Intelligent Systems: Reasoning and Recognition

James L. Crowley

ENSIMAG 3 Fall Semester 2017/2018

Lesson 6 6 December 2017

Clustering and non-supervised learning with K-Means and EM

Sources:

C. M. Bishop, "Pattern Recognition and Machine Learning", Springer Verlag, 2006.

Jeff Bilmes, A Gentle Tutorial of the EM Algorithm, Tech Report, Univ of Washington, 1998. (available for download from course website).

Notation

x	a variable
X	a random variable (unpredictable value)
V	The number of possible values for X (Can be infinite).
$\vec{x}$	A vector of D variables.
$\vec{X}$	A vector of D random variables.
D	The number of dimensions for the vector $\vec{x}$ or $\vec{X}$
k	index for cluster, data source or GMM Mode
K	Total number of clusters, or sources, of events
M	Total number of sample events. $M = \sum_{k=1}^{K} M_k$
$\{\vec{X}_m\}$	A set of M Sample Observations (a training set)
$\{\vec{y}_m\}$	A set of indicator vectors for the training samples in $\{\vec{X}_m\}$
	$\vec{y}_m$ indicates the source S_k for each training sample $\vec{X}_m$
Note that	$\vec{y}_m$ can be a binary vector with k rows (1 for S_k and 0 for others) or
	$\vec{y}_m$ can be the probability that $\vec{X}_m \in S_k$

$h(k,m) = (\vec{y}_1 \quad \cdots \quad \vec{y}_m)$ Indicator variables in matrix form. k rows, m columns

Expected Value: $E\{X\} = \frac{1}{M}\sum_{m=1}^{M} X_m$

Gaussian or Normal Density: $\mathcal{N}(\vec{X};\vec{\mu},\Sigma) = \frac{1}{(2\pi)^{\frac{D}{2}}\det(\Sigma)^{\frac{1}{2}}} e^{-\frac{1}{2}(\vec{X}-\vec{\mu})^T \Sigma^{-1}(\vec{X}-\vec{\mu})}$

Multivariate Normal Density Function

The "Central Limit Theorem" tells us that whenever the features an observation are the result of a sequence of N independent random events, the probability density of the features will tend toward a Normal or Gaussian density.

$$p(\vec{X}) = \mathcal{N}(\vec{X};\vec{\mu},\Sigma) = \frac{1}{(2\pi)^{\frac{D}{2}}\det(\Sigma)^{\frac{1}{2}}} e^{-\frac{1}{2}(\vec{X}-\vec{\mu})^T \Sigma^{-1}(\vec{X}-\vec{\mu})}$$

Where the parameters $\vec{\mu}, \Sigma$ and the mean and co-variance of the density. These are the first and second moments of the density.

Note that we use upper case for probabilities and lower case for functions.
Thus $P(\omega)$ is a value, $p(X)$ is a function.

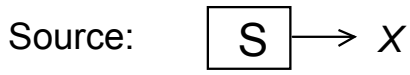
The mean is $\vec{\mu} = E\{\vec{X}\} = \begin{pmatrix} E\{X_1\} \\ E\{X_2\} \\ \dots \\ E\{X_D\} \end{pmatrix} = \begin{pmatrix} \mu_1 \\ \mu_2 \\ \dots \\ \mu_D \end{pmatrix}$

and the Covariance is $\Sigma = E\{(\vec{X} - E\{\vec{X}\})(\vec{X} - E\{\vec{X}\})^T\} = \begin{pmatrix} \sigma_{11}^2 & \sigma_{12}^2 & \dots & \sigma_{1D}^2 \\ \sigma_{21}^2 & \sigma_{22}^2 & \dots & \sigma_{2D}^2 \\ \dots & \dots & \dots & \dots \\ \sigma_{D1}^2 & \sigma_{D2}^2 & \dots & \sigma_{DD}^2 \end{pmatrix}$

