

*** Applied Machine Learning Fundamentals ***

Probability Density Estimation (PDE)

Daniel Wehner, M.Sc.

SAP SE / DHBW Mannheim

Winter term 2022/2023



Find all slides on [GitHub](#) (DaWe1992/Applied_ML_Fundamentals)

Lecture Overview

Unit I	Machine Learning Introduction
Unit II	Mathematical Foundations
Unit III	Bayesian Decision Theory
Unit IV	Probability Density Estimation
Unit V	Regression
Unit VI	Classification I
Unit VII	Evaluation
Unit VIII	Classification II
Unit IX	Clustering
Unit X	Dimensionality Reduction

Agenda for this Unit

1 Introduction

What about continuous Data?
Methods for PDE

2 Parametric Models

General Idea
Parameter Learning and Assumptions
Maximum Likelihood Estimation (MLE)

3 Non-parametric Models

Motivation
Non-parametric Approaches
Histograms
Kernel Density Estimation

k -Nearest Neighbors

4 Mixture Models

General Idea
Mixture of Gaussians (MoG)
Expectation Maximization for MoG
Recommendations

5 Wrap-Up

Summary
Self-Test Questions
Lecture Outlook
Recommended Literature and further Reading
Meme of the Day

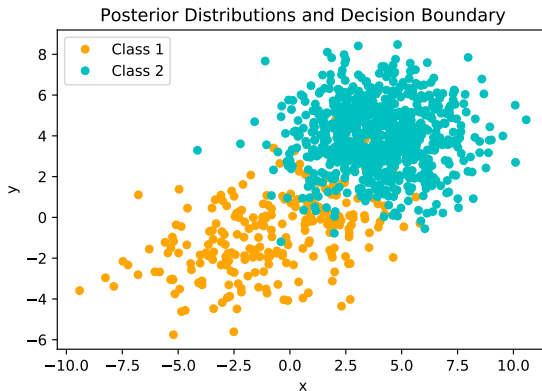
Section:
Introduction



Probability Density Estimation (PDE)

- We have learned about Bayes' optimal classifiers which classify data based on the probability distribution $p(\mathbf{x}|\mathcal{C}_k) \cdot p(\mathcal{C}_k)$
- (Multinomial) Naïve Bayes is an instance of PDE for **discrete data**
- **How to get these probabilities in the continuous case?**
 - The prior $p(\mathcal{C}_k)$ is still easy to compute
 - The estimation of class conditional probabilities $p(\mathbf{x}|\mathcal{C}_k)$ is more complicated
 - Assume labeled data; estimate the density separately for each class \mathcal{C}_k
- NB: For ease of notation: $p(\mathbf{x}) \equiv p(\mathbf{x}|\mathcal{C}_k)$

Training Data Example



Overview of the Methods for PDE

① Parametric models (maximum likelihood estimation)

- Assume a fixed parametric form (e. g. a Gaussian distribution)
- Estimate the parameters such that the model fits the data best

② Non-parametric models

- Often we do not know the functional form of the density
- Estimate probability directly from the data without an explicit model

③ Mixture models

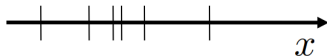
- Combination of ① and ②
- EM algorithm

Section:
Parametric Models

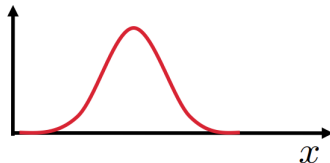


General Approach

- Given some (continuous) training data $\mathbf{X} = \{x^{(i)}\}_{i=1}^n$ (where all $x^{(i)}$ belong to the same class):



- Estimate $p(x)$ using a fixed parametric form:



Example: Gaussian Distribution

- One common case is the **Gaussian distribution**:

$$p(x|\mu, \sigma^2) = \mathcal{N}(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left\{ -\frac{(x - \mu)^2}{2\sigma^2} \right\} \quad (1)$$

- Notation for parametric models:

