Source: Bishop, Figure. 2.23

Learning and inference in GMM

Remember, that we are interested in two things

Learning

- We want to determine the "optimal" values of the parameters (e.g., find values that maximize the log-likelihood)

$$\boldsymbol{\pi}^*, \boldsymbol{\mu}^*, \boldsymbol{\Sigma}^* = \operatorname*{arg\,max}_{\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}} \log \mathrm{p}(\boldsymbol{X} \mid \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

Inference

- Given the parameters $\{\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}\}$, we want to assign the points to clusters, i.e. compute the posterior distribution of cluster indicators

$$\mathrm{p}(\boldsymbol{Z} \mid \boldsymbol{X}, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

(This is the actual goal of clustering.)

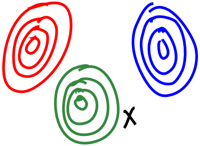
Inference in GMM

$\pi = [0.33, 0.33, \ldots]$

Uniform prior over clusters i.e all equally likely

higher density

$p(z_i \mid x_i, \theta) = [0.1, 0.1, 0.8]$



By straightforward application of Bayes' rule, we obtain the posterior

$$\mathrm{p}(\boldsymbol{z}_{ik} = 1 \mid \underbrace{\boldsymbol{x}_i, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}}_{\theta}) = \frac{\mathrm{p}(\boldsymbol{z}_{ik} = 1 \mid \boldsymbol{\pi})\ \mathrm{p}(\boldsymbol{x}_i \mid \boldsymbol{z}_{ik} = 1, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\mathrm{p}(\boldsymbol{x}_i \mid \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})}$$

$$= \frac{\boldsymbol{\pi}_k \mathcal{N}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \boldsymbol{\pi}_j \mathcal{N}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$$

$$p(Z \mid X, \theta) = \frac{p(X \mid Z, \theta)\ p(Z \mid \theta)}{p(X \mid \theta)}$$

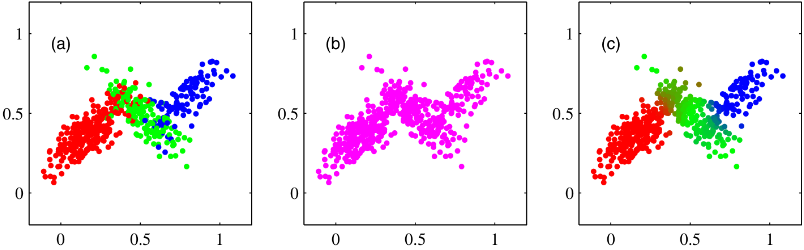
over the entries of $\boldsymbol{Z}$. We will call this quantity the responsibility of component k for the observation i.

We also introduce shorthand notation for it

$$\gamma(\boldsymbol{z}_{ik}) = \mathrm{p}(\boldsymbol{z}_{ik} = 1 \mid \boldsymbol{x}_i, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

* Unlike with K-means, where it was hard assignment, here it's soft assignment

Inference: Example



(a) Data colored according to the true underlying $\boldsymbol{Z}$.

(b) Observed data $\boldsymbol{X}$.

(c) Data colored according to the inferred $\gamma(\boldsymbol{Z})$ (with $\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$ known).

Source: Bishop, Figure. 9.5

Learning in GMM

Maximization of the log-likelihood w.r.t. the parameters appears to be more challenging. Recall that the log-likelihood is

$$\log \mathrm{p}(\boldsymbol{X} \,|\, \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{i=1}^{N} \log \left(\sum_{k=1}^{K} \boldsymbol{\pi}_k \cdot \mathcal{N}(\boldsymbol{x}_i \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right).$$

log(Σ ·) is always problematic

* Due to the summation over k, the logarithm doesn't act directly on the Gaussian density.

This happens to be a serious problem, as this means that the derivatives w.r.t. the model parameters $\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$ have no analytical roots.[3]

Is there anything that we can do in this case?

[3] i.e. we cannot obtain closed form expressions for the optimal values of $\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$.

Learning in GMM

Recall from the Linear Classification lecture (Linear Discriminant Analysis) that parameter estimation is rather straightforward if $\boldsymbol{Z}$ is known.

