Clustering

Max Welling

California Institute of Technology 136-93
Pasadena, CA 91125
welling@vision.caltech.edu

1 Introduction

In this class we will treat two clustering algorithms, "mixture of Gaussians" clustering (MoG) and K-means clustering. In the context of our treatment, K-means may be considered as a limit case of MoG clustering, but this implies by no means that MoG is better than K-means. As it turns out, they do have slightly different properties, the most important one being that classical K-means tends to cluster in groups of roughly equal size, while MoG does not care about the cluster sizes. We will not prove this property, but hope to show it experimentally in some demos. Arguments for this property of K-means are given in (**?**).

Clustering can be viewed as an unsupervised method for classification. If we have no prior information on the labels of the data (i.e. to which class they belong), clustering algorithms "naturally" assign the data to a usually prespecified number of clusters (each cluster represented by a different label). Of course, the definition of different distance measures influence the end result of such algorithms. The problem of finding the natural number of clusters is again an instance of choosing the correct complexity of the model given the data. Complexity terms, like the MDL criterium, or a Baysian analysis, may also help in this case finding the "correct" number of clusters.

We will pose the problem as follows. Given a dataset $\mathbf{d} = \mathbf{x}^N$, consisting of N datapoints in a D-dimensional space, assign each point to one out of K clusters.

2 MoG

The strategy employed by MoG is to fit a mixture of Gaussians to the data, using the ML criterium. The model for one datapoint is given by,

$$\begin{aligned} p(\mathbf{x}|\boldsymbol{\theta}) &= \sum_{i=1}^{K} p(z=i)\; p(\mathbf{x}|z=i) \\ &= \sum_{i=1}^{K} \pi_i\; \mathcal{G}_{\mathbf{x}}[\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i] \end{aligned} \tag{1}$$

where z is a hidden variable which assigns a label ($z = \{1, ...K\}$) to every data point. In the following we will simply write i instead of $z = i$ to simplify notation. We may therefore imagine that the data were generated in the following way. First sample from the probability mass table $p(i)$ a random number between 1 and K. Then, given the label, generate a datapoint from the corresponding Gaussian. What we will do next is learn the parameters $\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i$ and π_i from the observed data $\mathbf{x}^N$. The posterior density $p(i|\mathbf{x}_n)$ plays a crucial role since it tells us with what probability a datapoint is assigned to a cluster. We will call this the responsibilities of the clusters for a datapoint. If, at some point we have to make a final decision about the membership of a datapoint, we can use the maximum a posteriori rule (MAP), i.e. choose the cluster with the highest responsibility. These posterior probabilities are also crucial in the EM algorithm. First we write down Q,

$$\begin{aligned} Q(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{t-1}) &= \sum_{n=1}^{N}\sum_{i=1}^{K} p(i|\mathbf{x}_n, \boldsymbol{\theta}_{t-1})\; \log[p(\mathbf{x}_n, i|\boldsymbol{\theta}_t)] \\ &= \sum_{n=1}^{N}\sum_{i=1}^{K} p(i|\mathbf{x}_n, \boldsymbol{\theta}_{t-1}) \log[p(\mathbf{x}_n|i, \boldsymbol{\theta}_t)] + \log[p(i|\boldsymbol{\theta}_t)] \\ &= \sum_{n=1}^{N}\sum_{i=1}^{K} p(i|\mathbf{x}_n, \boldsymbol{\theta}_{t-1})\; \log[\mathcal{G}_{\mathbf{x}}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)] + \log[\pi_i] \end{aligned} \tag{2}$$