- $p(x|\theta)$
- In the case of a Gaussian: $\theta = \{\mu, \sigma^2\}$

$\mu \equiv \text{mean}$
 $\sigma^2 \equiv \text{variance}$

Learning the Parameters

- Learning means estimating the parameters θ given the data \mathbf{X}
- **Likelihood** of the parameters θ :
 - Is defined as the probability that \mathbf{X} was generated by a probability density function (pdf) with parameters θ

$$\mathcal{L}(\theta) = p(\mathbf{X}|\theta) \quad (2)$$

- We want to **maximize** the likelihood

⇒ **Maximum likelihood estimation (MLE)**

A fundamental Assumption

- How to compute $\mathcal{L}(\boldsymbol{\theta})$?
- The data is assumed to be **i.i.d.** (independent and identically distributed):
 - Two random variables x_1 and x_2 are independent, if

$$P(x_1 \leq \alpha, x_2 \leq \beta) = P(x_1 \leq \alpha) \cdot P(x_2 \leq \beta) \quad \forall \alpha, \beta \in \mathbb{R} \quad (3)$$

- Two random variables x_1 and x_2 are identically distributed, if

$$P(x_1 \leq \alpha) = P(x_2 \leq \alpha) \quad \forall \alpha \in \mathbb{R} \quad (4)$$

Computation of the Likelihood

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}) &= p(\mathbf{X}|\boldsymbol{\theta}) \\ &= p(x^{(1)}, x^{(2)}, \dots, x^{(n)}|\boldsymbol{\theta})\end{aligned}$$

data is independent:

$$= p(x^{(1)}|\boldsymbol{\theta}) \cdot p(x^{(2)}|\boldsymbol{\theta}) \cdot \dots \cdot p(x^{(n)}|\boldsymbol{\theta})$$

data is identically distributed:

$$= \prod_{i=1}^n p(x^{(i)}|\boldsymbol{\theta}) \quad \boxed{\text{What is the problem here?}} \quad (5)$$

Computation of the Likelihood (Ctd.)

- **Problem:** Large n might cause arithmetic underflows! (why?)
- Transform the likelihood using the logarithm \Rightarrow **log-likelihood**

$$\mathcal{LL}(\boldsymbol{\theta}) = \log \mathcal{L}(\boldsymbol{\theta})$$

Why is this an
allowed transformation?

$$= \log \prod_{i=1}^n p(x^{(i)} | \boldsymbol{\theta})$$

$$\log \Pi = \Sigma \log$$

$$= \sum_{i=1}^n \log p(x^{(i)} | \boldsymbol{\theta}) \quad (6)$$

Maximum Likelihood of a Gaussian

- $\theta = \{\mu, \sigma^2\}$

$$\mathcal{LL}(\{\mu, \sigma^2\}) = \sum_{i=1}^n \log \mathcal{N}(x^{(i)} | \mu, \sigma^2) \quad (7)$$

$$= \sum_{i=1}^n \log \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left\{ -\frac{(x^{(i)} - \mu)^2}{2\sigma^2} \right\} \quad (8)$$

- Find μ_{ml} and σ_{ml}^2 which maximize the log-likelihood:

$$\mu_{ml}, \sigma_{ml}^2 = \arg \max_{\mu, \sigma^2} \mathcal{LL}(\theta)$$

Maximum Likelihood of a Gaussian (Ctd.)

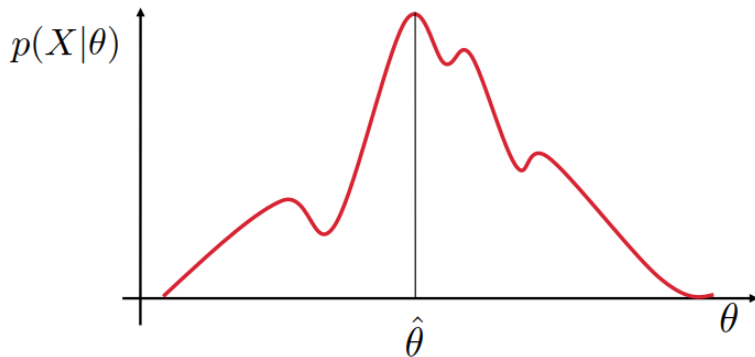
- Compute the partial derivatives with respect to the parameters θ
- Derivative w. r. t. μ :

$$\nabla_{\mu} \mathcal{L}(\theta) = \nabla_{\mu} \sum_{i=1}^n \log \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left\{ -\frac{(x^{(i)} - \mu)^2}{2\sigma^2} \right\} = \sum_{i=1}^n \frac{x^{(i)} - \mu}{\sigma^2}$$

