Machine Learning – Lecture 4

Probability Density Estimation III

17.10.2019

Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de

leibe@vision.rwth-aachen.de
Course Outline

• Fundamentals
   Bayes Decision Theory
   Probability Density Estimation

• Classification Approaches
   Linear Discriminants
   Support Vector Machines
   Ensemble Methods & Boosting
   Randomized Trees, Forests & Ferns

• Deep Learning
   Foundations
   Convolutional Neural Networks
   Recurrent Neural Networks
Recap: Maximum Likelihood Approach

- Computation of the likelihood
  - Single data point: \( p(x_n|\theta) \)
  - Assumption: all data points \( X = \{x_1, \ldots, x_n\} \) are independent
    \[
    L(\theta) = p(X|\theta) = \prod_{n=1}^{N} p(x_n|\theta)
    \]
  - Log-likelihood
    \[
    E(\theta) = -\ln L(\theta) = -\sum_{n=1}^{N} \ln p(x_n|\theta)
    \]

- Estimation of the parameters \( \theta \) (Learning)
  - Maximize the likelihood (=minimize the negative log-likelihood)
    \( \Rightarrow \) Take the derivative and set it to zero.
    \[
    \frac{\partial}{\partial \theta} E(\theta) = -\sum_{n=1}^{N} \frac{\partial}{\partial \theta} p(x_n|\theta) \cdot \frac{1}{p(x_n|\theta)} \overset{!}{=} 0
    \]
Recap: Histograms

- Basic idea:
  - Partition the data space into distinct bins with widths $\Delta_i$ and count the number of observations, $n_i$, in each bin.
  - Often, the same width is used for all bins, $\Delta_i = \Delta$.
  - This can be done, in principle, for any dimensionality $D$...

...but the required number of bins grows exponentially with $D$!
Recap: Kernel Density Estimation

- Approximation formula:
  \[ p(x) \approx \frac{K}{NV} \]

  - **Fixed \( V \)**: determine \( K \)
  - **Fixed \( K \)**: determine \( V \)

  **Kernel Methods**

  **K-Nearest Neighbor**

- **Kernel methods**
  - Place a *kernel window* \( k \) at location \( x \) and count how many data points fall inside it.

- **K-Nearest Neighbor**
  - Increase the volume \( V \) until the \( K \) nearest data points are found.

Slide adapted from Bernt Schiele

B. Leibe
Topics of This Lecture

• Mixture distributions
  - Mixture of Gaussians (MoG)
  - Maximum Likelihood estimation attempt

• K-Means Clustering
  - Algorithm
  - Applications

• EM Algorithm
  - Credit assignment problem
  - MoG estimation
  - EM Algorithm
  - Interpretation of K-Means
  - Technical advice

• Applications
Mixture Distributions

- A single parametric distribution is often not sufficient
  - E.g. for multimodal data

![Image of single Gaussian vs mixture of two Gaussians](image-source: C.M. Bishop, 2006)
Mixture of Gaussians (MoG)

- Sum of $M$ individual Normal distributions

$$f(x)$$

- In the limit, every smooth distribution can be approximated this way (if $M$ is large enough)

$$p(x | \theta) = \sum_{j=1}^{M} p(x | \theta_j)p(j)$$

Slide credit: Bernt Schiele
**Mixture of Gaussians**

\[
p(x|\theta) = \sum_{j=1}^{M} p(x|\theta_j) p(j)
\]

\[
p(x|\theta_j) = \mathcal{N}(x|\mu_j, \sigma_j^2) = \frac{1}{\sqrt{2\pi}\sigma_j} \exp \left\{ -\frac{(x - \mu_j)^2}{2\sigma_j^2} \right\}
\]

\[
p(j) = \pi_j \text{ with } 0 \cdot \pi_j \cdot 1 \text{ and } \sum_{j=1}^{M} \pi_j = 1
\]

**Notes**

- The mixture density integrates to 1:
  \[
  \int p(x) dx = 1
  \]

