RE: K-means and multivariate GMM algorithm implementations (2d and 3d)

Fu Amos <fs1984@msn.com>

Tue 2016-05-03 4:37 PM

To:Nizar Bouguila <nizar.bouguila@concordia.ca>;

Dear Doctor,
After reading the chapter 9 and 10 of bishop's <pattern recognition and machine learning> , wikipedia and some other documents, I got some ideas about the usage of maximum likelihood function in EM algorithm.

First of all, for an **independent Gaussian mixture module**, the likelihood proof can be found here : http://www.statlect.com/fundamentals-of-statistics/normal-distribution-maximum-likelihood. The main idea is:

- calculate the maximum likelihood function

$$\mathcal{L}(\theta;\, x_1, \ldots, x_n) = f(x_1, x_2, \ldots, x_n \mid \theta) = \prod_{i=1}^{n} f(x_i \mid \theta).$$

- Then calculate the log likelihood function:

$$\ln \mathcal{L}(\theta;\, x_1, \ldots, x_n) = \sum_{i=1}^{n} \ln f(x_i \mid \theta),$$

- Calculate maximum likelihood estimators by calculate the partial derivatives for every unknown parameters and assume the expression equals to zero to maximize the parameters:

$$\widehat{\mu}_n = \frac{1}{n}\sum_{j-1}^{n} x_j$$

$$\widehat{\sigma}_n^2 = \frac{1}{n}\sum_{j-1}^{n} (x_j - \widehat{\mu})^2$$

But for the **Gaussian mixture module**, the situation is more complex since we have more than one Gaussian distributions and need to also use Bayes rules to calculate priors and posteriors for every data point:

- Likelihood function:

$$L(\theta; \mathbf{x}) = \prod_{i=1}^{n} \sum_{j=1}^{2} \tau_j \; f(\mathbf{x}_i; \boldsymbol{\mu}_j, \Sigma_j)$$

- E-step:
  1. using Bayes rules to calculate current **posteriors**:

$$T_{j,i}^{(t)} := \mathrm{P}(Z_i = j | X_i = \mathbf{x}_i; \theta^{(t)}) = \frac{\tau_j^{(t)} \; f(\mathbf{x}_i; \boldsymbol{\mu}_j^{(t)}, \Sigma_j^{(t)})}{\tau_1^{(t)} \; f(\mathbf{x}_i; \boldsymbol{\mu}_1^{(t)}, \Sigma_1^{(t)}) + \tau_2^{(t)} \; f(\mathbf{x}_i; \boldsymbol{\mu}_2^{(t)}, \Sigma_2^{(t)})}.$$

  2. calculate log likilihood function for unlnown parameters:

$$\begin{aligned}
Q(\theta|\theta^{(t)}) &= \mathrm{E}_{\mathbf{Z}|\mathbf{X},\theta^{(t)}}[\log L(\theta; \mathbf{x}, \mathbf{Z})] \\
&= \mathrm{E}_{\mathbf{Z}|\mathbf{X},\theta^{(t)}}[\log \prod_{i=1}^{n} L(\theta; \mathbf{x}_i, \mathbf{z}_i)] \\
&= \mathrm{E}_{\mathbf{Z}|\mathbf{X},\theta^{(t)}}[\sum_{i=1}^{n} \log L(\theta; \mathbf{x}_i, \mathbf{z}_i)] \\
&= \sum_{i=1}^{n} \mathrm{E}_{\mathbf{Z}|\mathbf{X};\theta^{(t)}}[\log L(\theta; \mathbf{x}_i, \mathbf{z}_i)] \\
&= \sum_{i=1}^{n} \sum_{j=1}^{2} P(Z_i = j | X_i = \mathbf{x}_i; \theta^{(t)}) \log L(\theta_j; \mathbf{x}_i, \mathbf{z}_i) \\
&= \sum_{i=1}^{n} \sum_{j=1}^{2} T_{j,i}^{(t)} \left[\log \tau_j - \tfrac{1}{2} \log |\Sigma_j| - \tfrac{1}{2} (\mathbf{x}_i - \boldsymbol{\mu}_j)^\top \Sigma_j^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_j) - \tfrac{d}{2} \log(2\pi) \right]
\end{aligned}$$