The E-step now involves to calculate this posterior density. It is given by Bayes rule,

$$
\begin{aligned}
p(i|\mathbf{x}) &= \frac{p(\mathbf{x}|i)\ p(i)}{\sum_{j=1}^{K} p(\mathbf{x}|j)\ p(j)} \\
&= \frac{\mathcal{G}_{\mathbf{x}}[\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i]\ \pi_i}{\sum_{j=1}^{K} \mathcal{G}_{\mathbf{x}}[\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j]\ \pi_j}
\end{aligned}
\tag{3}
$$

where the parameters are from the previous iteration $t-1$. These are the responsibilities, discussed above, and we will denote them with $\mathbf{R}_{i,n}$ in the following. The M-step consists of maximizing Q over the parameters $\{\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \pi_i\}$, appearing in the joint density only. Taking derivatives and equating to zero gives,

$$
\begin{aligned}
\frac{\partial}{\partial \boldsymbol{\mu}_i} Q &= \sum_{n=1}^{N} \mathbf{R}_{i,n} \boldsymbol{\Sigma}_i^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_i) \Rightarrow \\
\boldsymbol{\mu}_i^{\mathbf{new}} &= \frac{\sum_{n=1}^{N} \mathbf{R}_{i,n} \mathbf{x}_n}{\sum_{n=1}^{N} \mathbf{R}_{i,n}}
\end{aligned}
\tag{4}
$$

$$
\begin{aligned}
\frac{\partial}{\partial \boldsymbol{\Sigma}_i^{-1}} Q &= \frac{1}{2} \sum_{n=1}^{N} \mathbf{R}_{i,n} \{\boldsymbol{\Sigma}_i - (\mathbf{x}_n - \boldsymbol{\mu}_i^{\mathbf{new}})(\mathbf{x}_n - \boldsymbol{\mu}_i^{\mathbf{new}})^T\} \Rightarrow \\
\boldsymbol{\Sigma}_i^{\mathbf{new}} &= \frac{\sum_{n=1}^{N} \mathbf{R}_{i,n} \mathbf{x}_n \mathbf{x}_n^T}{\sum_{n=1}^{N} \mathbf{R}_{i,n}} - \boldsymbol{\mu}_i^{\mathbf{new}} (\boldsymbol{\mu}_i^{\mathbf{new}})^T
\end{aligned}
\tag{5}
$$

We can think of

$$
\mathbf{w}_{in} = \frac{\mathbf{R}_{in}}{\sum_{n=1}^{M} \mathbf{R}_{in}}
\tag{6}
$$

as weights which weigh every datapoint with their "belonginess" to a certain cluster. Notice that in the case of one Gaussian the weights are all $\frac{1}{N}$ and we are back at the sample mean and covariance (see Estimation lecture). We could therefore think of the above quantities as weighted sample means and covariances.

For the mixing coefficients π_i we notice that they have to sum to one, $\sum_{i=1}^{K} \pi_i = 1$, which must be enforced with the use of a Lagrange multiplier term, $Q \to Q - \lambda(\sum_{i=1}^{K} \pi_i - 1)$. Taking derivatives and equating to one,

$$
\begin{aligned}
\frac{\partial}{\partial \pi_i} Q &= \sum_{n=1}^{N} \mathbf{R}_{i,n} \{\frac{1}{\pi_i}\} - \lambda \Rightarrow \\
\pi_i^{\mathbf{new}} &= \frac{1}{N} \sum_{n=1}^{N} \mathbf{R}_{i,n}
\end{aligned}
\tag{7}
$$

which fulfils the constraint since $\sum_{i=1}^{K} \mathbf{R}_{i,n} = 1 \ \ \forall\ n$ (the value of λ is determined by the constraint). This, then constitutes the M-step and alternating (3) with (4), (5) and (7), will converge to a ML soltion of the problem.

