

Machine Learning- Exercise 1

Python Tutorial, Probability Density, GMM, EM

Stephanie Kaes & George Lydakis
`<lastname>@vision.rwth-aachen.de`

RWTH Aachen University - Computer Vision Group
<http://www.vision.rwth-aachen.de/>

2023-11-06

Content

1. Warming Up!
2. Minimizing the Expected Loss
3. Maximum Likelihood
4. Kernel/ k -Nearest Neighborhood Density Estimators
5. Expectation Maximization (EM) Algorithm

Content

1. Warming Up!
2. Minimizing the Expected Loss
3. Maximum Likelihood
4. Kernel/ k -Nearest Neighborhood Density Estimators
5. Expectation Maximization (EM) Algorithm

Repetition: Important Statistical Formulas

- ▶ Rules of probability

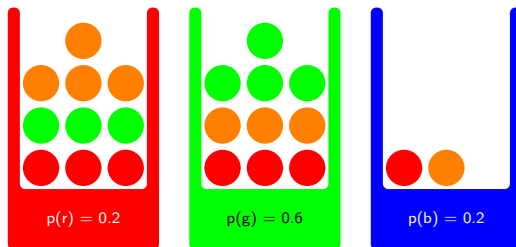
$$p(x) = \sum_y p(x, y)$$

$$p(x, y) = p(y|x)p(x)$$

- ▶ Bayes rule

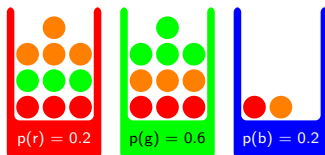
$$\begin{aligned} p(y|x) &= \frac{p(x|y)p(y)}{p(x)} \\ &= \frac{p(x|y)p(y)}{\sum_{y'} p(x|y')p(y')} \end{aligned}$$

Statement of problem



- ▶ Let $B \in \{r, g, b\}$ the random variable for the boxes.
- ▶ Let $F = \{a, o, l\}$ the random variable representing the fruits.

Selecting an apple



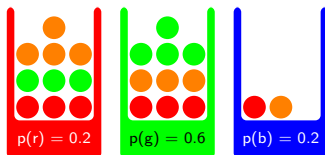
What is $p(F = a)$?

$$p(F = a) = \sum_{b \in B} p(F = a | B = b) p(B = b)$$

Law of total probability

$$\begin{aligned}
 &= p(F = a | B = r) p(B = r) + p(F = a | B = g) p(B = g) \\
 &\quad + p(F = a | B = b) p(B = b) \\
 &= \frac{3}{10} \cdot \frac{2}{10} + \frac{3}{10} \cdot \frac{6}{10} + \frac{1}{2} \cdot \frac{2}{10} \\
 &= \frac{17}{50} = 0.34
 \end{aligned}$$

Conditional probability for orange



What is $p(B = g|F = o)$?

$$\begin{aligned}
 p(B = g|F = o) &= \underbrace{\frac{p(F = o|B = g)p(B = g)}{p(F = o)}}_{\text{Bayes Rule}} \\
 &= \frac{p(F = o|B = g)p(B = g)}{\sum_{b \in B} p(B = b)p(F = o|B = b)} \\
 &= \frac{\frac{3}{10} \frac{6}{10}}{\frac{4}{10} \frac{2}{10} + \frac{3}{10} \frac{6}{10} + \frac{1}{2} \frac{2}{10}} \\
 &= \frac{1}{2}
 \end{aligned}$$

Content

1. Warming Up!
2. Minimizing the Expected Loss
3. Maximum Likelihood
4. Kernel/ k -Nearest Neighborhood Density Estimators
5. Expectation Maximization (EM) Algorithm

Minimizing the Expected Loss

- ▶ Optimal solution: Minimizing the loss by adapting decision regions
- ▶ Loss depends on true class
- ⇒ Minimize the **expected** loss

$$\begin{aligned}
 \mathbb{E}[L] &= \sum_k \sum_j \int_{\mathcal{R}_j} L_{kj} p(x, C_k) dx \\
 &= \sum_j \int_{\mathcal{R}_j} \sum_k L_{kj} p(x, C_k) dx \\
 &= \sum_j \int_{\mathcal{R}_j} \underbrace{\sum_k L_{kj} p(C_k|x) p(x)}_{\rightarrow \min} dx \rightarrow \min
 \end{aligned}$$

Minimizing Expected Loss with Rejection Choice

$$\text{classify}(x) \rightarrow \begin{cases} C_j & \text{if } \forall i : p(C_i|x) \leq p(C_j|x) \wedge p(C_j|x) \geq 1 - \frac{l_r}{l_s} \\ C_{\text{rej}} & \text{otherwise} \end{cases}$$

$$\begin{aligned} \operatorname{argmin}_j \left\{ \sum_k L_{kj} p(C_k|x) p(x) \right\} &= \operatorname{argmin}_j \left\{ \sum_k L_{kj} p(C_k|x) \right\} \\ &= \operatorname{argmin}_j \left\{ l_s \sum_{k \neq j} p(C_k|x) \right\} \\ &= \operatorname{argmin}_j \left\{ \sum_{k \neq j} p(C_k|x) \right\} \\ &= \operatorname{argmin}_j \left\{ (1 - p(C_j|x)) \right\} \\ &= \operatorname{argmax}_j \left\{ p(C_j|x) \right\} \end{aligned}$$