- The mixture parameters are
  \[
  \theta = (\pi_1, \mu_1, \sigma_1, \ldots, \pi_M, \mu_M, \sigma_M)
  \]
Mixture of Gaussians (MoG)

- "Generative model"

\[ p(x) = \sum_{j=1}^{M} p(x | \theta_j) p(j) \]

- "Weight" of mixture component

\[ p(j) = \pi_j \]

- Mixture component

- Mixture density

Slide credit: Bernt Schiele
Mixture of Multivariate Gaussians

(a) 

(b) 

(c) 

Image source: C.M. Bishop, 2006
Mixture of Multivariate Gaussians

- Multivariate Gaussians

\[ p(x|\theta) = \sum_{j=1}^{M} p(x|\theta_j)p(j) \]

\[ p(x|\theta_j) = \frac{1}{(2\pi)^{D/2}|\Sigma_j|^{1/2}} \exp \left\{ -\frac{1}{2}(x - \mu_j)^T \Sigma_j^{-1}(x - \mu_j) \right\} \]

- Mixture weights / mixture coefficients:

\[ p(j) = \pi_j \text{ with } 0 \cdot \pi_j \cdot 1 \text{ and } \sum_{j=1}^{M} \pi_j = 1 \]

- Parameters:

\[ \theta = (\pi_1, \mu_1, \Sigma_1, \ldots, \pi_M, \mu_M, \Sigma_M) \]
Mixture of Multivariate Gaussians

- “Generative model”

\[ p(j) = \pi_j \]

\[ p(x|\theta) = \sum_{j=1}^{3} \pi_j p(x|\theta_j) \]
Mixture of Gaussians – 1st Estimation Attempt

- **Maximum Likelihood**
  - Minimize \( E = - \ln L(\theta) = - \sum_{n=1}^{N} \ln p(x_n|\theta) \)
  - Let’s first look at \( \mu_j \):
    \[
    \frac{\partial E}{\partial \mu_j} = 0
    \]
  - We can already see that this will be difficult, since
    \[
    \ln p(X|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n|\mu_k, \Sigma_k) \right\}
    \]
    This will cause problems!
Mixture of Gaussians – 1st Estimation Attempt

- Minimization:
  \[
  \frac{\partial E}{\partial \mu_j} = - \sum_{n=1}^{N} \frac{\partial}{\partial \mu_j} p(x_n | \theta_j) = - \sum_{n=1}^{N} \frac{\Sigma^{-1}(x_n - \mu_j) p(x_n | \theta_j)}{\sum_{k=1}^{K} p(x_n | \theta_k)}
  \]

- We thus obtain
  \[
  \mu_j = \frac{\sum_{n=1}^{N} \gamma_j(x_n)x_n}{\sum_{n=1}^{N} \gamma_j(x_n)}
  \]

  “responsibility” of component \(j\) for \(x_n\)

\[\frac{\partial}{\partial \mu_j} \mathcal{N}(x_n | \mu_k, \Sigma_k) = \Sigma^{-1}(x_n - \mu_j) \mathcal{N}(x_n | \mu_k, \Sigma_k)\]
Mixture of Gaussians – 1st Estimation Attempt

- But...

\[ \mu_j = \frac{\sum_{n=1}^{N} \gamma_j(x_n) x_n}{\sum_{n=1}^{N} \gamma_j(x_n)} \]

\[ \gamma_j(x_n) = \frac{\pi_j \mathcal{N}(x_n | \mu_j, \Sigma_j)}{\sum_{k=1}^{K} \pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k)} \]

- I.e. there is no direct analytical solution!

\[ \frac{\partial E}{\partial \mu_j} = f(\pi_1, \mu_1, \Sigma_1, \ldots, \pi_M, \mu_M, \Sigma_M) \]

- Complex gradient function (non-linear mutual dependencies)
- Optimization of one Gaussian depends on all other Gaussians!
- It is possible to apply iterative numerical optimization here, but in the following, we will see a simpler method.
Mixture of Gaussians – Other Strategy

