Lecture Notes: Clustering and Mixture Models

Ted Meeds[1,2]

[1] Informatics Institute, University of Amsterdam
[2] The Centre for Integrative Bioinformatics, Vrije University
tmeeds@gmail.com

Abstract In this note we discuss two popular clustering algorithms: **k-means** clustering and **mixture of Gaussians**. Clustering falls under the scope of **unsupervised learning** because it assumes only a set of input vectors $\{\mathbf{x}_n\}$, whereas **supervised learning** assumes observed input-output pairs $\{t_n, \mathbf{x}_n\}$. K-means makes hard-assignments of input vectors to cluster indices (i.e. "hard" clustering) while mixture of Gaussians makes soft-assignments (i.e. "soft" clustering). Both models are optimized using iteratively using two steps per iteration: the first step assigns data to clusters and the second step updates cluster parameters according to assignments. For mixture models, this algorithm is called Expectation-Maximization.

1 Clustering

The inductive bias for clustering: data are split into groups or clusters. By assuming this model, the goal of unsupervised learning is to fit parameters that best explain this inductive bias. Clustering can sometimes be thought of as classification without observing the target labels, but assuming that there exists K classes. The assignment step can be thought of *completing* the data (assigning class labels given parameters) and the optimization step is just maximum likelihood given a complete labeling (sometimes called the *complete log-likelihood*). There are many ways of clustering data (let alone unsupervised learning), but K-means and mixtures of Gaussians are very useful and simple models.

2 K-means

We assume a data set of N input vectors $\{\mathbf{x}_n\}$. We assume a model with K cluster means $\{\boldsymbol{\mu}_k\}$, hence *K-means*. I.e. parameters $\boldsymbol{\theta} = \{\boldsymbol{\mu}_k\}$. We introduce assignment variables $r_{nk} \in \{0, 1\}$, $\sum_k r_{nk} = 1$ that are also called **responsibilities** because they represent the responsibility that cluster k has for explaining data vector $\mathbf{x}_n$. For K-means the responsibilities are all-or-none (hard-assignments), but for mixtures the responsibilities will be shared (soft-assignments). Responsibilities are also parameters of the model, but they are special because there is one set of responsibilities *per data point* and are generally used to fit the other parameters during learning. At test time, the values of the responsibilities are assigned using the mean parameters (not the training data responsibilities). It is a subtle difference.

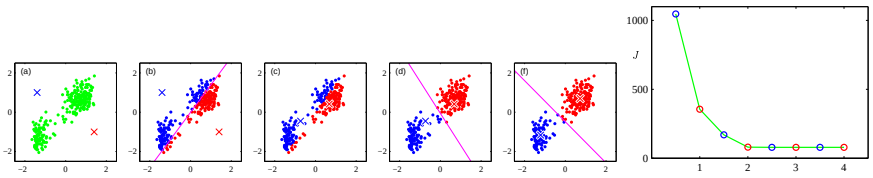

Figure 1. K-means algorithm. Left: initial means (x's) and data. Each other image is the result of either the E or the M step. Last image is the decrease in C (Bishop uses J) for each E or M.

2.1 Objective function

The objective function for K-means is the squared distance between data and their assigned cluster means:

$$C(\{r_{nk}\}, \{\boldsymbol{\mu}_k\}) = \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}||\mathbf{x}_n - \boldsymbol{\mu}_k||^2 \tag{1a}$$

$$= \sum_{k=1}^{K}\sum_{n\in S_k} ||\mathbf{x}_n - \boldsymbol{\mu}_k||^2 \tag{1b}$$

$$= \sum_{k=1}^{K}\sum_{n\in S_k} e_{nk} \tag{1c}$$

where S_k is the set of indices with hard assignment $r_{nk} = 1$.