Minimizing Expected Loss with Rejection Choice

$$\text{classify}(x) \rightarrow \begin{cases} C_j & \text{if } \forall i : p(C_i|x) \leq p(C_j|x) \wedge p(C_j|x) \geq 1 - \frac{l_r}{l_s} \\ C_{\text{rej}} & \text{otherwise} \end{cases}$$

$$l_s \sum_{k \neq j} p(C_k|x) \leq l_r$$

$$\Leftrightarrow l_s (1 - p(C_j|x)) \leq l_r$$

$$\Leftrightarrow 1 - p(C_j|x) \leq \frac{l_r}{l_s}$$

$$\Leftrightarrow -p(C_j|x) \leq \frac{l_r}{l_s} - 1$$

$$\Leftrightarrow p(C_j|x) \geq 1 - \frac{l_r}{l_s}$$

Minimizing the Expected Loss

- (b) Unless the $p(C_i|x) = 1$ always reject.
- (c) Never reject.

Content

1. Warming Up!
2. Minimizing the Expected Loss
3. Maximum Likelihood
4. Kernel/ k -Nearest Neighborhood Density Estimators
5. Expectation Maximization (EM) Algorithm

Maximum Likelihood

- ▶ Single data point

$$p(x_n|\theta)$$

- ▶ IID assumption: Likelihood

$$L(\theta) = p(X|\theta) = \prod_{n=1}^N p(x_n|\theta)$$

- ▶ Log-likelihood

$$E(\theta) = \ln L(\theta) = \ln \left(\prod_{n=1}^N p(x_n|\theta) \right) = \sum_{n=1}^N \ln p(x_n|\theta)$$

- ▶ Maximize Log-likelihood

$$\frac{\partial}{\partial \theta} E(\theta) = \frac{\partial}{\partial \theta} \sum_{n=1}^N \ln p(x_n|\theta) = \sum_{n=1}^N \frac{\frac{\partial}{\partial \theta} p(x_n|\theta)}{p(x_n|\theta)} \stackrel{!}{=} 0$$

Maximum Likelihood

$$p(x|\theta) = \theta^2 x \exp(-\theta x) g(x) = \theta^2 x \exp(-\theta x) \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$\begin{aligned} \frac{\partial}{\partial \theta} E(\theta) &= \sum_{n=1}^N \frac{\frac{\partial}{\partial \theta} p(x_n|\theta)}{p(x_n|\theta)} \\ &= \sum_{n=1}^N \frac{x_n g(x_n) \cdot [2\theta \exp(-\theta x_n) - \theta^2 x_n \exp(-\theta x_n)]}{\theta^2 x_n \exp(-\theta x_n) g(x_n)} \\ &= \sum_{n=1}^N \left(\frac{2}{\theta} - x_n \right) \\ &= - \sum_{n=1}^N x_n + \frac{2N}{\theta} \stackrel{!}{=} 0 \\ \Rightarrow \hat{\theta} &= \frac{2N}{\sum_{n=1}^N x_n} \end{aligned}$$

Content

1. Warming Up!
2. Minimizing the Expected Loss
3. Maximum Likelihood
4. Kernel/ k -Nearest Neighborhood Density Estimators
5. Expectation Maximization (EM) Algorithm

Kernel Density Estimation with Gaussian Kernel

- Kernel function

$$k(u) = \frac{1}{(2\pi)^{D/2}h} \exp\left(-\frac{u}{2h^2}\right)$$

$$K = \sum_{n=1}^N k(\|x - x_n\|) \quad V = \int_{-\infty}^{\infty} k(u) du = 1$$

- Probability density estimate

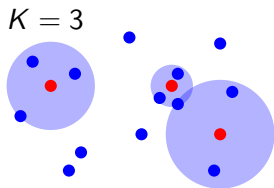
$$p(x) = \frac{K}{NV} = \frac{1}{N} \sum_{n=1}^N \frac{1}{(2\pi)^{D/2}h} \exp\left(-\frac{\|x - x_n\|}{2h^2}\right)$$

K-Nearest Neighbor Density Estimation

- ▶ Fix K , estimate V from data
- ▶ Consider a hypersphere centered at x and let it grow to volume V^* that includes K of the given N data points
- ▶ Then

$$p(x) = \frac{K}{NV^*}$$

- ▶ Strictly speaking, the model produced by K -NN is not a true density model, because the integral over all spaces may diverge
- ▶ E.g. consider $K = 1$ and a sample exactly on a data point $x = x_j$



