Clustering and Gaussian Mixture Models

Piyush Rai
IIT Kanpur

Probabilistic Machine Learning (CS772A)

Jan 25, 2016

Recap of last lecture..

### Clustering

- Usually an **unsupervised learning** problem
- Given: \( N \) **unlabeled** examples \( \{x_1, \ldots, x_N\} \); the number of partitions \( K \)
- Goal: Group the examples into \( K \) partitions

- Clustering groups examples based on their mutual similarities
- A good clustering is one that achieves:
  - **High within-cluster similarity**
  - **Low inter-cluster similarity**
- Examples: **K-means**, **Spectral Clustering**, **Gaussian Mixture Model**, etc.

### Refresher: K-means Clustering

- **Input**: \( N \) examples \( \{x_1, \ldots, x_N\} \); \( x_n \in \mathbb{R}^D \); the number of partitions \( K \)
- **Initialize**: \( K \) cluster means \( \mu_1, \ldots, \mu_K \). \( \mu_k \in \mathbb{R}^D \); many ways to initialize:
  - Usually initialized randomly, but good initialization is crucial; many smarter initialization heuristics exist (e.g., K-means++, Arthur & Vassilvitskii, 2007)
- **Iterate**:
  - **(Re)-Assign** each example \( x_n \) to its closest cluster center
    \[
    c_k = \{n: k = \arg \min_x \| x_n - \mu_k \|^2 \}
    \]
  - \( c_k \) is the set of examples assigned to cluster \( k \) with center \( \mu_k \)
  - **Update** the cluster means
    \[
    \mu_k = \text{mean}(c_k) = \frac{1}{|c_k|} \sum_{n \in c_k} x_n
    \]
- **Repeat** while not converged
- A possible convergence criteria: cluster means do not change anymore
The K-means Objective Function

- Notation: Size $K$ one-hot vector to denote membership of $x_n$ to cluster $k$
  $z_n = [0 \ 0 \ldots \ 1 \ 0 \ 0]$
  all zeros except the $k$th bit
- Also equivalent to just saying $z_n = k$
- $K$-means objective can be written in terms of the total distortion
  $J(\mu, Z) = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} \|x_n - \mu_k\|^2$
- Distortion: Loss suffered on assigning points $(x_n)_{n=1}^N$ to clusters $(\mu_k)_{k=1}^K$
- Goal: To minimize the objective w.r.t. $\mu$ and $Z$
- Note: Non-convex objective. Also, exact optimization is NP-hard
- The $K$-means algorithm is a heuristic; alternates b/w minimizing $J$ w.r.t. $\mu$ and $Z$; converges to a local minima

K-means: Some Limitations

- Makes hard assignments of points to clusters
  - A point either totally belongs to a cluster or not at all
  - No notion of a soft/fractional assignment (i.e., probability of being assigned to each cluster: say $K = 3$ and for some point $x$, $p_1 = 0.7, p_2 = 0.2, p_3 = 0.1$)
- $K$-means often doesn’t work when clusters are not round shaped, and/or may overlap, and/or are unequal

Gaussian Mixture Model: A probabilistic approach to clustering (and density estimation) addressing many of these problems

Mixture Models

- Data distribution $p(x)$ assumed to be a weighted sum of $K$ distributions
  $p(x) = \sum_{k=1}^{K} \pi_k p(x|\theta_k)$
  where $\pi_k$'s are the mixing weights: $\sum_{k=1}^{K} \pi_k = 1, \pi_k \geq 0$ (intuitively, $\pi_k$ is the proportion of data generated by the $k$-th distribution)
- Each component distribution $p(x|\theta_k)$ represents a “cluster” in the data
- Gaussian Mixture Model (GMM): component distributions are Gaussians
  $p(x) = \sum_{k=1}^{K} \pi_k N(x; \mu_k, \Sigma_k)$

- Mixture models used in many data modeling problems, e.g.,
  - Unsupervised Learning: Clustering (+density estimation)
  - Supervised Learning: Mixture of Experts models

GMM Clustering: Pictorially

- Notice the “mixed” colored points in the overlapping regions
GMM as a Generative Model of Data