2.2 Optimizing Assignments

Minimize C wrt r_{nk} keeping $\{\boldsymbol{\mu}_k\}$ fixed (aka the **E-step**). The cost due to the value of this parameter is independent of all other data vectors. Which means its cost is simply $\sum_{k=1}^{K} r_{nk}||\mathbf{x}_n - \boldsymbol{\mu}_k||^2$, which is minimized by assigning to the closest cluster mean:

$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_j ||\mathbf{x}_n - \boldsymbol{\mu}_j||^2 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

2.3 Optimizing Clusters

Minimize C wrt $\boldsymbol{\mu}_k$ keeping $\{r_{nk}\}$ fixed (aka the **M-step**). We can take the gradient of C and solve for μ_k exactly. Note that this is the same update as for the means of the generative classifier where class labels replace the responsibilities.

$$\frac{\partial C(\{r_{nk}\}, \{\boldsymbol{\mu}_k\})}{\partial \boldsymbol{\mu}_k} = 2\sum_{n=1}^{N} r_{nk}(\mathbf{x}_n - \boldsymbol{\mu}_k) = 0 \tag{3a}$$

$$\boldsymbol{\mu}_k = \frac{\sum_{n=1}^{N} r_{nk}\mathbf{x}_n}{\sum_{n=1}^{N} r_{nk}} \tag{3b}$$

2.4 K-means Algorithm

At a high-level, the algorithm iterates E-M-E-M-E-M... until convergence. Convergence is easy to detect for K-means because once the assignments no longer change, the means no longer change, so the cost function will no longer change. Figure 1 illustrates K-means on toy problem.

1. Initialize means / responsibilities (many ways, e.g. randomly assign then compute means).
2. Repeat until $\Delta C = 0$:
   (a) E-Step: assign data to (fixed) clusters.
   (b) M-Step: optimize clusters given (fixed) assignments.
3. Return $\{\boldsymbol{\mu}_k\}$.

2.5 Issues

- Many local optima. EM is a hill-climbing algorithm that will converge, but only to a local optimum. Many restarts may be required.
- The algorithm is $\mathcal{O}(NK)$. Faster online SGD-like updates possible and more efficient for very large datasets. See Bishop.
- Sensitive to the scale of the features in the data. If one dimension has much larger scale, then the clustering will be dominated by that dimension. Solutions: normalize your data or use mixture of Gaussians, which can take this into account in its covariances.

3 Mixture of Gaussians

Mixture of Gaussians is a soft-clustering version of K-means. It is "soft" because the assignments are not 0/1 but are equal to $P(C_k|\mathbf{x})$ (and therefore sum to 1). Although we focus on mixture of Gaussians, mixture models can be applied to any kind of data: Multinomial data, Poisson data, linear regression models, etc.

The inductive bias for mixture models is a generative model of the input vectors (NB: there are no "target" vectors like there are in classification). The generative process/model for the data is:

1. Pick a mixture component (a class-conditional Gaussian) according to $\pi_1, \ldots, \pi_K$ (e.g. $[0.1, 0.2, 0.7]$). I.e. $k \sim \boldsymbol{\pi}$.
2. Generate $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, where k was chosen in 1.

The probability density of this process (the likelihood of the data generated) is:

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{4}$$

The density of a data vector is therefore requires marginalizing over the mixture components.

To learn the parameters of a mixture model, we introduce a K-dimensional binary random variable $\mathbf{z}$, e.g. $\mathbf{z}^T = [0, 1, 0, 0, 0]$. This is an unobserved latent variable representing the cluster that generated the data. We do not know this, but since we assume the data a generated this way, we just need to infer its probability during the learning procedure.

$$P(z_k = 1) = \pi_k \tag{5a}$$
$$p(\mathbf{x}_n | z_{nk} = 1) = \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{5b}$$

or more generally

$$p(\mathbf{x}_n | \mathbf{z}_n) = \prod_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_{nk}} \tag{5c}$$

and the joint:

$$p(\mathbf{x}_n, \mathbf{z}_n) = p(\mathbf{z}_n) p(\mathbf{x}_n | \mathbf{z}_n) \tag{5d}$$
$$= \prod_k (\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k))^{z_{nk}} \tag{5e}$$

