Gaussian Mixture Models

David S. Rosenberg

New York University

April 24, 2018

Contents

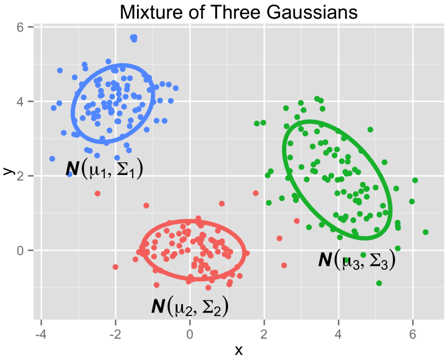
Gaussian Mixture Models

Probabilistic Model for Clustering

- Let's consider the following **generative model** (i.e. a way to generate data).
- Suppose
  1. There are k clusters (or "**mixture components**").
  2. We have a probability density for each cluster.
- Generate a point as follows
  1. Choose a random cluster $z \in \{1, 2, \ldots, k\}$.
  2. Choose a point from the distribution for cluster z.
- Data generated in this way is said to have a **mixture distribution.**

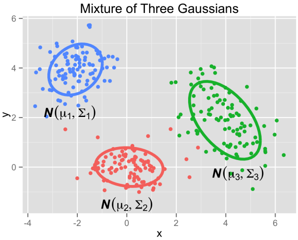
Gaussian Mixture Model ($k = 3$)

1. Choose $z \in \{1, 2, 3\}$ with $p(1) = p(2) = p(3) = \frac{1}{3}$.
2. Choose $x \mid z \sim \mathcal{N}(X \mid \mu_z, \Sigma_z)$.



Gaussian Mixture Model Parameters (k Components)

$$\begin{aligned} \text{Cluster probabilities :} \quad & \pi = (\pi_1, \ldots, \pi_k) \\ \text{Cluster means :} \quad & \mu = (\mu_1, \ldots, \mu_k) \\ \text{Cluster covariance matrices:} \quad & \Sigma = (\Sigma_1, \ldots \Sigma_k) \end{aligned}$$



For now, **suppose all these parameters are known**.
We'll discuss how to **learn** or **estimate** them later.

Gaussian Mixture Model: Joint Distribution

- Factorize the joint density:

$$
\begin{aligned}
p(x,z) &= p(z)p(x \mid z) \\
&= \pi_z \mathcal{N}(x \mid \mu_z, \Sigma_z)
\end{aligned}
$$

- π_z is probability of choosing cluster z.
- $x \mid z$ has distribution $\mathcal{N}(\mu_z, \Sigma_z)$.
- z corresponding to x is the true cluster assignment.
- Suppose we know the model parameters π_z, μ_z, Σ_z.
- Then we can easily evaluate the joint density $p(x,z)$.

Latent Variable Model

- We observe x.
- We don't observe z (the cluster assignment).
- Cluster assignment z is called a **hidden variable** or **latent variable**.

Definition

A **latent variable model** is a probability model for which certain variables are never observed.

e.g. The Gaussian mixture model is a latent variable model.

The GMM "Inference" Problem

- We observe x. We want to know its cluster assignment z.
- The conditional probability for cluster z given x is

$$p(z \mid x) = p(x,z)/p(x)$$

- The conditional distribution is a **soft assignment** to clusters.
- A **hard assignment** is

$$z^* = \underset{z \in \{1,\ldots,k\}}{\arg\min}\; p(z \mid x).$$

- So if we know the model parameters, clustering is trival.

Mixture Models

General Mixture Models: Generative Construction

- Let S be a set of k probability distributions ("**mixture components**").
- Let $\pi = (\pi_1, \ldots, \pi_k)$ be a distribution on $\{1, \ldots, k\}$ ("**mixture weights**")
- Suppose we generate x with the following procedure:
  1. Choose a distribution randomly from S according to π.
  2. Sample x from the chosen distribution.
- Then we say x has a **mixture distribution**.

Mixture Densities

- Suppose we have a mixture distribution with
  - **mixture components** represented as densities $p_1, \ldots, p_k$, and
  - **mixture weights** $\pi = (\pi_1, \ldots, \pi_k)$, then
- the corresponding probability density for x is

$$p(x) = \sum_{i=1}^{k} \pi_i p_i(x).$$

- Note that p is a convex combination of the mixture component densities.
- $p(x)$ is called a **mixture density**.
- Conversely, if x has a density of this form, then x has a mixture distribution.