3 Vector Quantization through K-means

In the following we will take a particular limit of the MoG update rules and show that the resultant algorithm is a generalization of the K-means algorithm. First recal that the MoG model had the general structure,

$$
p(\mathbf{x}) = \sum_i p(\mathbf{x}, i) = \sum_i p(\mathbf{x}|i)\ p(i)
\tag{8}
$$

where we omitted dependence on the parameters $\boldsymbol{\theta}$ for convenience. In the following we will take $p(\mathbf{x}, i)$ to the power α (a positive factor), and study the limit were,

$$
p(\mathbf{x}, i)^{\alpha} \quad \alpha \to \infty
\tag{9}
$$

First let's study its effect on the posterior density,

$$p(i|\mathbf{x}) = \frac{p(\mathbf{x}, i)}{\sum_j p(\mathbf{x}, j)} \rightarrow p_\alpha(i|\mathbf{x}) = \frac{p(\mathbf{x}, i)^\alpha}{\sum_j p(\mathbf{x}, j)^\alpha} \tag{10}$$

So for the inverse we have,

$$\frac{1}{p_\alpha(i|\mathbf{x})} = \sum_j \left(\frac{p(\mathbf{x}, j)}{p(\mathbf{x}, i)}\right)^\alpha \qquad \alpha \rightarrow \infty \tag{11}$$

Now consider a cluster i, for which $p(\mathbf{x}, i)$ is larger than the corresponding value for all other clusters. This implies that all ratios $\frac{p(\mathbf{x},j)}{p(\mathbf{x},i)}$ for $i \neq j$ go to zero in the limit $\alpha \rightarrow \infty$. The only surviving term is the cluster $j = i$, which has a value 1. Summarizing, the posterior for the cluster for which $p(\mathbf{x}, i)$ is largest, has a posterior probability of one. Now consider a cluster k, such that there is at least one cluster i for which $p(\mathbf{x}, i)$ is larger. The ratio $\frac{p(\mathbf{x},i)}{p(\mathbf{x},k)}$ appearing in the sum is larger than one, which implies that it tends to infinity in the limit $\alpha \rightarrow \infty$. This implies in turn that its posterior probability must be zero. We may interpret this result as follows. The responsibilities of the clusters for the datapoints are *all or nothing*, i.e. a cluster has full responsibility for a datapoint, in which case the data point is assigned to that cluster, or no responsibility, in which case it is assigned to another cluster. At every E-step of EM a hard decision is made about the labeling. Notice that MoG allowed for "soft" assignments.

We will define a new objective function for any value of α, which is not a log-likelihood, but which can still be efficiently optimized using EM. Then we will take the limit $\alpha \rightarrow \infty$. Our objective will be,

$$L_\alpha(\mathbf{x}^N) = \frac{1}{\alpha} \sum_{n=1}^{N} \log \left[\sum_{i=1}^{K} p(\mathbf{x}_n, i)^\alpha\right] \tag{12}$$

The $\frac{1}{\alpha}$ will become clear in a moment when we take the limit. Now recall the comment we made in the EM lecture which stated that any costfunction of the sort,

$$L' = \sum_{n=1}^{N} \log\left[g(\mathbf{x}_n)\right], \tag{13}$$

with

$$g(\mathbf{x}) = \int d\mathbf{y} \; f(\mathbf{x}, \mathbf{y}) \tag{14}$$

could be optimized using EM, since as long as the quantity,

$$p(\mathbf{y}|\mathbf{x}) = \frac{f(\mathbf{x}, \mathbf{y})}{g(\mathbf{x})} = \frac{f(\mathbf{x}, \mathbf{y})}{\int d\mathbf{y} \; f(\mathbf{x}, \mathbf{y})} \tag{15}$$

is a probability density, which it automatically is as long as $f(\mathbf{x}, \mathbf{y})$ is positive everywhere. It is thus also easily seen that the posterior $p_\alpha(i|\mathbf{x})$ is indeed a probability function. We therefore conclude that the ususal EM-steps apply and maximize L_α. Then, let us take the limit $\alpha \rightarrow \infty$. First, we rewrite (see EM-lecture),