- M-step

- Calculate new **priors** based on old priors using log likelihood function by calculate the partial derivative of it:

$$\begin{aligned}\boldsymbol{\tau}^{(t+1)} &= \arg\max_{\boldsymbol{\tau}} Q(\theta|\theta^{(t)}) \\ &= \arg\max_{\boldsymbol{\tau}} \left\{ \left[\sum_{i=1}^{n} T_{1,i}^{(t)}\right] \log \tau_1 + \left[\sum_{i=1}^{n} T_{2,i}^{(t)}\right] \log \tau_2 \right\}\end{aligned}$$

This has the same form as the MLE for the binomial distribution, so

$$\tau_j^{(t+1)} = \frac{\sum_{i=1}^{n} T_{j,i}^{(t)}}{\sum_{i=1}^{n}(T_{1,i}^{(t)} + T_{2,i}^{(t)})} = \frac{1}{n}\sum_{i=1}^{n} T_{j,i}^{(t)}.$$

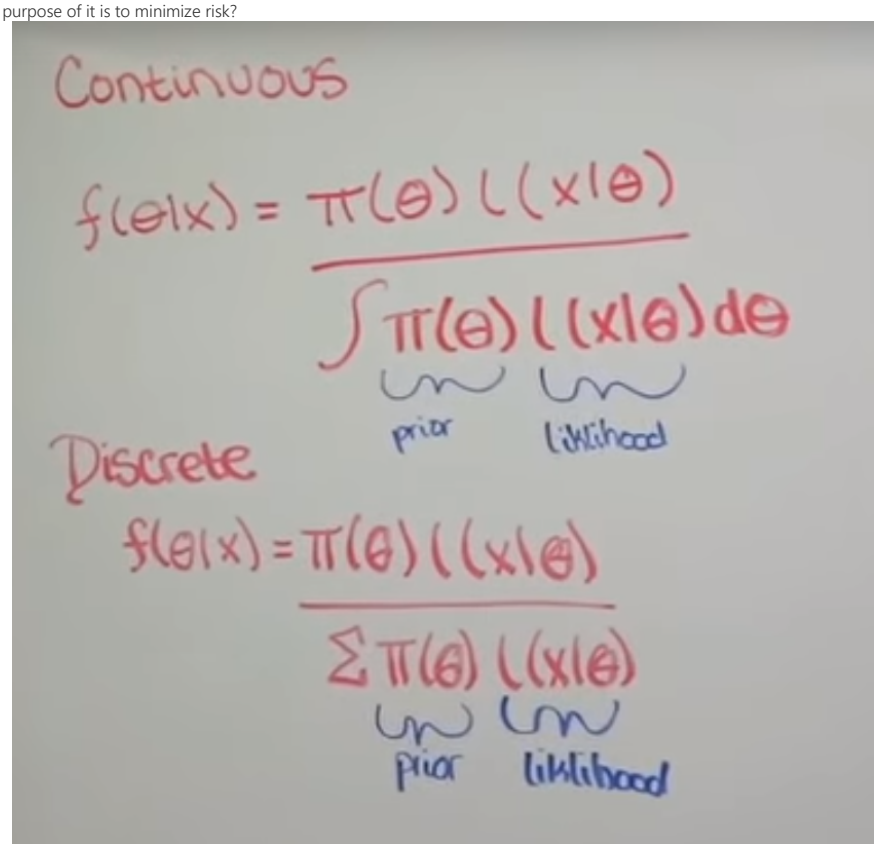
- Also maximize **mu and sigma** following the same procedure as priors:

$$\begin{aligned}(\boldsymbol{\mu}_1^{(t+1)}, \Sigma_1^{(t+1)}) &= \arg\max_{\boldsymbol{\mu}_1,\Sigma_1} Q(\theta|\theta^{(t)}) \\ &= \arg\max_{\boldsymbol{\mu}_1,\Sigma_1} \sum_{i=1}^{n} T_{1,i}^{(t)} \left\{-\tfrac{1}{2}\log|\Sigma_1| - \tfrac{1}{2}(\mathbf{x}_i - \boldsymbol{\mu}_1)^\top \Sigma_1^{-1}(\mathbf{x}_i - \boldsymbol{\mu}_1)\right\}\end{aligned}$$