That is, we know how to solve the "easy" problem

$$\boldsymbol{\pi}^*, \boldsymbol{\mu}^*, \boldsymbol{\Sigma}^* = \arg\max_{\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}} \log \mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) \qquad (*)$$

Too bad we don't know anything about $\boldsymbol{Z}$... Or do we?

Learning in GMM

Recall from the Linear Classification lecture (Linear Discriminant Analysis) that parameter estimation is rather straightforward if $\boldsymbol{Z}$ is known.

That is, we know how to solve the "easy" problem

$$\boldsymbol{\pi}^*, \boldsymbol{\mu}^*, \boldsymbol{\Sigma}^* = \underset{\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}}{\arg\max} \ \log \mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) \qquad (*)$$

Too bad we don't know anything about $\boldsymbol{Z}$... Or do we?

Idea: use our initial beliefs $\gamma_0(\boldsymbol{Z}) = \mathrm{p}(\boldsymbol{Z} \mid \boldsymbol{X}, \boldsymbol{\pi}^{(0)}, \boldsymbol{\mu}^{(0)}, \boldsymbol{\Sigma}^{(0)})$ and solve the easier problem (*).

Such initial posterior turns to be alower bound

Learning in GMM: from $\mathrm{p}(\boldsymbol{X})$ to $\mathrm{p}(\boldsymbol{X}, \boldsymbol{Z})$

$\log p(X|\theta) = \log \sum_{Z} p(X, Z|\theta)$

Instead of optimizing $\mathrm{p}(\boldsymbol{X})$ directly, we introduce the latent variables $\boldsymbol{Z}$ by taking their expectation and formulate the proxy objective

$$\mathop{\mathbb{E}}_{\boldsymbol{Z} \sim \gamma_t(\boldsymbol{Z})} [\log \mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})] \,. \qquad = \sum_{K} \gamma_t(z_{iK}) \cdot \log p(x_i, z_i | \theta)$$

Now the $\log$ acts directly on $\mathrm{p}(\boldsymbol{X}, \boldsymbol{Z})$ and we can optimize the whole expression. Note, that $\gamma_t(\boldsymbol{Z})$ and the optimizers $\boldsymbol{\pi}^*$, $\boldsymbol{\mu}^*$ and $\boldsymbol{\Sigma}^*$ recursively depend on each other.

This motivates the expectation maximization (EM) algorithm that iterates the following two steps until convergence:

1. Evaluate the posterior distribution $\gamma_t(\boldsymbol{Z})$ with respect to the current estimate of the parameters $\{\boldsymbol{\pi}^{(t)}, \boldsymbol{\mu}^{(t)}, \boldsymbol{\Sigma}^{(t)}\}$
2. Obtain the next iteration of the parameters by solving

$$\boldsymbol{\pi}^{(t+1)}, \boldsymbol{\mu}^{(t+1)}, \boldsymbol{\Sigma}^{(t+1)} = \mathop{\arg\max}_{\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}} \mathop{\mathbb{E}}_{\boldsymbol{Z} \sim \gamma_t(\boldsymbol{Z})} [\log \mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})]$$

EM algorithm for GMM

1. Initialize model parameters $\{\boldsymbol{\pi}^{(0)}, \boldsymbol{\mu}_1^{(0)}, ..., \boldsymbol{\mu}_K^{(0)}, \boldsymbol{\Sigma}_1^{(0)}, ..., \boldsymbol{\Sigma}_K^{(0)}\}$.
2. **E step**. Evaluate the responsibilities

Inference Step

$$\gamma_t(\boldsymbol{z}_{ik}) = \frac{\boldsymbol{\pi}_k^{(t)} \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})}{\sum_{j=1}^{K} \boldsymbol{\pi}_j^{(t)} \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{\mu}_j^{(t)}, \boldsymbol{\Sigma}_j^{(t)})}.$$

applying Bayes

3. **M step**. Re-estimate the parameters

$$\operatorname*{argmax}_{\pi, \mu, \Sigma} \mathbb{E}_{Z \sim \gamma_t(Z)} \log p(X, Z | \pi, \mu, \Sigma)$$

$$\boldsymbol{\mu}_k^{(t+1)} = \frac{1}{N_k} \sum_{i=1}^{N} \gamma_t(\boldsymbol{z}_{ik}) \boldsymbol{x}_i$$

$$\boldsymbol{\Sigma}_k^{(t+1)} = \frac{1}{N_k} \sum_{i=1}^{N} \gamma_t(\boldsymbol{z}_{ik}) (\boldsymbol{x}_i - \boldsymbol{\mu}_k^{(t+1)}) (\boldsymbol{x}_i - \boldsymbol{\mu}_k^{(t+1)})^T$$