Gaussian Mixture Model (GMM): Marginal Distribution

For example:

- The **marginal distribution** for a single observation x in a GMM is

$$
\begin{aligned}
p(x) &= \sum_{z=1}^{k} p(x,z) \\
&= \sum_{z=1}^{k} \pi_z \mathcal{N}(x \mid \mu_z, \Sigma_z)
\end{aligned}
$$

Learning in Gaussian Mixture Models

The GMM "Learning" Problem

- Given data $x_1, \ldots, x_n$ drawn from a GMM,
- Estimate the parameters:

$$\begin{aligned}
\text{Cluster probabilities}: &\quad \pi = (\pi_1, \ldots, \pi_k) \\
\text{Cluster means}: &\quad \mu = (\mu_1, \ldots, \mu_k) \\
\text{Cluster covariance matrices}: &\quad \Sigma = (\Sigma_1, \ldots \Sigma_k)
\end{aligned}$$

- Once we have the parameters, we're done.
- Just do "inference" to get cluster assignments.

Estimating/Learning the Gaussian Mixture Model

- One approach to learning is **maximum likelihood**
  - find parameter values with **highest likelihood** for the **observed data**.
- The model likelihood for $\mathcal{D} = (x_1, \ldots, x_n)$ sampled iid from a GMM is

$$
\begin{aligned}
L(\pi, \mu, \Sigma) &= \prod_{i=1}^{n} p(x_i) \\
&= \prod_{i=1}^{n} \sum_{z=1}^{k} \pi_z \mathcal{N}(x_i \mid \mu_z, \Sigma_z).
\end{aligned}
$$

- As usual, we'll take our objective function to be the log of this:

$$
J(\pi, \mu, \Sigma) = \sum_{i=1}^{n} \log \left\{ \sum_{z=1}^{k} \pi_z \mathcal{N}(x_i \mid \mu_z, \Sigma_z) \right\}
$$

Review: Estimating a Gaussian Distribution

- Recall that the density for $x \sim \mathcal{N}(\mu, \Sigma)$ is

$$p(x \mid \mu, \Sigma) = \frac{1}{\sqrt{|2\pi\Sigma|}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right)$$

- And the log-density is

$$\log p(x \mid \mu, \Sigma) = -\frac{1}{2}\log|2\pi\Sigma| - \frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)$$

- To estimate μ and Σ from a sample $x_1, \ldots, x_n$ i.i.d. $\mathcal{N}(\mu, \Sigma)$, we'll maximize the log joint density:

$$\sum_{i=1}^{n} \log p(x_i \mid \mu, \Sigma) = -\frac{n}{2}\log|2\pi\Sigma| - \frac{1}{2}\sum_{i=1}^{n}(x_i-\mu)^T \Sigma^{-1}(x_i-\mu)$$

Review: Estimating a Gaussian Distribution

- To estimate μ and Σ from a sample $x_1,\dots,x_n$ i.i.d. $\mathcal{N}(\mu,\Sigma)$, we'll maximize the log joint density:

$$J(\mu,\Sigma)=\sum_{i=1}^{n}\log p(x\mid\mu,\Sigma)=-\frac{n}{2}\log|2\pi\Sigma|-\frac{1}{2}\sum_{i=1}^{n}(x_i-\mu)^T\Sigma^{-1}(x_i-\mu)$$

- This is a solid exercise in vector and matrix differentiation. Find $\hat{\mu}$ and $\hat{\Sigma}$ satisfying

$$\nabla_\mu J(\mu,\Sigma)=0 \qquad \nabla_\Sigma J(\mu,\Sigma)=0$$

- We get a closed form solution:

$$\begin{aligned}\hat{\mu}_{\text{MLE}} &= \frac{1}{n}\sum_{i=1}^{n}x_i \\ \hat{\Sigma}_{\text{MLE}} &= \frac{1}{n}\sum_{i=1}^{n}(x_i-\hat{\mu}_{\text{MLE}})^T(x_i-\hat{\mu}_{\text{MLE}})\end{aligned}$$

Properties of the GMM Log-Likelihood

- GMM log-likelihood:

$$J(\pi,\mu,\Sigma) \quad = \quad \sum_{i=1}^{n}\log\left\{\sum_{z=1}^{k}\frac{\pi_z}{\sqrt{|2\pi\Sigma_z|}}\exp\left(-\frac{1}{2}(x-\mu_z)^T\Sigma_z^{-1}(x-\mu_z)\right)\right\}$$

- Let's compare to the log-likelihood for a single Gaussian:

$$J(\mu,\Sigma) = -\frac{n}{2}\log|2\pi\Sigma| - \frac{1}{2}\sum_{i=1}^{n}(x_i-\mu)^T\Sigma^{-1}(x_i-\mu)$$

- For a single Gaussian, the log cancels the exp in the Gaussian density.
  - $\implies$ Things simplify a lot.
- For the GMM, the sum inside the log prevents this cancellation.
  - $\implies$ Expression more complicated. No closed form expression for MLE.