We marginalize to obtain the likelihood of the observation:

$$p(\mathbf{x}_n) = \sum_{\mathbf{z}_n} p(\mathbf{z}_n) p(\mathbf{x}_n | \mathbf{z}_n) \tag{5f}$$
$$= \sum_{\mathbf{z}_n} \underbrace{\prod_k (\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k))^{z_{nk}}}_{\text{each } \mathbf{z}_n \text{selects one cluster}} \tag{5g}$$
$$= \sum_k \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{5h}$$

We therefore have a latent variable $\mathbf{z}_n$ for each training vector $\mathbf{x}_n$. Even though the likelihood marginalizes over $\mathbf{z}$, the joint distribution will be used to iteratively learn the parameters of the model, just like the joint $p(\mathbf{t}, \mathbf{x})$ was used in the generative model for classification.

The model of the data is:

$$p(\mathbf{x}_n) = \sum_k \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \qquad \boldsymbol{\theta} = \{\boldsymbol{\pi}, \{\boldsymbol{\mu}_k\}, \{\boldsymbol{\Sigma}_k\}\} \tag{6a}$$

the likelihood of the data set is:

$$p(\mathbf{X}) = \prod_n \sum_k \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{6b}$$

3.1 Objective Function

Our objective function is again the log-likelihood of the training set (implicitly conditioning on $\boldsymbol{\theta}$):

$$\log p(\mathbf{X}) = \sum_n \log \sum_k \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{6c}$$

Note that is we observed $\mathbf{z}$, as $\mathbf{t}$ are observed in classification, then the logarithm would go inside the sum over clusters.

3.2 Optimization of Parameters (M-step)

Optimizing cluster centers :

$$\frac{\partial \log p(\mathbf{X})}{\partial \mu_k} = \sum_n \underbrace{\frac{\pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_k \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}}_{\frac{p(\mathbf{x}_n, z_{nk}=1)}{\sum_{\mathbf{z}_n} p(\mathbf{x}_n, \mathbf{z}_n)} = P(z_{nk}=1|\mathbf{x}_n) = r_{nk}} \boldsymbol{\Sigma}_k^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_k) \tag{7a}$$

$$0 = \sum_n r_{nk} \boldsymbol{\Sigma}_k^{-1}(\mathbf{x}_n - \boldsymbol{\mu}_k) \tag{7b}$$

$$\boldsymbol{\mu}_k = \frac{\sum_n r_{nk}\mathbf{x}_n}{\sum_n r_{nk}} = \frac{1}{N_k}\sum_n r_{nk}\mathbf{x}_n \tag{7c}$$

It is important to notice that we have used Bayes rule to find $p(z_k|\mathbf{x})$ and call this the responsibility of cluster k for input $\mathbf{x}$. Note this is a posterior probability and is called $\gamma(z_{nk})$ in Bishop. This is also the *expected* value of $z_k|\mathbf{x}$ (hence expectation step). If you dig into EM further in Bishop, this is used fill in the *expected complete data likelihood*. Remember for classification our data was complete since we observed the class labels. Here they are filled in with their expectation under the model then the model is optimized given the expected labels (this is the EM algo).

Optimizing cluster covariances :

$$\boldsymbol{\Sigma}_k = \frac{1}{N_k}\sum_n r_{nk}(\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^T \tag{7d}$$

i.e. a soft-covariance.

Optimizing cluster weights : The mixture weights need to be constrained $\sum_k \pi_k = 1$. To do this we use a Lagrange multiplier λ. The objective function is also modified: $f = \log p(\mathbf{X}) + \lambda(\sum_k \pi_k - 1)$, so that the derivative for π_k becomes:

$$\frac{\partial \log p(\mathbf{X})}{\partial \pi_k} = \sum_n \frac{\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} + \lambda \tag{7e}$$

$$\text{multiply by } \pi_k \tag{7f}$$

$$= \sum_n r_{nk} + \pi_k\lambda = 0 \tag{7g}$$

$$\pi_k\lambda = -N_k \tag{7h}$$

$$\sum_k \pi_k\lambda = \sum_k -N_k \tag{7i}$$

$$\lambda = -N \qquad \pi_k = \frac{N_k}{N} \tag{7j}$$

The M-step therefore provides an update of parameters $\boldsymbol{\theta}$ while fixing the responsibilities.