- Set derivative to zero and solve:

$$\sum_{i=1}^n (x^{(i)} - \mu) \stackrel{!}{=} 0 \Leftrightarrow n \cdot \mu = \sum_{i=1}^n x^{(i)} \Leftrightarrow \mu = \frac{1}{n} \sum_{i=1}^n x^{(i)}$$

Maximization of the Likelihood



We can classify!

- Maximum likelihood parameters:

Looks familiar?

$$\mu_{ml} = \frac{1}{n} \sum_{i=1}^n x^{(i)}$$

$$\sigma_{ml}^2 = \frac{1}{n} \sum_{i=1}^n (x^{(i)} - \mu_{ml})^2$$

- Now we can use Bayes' rule to predict class labels
 - We have the priors...
 - ...and the class conditionals
- Also, the **decision boundary** can be computed

Multivariate Case

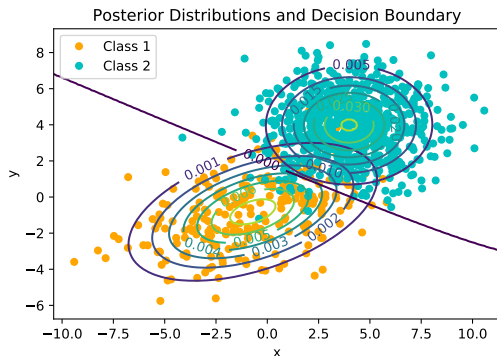
- The solution above is for 1-D data; what if we have more dimensions?
- **Multivariate Gaussian distribution:**

$$\mathcal{N}_D(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^D |\boldsymbol{\Sigma}|}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\} \quad (9)$$

- Luckily, the derivations don't change:

$$\boldsymbol{\mu}_{ml} = \frac{1}{n} \sum_{i=1}^n \mathbf{x}^{(i)} \quad \boldsymbol{\Sigma}_{ml} = \frac{1}{n} \sum_{i=1}^n (\mathbf{x}^{(i)} - \boldsymbol{\mu}_{ml})(\mathbf{x}^{(i)} - \boldsymbol{\mu}_{ml})^\top \quad (10)$$

Gaussian naïve Bayes – Final Model



$$p(\mathcal{C}_k|\mathbf{x}) = \mathcal{N}_D(\mathbf{x}|\boldsymbol{\mu}_{\mathcal{C}_k}, \boldsymbol{\Sigma}_{\mathcal{C}_k}) \cdot p(\mathcal{C}_k)$$

NB: $\mathcal{N}_D(\mathbf{x}|\boldsymbol{\mu}_{\mathcal{C}_k}, \boldsymbol{\Sigma}_{\mathcal{C}_k})$ denotes the Gaussian distribution estimated for class \mathcal{C}_k (using MLE). $p(\mathcal{C}_k)$ is the prior probability of class \mathcal{C}_k (as in the discrete case).

Generative vs. Discriminative Models

Generative Model

The artist



A **generative** algorithm models **how** the data was generated. **It models the respective probability distributions.**

Discriminative Model

The lousy painter



A **discriminative** algorithm does not care about how the data was generated. **It only knows how to distinguish the classes.**

Section:
Non-parametric Models





Disadvantages of parametric Models

- Until now we used a fixed parametric form (e.g. a Gaussian) which is governed by a small amount of parameters
- **This assumption may be wrong:**
 - Another distribution (exponential, gamma, ...) may fit better
 - A suitable 'text-book distribution' may not exist

We don't want to make any assumptions about the underlying distribution!