$$\begin{aligned}
L_\alpha(\mathbf{x}^N) &= \frac{1}{\alpha} \sum_{n=1}^{N} \log \left[\sum_{j=1}^{K} p(\mathbf{x}_n, j)^\alpha\right] \\
&= L_\alpha(\mathbf{x}^N) = \frac{1}{\alpha} \sum_{n=1}^{N} \sum_{i=1}^{K} p_\alpha(i|\mathbf{x}_n) \log \left[\sum_{j=1}^{K} p(\mathbf{x}_n, j)^\alpha\right] \\
&= L_\alpha(\mathbf{x}^N) = \frac{1}{\alpha} \sum_{n=1}^{N} \sum_{i=1}^{K} p_\alpha(i|\mathbf{x}_n) \log \left[\frac{p(\mathbf{x}_n, i)^\alpha}{p_\alpha(i|\mathbf{x}_n)}\right] \\
&= \frac{1}{\alpha} \sum_{n=1}^{N} \sum_{i=1}^{K} p_\alpha(i|\mathbf{x}_n) \; \log[p(\mathbf{x}_n, i)^\alpha] - \frac{1}{\alpha} \sum_{n=1}^{N} \sum_{i=1}^{K} p_\alpha(i|\mathbf{x}_n) \; \log[p_\alpha(i|\mathbf{x}_n)]
\end{aligned}$$

$$
\begin{aligned}
&\overset{\alpha>>1}{\approx} \quad \frac{1}{\alpha}\sum_{i=1}^{K}\sum_{n_i=1}^{N_i}\log[p(\mathbf{x}_{n_i},i)^{\alpha}] - \frac{1}{\alpha}\sum_{i=1}^{K}\sum_{n_i=1}^{N_i}\log[1] \\
&\overset{\alpha\to\infty}{\to} \quad \sum_{i=1}^{K}\sum_{n_i=1}^{N_i}\log[p(\mathbf{x}_{n_i},i)] \Rightarrow \\
L_{\infty}(\mathbf{x}^N) \quad &= \quad \sum_{i=1}^{K}\sum_{n_i=1}^{N_i}\log[p(\mathbf{x}_{n_i}|i,\boldsymbol{\theta})\ p(i|\boldsymbol{\theta})] \qquad (16)
\end{aligned}
$$

where we used $\sum_{i=1}^{K} p_{\alpha}(i|\mathbf{x}_n) = 1$ in going to the second line, while $p_{\alpha}(i|\mathbf{x}_n) = \frac{p(\mathbf{x}_n,i)^{\alpha}}{\sum_{j=1}^{K} p(\mathbf{x}_n,j)^{\alpha}}$ in going to the third line. Furthermore, one must use $\lim_{x\to 0} x\log[x] = 0$. Also, n_i is an index which labels the datapoints of cluster i, and N_i is the total number of points in that cluster. The double sum in the last line thus sums over all clusters, and sums over all samples inside that cluster.

The E-step thus consists of finding the optimal labels for every data point by choosing the cluster i for which $p(\mathbf{x}|i)\ p(i)$ is largest, and the M-step consists of,

$$
\begin{aligned}
\boldsymbol{\theta}_i^{\mathbf{new}} \quad &= \quad \mathbf{argmax}_{\boldsymbol{\theta}_i} \sum_{n_i=1}^{N_i}\log[p(\mathbf{x}_{n_i}|i,\boldsymbol{\theta}_i)] \qquad (17) \\
\pi_i^{\mathbf{new}} \quad &= \quad \mathbf{argmax}_{\pi_i}\ N_i\log[\pi_i] - \lambda(\sum_j \pi_j - 1) = \frac{N_i}{N} \qquad (18)
\end{aligned}
$$

where λ is a Lagrange multiplier. This is the most general form of weighted K-means and it consists of alternating the assignment rule and the parameter estimation step for every cluster. The fact that both steps do indeed maximize the cost function is quite intuitive. The assignment rule says that, given a fixed set of parameters, the best way to assign each data point is to assign it to the most likely cluster. That this maximizes L_{∞}, is easily seen by taking one datapoint and assigning it to another cluster, obviously, L_{∞} will decrease in that case. On the other hand, let's fix the assignments, then what are the optimal parameters that maximize L_{∞}? Again, since there is no overlap, the best we can do is maximize the parameters over the set of datapoints assigned to the cluster. The maximum likelihood estimate is then the optimal choice.