Gaussian Mixture Models

Gaussian Mixtures as a Sum of Independent Sources

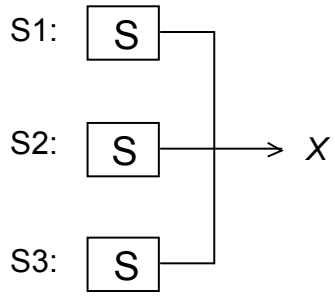
We can consider a sequence of random trials as a "source" of event



The central limit theorem tells us that in this case, the result of many independent random events will converge to a Normal density function:

$$p(\vec{X}) = \mathcal{N}(\vec{X}; \vec{\mu}, \Sigma)$$

Sometimes a population will result from a set of K different sources, S_k, each with it own unique independent random variables.



In this case, the probability density is often better represented as a weighted sum of normal densities.

$$p(\vec{X}) = \sum_{k=1}^{K} \alpha_k \mathcal{N}(\vec{X}; \vec{\mu}_k, \Sigma_k)$$

Such a sum is referred to as a Gaussian Mixture Model (GMM). A GMM can be used to represent density functions from multiple sources. It can also be used to discover a set of subclasses within a global class.

The weights, α_k, are the relative frequencies of events from each source S_k.

The coefficients α_k to be a probability, we must assure that $\sum_{k=1}^{K} \alpha_k = 1$

In this case, the α_k form a probability Distribution.

Our problem is to discover the source for each sample, and to estimate the mean and covariance $(\vec{\mu}_k, \Sigma_k)$ for each source. We will look at two possible algorithms for this: K-Means Clustering, and Expectation Maximization. In both cases, the algorithm will iteratively construct a table, *h(k,m)* that assigns each sample to one of K clusters or sources.

K-Means and EM can be used to discover the classes for each training sample, and are thus used for Unsupervised Learning. They can also be used to estimate a multimodal density for a single class.

For K-Means, the assignment of a source to a sample, *h(k,m)*, is be a hard assignment, with *h(k, m) = 1* if observation $\vec{X}_m$ is assigned to cluster S_k and 0 otherwise.

When used for unsupervised learning, this can be seen as equivalent to discovering the indicator variable $\vec{y}_m$

$$h(k,m) = \begin{cases} 1 & \text{if sample } \vec{X}_m \in S_k \\ 0 & \text{Otherwise} \end{cases}$$

h(k, m) = 1 if $\vec{X}_m$ is assigned to cluster *k*, 0 otherwise.

In the case of EM, this will be a soft assignment, in which *h(k,m)* represents the probability that sample $\vec{X}_m$ comes from source (or cluster), S_k.

$$h(k,m) = P(X_m \in S_k)$$

In either case we must initialize the estimated clusters. The better the initial estimate, the faster and more reliable the result. In the absence of any initial estimate, we can use $\vec{\mu}_k^1 = k\vec{\mu}_0^1$, $\Sigma_k^1 = I$. However, it is better to use domain knowledge when possible.

Both K-means and EM are sensitive to the starting point and can converge to a local minimum that is not the best estimate. EM is less sensitive but does not always converge to the global best estimate.

K-Means Clustering

Assume a set of M sample observations $\{\vec{X}_m\}$, with each observation drawn from one of K clusters S_k. Our problem is to discover an assignment table *h(k, m)* that assigns each observation, $\vec{X}_m$ in the sample set to the "best" cluster, S_k.

$$h(k,m) = \begin{cases} 1 & \text{if sample } \vec{X}_m \in S_k \\ 0 & \text{Otherwise} \end{cases}$$

Given an estimate of the mean, $\vec{\mu}_k$, and covariance Σ_k for each cluster, S_k. we can use the Mahalanobis Distance to determine the best cluster.

For each cluster we can then refine the estimate of the mean, $\vec{\mu}_k$, and covariance Σ_k.