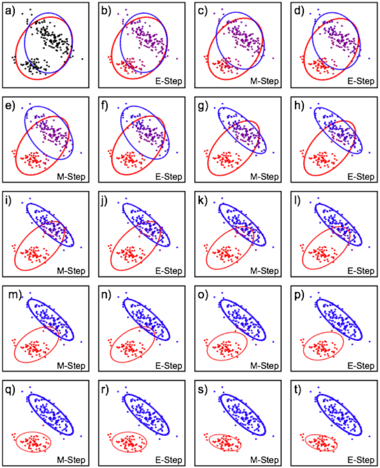
$$\boldsymbol{\pi}_k^{(t+1)} = \frac{N_k}{N} \qquad \text{where } N_k = \sum_{i=1}^{N} \gamma_t(\boldsymbol{z}_{ik}).$$

4. Iterate steps 2 & 3 until $\mathbb{E}_{\boldsymbol{Z} \sim \gamma_t(\boldsymbol{Z})} \left[\log \mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\pi}^{(t)}, \boldsymbol{\mu}^{(t)}, \boldsymbol{\Sigma}^{(t)}) \right]$ converges

EM algorithm for GMM: Demo

* Instead of centroids of K-means, Gaussian is used.

* Instead of hard assignment, posterior dist is used where each point shall be assigned to which cluster.



Gif: `https://upload.wikimedia.org/wikipedia/commons/6/69/EM_Clustering_of_Old_Faithful_data.gif`

Source: Computer Vision: Models, Learning, and Inference - S.J.Prince

EM algorithm in general

$\log p(X \mid \theta) = \log \sum_{Z} p(X, Z \mid \theta)$ $\int(\cdot)$

The Expectation-Maximization algorithm (EM) is applicable to more models than just GMMs.

In its most general form, the two steps are

- **E step** - evaluate the posterior $\gamma_t(\boldsymbol{Z}) = \mathrm{p}(\boldsymbol{Z} \mid \boldsymbol{X}, \boldsymbol{\theta}^{(t)})$.
- **M step** - maximize the expected joint log-likelihood $\log \mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\theta})$ w.r.t. $\boldsymbol{\theta}$ under the current beliefs $\gamma_t(\boldsymbol{Z})$

which is easier than $\log p(X \mid \theta)$

$$\boldsymbol{\theta}^{(t+1)} = \arg\max_{\boldsymbol{\theta}} \mathop{\mathbb{E}}_{\boldsymbol{Z} \sim \gamma_t(\boldsymbol{Z})} [\log \mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\theta})]$$

Note: we do **not** get a closed form solution neither for $\gamma(\boldsymbol{Z})$, nor for the parameters, as $\gamma(\boldsymbol{Z})$ and $\boldsymbol{\theta}$ recursively depend on each other.

Therefore, we have to iterate between these two steps until the objective $\mathbb{E}_{\boldsymbol{Z} \sim \gamma(\boldsymbol{Z})} [\log \mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\theta})]$ converges.

being alower bound $\log p(X \mid \Theta)$

EM algorithm: A justification

Let $\mathrm{p}(\boldsymbol{X} \mid \boldsymbol{\theta})$ be a parametric probabilistic model with latent variables $\boldsymbol{Z}$ and $\mathrm{q}(\boldsymbol{Z})$ any probability distribution. Then we have

$\sum_{Z} \cdot = 1$

this is not marginalization !!