Issues with MLE for GMM

Identifiability Issues for GMM

- Suppose we have found parameters

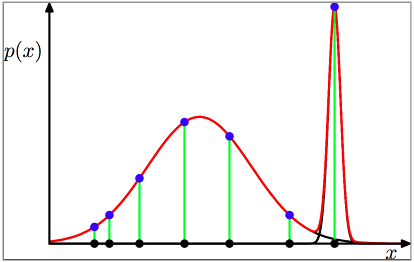
$$\begin{aligned} \text{Cluster probabilities}: \quad & \pi = (\pi_1, \ldots, \pi_k) \\ \text{Cluster means}: \quad & \mu = (\mu_1, \ldots, \mu_k) \\ \text{Cluster covariance matrices}: \quad & \Sigma = (\Sigma_1, \ldots \Sigma_k) \end{aligned}$$

  that are at a local minimum.

- What happens if we shuffle the clusters? e.g. Switch the labels for clusters 1 and 2.
- We'll get the same likelihood. How many such equivalent settings are there?
- Assuming all clusters are distinct, there are $k!$ equivalent solutions.
- Not a problem *per se*, but something to be aware of.

Singularities for GMM

- Consider the following GMM for 7 data points:



- Let σ^2 be the variance of the skinny component.
- What happens to the likelihood as $\sigma^2 \to 0$?
- In practice, we end up in local minima that do not have this problem.
  - Or keep restarting optimization until we do.
- Bayesian approach or regularization will also solve the problem.

From Bishop's *Pattern recognition and machine learning*, Figure 9.7.

Gradient Descent / SGD for GMM

- What about running gradient descent or SGD on

$$J(\pi, \mu, \Sigma) = -\sum_{i=1}^{n} \log\left\{\sum_{z=1}^{k} \pi_z \mathcal{N}(x_i \mid \mu_z, \Sigma_z)\right\}?$$

- Can be done, in principle – but need to be clever about it.
- Each matrix $\Sigma_1, \ldots, \Sigma_k$ has to be positive semidefinite.
- How to maintain that constraint?
    - Rewrite $\Sigma_i = M_i M_i^T$, where M_i is an unconstrained matrix.
    - Then Σ_i is positive semidefinite.
- Even then, pure gradient-based methods have trouble.[1]

[1] See Hosseini and Sra's Manifold Optimization for Gaussian Mixture Models for discussion and further references.

The EM Algorithm for GMM

MLE for Gaussian Model

- Let's start by considering the MLE for the Gaussian model.
- For data $\mathcal{D} = \{x_1, \ldots, x_n\}$, the log likelihood is given by

$$\sum_{i=1}^{n} \log \mathcal{N}(x_i \mid \mu, \Sigma) = -\frac{nd}{2}\log(2\pi) - \frac{n}{2}\log|\Sigma| - \frac{1}{2}\sum_{i=1}^{n}(x_i - \mu)'\Sigma^{-1}(x_i - \mu).$$

- With some calculus, we find that the MLE parameters are

$$\begin{aligned} \mu_{\text{MLE}} &= \frac{1}{n}\sum_{i=1}^{n} x_i \\ \Sigma_{\text{MLE}} &= \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu_{\text{MLE}})(x_i - \mu_{\text{MLE}})^T \end{aligned}$$

- For GMM, If we knew the cluster assignment z_i for each x_i,
  - we could compute the MLEs for each cluster.

Estimating a Fully-Observed GMM

- Suppose we observe $(x_1, z_1), \ldots, (x_n, z_n)$ i.i.d. from GMM $p(x, z)$.
- Them find MLE is easy:

$$
\begin{aligned}
n_z &= \sum_{i=1}^{n} 1(z_i = z) \\
\hat{\pi}(z) &= \frac{n_z}{n} \\
\hat{\mu}_z &= \frac{1}{n_z} \sum_{i:z_i = z} x_i \\
\hat{\Sigma}_z &= \frac{1}{n_z} \sum_{i:z_i = z} (x_i - \hat{\mu}_z)(x_i - \hat{\mu}_z)^T.
\end{aligned}
$$

- In the EM algorithm we will modify the equations to handle our evolving **soft assignments**, which we will call **responsibilities**.

Cluster Responsibilities: Some New Notation

- Denote the probability that observed value x_i comes from cluster j by

$$\gamma_i^j = p\left(z = j \mid x = x_i\right).$$

- The **responsibility** that cluster j takes for observation x_i.
- Computationally,

$$\begin{aligned}
\gamma_i^j &= p\left(z = j \mid x_i\right). \\
&= p\left(z = j, x_i\right)/p(x_i) \\
&= \frac{\pi_j \mathcal{N}\left(x_i \mid \mu_j, \Sigma_j\right)}{\sum_{c=1}^k \pi_c \mathcal{N}\left(x_i \mid \mu_c, \Sigma_c\right)}
\end{aligned}$$

- The vector $\left(\gamma_i^1, \ldots, \gamma_i^k\right)$ is exactly the **soft assignment** for x_i.
- Let $n_c = \sum_{i=1}^n \gamma_i^c$ be the "number" of points "soft assigned" to cluster c.

EM Algorithm for GMM: Overview

- If we know μ_j, Σ_j, π_j for all clusters j, then easy to find

$$\gamma_i^j = p(z = j \mid x_i)$$

- If we know the (soft) assignments, we can easily find estimates for π, Σ, μ.
- Repeatedly alternate these two steps.

EM Algorithm for GMM: Overview

1. Initialize parameters μ, Σ, π (e.g. using k-means).
2. "E step". Evaluate the responsibilities using current parameters:

$$\gamma_i^j = \frac{\pi_j \mathcal{N}(x_i \mid \mu_j, \Sigma_j)}{\sum_{c=1}^k \pi_c \mathcal{N}(x_i \mid \mu_c, \Sigma_c)},$$

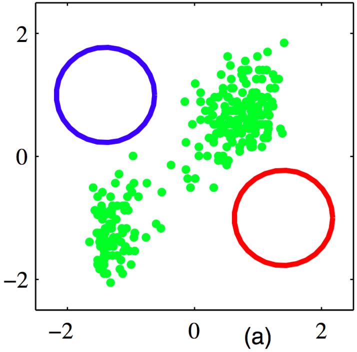
   for $i = 1, \ldots, n$ and $j = 1, \ldots, k$.
3. "M step". Re-estimate the parameters using responsibilities:

$$\begin{aligned}
\mu_c^{\text{new}} &= \frac{1}{n_c} \sum_{i=1}^n \gamma_i^c x_i \\
\Sigma_c^{\text{new}} &= \frac{1}{n_c} \sum_{i=1}^n \gamma_i^c (x_i - \mu_c^{\text{new}})(x_i - \mu_c^{\text{new}})^T \\
\pi_c^{\text{new}} &= \frac{n_c}{n},
\end{aligned}$$

4. Repeat from Step 2, until log-likelihood converges.

EM for GMM

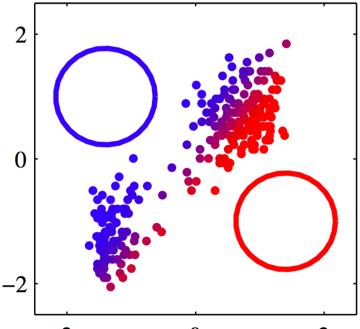
- Initialization



From Bishop's *Pattern recognition and machine learning*, Figure 9.8.

EM for GMM

- First soft assignment:

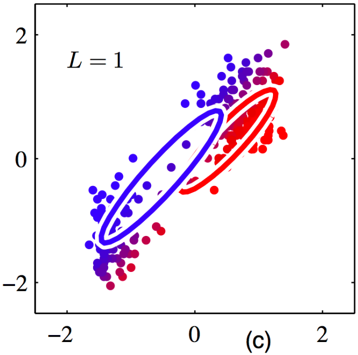


(b)

From Bishop's *Pattern recognition and machine learning*, Figure 9.8.

EM for GMM

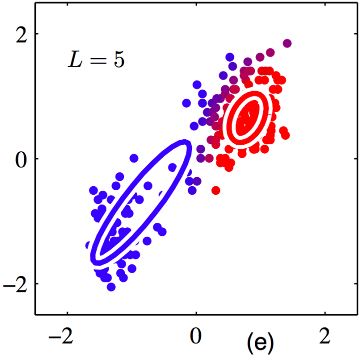
- First soft assignment:



From Bishop's *Pattern recognition and machine learning*, Figure 9.8.

EM for GMM

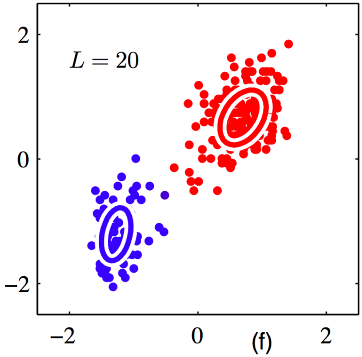
- After 5 rounds of EM:



From Bishop's *Pattern recognition and machine learning*, Figure 9.8.

EM for GMM

- After 20 rounds of EM:



From Bishop's *Pattern recognition and machine learning*, Figure 9.8.

Relation to k-Means

- EM for GMM seems a little like k-means.
- In fact, k-means is a limiting case of a **restricted** version of GMM.
- First, fix each cluster covariance matrix to be $\sigma^2 I$.
  - (This is the restriction: covariance matrices are fixed, and not iteratively estimated.)
- As we take $\sigma^2 \to 0$, the update equations converge to doing k-means.
- If you do a quick experiment yourself, you'll find
  - Soft assignments converge to hard assignments.
  - Has to do with the tail behavior (exponential decay) of Gaussian.