3.3 Setting the responsibilities (E-step)

The E-step simply using the updated parameters to assign responsibilities. This too is an optimization step, though the theory why is not covered in the course, but it is in Bishop.

3.4 Expectation-Maximization Algorithm

1. Initialize $\boldsymbol{\theta} = \{\boldsymbol{\pi}, \{\boldsymbol{\mu}_k\}, \{\boldsymbol{\Sigma}_k\}\}$.
2. Repeat until convergence (e.g. check change in log-likelihood):
   (a) E-step (for all k, n):

$$r^\star_{nk} = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} \tag{8a}$$

   (b) M-step (for all k):

$$\pi^\star_k = \frac{1}{N} \sum_n r^\star_{nk} \tag{8b}$$

$$\boldsymbol{\mu}^\star_k = \frac{1}{N_k} \sum_n r^\star_{nk} \mathbf{x}_n \tag{8c}$$

$$\boldsymbol{\Sigma}^\star_k = \frac{1}{N_k} \sum_n r^\star_{nk} (\mathbf{x}_n - \boldsymbol{\mu}^\star_k)(\mathbf{x}_n - \boldsymbol{\mu}^\star_k)^T \tag{8d}$$

After each EM step, compute the loglikelihood $\log p(\mathbf{X}|\boldsymbol{\theta}^\star)$ (in a numerically stable way using logsumexp). A useful convergence test is the delta log-likelihood because the log-likelihood is guaranteed to increase after each step. One could also look at changes in parameters and stop if they have not changed very much. Log-likelihood is the most common test.

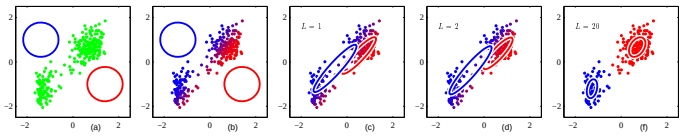

Figure 2. MoG algorithm.

3.5 General EM for Mixtures

1. Initialize $\boldsymbol{\theta}$.
2. Repeat until convergence (e.g. check change in log-likelihood):
   (a) E-step (for all k, n):

$$r_{nk}^{\star} = \frac{\pi_k p(\mathbf{x}_n|\boldsymbol{\theta}_k)}{\sum_j \pi_j p(\mathbf{x}_n|\boldsymbol{\theta}_j)} \tag{9a}$$

   (b) M-step (for all k):

$$\pi_k^{\star} = \frac{1}{N}\sum_n r_{nk}^{\star} \tag{9b}$$

$$\boldsymbol{\theta}_k^{\star} \leftarrow \arg\min_{\boldsymbol{\theta}_k} \log p(\mathbf{X}|\boldsymbol{\theta}) \qquad \text{(with } r_{nk}^{\star} \text{ and } \pi_k^{\star} \text{ fixed)} \tag{9c}$$

3.6 Summary of EM

- Log-likelihood increases after each iteration. It can be seen as performing block-wise coordinate ascent, first on responsibilities given parameters, then parameters given responsibilities.
- There are **many** local optima (though many will be good) since we just performing gradient ascent.
- If there is one datapoint at one cluster, then degenerate solutions appear: $\boldsymbol{\Sigma}_k \to 0$, $\log p(\mathbf{X}) \to \infty$.
- Sometimes people think of the cluster indices as labels 1 through K. This is dangerous as the labels are non-identifiable (there are $K!$ labelings).
- Choosing K: either by model selection techniques like cross-validation, or a Bayesian approach called Dirichlet processes.
- Initialize with K-means or randomly