Clustering: K-means and Gaussian Mixture Models

Brian Azizi, Jia Guang Choo

Cavendish Laboratory, Department of Physics, J J Thomson Avenue, Cambridge. CB3 0HE

Clustering

We have an unlabelled training set of N data points, $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N)}\}$. Each data has D features, i.e. $\boldsymbol{x}^{(i)} \in \mathbb{R}^D$ for each i. We compactly represent the training data as an $N \times D$ matrix $\boldsymbol{X}$.

Our goal is to identify clusters in the data. Data points within the same cluster should be similar to each other and unlike data points in different clusters.

We will discuss two clustering models: K-means and GMM. Both models are usually trained using the *EM-algorithm*.

1. K-means

Defining the Model

Each of the K clusters can be characterized by its centre $\boldsymbol{\mu}_k \in \mathbb{R}^D, k = 1, \ldots, K$. Each point $\boldsymbol{x}^{(i)}$ has a corresponding binary vector $\boldsymbol{r}^{(i)} \in \mathbb{Z}_2^D$, where $r_k^{(i)} = 1$ if $\boldsymbol{x}^{(i)}$ belongs to cluster k, 0 otherwise. In general, we aim to minimize the cost function

$$C = \sum_{i=1}^{N} \sum_{k=1}^{K} r_k^{(i)} V(\boldsymbol{x}^{(i)}, \boldsymbol{\mu}_k) \tag{1}$$

where $V(\boldsymbol{x}^{(i)}, \boldsymbol{\mu}_k)$ measures the 'distance' between $\boldsymbol{x}^{(i)}$ and $\boldsymbol{\mu}_k$. For the purpose of this presentation, we let $V(\boldsymbol{x}^{(i)}, \boldsymbol{\mu}_k) = \|\boldsymbol{x}^{(i)} - \boldsymbol{\mu}_k\|^2$.

Minimizing the cost function

A local (and hopefully global) minimum for C is obtained by repeatedly minimizing C with respect to $\boldsymbol{r}^{(i)}$ while keeping $\boldsymbol{\mu}_k$ fixed, then doing the same with respect to $\boldsymbol{\mu}_k$ while keeping $\boldsymbol{r}^{(i)}$ fixed, until convergence.

For an initial set of $\boldsymbol{\mu}_k$, we minimize C with respect to $\boldsymbol{r}^{(i)}$ by setting

$$r_k^{(i)} = \begin{cases} 1 \text{ if } j = \arg\min_j \|\boldsymbol{x}^{(i)} - \boldsymbol{\mu}_j\|^2, \\ 0 \text{ otherwise.} \end{cases} \tag{2}$$

We then minimize C with respect to $\boldsymbol{\mu}_k$ by differentiating C with respect to $\boldsymbol{\mu}_k$ to get

$$\frac{\partial C}{\partial \boldsymbol{\mu}_k} = 2 \sum_{i=1}^{N} r_k^{(i)} (\boldsymbol{x}^{(i)} - \boldsymbol{\mu}_k) = 0 \Rightarrow \boldsymbol{\mu}_k = \frac{\sum_{i=1}^{N} r_k^{(i)} \boldsymbol{x}^{(i)}}{\sum_{i=1}^{N} r_k^{(i)}}. \tag{3}$$

Applications in image compression

Suppose we have an image with N pixels, each comprising of 3 numbers representing the intensity of red, blue and green, and each of these numbers is stored with 8 bits (i.e. values range from 0 to 255). Hence, the total number of bits required to store the image directly would be $24N$.

Now suppose the K-means algorithm is applied, where the features are the values of red, blue and green. If there are K clusters, then only the K vectors corresponding to $\boldsymbol{\mu}_k$ are stored. In addition, the cluster that each pixel belongs to can also be stored in $\log_2 K$ bits. Hence the total number of bits required is $24K + N \log_2 K$, which is much smaller than $24N$.

2. Gaussian Mixture Model

Defining the Model

In latent variable models, we model the probability density function (pdf) of the data as

$$p(\boldsymbol{x}) = \sum_{z} p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z}) \tag{4}$$

The random variable $\boldsymbol{x}$ represents our data and the random variable $\boldsymbol{z}$ is a so called *latent* variable, i.e. it is not observed.