- Other strategy:

  - Observed data:
  - Unobserved data:
    - Unobserved = “hidden variable”: $j|x$

\[
\begin{align*}
  h(j = 1 | x_n) &= 1 \ 1 \ 1 \ 1 \ 0 \ 0 \ 0 \\
  h(j = 2 | x_n) &= 0 \ 0 \ 0 \ 0 \ 1 \ 1 \ 1
\end{align*}
\]

Slide credit: Bernt Schiele
Mixture of Gaussians – Other Strategy

- Assuming we knew the values of the hidden variable...

\[ f(x) \]

\[ x \]

ML for Gaussian #1

ML for Gaussian #2

\[ h(j = 1 | x_n) = \begin{cases} 1 & 111 \\ 0 & 000 \end{cases} \]

\[ h(j = 2 | x_n) = \begin{cases} 0 & 000 \\ 1 & 111 \end{cases} \]

\[ \mu_1 = \frac{\sum_{n=1}^{N} h(j = 1 | x_n) x_n}{\sum_{i=1}^{N} h(j = 1 | x_n)} \]

\[ \mu_2 = \frac{\sum_{n=1}^{N} h(j = 2 | x_n) x_n}{\sum_{i=1}^{N} h(j = 2 | x_n)} \]

Slide credit: Bernt Schiele
Mixture of Gaussians – Other Strategy

• Assuming we knew the mixture components…

\[ f(x) \]

\[ p(j = 1| x) \]

\[ p(j = 2| x) \]

\[ p(j = 1| x_n) > p(j = 2| x_n) \]

• Bayes decision rule: Decide \( j = 1 \) if

\[ p(j = 1| x_n) > p(j = 2| x_n) \]

assumed known
Clustering with Hard Assignments

- Let's first look at clustering with "hard assignments"
Topics of This Lecture

• Mixture distributions
  ➢ Mixture of Gaussians (MoG)
  ➢ Maximum Likelihood estimation attempt

• K-Means Clustering
  ➢ Algorithm
  ➢ Applications

• EM Algorithm
  ➢ Credit assignment problem
  ➢ MoG estimation
  ➢ EM Algorithm
  ➢ Interpretation of K-Means
  ➢ Technical advice

• Applications
K-Means Clustering

- Iterative procedure
  1. Initialization: pick $K$ arbitrary centroids (cluster means)
  2. Assign each sample to the closest centroid.
  3. Adjust the centroids to be the means of the samples assigned to them.
  4. Go to step 2 (until no change)

- Algorithm is guaranteed to converge after finite #iterations.
  - Local optimum
  - Final result depends on initialization.

Slide credit: Bernt Schiele
K-Means – Example with K=2

Image source: C.M. Bishop, 2006
K-Means Clustering

- K-Means optimizes the following objective function:

\[
J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \left\| x_n - \mu_k \right\|^2
\]

where

\[
r_{nk} = \begin{cases} 
1 & \text{if } k = \arg\min_j \left\| x_n - \mu_j \right\|^2 \\
0 & \text{otherwise.}
\end{cases}
\]

- I.e., \(r_{nk}\) is an indicator variable that checks whether \(\mu_k\) is the nearest cluster center to point \(x_n\).
- In practice, this procedure usually converges quickly to a local optimum.
Example Application: Image Compression

Take each pixel as one data point.

Set the pixel color to the cluster mean.

