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

Daniel Wehner, M.Sc.

SAP SE / DHBW Mannheim

Winter term 2020/2021

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

Section:

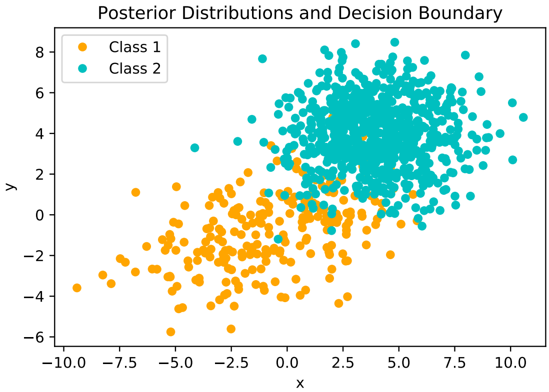
Introduction

Duale Hochschule Baden-Württemberg

Probability Density Estimation (PDE)

- We have learned about Bayes' optimal classifiers which classify data based on the probability distribution $p(\boldsymbol{x}|\mathcal{C}_k) \cdot p(\mathcal{C}_k)$
- (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 $p(\boldsymbol{x}|\mathcal{C}_k)$ is more complicated
  - Assume labeled data; estimate the density separately for each class $\mathcal{C}_k$
- NB: For ease of notation: $p(x) \equiv p(x|\mathcal{C}_k)$

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 ❶ and ❷
   - EM algorithm

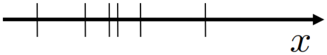
Section:

Parametric Models

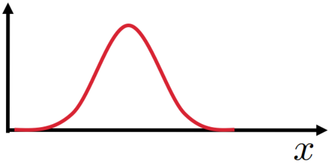
Duale Hochschule
Baden-Württemberg

General Approach

- Given some (continuous) training data $\boldsymbol{X} = \{x^{(i)}\}_{i=1}^{n}$ (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**:

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

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

$\mu \equiv$ mean
$\sigma^2 \equiv$ variance

Learning the Parameters

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

$$\mathcal{L}(\boldsymbol{\theta}) = p(\boldsymbol{X}|\boldsymbol{\theta}) \tag{2}$$

  - We want to **maximize** the likelihood

$\Rightarrow$ **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

$$P(x_1 \leqslant \alpha, x_2 \leqslant \beta) = P(x_1 \leqslant \alpha) \cdot P(x_2 \leqslant \beta) \qquad \forall \alpha, \beta \in \mathbb{R} \tag{3}$$

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

$$P(x_1 \leqslant \alpha) = P(x_2 \leqslant \alpha) \qquad \forall \alpha \in \mathbb{R} \tag{4}$$

Computation of the Likelihood

$$\mathcal{L}(\boldsymbol{\theta}) = p(\boldsymbol{X}|\boldsymbol{\theta})$$

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

data is independent:

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

data is identically distributed:

$$= \prod_{i=1}^{n} p(x^{(i)}|\boldsymbol{\theta}) \tag{5}$$

What is the problem here?

Computation of the Likelihood (Ctd.)

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

Why is this an allowed transformation?

$$\begin{aligned}\mathcal{LL}(\boldsymbol{\theta}) &= \log \mathcal{L}(\boldsymbol{\theta}) \\ &= \log \prod_{i=1}^{n} p(x^{(i)}|\boldsymbol{\theta}) \\ &= \sum_{i=1}^{n} \log p(x^{(i)}|\boldsymbol{\theta}) \qquad (6)\end{aligned}$$

$\boxed{\log \prod = \sum \log}$

Maximum Likelihood of a Gaussian

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

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

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

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

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

Maximum Likelihood of a Gaussian (Ctd.)

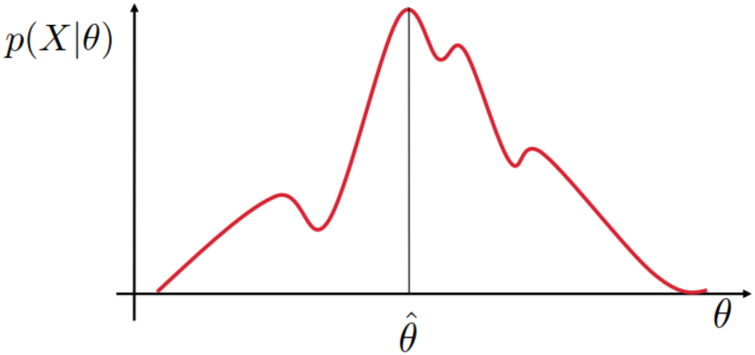
- Compute the partial derivatives with respect to the parameters $\boldsymbol{\theta}$
- Derivative w. r. t. μ:

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

- 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!

Looks familiar?