$$
\begin{aligned}
\log \mathrm{p}(\boldsymbol{X} \mid \boldsymbol{\theta}) &= \sum_{\boldsymbol{Z}} \mathrm{q}(\boldsymbol{Z}) \log \mathrm{p}(\boldsymbol{X} \mid \boldsymbol{\theta}) \\
&= \sum_{\boldsymbol{Z}} \mathrm{q}(\boldsymbol{Z}) \log \frac{\mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\theta})}{\mathrm{p}(\boldsymbol{Z} \mid \boldsymbol{X}, \boldsymbol{\theta})} \\
&= \sum_{\boldsymbol{Z}} \mathrm{q}(\boldsymbol{Z}) \log \frac{\mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\theta})}{\mathrm{p}(\boldsymbol{Z} \mid \boldsymbol{X}, \boldsymbol{\theta})} \cdot \frac{\mathrm{q}(\boldsymbol{Z})}{\mathrm{q}(\boldsymbol{Z})} \\
&= \sum_{\boldsymbol{Z}} \mathrm{q}(\boldsymbol{Z}) \log \frac{\mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\theta})}{\mathrm{q}(\boldsymbol{Z})} - \sum_{\boldsymbol{Z}} \mathrm{q}(Z) \log \frac{\mathrm{p}(\boldsymbol{Z} \mid \boldsymbol{X}, \boldsymbol{\theta})}{\mathrm{q}(\boldsymbol{Z})} \\
&= \underbrace{\mathop{\mathbb{E}}_{\boldsymbol{Z} \sim \mathrm{q}} \left[\log \frac{\mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\theta})}{\mathrm{q}(\boldsymbol{Z})} \right]}_{\mathcal{L}(\mathrm{q}, \boldsymbol{\theta})} + \underbrace{\mathbb{KL}\left(\mathrm{q} \,||\, \mathrm{p}(\cdot \mid \boldsymbol{X}, \boldsymbol{\theta})\right)}_{\geqslant 0} .
\end{aligned}
$$

$\mathcal{L}(\mathrm{q}, \boldsymbol{\theta})$ is a lower-bound on $\log \mathrm{p}(\boldsymbol{X} \mid \boldsymbol{\theta})$ for *any* q.

EM algorithm: The E-step

$$\log \mathrm{p}(\boldsymbol{X} \mid \boldsymbol{\theta}) = \mathcal{L}(\mathrm{q}, \boldsymbol{\theta}) + \mathbb{KL}\left(\mathrm{q} \,||\, \mathrm{p}(\,\cdot \mid \boldsymbol{X}, \boldsymbol{\theta})\right)$$

The lefthand side is constant with respect to q and the $\mathbb{KL}$ divergence is non-negative. Therefore $\mathcal{L}(\mathrm{q}, \boldsymbol{\theta})$ is maximal when the $\mathbb{KL}$ divergence is 0 which happens when $\mathrm{q}(\boldsymbol{Z}) = \gamma(\boldsymbol{Z}) = \mathrm{p}(\boldsymbol{Z} \mid \boldsymbol{X}, \boldsymbol{\theta})$.

This constitutes the E-step and ensures that the lower bound $\mathcal{L}(\mathrm{q}, \boldsymbol{\theta})$ is tight, i.e.

tight

$$\log \mathrm{p}(\boldsymbol{X} \mid \boldsymbol{\theta}) = \mathcal{L}(\mathrm{q}, \boldsymbol{\theta}).$$

Note that

$$\mathcal{L}(\mathrm{q}, \boldsymbol{\theta}) = \mathop{\mathbb{E}}_{\boldsymbol{Z} \sim \mathrm{q}}\left[\log \frac{\mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\theta})}{\mathrm{q}(\boldsymbol{Z})}\right] = \underbrace{\mathop{\mathbb{E}}_{\boldsymbol{Z} \sim \mathrm{q}}\left[\log \mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\theta})\right]}_{\text{our proxy from GMMs}} + \underbrace{\mathbb{H}[\mathrm{q}(\,\cdot\,)]}_{\text{independent of } \boldsymbol{\theta}}.$$

entropy of q

EM algorithm: The M-step

$$\log \mathrm{p}(\boldsymbol{X} \mid \boldsymbol{\theta}) = \mathcal{L}(\mathrm{q}, \boldsymbol{\theta}) + \mathbb{KL}\left(\mathrm{q} \,||\, \mathrm{p}(\cdot \mid \boldsymbol{X}, \boldsymbol{\theta})\right)$$

