CS229 Lecture notes

原作者：[Andrew Ng](http://cs229.stanford.edu/)（[吴恩达](http://open.163.com/movie/2008/1/M/C/M6SGF6VB4_M6SGHFBMC.html)）

翻译：[CycleUser](https://zhuanlan.zhihu.com/python-kivy)

# 混用高斯 (Gaussians) 和最大期望算法(the EM algorithm)

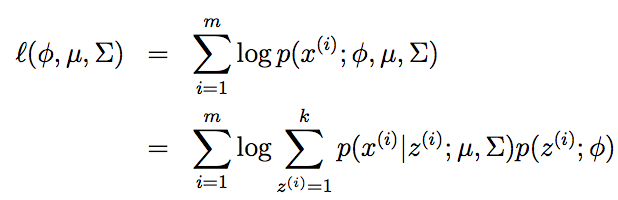
在本章讲义中，我们要讲的是使用期望最大化算法（EM，Expectation-Maximization）来进行密度估计（density estimation）。

一如既往，还是假设我们得到了某一个训练样本集 {x(1), ... , x(m)}。由于这次是非监督学习（unsupervised learning）环境，所以这些样本就没有什么分类标签了。

We wish to model the data by specifying a joint distribution p(x(i), z(i)) = p(x(i)|z(i))p(z(i)). Here, z(i) ∼ Multinomial(φ) (where φj ≥ 0, Σ kj=1 φj = 1, and the parameter φj gives p(z(i) = j),), and x(i)|z(i) = j ∼ N(μj,Σj). We let k denote the number of values that the z(i)’s can take on. Thus, our model posits that each x(i) was generated by randomly choosing z(i) from {1, ..., k}, and then x(i) was drawn from one of k Gaussians depending on z(i). This is called the mixture of Gaussians model. Also, note that the z(i)’s are latent random variables, meaning that they’re hidden/unobserved. This is what will make our estimation problem difficult.

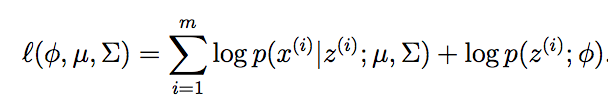
The parameters of our model are thus φ, μ and Σ. To estimate them, we can write down the likelihood of our data:

我们希望能够获得一个联合分布 p(x(i), z(i)) = p(x(i)|z(i))p(z(i)) 来对数据进行建模。其中的 z(i) ∼ Multinomial(φ) (即 z(i) 是一个以 φ 为参数的多项式分布，其中 φj ≥ 0, Σ kj=1 φj = 1，而参数 φj 给出了 p(z(i) = j))，另外 x(i)|z(i) = j ∼ N(μj,Σj) （一个以 μj 和 Σj 为参数的正态分布）。我们设 k 来表示 z(i) 能取的值的个数。

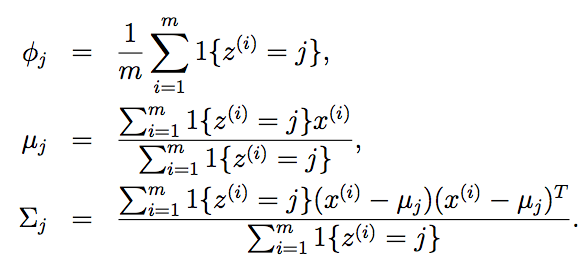


However, if we set to zero the derivatives of this formula with respect to the parameters and try to solve, we’ll find that it is not possible to find the maximum likelihood estimates of the parameters in closed form. (Try this yourself at home.)

The random variables z(i) indicate which of the k Gaussians each x(i) had come from. Note that if we knew what the z(i)’s were, the maximum likelihood problem would have been easy. Specifically, we could then write down the likelihood as



Maximizing this with respect to φ, μ and Σ gives the parameters:





Indeed, we see that if the z(i)’s were known, then maximum likelihood estimation becomes nearly identical to what we had when estimating the parameters of the Gaussian discriminant analysis model, except that here the z(i)’s playing the role of the class labels.1

However, in our density estimation problem, the z(i)’s are not known. What can we do?

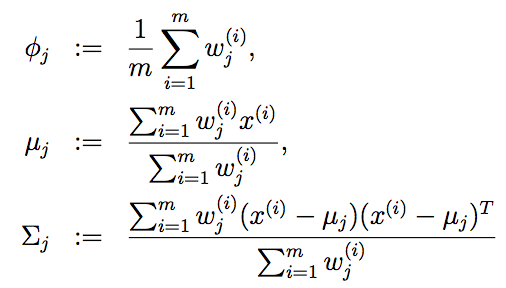
The EM algorithm is an iterative algorithm that has two main steps. Applied to our problem, in the E-step, it tries to “guess” the values of the z(i)’s. In the M-step, it updates the parameters of our model based on our guesses. Since in the M-step we are pretending that the guesses in the first part were correct, the maximization becomes easy. Here’s the algorithm:

Repeat until convergence: {

(E-step) For each i, j, set



(M-step) Update the parameters:

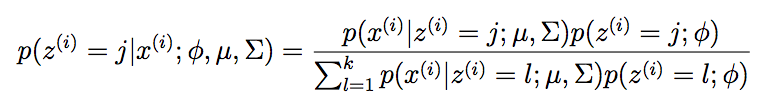


}

1There are other minor differences in the formulas here from what we’d obtained in PS1 with Gaussian discriminant analysis, first because we’ve generalized the z(i)’s to be multinomial rather than Bernoulli, and second because here we are using a different Σj for each Gaussian.



In the E-step, we calculate the posterior probability of our parameters the z(i)’s, given the x(i) and using the current setting of our parameters. I.e., using Bayes rule, we obtain:



Here, p(x(i)|z(i) = j;μ,Σ) is given by evaluating the density of a Gaussian with mean μj and covariance Σj at x(i); p(z(i) = j;φ) is given by φj, and so on. The values w(i) calculated in the E-step represent our “soft” guesses2 for j the values of z(i). Also, you should contrast the updates in the M-step with the formulas we had when the z(i)’s were known exactly. They are identical, except that instead of the indicator functions “1{z(i) = j}” indicating from which Gaussian each datapoint had come, we now instead have the w(i)’s. j The EM-algorithm is also reminiscent of the K-means clustering algorithm, except that instead of the “hard” cluster assignments c(i), we instead have the “soft” assignments w(i). Similar to K-means, it is also susceptible j to local optima, so reinitializing at several different initial parameters may be a good idea.

It’s clear that the EM algorithm has a very natural interpretation of repeatedly trying to guess the unknown z(i)’s; but how did it come about, and can we make any guarantees about it, such as regarding its convergence? In the next set of notes, we will describe a more general view of EM, one that will allow us to easily apply it to other estimation problems in which there are also latent variables, and which will allow us to give a convergence guarantee.

2The term “soft” refers to our guesses being probabilities and taking values in [0, 1]; in contrast, a “hard” guess is one that represents a single best guess (such as taking values in {0,1} or {1, ..., k}).