- Can think of the data \( \{x_1, x_2, \ldots, x_N\} \) using a "generative story"
  - For each example \( x_n \), first choose its cluster assignment \( z_n \in \{1, 2, \ldots, K\} \) as
    \[
    z_n \sim \text{Multinoulli}(\pi_1, \pi_2, \ldots, \pi_K)
    \]
  - Now generate \( x \) from the Gaussian with id \( z_n \)
    \[
    x_n | z_n \sim N(\mu_{z_n}, \Sigma_{z_n})
    \]

- Note: \( p(z_n = 1) = \pi_k \) is the prior probability of \( x_n \) going to cluster \( k \) and
  \[
  p(z_n) = \prod_{k=1}^{K} \pi_k
  \]

GMM: Learning Cluster Assignment Probabilities

- For now, assume \( \pi = \{\pi_1, \ldots, \pi_K\} \) and \( \theta = \{\mu_k, \Sigma_k\}_{k=1}^{K} \) are known
- Given \( \theta \), the posterior probabilities of cluster assignments, using Bayes rule
  \[
  \gamma_{nk} = p(z_n = k | x_n) = \frac{p(x_n | z_n = k) p(z_n = k)}{\sum_{k=1}^{K} p(x_n | z_n = k) p(z_n = k)} = \frac{\pi_k N(x_n | \mu_k, \Sigma_k)}{\sum_{k=1}^{K} \pi_k N(x_n | \mu_k, \Sigma_k)}
  \]
  - Here \( \gamma_{nk} \) denotes the posterior probability that \( x_n \) belongs to cluster \( k \)
  - Posterior prob. \( \gamma_{nk} \propto \text{prior probability } \pi_k \times \text{likelihood } N(x_n | \mu_k, \Sigma_k) \)
  - Note that unlike K-means, there is a non-zero posterior probability of \( x_n \) belonging to each of the \( K \) clusters (i.e., probabilistic/soft clustering)
  - Therefore for each example \( x_n \), we have a vector \( \gamma_n \) of cluster probabilities
    \[
    \gamma_n = [\gamma_{n1}, \gamma_{n2}, \ldots, \gamma_nK], \quad \sum_{k=1}^{K} \gamma_{nk} = 1, \gamma_{nk} > 0
    \]

Learning GMM

- Given \( N \) observations \( \{x_1, x_2, \ldots, x_N\} \) drawn from mixture distribution \( p(x) \)
  \[
  p(x) = \sum_{k=1}^{K} \pi_k N(x | \mu_k, \Sigma_k)
  \]
- Learning the GMM involves the following:
  - Learning the cluster assignments \( \{z_1, z_2, \ldots, z_N\} \)
  - Estimating the mixing weights \( \pi = \{\pi_1, \ldots, \pi_K\} \) and the parameters \( \theta = \{\mu_k, \Sigma_k\}_{k=1}^{K} \) of each of the \( K \) Gaussians

GMM as a Generative Model of Data

- Joint distribution of data and cluster assignments
  \[
  p(x, z) = p(z)p(x|z)
  \]
- Marginal distribution of data
  \[
  p(x) = \sum_{k=1}^{K} p(z_k = 1)p(x|z_k = 1) = \sum_{k=1}^{K} \pi_k N(x | \mu_k, \Sigma_k)
  \]
- Thus the generative model leads to exactly the same \( p(x) \) that we defined
GMM: Estimating Parameters

- Now assume the cluster probabilities \( \gamma_1, \ldots, \gamma_K \) are known.
- Let us write down the log-likelihood of the model:
  \[
  L = \log p(X) = \log \prod_{n=1}^{N} p(x_n) = \sum_{n=1}^{N} \log p(x_n) = \sum_{n=1}^{N} \log \left( \sum_{k=1}^{K} \gamma_k N(x_n; \mu_k, \Sigma_k) \right)
  \]
- Taking derivative w.r.t. \( \mu_k \) (done on blackboard) and setting to zero for \( k = 1, \ldots, K \):
  \[
  \sum_{n=1}^{N} \gamma_k N(x_n; \mu_k, \Sigma_k) \Sigma_k^{-1} (x_n - \mu_k) = 0
  \]
- Plugging and chugging, we get:
  \[
  \mu_k = \frac{\sum_{n=1}^{N} \gamma_n x_n}{\sum_{n=1}^{N} \gamma_n} = \frac{1}{N} \sum_{n=1}^{N} \gamma_n x_n
  \]
- Thus mean of \( k \)-th Gaussian is the weighted empirical mean of all examples.
- \( N_k = \sum_{n=1}^{N} \gamma_n \): "effective" num. of examples assigned to \( k \)-th Gaussian (note that each example belongs to each Gaussian, but "partially")