This suggests an iterative process composed of two steps:

1) Expectation: For each sample, $\vec{X}_m$, determine the most likely cluster S_k. using the distance to the current estimate of the mean, $\vec{\mu}_k$, and covariance Σ_k.

2) Maximization: For each cluster re-calcuate the mean, $\vec{\mu}_k$, and covariance Σ_k using sample assignments in *h(k,m)*.

We can initialize the process to any value. For example, $\vec{\mu}_k^{(0)} = k\vec{\mu}_0$, $\Sigma_k^{(0)} = I$

However, it IS possible for K-means to be stuck in a local minimum, and the closer we start to the best values, the faster the process converges.

We will seek to minimize a quality metric:
For K-Means this is the sum of the Mahalanobis distances. (
Distance normalized by Covariance)

$$Q^{(i)} = \sum_{m=1}^{M} \sum_{k=1}^{K} h^{(i)}(m,k)(\vec{X}_m - \vec{\mu}_k^{(i)})^T \Sigma_k^{(i)-1} (\vec{X}_m - \vec{\mu}_k^{(i)})$$

Initially $h^{(o)}(m, k) = 0$, $i=0$.
We can stop the process after a fixed number of iterations, or when the assignment table does not change or when $Q^{(i)}$ does not change.

Expectation:

$$i \leftarrow i+1$$
$$\forall m = 1, M:$$
$$\forall k = h^{(i)}(k,m) = 0$$
$$\mathrm{k} = \arg\min_{k} \left\{ (\vec{X}_m - \vec{\mu}_k)^T \Sigma_k^{-1} (\vec{X}_m - \vec{\mu}_k) \right\}$$
$$h^{(i)}(k,m) \leftarrow 1$$

Maximization

Mass: $M_k = \sum_{m=1}^{M} h^{(i)}(k,m)$ is the number of samples attributed to source k.

If $M_k \neq 0$:

Mean: $$\mu_k^{(i)} = \frac{1}{M_k} \sum_{m=1}^{M} h^{(i)}(k,m) \cdot \vec{X}_m$$

Covariance: $$\Sigma_k^{(i)} = \frac{1}{M_k} \sum_{m=1}^{M} h^{(i)}(k,m) \cdot (\vec{X}_m - \vec{\mu}_k)(\vec{X}_m - \vec{\mu}_k)^T$$

That is, for each component of the covariance, $\sigma_{ij}^{(i)}$:

$$\sigma_{ij}^{2(i)} = \frac{1}{M_k} \sum_{m=1}^{M} h^{(i)}(k,m) \cdot (x_{mi} - \mu_{ki})(x_{mj} - \mu_{kj})$$

At the end of each cycle:

Quality: $$Q^{(i)} = \sum_{m=1}^{M} \sum_{k=1}^{K} h^{(i)}(m,k) (\vec{X}_m - \vec{\mu}_k^{(i)})^T \Sigma_k^{(i)-1} (\vec{X}_m - \vec{\mu}_k^{(i)})$$

The process stops after a fixed number of cycles, or when the sample assignment does not change or the quality metric does not change.

Each source can be interpreted as a separate class or as a mode in a Gaussian Mixture model, depending on the application.

The Expectation Maximization Algorithm (EM)

As before, assume a set of *M* sample observations $\{\vec{X}_m\}$, with each observation drawn from one of K sources S_k. Our problem is to discover an assignment table *h(k, m)* that assigns each observation, $\vec{X}_m$ in the sample set to the "best" cluster, S_k. For EM this will be a probability.

EM iteratively estimates the probability for the assignment of each observation to each source.

Expectation Maximization has many uses, including estimating the density functions for a Hidden Markov Model (HMM) as well as for estimating the parameters for a Gaussian Mixture model.