In mixture models, $\boldsymbol{x}$ is generated from one of K possible base distributions and $\boldsymbol{z}$ tells us from which. Here, K is analogous to number of clusters K in K-means,. So, to get the required behaviour, $\boldsymbol{z}$ has a categorical nature. We generally use the so-called 1-of-K representation of $\boldsymbol{z}$, where

$$\boldsymbol{z} \in \{0, 1\}^K, \qquad \sum_{k=1}^{K} z_k = 1 \tag{5}$$

For example, if $K = 3$ and for a particular point $\boldsymbol{x}^{(i)}$ we have $\boldsymbol{z}^{(i)} = (0, 1, 0)$, then this means that $\boldsymbol{x}^{(i)}$ came from the 2nd base distribution (think: 2nd cluster).

Of course, we don't know $\boldsymbol{z}$, since it is a latent variable. So we define a distribution for it:

$$p(z_k = 1) = \pi_k \tag{6}$$

or equivalently

$$p(\boldsymbol{z}) = \prod_{k=1}^{K} \pi_k^{z_k} \tag{7}$$

Equation (4) then gives:

$$\begin{aligned} p(\boldsymbol{x}) &= \sum_{z} p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z}) \\ &= \sum_{z} p(\boldsymbol{x}|\boldsymbol{z}) \prod_{k=1}^{K} \pi_k^{z_k} \\ &= \sum_{k=1}^{K} \pi_k p(\boldsymbol{x}|z_k = 1) \end{aligned} \tag{8}$$

Now we have general mixture models of K base distribution. In *Gaussian Mixture Models*, each base distribution is a multivariate Gaussian. The kth base distribution has parameters $(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ and has pdf

$$\mathcal{N}(\boldsymbol{x} \,|\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}_k|^{1/2}} \exp\{-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_k)\} \tag{9}$$

The final Gaussian mixture model is then

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x} \,|\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{10}$$

Training the model

To train the model, i.e. find the parameters $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ for $k = 1, \ldots, K$, we need a set of training data $\boldsymbol{X} \in \mathbb{R}^{N \times D}$. We assume that the samples were idenpendently generated from our model, which allows us to write the likelihood function as

$$p(\boldsymbol{X} \,|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{i=1}^{N} \sum_{k=1}^{K} pi_k \mathcal{N}(\boldsymbol{x}^{(i)} \,|\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k). \tag{11}$$

We maximize the (log) likelihood to get the parameters. The solution can be expressed as

$$\begin{aligned}
\boldsymbol{\mu}_k &= \frac{1}{N_k} \sum_{i=1}^{N} \gamma(z_k^{(i)}) \boldsymbol{x}^{(i)}, \\
\boldsymbol{\Sigma}_k &= \frac{1}{N_k} \sum_{i=1}^{N} \gamma(z_k^{(i)}) (\boldsymbol{x}^{(i)} - \boldsymbol{\mu}_k)(\boldsymbol{x}^{(i)} - \boldsymbol{\mu}_k)^T, \\
\pi_k &= \frac{N_k}{N}, \\
N_k &= \sum_{i=1}^{N} \gamma(z_k^{(i)}), \\
\gamma(z_k^{(i)}) &= \frac{\pi_k \mathcal{N}(\boldsymbol{x}^{(i)} \,|\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\boldsymbol{x}^{(i)} \,|\, \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_k)}.
\end{aligned} \tag{12}$$

The quantity $\gamma(z_k^{(i)})$ is referred to as the *responsibility* that cluster k takes in explaining sample $\boldsymbol{x}^{(i)}$, while N_k is the effective number of data points in cluster k.

Note that this is not a closed-form solution since the quantities depend on each other. However, we can try to find the solution iteratively as follows: First, randomly initialize the quantities $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$ and π_k. Then calculate $\gamma(z_k^{(i)})$ and N_k. Then calculate $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$ and π_k and so on until convergence. This iterative scheme is known as the *Expectation-Maximization algorithm* for the Gaussian mixture model.

For more information and details on the derivation, see [1] [2].

References

[1] Georgios Pilikos, Unsupervised Learning, Lecture Notes in Machine Learning, Chapter 2, 2015

[2] Christopher Bishop, Pattern Recognition and Machine Learning, Chapter 9, 2015, Springer