After the E-step we have $\log \mathrm{p}(\boldsymbol{X} \mid \boldsymbol{\theta}) = \mathcal{L}(\mathrm{q}, \boldsymbol{\theta})$. So if we now perform the M-step and maximize $\mathcal{L}(\mathrm{q}, \boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$, either

- $\mathcal{L}(\mathrm{q}, \boldsymbol{\theta})$ does not change, we are at a local maximum of $\log \mathrm{p}(\boldsymbol{X} \mid \boldsymbol{\theta})$ and EM converges
- or $\mathcal{L}(\mathrm{q}, \boldsymbol{\theta})$ "pushes" up against $\log \mathrm{p}(\boldsymbol{X} \mid \boldsymbol{\theta})$.

In the latter case, we receive parameters $\boldsymbol{\theta}'$ that have a higher data log-likelihood. Furthermore, the same equation that we started with holds again

$$\log \mathrm{p}(\boldsymbol{X} \mid \boldsymbol{\theta}') = \mathcal{L}(\mathrm{q}, \boldsymbol{\theta}') + \mathbb{KL}\left(\mathrm{q} \,||\, \mathrm{p}(\cdot \mid \boldsymbol{X}, \boldsymbol{\theta}')\right)$$

and we can perform another iteration.

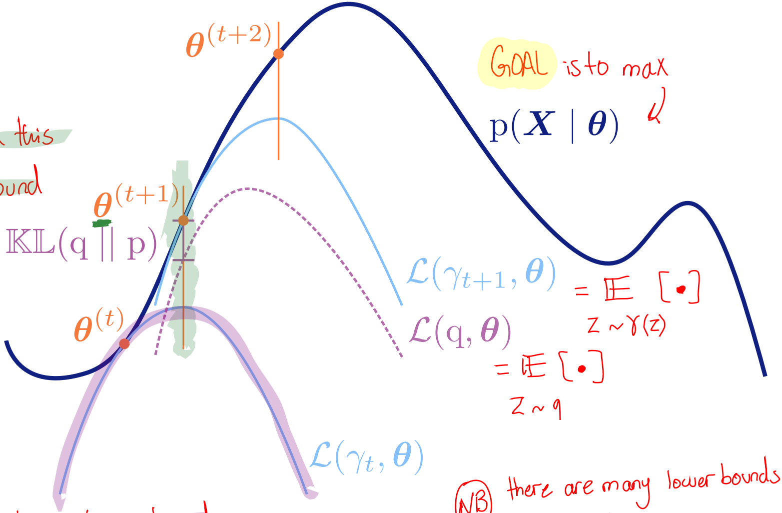
At each iter, $\log p(X|\theta)$ either

- increases
- stays the same $\rightarrow$ converged to local max

EM algorithm: Illustration

Local opt → guaranteed

global → not necessarily!



GOAL is to max

max this lower bound

$= \mathbb{E}_{z \sim \gamma(z)} [\bullet]$

$= \mathbb{E}_{z \sim q} [\bullet]$

Constructing a lower bound

NB there are many lower bounds, yet we choose the tighest!

EM algorithm: Summary

When should we use EM?

- Intractable to optimize the log-likelihood due to latent variables $\boldsymbol{Z}$

$$\log \mathrm{p}(\boldsymbol{X} \mid \boldsymbol{\theta}) = \log \left(\sum_{\boldsymbol{Z}} \mathrm{p}(\boldsymbol{X} \mid \boldsymbol{Z}, \boldsymbol{\theta})\, \mathrm{p}(\boldsymbol{Z} \mid \boldsymbol{\theta}) \right)$$

- Easy to optimize the joint probability

$$\log \mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\theta}) = \log \mathrm{p}(\boldsymbol{Z} \mid \boldsymbol{\theta}) + \log \mathrm{p}(\boldsymbol{X} \mid \boldsymbol{Z}, \boldsymbol{\theta})$$

The main idea of EM

- Use our current beliefs $\mathrm{p}(\boldsymbol{Z} \mid \boldsymbol{X}, \boldsymbol{\theta}^{(t)})$ to "pretend" that we know $\boldsymbol{Z}$.
- E step - update the responsibilities $\gamma_t(\boldsymbol{Z}) = \mathrm{p}(\boldsymbol{Z} \mid \boldsymbol{X}, \boldsymbol{\theta}^{(t)})$.
- M step - update parameters

$$\boldsymbol{\theta}^{(t+1)} = \arg\max_{\boldsymbol{\theta}} \mathop{\mathbb{E}}_{\boldsymbol{Z} \sim \gamma_t(\boldsymbol{Z})} \left[\log \mathrm{p}(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\theta})\right]$$

Choosing number of clusters K: Heuristic methods

- Elbow/knee heuristic:
  Plot within-cluster sum of squared distances for varying K. Look for a bent (knee/elbow).