K-Means Clustering

Image source: C.M. Bishop, 2006
Example Application: Image Compression

$K = 2$  $K = 3$  $K = 10$  Original image

Image source: C.M. Bishop, 2006
Summary K-Means

- **Pros**
  - Simple, fast to compute
  - Converges to local minimum of within-cluster squared error

- **Problem cases**
  - Setting k?
  - Sensitive to initial centers
  - Sensitive to outliers
  - Detects spherical clusters only

- **Extensions**
  - Speed-ups possible through efficient search structures
  - General distance measures: k-medoids

---

Slide credit: Kristen Grauman
Topics of This Lecture

• Mixture distributions
  - Recap: Mixture of Gaussians (MoG)
  - Maximum Likelihood estimation attempt

• K-Means Clustering
  - Algorithm
  - Applications

• EM Algorithm
  - Credit assignment problem
  - MoG estimation
  - EM Algorithm
  - Interpretation of K-Means
  - Technical advice

• Applications
EM Clustering

• Clustering with “soft assignments”
  - Expectation step of the EM algorithm
EM Clustering

- Clustering with “soft assignments”
  - Maximization step of the EM algorithm

\[
\mu_j = \frac{\sum_{n=1}^{N} p(j|x_n) x_n}{\sum_{n=1}^{N} p(j|x_n)}
\]

Maximum Likelihood estimate

\[
p(1|x) = \begin{cases} 0.99 & \text{if } x \in \text{class 1} \\ 0.01 & \text{if } x \in \text{class 2} \end{cases}
\]

\[
p(2|x) = \begin{cases} 0.01 & \text{if } x \in \text{class 1} \\ 0.99 & \text{if } x \in \text{class 2} \end{cases}
\]
Credit Assignment Problem

• “Credit Assignment Problem”
  - If we are just given \( x \), we don’t know which mixture component this example came from
    \[
    p(x|\theta) = \sum_{j=1}^{2} \pi_j p(x|\theta_j)
    \]
  - We can however evaluate the posterior probability that an observed \( x \) was generated from the first mixture component.
    \[
    p(j = 1|x, \theta) = \frac{p(j = 1, x|\theta)}{p(x|\theta)}
    \]
    \[
    p(j = 1, x|\theta) = p(x|j = 1, \theta)p(j = 1) = p(x|\theta_1)p(j = 1)
    \]
    \[
    p(j = 1|x, \theta) = \frac{p(x|\theta_1)p(j = 1)}{\sum_{j=1}^{2} p(x|\theta_j)p(j)} = \gamma_j(x)
    \]
    “responsibility” of component \( j \) for \( x \).
EM Algorithm

- Expectation-Maximization (EM) Algorithm
  - **E-Step**: softly assign samples to mixture components
    \[ \gamma_j(x_n) \leftarrow \frac{\pi_j \mathcal{N}(x_n | \mu_j, \Sigma_j)}{\sum_{k=1}^{K} \pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k)} \quad \forall j = 1, \ldots, K, \ n = 1, \ldots, N \]
  - **M-Step**: re-estimate the parameters (separately for each mixture component) based on the soft assignments
    \[
    \hat{N}_j \leftarrow \sum_{n=1}^{N} \gamma_j(x_n) = \text{soft number of samples labeled } j
    \]
    \[
    \hat{\pi}_j^{\text{new}} \leftarrow \frac{\hat{N}_j}{N}
    \]
    \[
    \hat{\mu}_j^{\text{new}} \leftarrow \frac{1}{\hat{N}_j} \sum_{n=1}^{N} \gamma_j(x_n)x_n
    \]
    \[
    \hat{\Sigma}_j^{\text{new}} \leftarrow \frac{1}{\hat{N}_j} \sum_{n=1}^{N} \gamma_j(x_n)(x_n - \hat{\mu}_j^{\text{new}})(x_n - \hat{\mu}_j^{\text{new}})^T
    \]

Slide adapted from Bernt Schiele
EM Algorithm – An Example

Image source: C.M. Bishop, 2006
EM – Technical Advice

• When implementing EM, we need to take care to avoid singularities in the estimation!
  - Mixture components may collapse on single data points.
  - E.g. consider the case \( \Sigma_k = \sigma_k^2 I \) (this also holds in general)
  - Assume component \( j \) is exactly centered on data point \( x_n \). This data point will then contribute a term in the likelihood function
    \[
    \mathcal{N}(x_n | x_n, \sigma_j^2 I) = \frac{1}{\sqrt{2\pi} \sigma_j}
    \]
    - For \( \sigma_j \to 0 \), this term goes to infinity!