GMM: Estimating Parameters

- Doing the same, this time w.r.t. the covariance matrix \( \Sigma_k \) of \( k \)-th Gaussian:
  \[
  \Sigma_k = \frac{1}{N} \sum_{n=1}^{N} \gamma_n (x_n - \mu_k)(x_n - \mu_k)^T
  \]
- .. using similar computations as MLE of the covariance matrix of a single Gaussian (shown on board)
- \( \Sigma_k \) is the weighted empirical covariance of all examples.
- Finally, the MLE objective for estimating \( \pi = (\pi_1, \pi_2, \ldots, \pi_K) \):
  \[
  \sum_{n=1}^{N} \log \sum_{k=1}^{K} \pi_k N(x_n; \mu_k, \Sigma_k) = \lambda \sum_{k=1}^{K} \gamma_k - 1 \quad (\lambda \text{ is the Lagrange multiplier for } \sum_{k=1}^{K} \gamma_k = 1)
  \]
- Taking derivative w.r.t. \( \pi_k \) and setting it to zero gives Lagrange multiplier \( \lambda = -N \). Plugging it back and chugging, we get:
  \[
  \pi_k = \frac{N_k}{N}
  \]
- which makes intuitive sense (fraction of examples assigned to cluster \( k \))

Summary of GMM Estimation

- Initialize parameters \( \theta = (\mu_1, \Sigma_1)^K \), and mixing weights \( \pi = (\pi_1, \ldots, \pi_K) \), and alternate between the following steps until convergence:
  - Given current estimates of \( \theta = (\mu_k, \Sigma_k)^K \) and \( \pi \):
    - Estimate the posterior probabilities of cluster assignments:
      \[
      \gamma_{nk} = \frac{\pi_k N(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j N(x_n; \mu_j, \Sigma_j)} \quad \forall n, k
      \]
    - Given the current estimates of cluster assignment probabilities \( \{\gamma_{nk}\} \):
      - Estimate the mean of each Gaussian:
        \[
        \mu_k = \frac{1}{N_k} \sum_{n=1}^{N_k} \gamma_{nk} x_n \quad \forall k, \text{ where } N_k = \sum_{n=1}^{N} \gamma_{nk}
        \]
      - Estimate the covariance matrix of each Gaussian:
        \[
        \Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N_k} \gamma_{nk} (x_n - \mu_k)(x_n - \mu_k)^T \quad \forall k
        \]
      - Estimate the mixing proportion of each Gaussian:
        \[
        \pi_k = \frac{N_k}{N} \quad \forall k
        \]
- K-means: A Special Case of GMM

- Assume the covariance matrix of each Gaussian to be spherical:
  \[
  \Sigma_k = \sigma^2 I
  \]
- Consider the posterior probabilities of cluster assignments:
  \[
  \gamma_{nk} = \frac{\pi_k N(x_n; \mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j N(x_n; \mu_j, \Sigma_j)} = \frac{\pi_k \exp \left( -\frac{1}{2\sigma^2} ||x_n - \mu_k||^2 \right)}{\sum_{j=1}^{K} \pi_j \exp \left( -\frac{1}{2\sigma^2} ||x_n - \mu_j||^2 \right)}
  \]
- As \( \sigma^2 \to 0 \), the summation of denominator will be dominated by the term with the smallest \( ||x_n - \mu_j||^2 \). For that \( j \):
  \[
  \gamma_{nj} = \frac{\pi_j \exp \left( -\frac{1}{2\sigma^2} ||x_n - \mu_j||^2 \right)}{\pi_j \exp \left( -\frac{1}{2\sigma^2} ||x_n - \mu_j||^2 \right)} = 1
  \]
- For \( \ell \neq j \), \( \gamma_{nj} \approx 0 \Rightarrow \text{hard assignment with } \gamma_{nj} \approx 1 \text{ for a single cluster } j \)
- Thus, for \( \Sigma_k = \sigma^2 I \) (spherical) and \( \sigma^2 \to 0 \), GMM reduces to K-means
Next class: The Expectation Maximization Algorithm