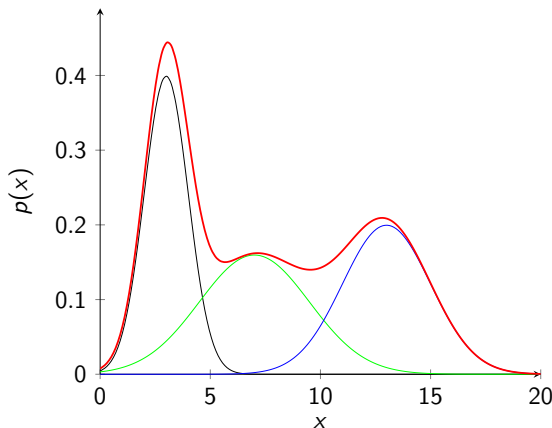
Content

1. Warming Up!
2. Minimizing the Expected Loss
3. Maximum Likelihood
4. Kernel/ k -Nearest Neighborhood Density Estimators
5. Expectation Maximization (EM) Algorithm

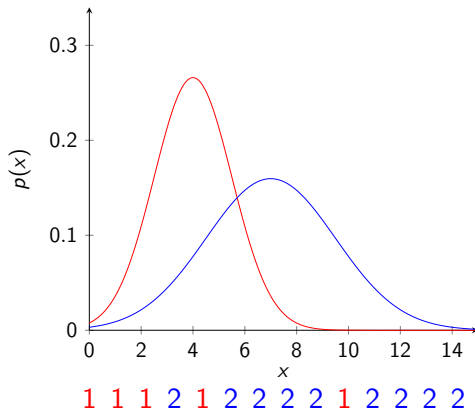
Expectation Maximization Algorithm for GMM

► Generative model

$$p(x|\theta) = \sum_{k=1}^K p(k)p(x|\theta_k, k)$$



Maximum Likelihood - known assignments

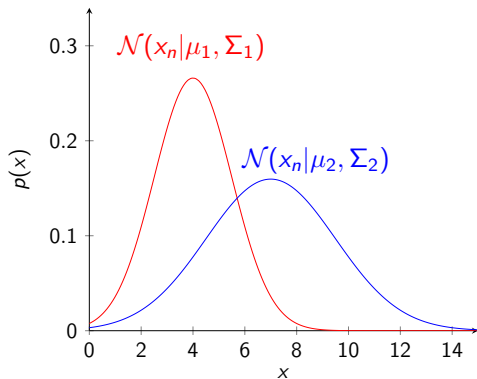


If assignment $h(k|x)$ is known, we can estimate parameters:

$$\mu_1 = \frac{\sum_{n=1}^N h(\textcolor{red}{k} = 1|x_n)x_n}{\sum_{n=1}^N h(\textcolor{red}{k} = 1|x_n)}$$

$$\mu_2 = \frac{\sum_{n=1}^N h(\textcolor{blue}{k} = 2|x_n)x_n}{\sum_{n=1}^N h(\textcolor{blue}{k} = 2|x_n)}$$

Maximum Likelihood - known mixtures



If parameters μ_k, Σ_k are known, we can estimate $\mathcal{N}(x_n | \mu_k, \Sigma_k)$ for each mixture component and sample x_n .

Iterative Optimization: Expectation Maximization (EM)

- E-Step: **softly** assign samples to mixture components

$$\gamma_k(x_n) = \frac{\pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k)}{\sum_{k'=1}^K \pi_{k'} \mathcal{N}(x_n | \mu_{k'}, \Sigma_{k'})}$$

- M-Step: re-estimate mixture parameters based on soft assignment

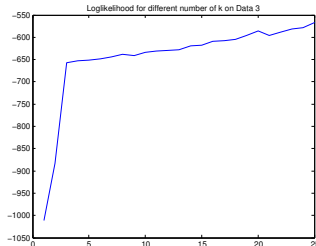
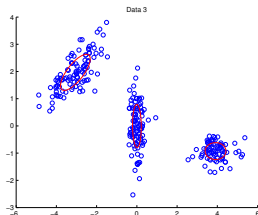
$$\hat{N}_k = \sum_{n=1}^N \gamma_k(x_n) \quad \hat{\pi}_k = \frac{\hat{N}_k}{N}$$

$$\hat{\mu}_k = \frac{1}{\hat{N}_k} \sum_{n=1}^N \gamma_k(x_n) x_n$$

$$\hat{\Sigma}_k = \frac{1}{\hat{N}_k} \sum_{n=1}^N \gamma_k(x_n) (x_n - \hat{\mu}_k)(x_n - \hat{\mu}_k)^T$$

Optimal number of clusters k

- What is the optimal number of clusters?



- You can **NOT** use the log-likelihood!
 - Potentially each point forms its own cluster: $\mathcal{L}(\theta) = 0$
- There is no “good” evaluation of clustering.