⇒ Need to introduce regularization
  - Enforce minimum width for the Gaussians
  - E.g., instead of \( \Sigma^{-1} \), use \( (\Sigma + \sigma_{\text{min}} I)^{-1} \)
EM – Technical Advice (2)

• EM is very sensitive to the initialization
  ➢ Will converge to a local optimum of $E$.
  ➢ Convergence is relatively slow.

⇒ Initialize with k-Means to get better results!
  ➢ k-Means is itself initialized randomly, will also only find a local optimum.
  ➢ But convergence is much faster.

• Typical procedure
  ➢ Run k-Means $M$ times (e.g. $M = 10–100$).
  ➢ Pick the best result (lowest error $J$).
  ➢ Use this result to initialize EM
    – Set $\mu_j$ to the corresponding cluster mean from k-Means.
    – Initialize $\Sigma_j$ to the sample covariance of the associated data points.
K-Means Clustering Revisited

- Interpreting the procedure
  1. Initialization: pick $K$ arbitrary centroids (cluster means)
  2. Assign each sample to the closest centroid. (E-Step)
  3. Adjust the centroids to be the means of the samples assigned to them. (M-Step)
  4. Go to step 2 (until no change)
K-Means Clustering Revisited

- K-Means clustering essentially corresponds to a Gaussian Mixture Model (MoG or GMM) estimation with EM whenever
  - The covariances are of the $K$ Gaussians are set to $\Sigma_j = \sigma^2 I$
  - For some small, fixed $\sigma^2$
Summary: Gaussian Mixture Models

• Properties
  - Very general, can represent any (continuous) distribution.
  - Once trained, very fast to evaluate.
  - Can be updated online.

• Problems / Caveats
  - Some numerical issues in the implementation
    ⇒ Need to apply regularization in order to avoid singularities.
  - EM for MoG is computationally expensive
    – Especially for high-dimensional problems!
    – More computational overhead and slower convergence than k-Means
    – Results very sensitive to initialization
    ⇒ Run k-Means for some iterations as initialization!
  - Need to select the number of mixture components K.
    ⇒ Model selection problem (see later lecture)
Topics of This Lecture

• Mixture distributions
  - Recap: Mixture of Gaussians (MoG)
  - Maximum Likelihood estimation attempt

• K-Means Clustering
  - Algorithm
  - Applications

• EM Algorithm
  - Credit assignment problem
  - MoG estimation
  - EM Algorithm
  - Interpretation of K-Means
  - Technical advice

• Applications
Applications

• Mixture models are used in many practical applications.
  ➢ Wherever distributions with complex or unknown shapes need to be represented…

• Popular application in Computer Vision
  ➢ Model distributions of pixel colors.
  ➢ Each pixel is one data point in, e.g., RGB space.
  ⇒ Learn a MoG to represent the class-conditional densities.
  ⇒ Use the learned models to classify other pixels.

Image source: C.M. Bishop, 2006
Application: Background Model for Tracking

- Train background MoG for each pixel
  - Model “common“ appearance variation for each background pixel.
  - Initialization with an empty scene.
  - Update the mixtures over time
    - Adapt to lighting changes, etc.

- Used in many vision-based tracking applications
  - Anything that cannot be explained by the background model is labeled as foreground (=object).
  - Easy segmentation if camera is fixed.


Image Source: Daniel Roth, Tobias Jäggli
Application: Image Segmentation

- User assisted image segmentation
  - User marks two regions for foreground and background.
  - Learn a MoG model for the color values in each region.
  - Use those models to classify all other pixels.
  - Simple segmentation procedure
    (building block for more complex applications)
References and Further Reading

• More information about EM and MoG estimation is available in Chapter 2.3.9 and the entire Chapter 9 of Bishop’s book (recommendable to read).

  Christopher M. Bishop
  Pattern Recognition and Machine Learning
  Springer, 2006

• Additional information

  ➢ Original EM paper:

  ➢ EM tutorial: