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

Clemens Biehl, Daniel Wehner

SAP SE

November 29, 2019

DHBW Duale Hochschule Baden-Württemberg

Find all slides on GitHub

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 November 29, 2019

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
4. Mixture Models
   - General Idea
   - Mixture of Gaussians
   - Expectation Maximization for MoG
5. Wrap-Up
   - Summary
   - Self-Test Questions
   - Lecture Outlook
   - Recommended Literature and further Reading
   - Meme of the Day

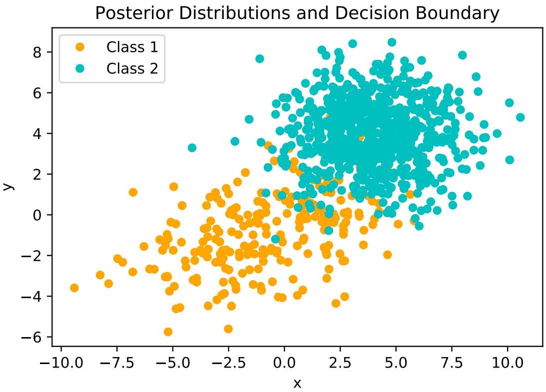
Section:

Introduction

Probability Density Estimation (PDE)

- We have learned about Bayes' optimal classifiers which classify data based on the probability distribution $p(\boldsymbol{x}|\mathcal{C}_k) \cdot p(\mathcal{C}_k)$
- 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

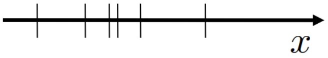
Section:

Parametric Models

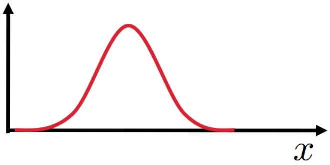
Duale Hochschule Baden-Württemberg

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

$\mu \mathrel{\widehat{=}}$ mean

$\sigma^2 \mathrel{\widehat{=}}$ variance

Learning the Parameters

- Learning = Estimation 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

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

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

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

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

Why is this an allowed transformation?

$\log \prod = \sum \log$

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.)

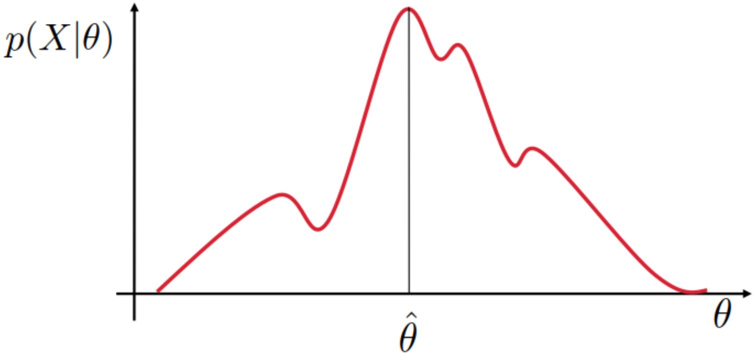
- Compute the partial derivatives with respect to the parameters $\boldsymbol{\theta}$
- Derivative w. r. t. μ:

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

- Set derivative to zero and solve:

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

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

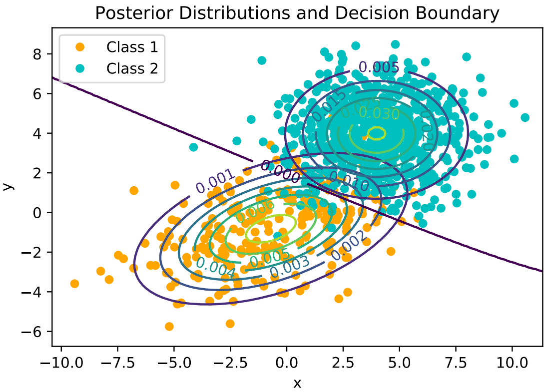
- The solution above is for 1-D data, what if we have more dimensions?
- **Multivariate Gaussian distribution**:

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

- Luckily, the derivations don't change:

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

MLE for the Example Data Set



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

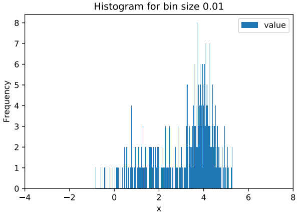
- Histograms partition the data $\boldsymbol{X} = \{\boldsymbol{x}^{(i)}\}_{i=1}^{n}$ into distinct **bins** of volume v_j...
- ...and subsequently count the number of instances k_j falling into the j-th bin
- Approximate the probability $p(\boldsymbol{x})$ by:

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

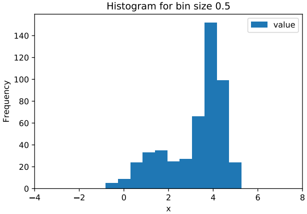
- The sum of all probabilities equals 1: $\sum_{j=1}^{m} \frac{k_j}{n \cdot v_j} = 1$
- v_j is a **hyper-parameter** (usually, all bins have equal size)

Histograms (Ctd.)

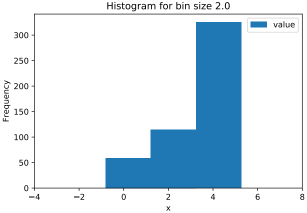
Too narrow



About right

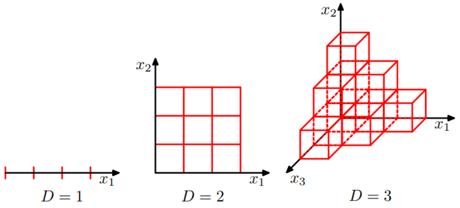


Too wide



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.)

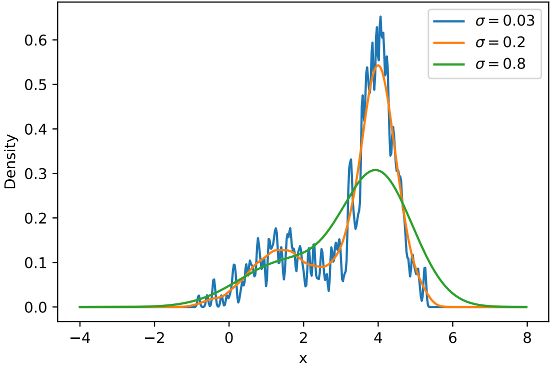
- The volume v is simple to compute:

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

- Putting it all together we get:

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

Kernel Density Estimation: Parzen Window (Ctd.)



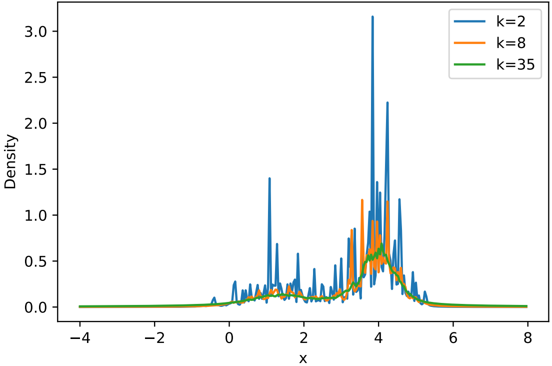
k-Nearest Neighbors

- Increase the size of a sphere until k data points fall into this sphere, keep the number K of data points fixed

$$p(\boldsymbol{x}) \approx \frac{k}{n \cdot v(\boldsymbol{x})} \tag{17}$$

- We will also look at k-Nearest Neighbors as a classification model later $\rightarrow$ you can use a majority vote among the k closest training data points to classify a new data point $\boldsymbol{x}$

k-Nearest Neighbors (Ctd.)



Section:

Mixture Models

Duale Hochschule Baden-Württemberg

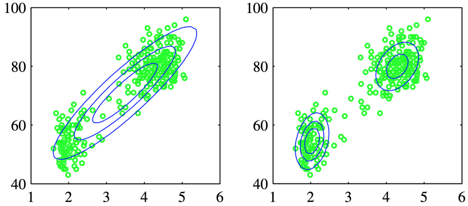
Why do we need mixture models?

- 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!

$$p(\mathbf{x}) = \sum_{j=1}^{M} p(\mathbf{x}|j)p(j) \tag{18}$$

Why do we need mixture models?

- A single parametric model might not be able to capture the structure of the dataset at hand
- Mixture distributions (e. g. a linear combination of Gaussians) can approximate almost any continuous density to arbitrary accuracy (given a sufficient number of Gaussians is used)

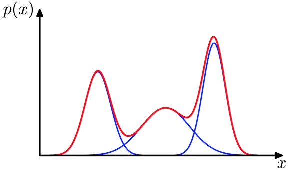


Mixture of Gaussians Definition

- Sum up the M individual Gaussian distributions

$$p(x) = \sum_{j=1}^{M} p(x|z_j) \tag{19}$$

$$p(x|z_j) = \mathcal{N}(x|\mu_j, \sigma_j) = \frac{1}{\sqrt{2\pi}\sigma_j} \exp\left(-\frac{(x-\mu_j)^2}{2\sigma_j^2}\right) \tag{20}$$



Mixture of Gaussians Definition

- Sum up the M individual Gaussian distributions

$$p(x) = \sum_{j=1}^{M} p(x|z_j) \tag{21}$$

$$p(x|z_j) = \mathcal{N}(x|\mu_j, \sigma_j) = \frac{1}{\sqrt{2\pi}\sigma_j} \exp\left(-\frac{(x-\mu_j)^2}{2\sigma_j^2}\right) \tag{22}$$

- z_j for $j \in \{1, \ldots, M\}$ are the *mixing coefficients*
- $z_j \sim \text{Multinomial}(\phi)$, where $\phi_j \geqslant 0$ and $\sum_{j=1}^{M} \phi_j = 1$

Mixture of Gaussians Definition

- Sum up the M individual Gaussian distributions

$$p(x) = \sum_{j=1}^{M} p(x|z_j) \tag{23}$$

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

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

Which function do we need to maximize to estimate the parameters?

Maximum Likelihood Estimation for MoG

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

$$\mathcal{L} = \log L(\theta) = \sum_{i=1}^{N} \log p(x_i|\theta) \tag{25}$$

$$\frac{\partial \mathcal{L}}{\partial \mu_j} = 0 \tag{26}$$

$$\mu_j = \frac{\sum_{i=1}^{N} p(z_j|x_i) x_i}{\sum_{i=1}^{N} p(z_j|x_i)} \qquad \Leftarrow \text{Do you see the issue?} \tag{27}$$

Maximum Likelihood Estimation for MoG

- Maximize the log-likelihood to estimate the parameters $\boldsymbol{\theta}$:

$$\mathcal{L} = \log L(\theta) = \sum_{i=1}^{N} \log p(x_i|\theta) \tag{28}$$

$$\frac{\partial \mathcal{L}}{\partial \mu_j} = 0 \tag{29}$$

$$\mu_j = \frac{\sum_{i=1}^{N} p(z_j|x_i)x_i}{\sum_{i=1}^{N} p(z_j|x_i)} \quad \Leftarrow \text{Do you see the issue?} \tag{30}$$

- There is circular dependency, μ_j depends on z_j $\Rightarrow$ there is no analytical solution!

Maximum Likelihood Estimation for MoG

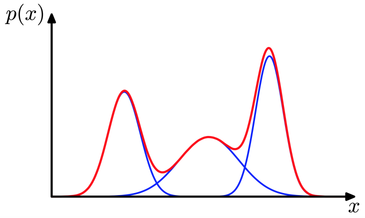
- There is circular dependency, μ_j depends on $z_j \Rightarrow$ there is no analytical solution!
- How about using gradient descent?
  - Complex gradient (circular dependency, non-linearity)
  - Optimization of each Gaussian mixture component depends on all of the other mixture components!
- We have to take a different approach: *Expectation Maximization*

Expectation Maximization

- We have observed data (without labels) x_i and unobserved / hidden / latent variables z_j
- If we knew which Gaussian has generated which data point x_i, then we could do a maximum likelihood estimation for each mixture component j
- If we knew the Gaussians already, i. e. their means and (co)variances, then we could assign each data point x_i to mixture component η if $p(z_\eta = 1 | x_i) > p(z_j = 2 | x_i) \quad \forall j$

Expectation Maximization

- This is a chicken-and-egg problem:
  - We don't know which mixture component generated which x_i
  - We don't know the distributions
- How can we estimate the parameters of the distribution and the mixture coefficients z_j?



Expectation Maximization

- How can we estimate the parameters of the distribution and the mixture coefficients z_j?
- Idea:
  - Assign a *mixture label* to each data point x_i, cluster the data points and assign them to the mixture components
  - The mixture coefficients z_j are *soft assignments* of data points to mixture components
  - Iterate between updating the estimate of the mixture coefficients z_j and the distribution parameters μ_j, σ_j

Expectation Maximization

EM for Gaussian Mixture Models:

1. Initialize μ_j, σ_j, z_j and evaluate the initial log-likelihood
2. **E step:** Compute the posterior distribution (*responsibilities* γ_{ij}) for each mixture coefficient z_j and all data points x_i using the current parameter values

$$\gamma_{ij} = p(z_j|x_i) = \frac{z_j \mathcal{N}(x_i|\mu_j, \sigma_j)}{\sum_{m=1}^{M} z_m \mathcal{N}(x_i|\mu_m, \sigma_m)} \tag{31}$$

Expectation Maximization

EM for Gaussian Mixture Models:

3. **M step:** Compute the new parameters using weighted estimates

$$N_j = \sum_{i=1}^{N} \gamma_{ij} \qquad z_j^{\text{new}} = \frac{N_j}{N} \tag{32}$$

$$\mu_j^{\text{new}} = \frac{1}{N_j} \sum_{i=1}^{N} \gamma_{ij} x_i \qquad (\sigma_j^{\text{new}})^2 = \frac{1}{N_j} \sum_{i=1}^{N} \gamma_{ij} (x_i - \mu_j^{\text{new}})^2 \tag{33}$$

Expectation Maximization

EM for Gaussian Mixture Models:

4. Iterate **E Step** and **M Step** until convergence, i. e. evaluate the log-likelihood after each run and check if the parameters converge:

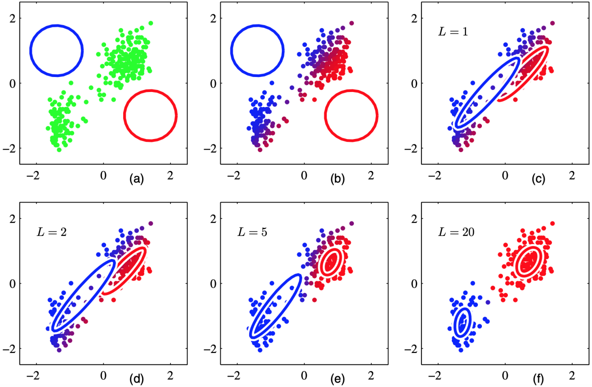
$$\mathcal{L} = \sum_{i=1}^{N} \log p(x_i|\boldsymbol{\theta}) = \sum_{i=1}^{N} \log \left(\sum_{j=1}^{M} z_j \mathcal{N}(x_i|\mu_j, \sigma_j) \right) \tag{34}$$

Expectation Maximization

- We can use EM for performing maximum likelihood estimation even when the data is incomplete (missing features)
- The incomplete log-likelihood $\mathcal{L}$ is maximized locally
- The log-likelihood is guaranteed to improve or stay the same in every EM iteration
- There are great visualizations of EM for Gaussian mixture models:
  - `EM density estimation animation`
  - `2-dimensional EM animation`

Expectation Maximization

Let's look at some iterations of expectation maximization on a simple dataset:



Expectation Maximization

- How do we initialize the parameters for EM?
  - EM depends on a good initialization of the parameters, a poor initialization can lead to getting stuck in bad local optima
  - We can use *k-means* to get an initial clustering
- How many mixture components do we need?
  - There are different criteria, we can for instance maximize the *Bayesian Information Criterion* with respect to K:

$$\log p(X|\theta_{ML}) - \frac{1}{2} K \log N \tag{35}$$

    - 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 for density estimation
- This allows us to estimate the probabilities needed for e. g. a Naïve Bayes model to work with continuous features
- Parametric models assume a certain form of the density, governed by parameters like mean and variance for a Gaussian
- Maximum likelihood estimation allows us to determine the parameters based on our dataset
- Non-parametric models directly use the (training) data points themselves
- We can use expectation maximization to optimize the parameters of mixture models

Self-Test Questions

- What is maximum likelihood estimation? How can you get the maximum likelihood estimate for a Gaussian distribution?
- What does *non-parametric* mean in *non-parametric models*?
- What is different between kernel density estimation and k-nearest neighbors?
- Why can't we use a simple maximum likelihood estimate for a mixture model as with a single Gaussian distribution?
- What happens in the E and M steps in expectation maximization?

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

- **[1] Pattern Recognition and Machine Learning - Chapter 1.2.4 *The Gaussian Distribution***
  Bishop. 2006.
  → Link

- **[1] Pattern Recognition and Machine Learning - Chapter 2.5 *Non-Parametric Methods***
  Bishop. 2006.
  → Link

- **[1] Pattern Recognition and Machine Learning - Chapter 9.2 *Mixtures of Gaussians***
  Bishop. 2006.
  → Link

Meme of the Day

Thank you very much for the attention!

Topic: *** Applied Machine Learning Fundamentals *** Probability Density Estimation (PDE)
Term: November 29, 2019

Contact:
Clemens Biehl
Moodle Forum

Do you have any questions?