Non-parametric Approaches

- ① **Histograms** (Binning)
- ② **Kernel density estimation** (KDE)
- ③ **Nearest neighbors** (kNN)

Histograms

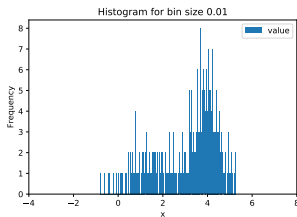
- Histograms partition the data $\mathbf{X} = \{\mathbf{x}^{(i)}\}_{i=1}^n$ into distinct **bins** of volume v_j ...
- ...and subsequently count the number of instances k_j falling into the j -th bin
- Approximate the probability $p(\mathbf{x})$ by:

$$p(\mathbf{x}) \approx \frac{k_j}{n \cdot v_j} \quad \text{for } \mathbf{x} \text{ in bin } j \quad (11)$$

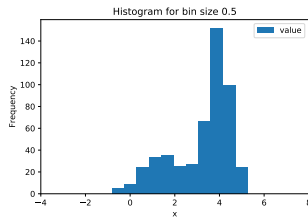
- The sum of all probabilities equals 1: $\sum_j \frac{k_j}{n \cdot v_j} = 1$
- v_j is a **hyper-parameter** (usually all bins have equal size)

Histograms (Ctd.)

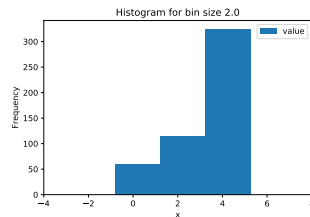
Too narrow



About right

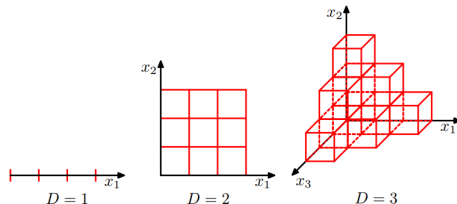


Too wide



Drawbacks of Histograms

- Histograms are mostly unsuited for many applications
- Drawbacks:**
 - Discontinuities due to bin edges
 - Number of bins **explodes** with growing number of dimensions D



The latter issue is known as the curse of dimensionality

An alternative Approach

- Don't use a fixed number of pre-determined bins
- Instead, employ a **sliding window** approach by centering a region \mathcal{R} (bin) around the data point of interest \mathbf{x}

$$p(\mathbf{x}) \approx \frac{k}{n \cdot v} \quad (12)$$

- This gives rise to two different techniques:
 - ① **Kernel density estimation** (Fix v and determine k)
 - ② **k-nearest neighbors** (Fix k and determine v)

Kernel Density Estimation: Parzen Window

- \mathcal{R} is a D -dimensional **hyper-cube** of edge length h centered on \mathbf{x}
- Determine if a data point falls into region \mathcal{R} :

$$H(\mathbf{u}) = \begin{cases} 1 & \text{if } |u_d| \leq h/2, d = 1, 2, \dots, D \\ 0 & \text{otherwise} \end{cases} \quad (13)$$

- The total number of data points falling into region \mathcal{R} is given by:

$$k(\mathbf{x}) = \sum_{i=1}^n H(\mathbf{x} - \mathbf{x}^{(i)}) \quad (14)$$

Kernel Density Estimation: Parzen Window (Ctd.)

- The volume v is simple to compute:

$$v = \int H(\mathbf{u}) d\mathbf{u} = h^D \quad (15)$$

- Putting it all together we get:

$$p(\mathbf{x}) \approx \frac{k(\mathbf{x})}{n \cdot v} = \frac{1}{n \cdot h^D} \sum_{i=1}^n H(\mathbf{x} - \mathbf{x}^{(i)}) \quad (16)$$