- Maximum likelihood parameters:

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

- 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

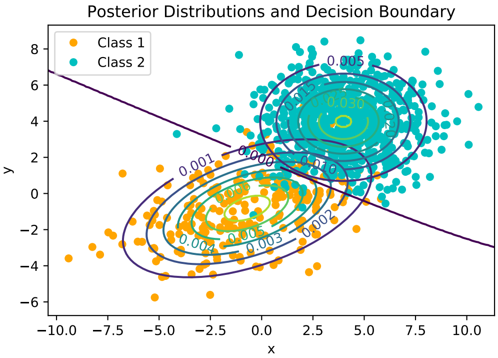
- The solution above is for 1-D data; what if we have more dimensions?
- **Multivariate Gaussian distribution**:

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

- Luckily, the derivations don't change:

$$\boldsymbol{\mu}_{ml} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}^{(i)} \qquad \boldsymbol{\Sigma}_{ml} = \frac{1}{n}\sum_{i=1}^{n} (\boldsymbol{x}^{(i)}-\boldsymbol{\mu}_{ml})(\boldsymbol{x}^{(i)}-\boldsymbol{\mu}_{ml})^\intercal \tag{10}$$

Gaussian naïve Bayes – Final Model



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

NB: $\mathcal{N}_D(\boldsymbol{x}|\boldsymbol{\mu}_{\mathcal{C}_k}, \boldsymbol{\Sigma}_{\mathcal{C}_k})$ 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

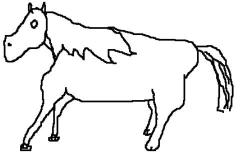
Generative Model

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

Duale Hochschule
Baden-Württemberg

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

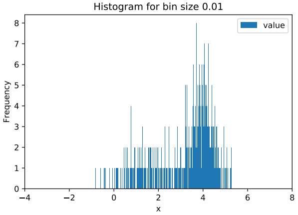
- Histograms partition the data $\boldsymbol{X} = \{\boldsymbol{x}^{(i)}\}_{i=1}^{n}$ into distinct **bins** of volume v_j...
- ...and subsequently count the number of instances k_j falling into the j-th bin
- Approximate the probability $p(\boldsymbol{x})$ by:

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

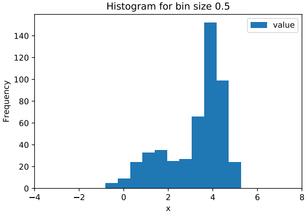
- The sum of all probabilities equals 1: $\int_j \frac{k_j}{n \cdot v_j} = 1$
- v_j is a **hyper-parameter** (usually all bins have equal size)

Histograms (Ctd.)

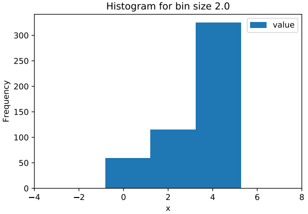
Too narrow



About right

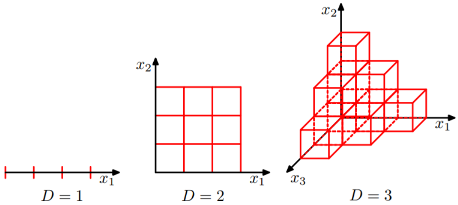


Too wide



Drawbacks of Histograms

- Histograms are mostly unsuited for many applications
- Drawbacks:
  1. **Discontinuities** due to bin edges
  2. 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 $\boldsymbol{x}$

$$p(\boldsymbol{x}) \approx \frac{k}{n \cdot v} \tag{12}$$

- 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 $\boldsymbol{x}$
- Determine if a data point falls into region $\mathcal{R}$:

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

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

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

Kernel Density Estimation: Parzen Window (Ctd.)

- The volume v is simple to compute:

$$v = \int H(\boldsymbol{u})\,\mathrm{d}\boldsymbol{u} = h^D \tag{15}$$

- Putting it all together we get:

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

- **Problem:** There are still discontinuities

Kernel Density Estimation: Gaussian Kernel

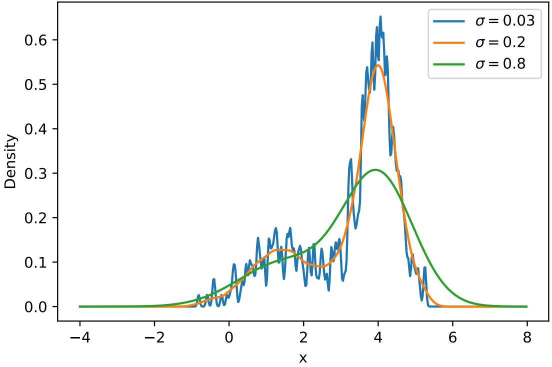
$$H(\boldsymbol{u}) = \frac{1}{\left(\sqrt{2\pi h^2}\right)^D} \exp\left\{-\frac{\|\boldsymbol{u}\|^2}{2h^2}\right\} \tag{17}$$

$$v = \int H(\boldsymbol{u})\,\mathrm{d}\boldsymbol{u} = 1 \tag{18}$$

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

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

Kernel Density Estimation: Gaussian Kernel (Ctd.)



k-Nearest Neighbors

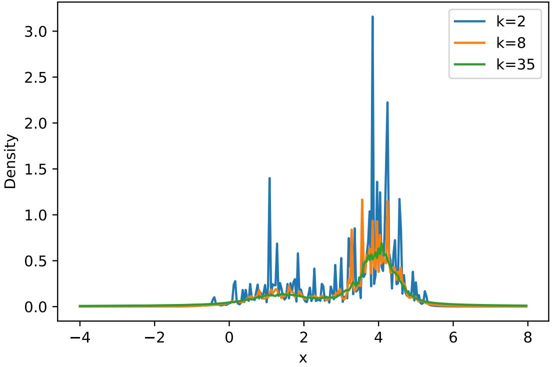
Different strategy:

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

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

- **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

DHBW
Duale Hochschule
Baden-Württemberg

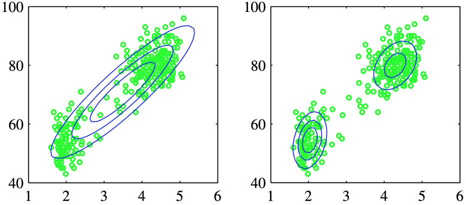
Why do we need Mixture Models?

- 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!**

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

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)

Important

Mixture of Gaussians (MoG)

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

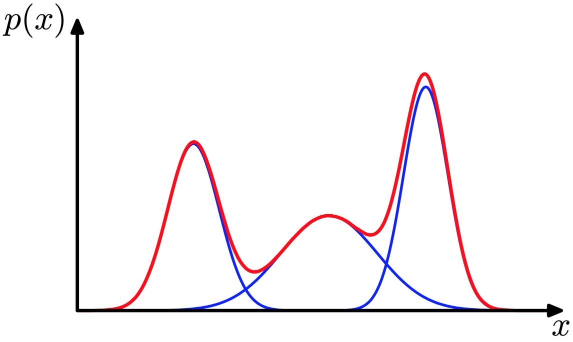
$$p(x|j) = \mathcal{N}(x|\mu_j, \sigma_j) = \frac{1}{\sqrt{2\pi\sigma_j^2}} \exp\left\{-\frac{(x-\mu_j)^2}{2\sigma_j^2}\right\} \tag{24}$$

$$p(j) = \pi_j \qquad \text{with} \qquad 0 \leqslant \pi_j \leqslant 1 \qquad \text{and} \qquad \sum_{j=1}^{M} \pi_j = 1 \tag{25}$$

Remarks:

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

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 $\boldsymbol{\theta}$:

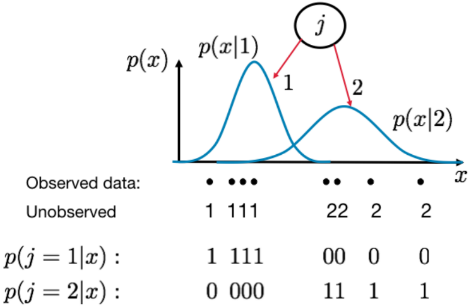
$$\mathcal{LL} = \log \mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log p(x^{(i)}|\boldsymbol{\theta}) \tag{26}$$

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

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

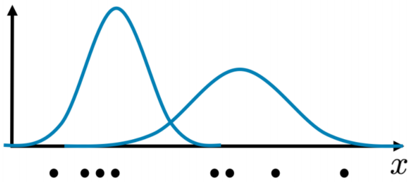
Expectation Maximization (EM)

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

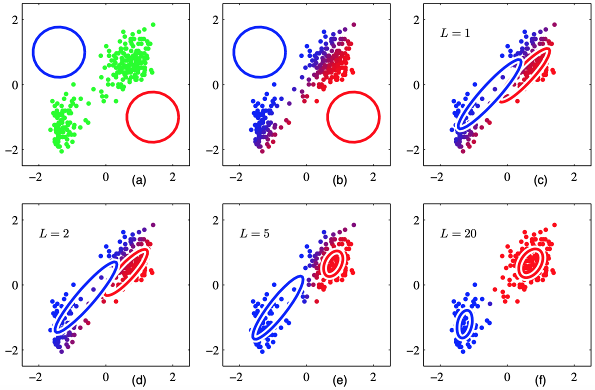


- **We have neither! $\Rightarrow$ Chicken-Egg-Problem!**

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 ❷

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:

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

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

M-Step in Detail

- Update means:

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

- Update variance:

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

- 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):

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

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

Section:

Wrap-Up

DHBW
Duale Hochschule
Baden-Württemberg

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

Important

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 2020/2021

Contact:
Daniel Wehner, M.Sc.
SAP SE / DHBW Mannheim
daniel.wehner@sap.com

Do you have any questions?