As an aside we mention that we actually have defined a whole sequence of clustering algorithms, one for each α. $\alpha = 1$ corresponds to the MoG model while $\alpha = \infty$ corresponds to K-means. It is also well known that K-means can often get stuck in local maxima, while MoG is more robust to this problem. This inspired people to start at an even lower value of $\alpha \to 0$ and perform EM. In turns out that for low values of α the optimization surface contains less local maxima, i.e. it is smoothed out. It is as if the we had added noise to the data. The solutions look fuzzier, i.e. the clusters are larger and may overlap more. If we want to avoid local maxima a good procedure is therefore to start at low values of α, perform EM until convergence, increase α by some small amount, perform EM again until convergence etc, until one has reached the desired value of α. This process is called deterministic annealing in the literature and is used to increase ones chances of ending up in the global maximum instead of some local maximum.

We will now discuss some special cases. First, let's choose Gaussians for the conditional densities,

$$
p(\mathbf{x}|i) = \mathcal{G}_{\mathbf{x}}[\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i] \qquad (19)
$$

The M-step now consists of,

$$
\begin{aligned}
\boldsymbol{\mu}_i^{\mathbf{new}} \quad &= \quad \frac{1}{N_i}\sum_{n_i=1}^{N_i}\mathbf{x}_{n_i} \qquad (20) \\
\boldsymbol{\Sigma}_i^{\mathbf{new}} \quad &= \quad \frac{1}{N_i}\sum_{n_i=1}^{N_i}\mathbf{x}_{n_i}\mathbf{x}_{n_i}^T - \boldsymbol{\mu}_i^{\mathbf{new}}(\boldsymbol{\mu}_i^{\mathbf{new}})^T \qquad (21) \\
\pi_i^{\mathbf{new}} \quad &= \quad \frac{N_i}{N} \qquad (22)
\end{aligned}
$$

while the E-step assigns data again to the Gaussian with the largest likelihood. This can be viewed as a "hard assignment variant" of the MoG update rules.

Next assume that the prior densities π_i are fixed. If one is larger than the other, this will bias the algorithm to assign more points to the cluster with the highest π_i, i.e. we can control the size of the clusters in this way. In the case when all π_i are equally large the term $\sum_i N_i \pi_i \to \frac{N}{K}$ and can be dropped from the objective function.

Finally, we can take a special case of this by assuming that the covariance matrix is the identity. We have then arrived to what is classically known as the K-means algorithm (k stands for number of clusters). In that case, the cluster with maximum likelihood for a datapoint, is also the cluster for which the Euclidean distance $\| \mathbf{x} - \boldsymbol{\mu}_i \|^2$ is smallest. Therefore the update rules are,

E-step Assign all datapoints to the clusters for which the Euclidean distance $\| \mathbf{x} - \boldsymbol{\mu}_i \|^2$ is smallest.

M-step Compute for every new cluster its sample mean, $\boldsymbol{\mu}_i^{\mathbf{new}} = \frac{1}{N_i} \sum_{n_i=1}^{N_i} \mathbf{x}_{n_i}$.

A brief look at the objective function confirms that,

$$L_\infty = -\frac{1}{2} \sum_{i=1}^{K} \sum_{n_i=1}^{N_i} \| \mathbf{x}_{n_i} - \boldsymbol{\mu}_i \|^2 \tag{23}$$

The $\boldsymbol{\mu}_i$ are sometimes interpreted as "prototypes" for a certain cluster. The objective is then to minimize the total square distance between the data and the prototypes. An easy way to understand that the above K-means algorithm does indeed minimize this objective is to imagine that every datapoint is connected with a "rod" to one of the prototypes. One should minimize the total rod-length. If the prototypes are fixed, one obviously needs to use the shortest rod connecting to the nearest prototype. If the assignments are fixed one must update the prototype as the sample mean of the data, which minimizes the square distances to all data in the cluster.