  $K \to N \Rightarrow$ each point gets its own cluster
- Gap statistic:
  Compare within-cluster variation to uniform data. Choose smallest K such that gap statistic is within one std. dev. of gap at $K+1$.
- Silhouette:
  Per point difference of mean distance within cluster to points in closest other cluster. Maximize mean of all points.

  (avg to other instances within own cluster) $-$ (avg ... within 2nd closest cluster)

* Don't forget cross-validation to prevent overfitting!

Choosing number of clusters K: Probabilistic methods

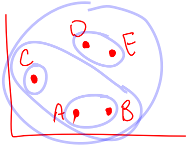
Needs generative model that defines the data likelihood $\hat{L} = \mathrm{p}(\boldsymbol{X} \mid \hat{\boldsymbol{Z}}, \hat{\boldsymbol{\theta}})$ (with optimal parameters $\hat{\boldsymbol{Z}}$, $\hat{\boldsymbol{\theta}}$).

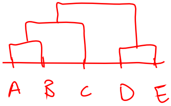
- Bayesian information criterion (BIC):
  Approximate model likelihood $\mathrm{p}(\boldsymbol{X} \mid \text{model})$. Balances number of parameters vs. likelihood, $\mathrm{BIC} = M \log N - 2 \log \hat{L}$.

- Akaike information criterion (AIC):
  Estimate information lost by given model, $\mathrm{AIC} = 2M - 2 \log \hat{L}$

- Here M is the number of free parameters (e.g $K \cdot (D + D^2 + 1)$ for GMM) and N is the number of samples

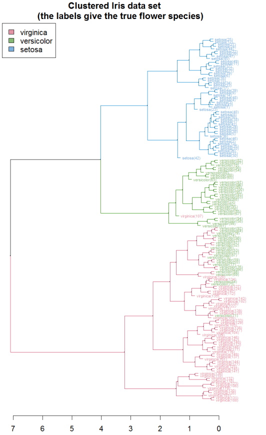
Hierarchical clustering





- Agglomerative: Bottom-up, merge cluster pairs iteratively
- Divisive: Top-down, split data recursively

NB No need to choose K



From https://en.wikipedia.org/wiki/File:Iris_dendrogram.png

Agglomorative clustering: How to choose pairs?

Cluster pairs are chosen by minimum cluster distance.

Linkage criterion defines cluster distance from sample distance:

- Complete-linkage clustering: Maximum sample distance
- Single-linkage clustering: Minimum sample distance
- Unweighted average linkage clustering (UPGMA): Mean sample distance
- Centroid linkage clustering (UPGMC): Centroid distance
- Weighted average linkage clustering (WPGMA): Distance defined recursively as average of distances in previous level (before merging)

Summary

- Clustering: Group objects into K groups
- K-means: Alternatingly (1) group objects by the closest centroids and (2) set the centroids to the group means

  Note: K-means update equivalent to isotropic GMM with $\sigma \to 0$! (exercise)
- GMM: Fit set of Gaussians to the data (via EM), group is the Gaussian with highest probability
- EM for GMM: Alternatingly update (1) responsibilities based on parameters and (2) parameters based on responsibilities.
- EM in general: Alternatingly (1) evaluate the expected posterior $\gamma_t(\boldsymbol{Z})$ (E-step) and (2) update parameters by maximizing $\mathcal{L}(\gamma_t, \boldsymbol{\theta})$ (M-step).
- Agglomorative clustering: Merge clusters recursively

Reading material

Reading material

- Bishop: chapters 9.1, 9.2, 9.3.0, 9.3.1
- Murphy: chapter 11.4.1, 11.4.2, 11.4.7 (optional)