- Problem:** There are still discontinuities



Kernel Density Estimation: Gaussian Kernel

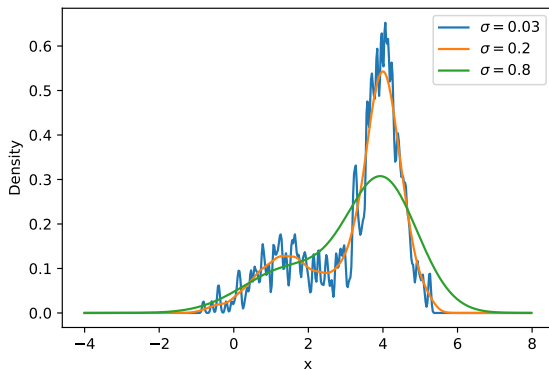
$$H(\mathbf{u}) = \frac{1}{\left(\sqrt{2\pi h^2}\right)^D} \exp \left\{ -\frac{\|\mathbf{u}\|^2}{2h^2} \right\} \quad (17)$$

$$v = \int H(\mathbf{u}) d\mathbf{u} = 1 \quad (18)$$

$$k(\mathbf{x}) = \sum_{i=1}^n H(\mathbf{x} - \mathbf{x}^{(i)}) \quad (19)$$

$$p(\mathbf{x}) \approx \frac{k(\mathbf{x})}{n \cdot v} = \frac{1}{n \cdot \left(\sqrt{2\pi h^2}\right)^D} \sum_{i=1}^n \exp \left\{ -\frac{\|\mathbf{x} - \mathbf{x}^{(i)}\|^2}{2h^2} \right\} \quad (20)$$

Kernel Density Estimation: Gaussian Kernel (Ctd.)



k-Nearest Neighbors

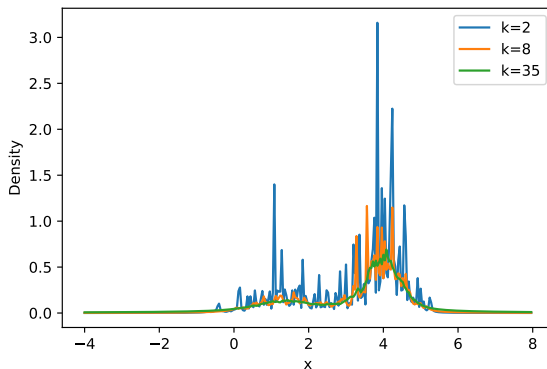
Different strategy:

- Fix k and increase the volume, until k data points fall into region \mathcal{R}

$$p(\mathbf{x}) \approx \frac{k}{n \cdot v(\mathbf{x})} \quad (21)$$

- **Usually, kernel density estimation gives better results!**
- We will also look at k -nearest neighbors as a classification method later!

k-Nearest Neighbors (Ctd.)



Section:
Mixture Models



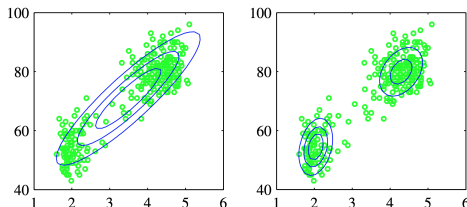
Why do we need Mixture Models?

- Parametric models have low memory footprint, are quick at runtime and often have nice analytic properties
- Non-parametric models make fewer assumptions about the data, but are slower and have a high memory footprint
- **We can combine different models in a mixture model!**

$$p(\mathbf{x}) = \sum_{j=1}^M p(\mathbf{x}|j)p(j) \quad (22)$$

Why do we need Mixture Models? (Ctd.)

- A single parametric model might fail to capture the structure of the data set
Solution: Use more components



- Mixture distributions (e. g. combination of Gaussians) can approximate almost any continuous density to arbitrary accuracy (given a sufficient number of Gaussians is used)

Mixture of Gaussians (MoG)

$$p(x) = \sum_{j=1}^M p(x|j)p(j) \quad \text{probability of data given comp. } j \times \text{probability of comp. } j \quad (23)$$