For a Gaussian Mixture model, a probability density is represented as a weighted sum of normal densities.

$$p(\vec{X}) = \sum_{k=1}^{K} \alpha_k \mathcal{N}(\vec{X}; \vec{\mu}_k, \Sigma_k)$$

It is sometimes convenient to group the parameters for each source into a single vector:

$$\vec{v}_k = (\alpha_k, \vec{\mu}_k, \Sigma_k)$$

The complete set of parameters is a vector with K·P coefficients.
For a feature vector of D dimensions, $\vec{v}_k$ has P = 1 + D + D(D+1)/2 coefficients.

To estimate $\{\alpha_k, \vec{\mu}_k, \Sigma_k\}$ we need the assignment of samples to source, *h(k,m)*.
To estimate *h(k,m)* we need the parameters $\{\alpha_k, \vec{\mu}_k, \Sigma_k\}$

This leads to an iterative two-step process in which we alternately estimate *h(k,m)*. and then $\{\alpha_k, \vec{\mu}_k, \Sigma_k\}$.

The EM algorithms constructs a table, *h(k,m)*
Unlike K-Means, *h(k,m)* will contain probabilities.

$$h(k,m) = P(\vec{X}_m \in S_k)$$

Initialization:

Choose K (the number of sources). Use domain knowledge if possible.
set *i=0*.
Form an initial estimate for $\vec{v}^{(0)} = (\alpha_k^{(0)}, \vec{\mu}_k^{(0)}, \Sigma_k^{(0)})$ *for k = 1 to K.*

The closer the initial estimate, the faster the algorithm converges. Domain knowledge is useful here.

Expectation step (E)

let $i \leftarrow i+1$

Calculate the table $h^{(i)}(k,m)$ using the training data and estimated parameters.

$$h^{(i)}(k,m) = P(\vec{X}_m \in S_k \mid \{X_m\}, \vec{v}^{(i-1)})$$

which gives:

$$h^{(i)}(k,m) \leftarrow \frac{\alpha_k^{(i-1)} \mathcal{N}(\vec{X}_m, \vec{\mu}_k^{(i-1)}, \Sigma_k^{(i-1)})}{\sum_{j=1}^{K} \alpha_j^{(i-1)} \mathcal{N}(\vec{X}_m, \vec{\mu}_j^{(i-1)}, \Sigma_j^{(i-1)})}$$

Maximization Step (M)

Estimate the parameters $\vec{v}^{(i)}$ using $h^{(i)}(k,m)$

Mass:	$M_k^{(i)} \leftarrow \sum_{m=1}^{N} h^{(i)}(k,m)$ (Note: M_k is a real)
Probability:	$\alpha_k^{(i)} \leftarrow \frac{1}{M} \sum_{m=1}^{M} h^{(i)}(k,m) = \frac{M_k^{(i)}}{M}$
Mean:	$\vec{\mu}_k^{(i)} \leftarrow \frac{1}{M_k^{(i)}} \sum_{m=1}^{M} h^{(i)}(k,m) \vec{X}_m$
Covariance:	$\Sigma_k^{(i)} \leftarrow \frac{1}{M_k^{(i)}} \sum_{m=1}^{M} h^{(i)}(k,m)(\vec{X}_m - \vec{\mu}_k^{(i)})(\vec{X}_m - \vec{\mu}_k^{(i)})^T$

Convergence Criteria

The quality metric is the Log-likelihood of the probability of obtaining the data given the parameters.

$$Q^{(i)} = \ln\{p(\{\vec{X}_n\} \mid \vec{v}^{(i)})\} = \sum_{m=1}^{M} \ln\left\{\sum_{j=1}^{K} \alpha_j^{(i)} \mathcal{N}(\vec{X}_m \mid \mu_j^{(i)}, \Sigma_j^{(i)})\right\}$$

It can be shown that, for EM, the log likelihood will converge to a stable maximum. The change in Q will monotonically decrease. This can be used to define a halting condition:

If $\Delta Q = Q^{(i)} - Q^{(i-1)}$ is less than a threshold, halt.