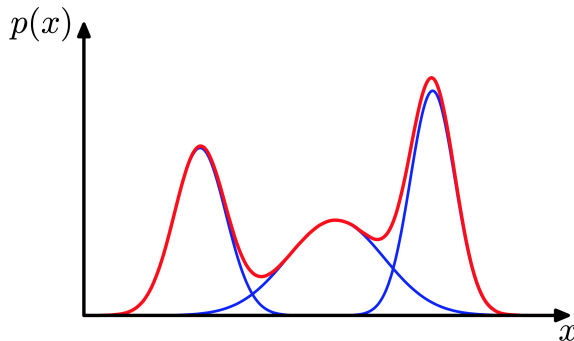
$$p(x|j) = \mathcal{N}(x|\mu_j, \sigma_j) = \frac{1}{\sqrt{2\pi\sigma_j^2}} \exp \left\{ -\frac{(x - \mu_j)^2}{2\sigma_j^2} \right\} \quad (24)$$

$$p(j) = \pi_j \quad \text{with} \quad 0 \leq \pi_j \leq 1 \quad \text{and} \quad \sum_{j=1}^M \pi_j = 1 \quad (25)$$

Remarks:

- The mixture density integrates to 1: $\int p(x) dx = 1$
- The mixture parameters are: $\theta = \{\mu_1, \sigma_1, \pi_1, \dots, \mu_M, \sigma_M, \pi_M\}$

Mixture of Gaussians (Ctd.)



The mixture of Gaussians (red) is obtained by summing over individual Gaussians (blue)

Maximum Likelihood Estimation for MoG

- We have defined our Gaussian mixture model: $p(x) = \sum_{j=1}^M p(x|j)p(j)$
- Maximize the **log-likelihood** to estimate the parameters θ :

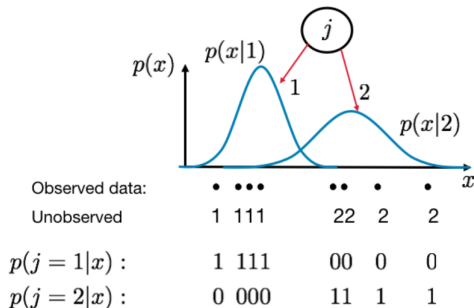
$$\mathcal{LL} = \log \mathcal{L}(\theta) = \sum_{i=1}^n \log p(x^{(i)}|\theta) \quad (26)$$

$$\nabla_{\mu_j} \mathcal{LL} \stackrel{!}{=} 0 \quad \mu_j = \frac{\sum_{i=1}^n p(j|x^{(i)})x^{(i)}}{\sum_{i=1}^n p(j|x^{(i)})} \quad (27)$$

- Do you see the issue? \Rightarrow **Circular dependency, no analytical solution!**

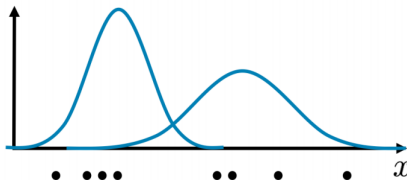
Expectation Maximization (EM)

Different strategy: We have observed data (without labels) $x^{(i)}$ and unobserved / hidden / latent variables $j|x$



Expectation Maximization (Ctd.)

- **Suppose we knew the observed and the unobserved data set:**
We could compute the maximum likelihood solution of all components
- **Suppose we knew the distributions:**
We could infer the labels for the unobserved data

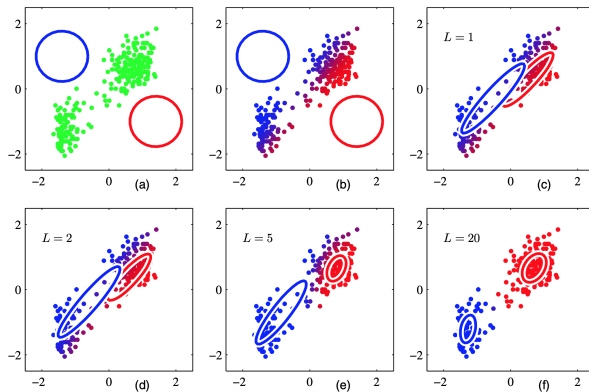


- **We have neither! \Rightarrow Chicken-Egg-Problem!**

Expectation Maximization: General Procedure

- So, how can we estimate the mixture parameters?
- **EM algorithm:**
 - ① Start with an initial guess for the parameters
 - ② **E-step:** Assign each data point $x^{(i)}$ to a component and compute $p(j|x^{(i)})$:
 - *Hard assignment:* Each data point is assigned to exactly one component
 - *Soft assignment:* Use soft probabilities instead
 - ③ **M-step:** Update the parameters based on the assignments
 - ④ If not converged: Go to ②

Expectation Maximization: General Procedure (Ctd.)



Expectation Maximization for MoG

EM for Gaussian Mixture Models:

- Initialize μ_j, σ_j, π_j
- While stop-condition is not met:
 - **E-step:** Compute the posterior distribution (a. k. a. responsibility α) for each mixture component and all data points:

$$\alpha_{ij} = p(j|x^{(i)}) = \frac{\pi_j \mathcal{N}(x^{(i)}|\mu_j, \sigma_j)}{\sum_{k=1}^M \pi_k \mathcal{N}(x^{(i)}|\mu_k, \sigma_k)} \quad (28)$$

- **M-step:** Compute new parameters using the responsibilities (cf. next slide)
- Iterate until converged

M-Step in Detail

- Update means:

$$\mu_j^{(new)} = \frac{1}{n_j} \sum_{i=1}^n \alpha_{ij} x^{(i)} \quad \text{with} \quad n_j = \sum_{i=1}^n \alpha_{ij} \quad (29)$$

- Update variance:

$$(\sigma_j^{(new)})^2 = \frac{1}{n_j} \sum_{i=1}^n \alpha_{ij} (x^{(i)} - \mu_j^{(new)})^2 \quad (30)$$

- Update π_j : $\pi_j^{(new)} = \frac{n_j}{n}$

Expectation Maximization: General Remarks

- EM is a general framework and not limited to mixture models
- We can use EM for performing maximum likelihood estimation, even when the data is incomplete (missing features)
- The log-likelihood is guaranteed to improve or stay the same in every EM iteration \Rightarrow **Convergence guarantee!**
- Visualizations of EM for Gaussian mixture models:
 - EM density estimation animation
 - 2-dimensional EM animation

Expectation Maximization: Some Recommendations

- **How do we initialize the parameters for EM?**
 - EM depends on a good initialization of the parameters, a poor initialization can lead to bad local optima
 - We can use *k-means* to get an initial clustering
- **How many mixture components do we need?**
 - Use M which maximizes the **Bayesian information criterion (BIC)**:

$$\log p(\mathbf{X}|\boldsymbol{\theta}_{ML}) - \frac{1}{2}K \log n \quad (31)$$

- K : Number of parameters
- n : Number of data points

Section:
Wrap-Up



Summary

- We can use **parametric**, **non-parametric** and **mixture models** to estimate the density
- This allows us to estimate the probabilities needed by e. g. a naïve Bayes model to work with **continuous features**
- Parametric models assume a certain **parametric form**, e. g. a Gaussian
- **MLE** allows us to determine the parameters based on our dataset
- Non-parametric models directly **use the data points themselves**
- Use the **EM algorithm** to optimize the parameters of mixture models



Self-Test Questions

- 1 What is maximum likelihood estimation? How can you get the maximum likelihood estimate for a Gaussian distribution?
- 2 What does the term '*non-parametric*' mean? How many parameters does such a model have?
- 3 What distinguishes kernel density estimation and k -nearest neighbors?
- 4 Why can't we use a simple maximum likelihood estimate for mixture models?
- 5 What happens in the E and M steps in the EM algorithm?

What's next...?

Unit I	Machine Learning Introduction
Unit II	Mathematical Foundations
Unit III	Bayesian Decision Theory
Unit IV	Probability Density Estimation
Unit V	Regression
Unit VI	Classification I
Unit VII	Evaluation
Unit VIII	Classification II
Unit IX	Clustering
Unit X	Dimensionality Reduction

Recommended Literature and further Reading I



[1] Pattern Recognition and Machine Learning

Christopher Bishop. Springer. 2006.

→ [Link](#), cf. chapters 1.2.4, 2.5, 9.2

Meme of the Day



Thank you very much for the attention!

Topic: *** Applied Machine Learning Fundamentals *** Probability Density Estimation (PDE)

Term: Winter term 2022/2023

Contact:

Daniel Wehner, M.Sc.

SAP SE / DHBW Mannheim

daniel.wehner@sap.com

Do you have any questions?