This has the same form as a weighted MLE for a normal distribution, so

$$\boldsymbol{\mu}_1^{(t+1)} = \frac{\sum_{i=1}^{n} T_{1,i}^{(t)} \mathbf{x}_i}{\sum_{i=1}^{n} T_{1,i}^{(t)}} \text{ and } \Sigma_1^{(t+1)} = \frac{\sum_{i=1}^{n} T_{1,i}^{(t)} (\mathbf{x}_i - \boldsymbol{\mu}_1^{(t+1)})(\mathbf{x}_i - \boldsymbol{\mu}_1^{(t+1)})^\top}{\sum_{i=1}^{n} T_{1,i}^{(t)}}$$

Also for Bayesian estimation, I read some papers and materials but I'm not totally understand the usage of it, it seems like a method to estimate probability using likelihood function and priors. The purpose of it is to minimize risk?



Looking forward to talk with you about it further more.

Thanks,
Shuai Fu

From: nizar.bouguila@concordia.ca
To: fs1984@msn.com
Subject: Re: K-means and multivariate GMM algorithm implementations (2d and 3d)
Date: Fri, 22 Apr 2016 19:09:19 +0000

No it is different from Bayesian estimation. We will discuss it later.

Now you need to understand the derivations of the likelihood function to understand how we got the equations in the EXPECTATION and MAXIMIZATION steps.

Nizar Bouguila, Associate Professor, PhD, PEng
nizar.bouguila@concordia.ca
Phone: 514-848 2424 ext. 5663

Mailing Address

Concordia Institute for Information Systems Engineering,
1455 de Maisonneuve Blvd. West, EV7.632
Montréal, Québec,
Canada, H3G 1M8

Physical Address:

Office EV-007-632
Concordia Institute for Information Systems Engineering
Concordia University
1515 St.Catherine Street West, EV.007.632
H3G 2W1, Montreal, Canada

From: Fu Amos <fs1984@msn.com>
Sent: April 22, 2016 3:06 PM
To: Nizar Bouguila
Subject: RE: K-means and multivariate GMM algorithm implementations (2d and 3d)

Dear Doctor,
From best of my understanding, I'm using EM algorithm which should be **based on Bayesian estimation?**

- Membership weight:
-
- Mean and Sigma

$$\ln \mathcal{L}(\theta; x_1, \ldots, x_n) = \sum_{i=1}^{n} \ln f(x_i \mid \theta),$$

-

About Maximization likelihood, I read some article like https://en.wikipedia.org/wiki/Maximum_likelihood , I'm not sure that I could understand all the details about it because there are too many math derivations behind it but from me, it's similar to EM, just without the posterior.

If my understanding is wrong please correct me, thank you!

BTW: I followed the concepts about the paper attached to implement the EM with GMM.

BRs,
Shuai Fu

From: nizar.bouguila@concordia.ca
To: fs1984@msn.com
Subject: Re: K-means and multivariate GMM algorithm implementations (2d and 3d)
Date: Fri, 22 Apr 2016 17:38:28 +0000

Great.

Are you able to develop by yourself the estimation equations of the mean and variance by calculating the derivative of the log likelihood?

If yes great, we will move to the next step (Bayesian estimation). If not try to do it and then let me know.

Nizar Bouguila, Associate Professor, PhD, PEng
nizar.bouguila@concordia.ca
Phone: 514-848 2424 ext. 5663

Mailing Address

Concordia Institute for Information Systems Engineering,
1455 de Maisonneuve Blvd. West, EV7.632
Montréal, Québec,
Canada, H3G 1M8

Physical Address:

Office EV-007-632
Concordia Institute for Information Systems Engineering
Concordia University
1515 St.Catherine Street West, EV.007.632
H3G 2W1, Montreal, Canada

From: Fu Amos <fs1984@msn.com>
Sent: April 22, 2016 10:37 AM
To: Nizar Bouguila
Subject: K-means and multivariate GMM algorithm implementations (2d and 3d)

Dear Doctor,

Thank you for your information about EM algorithm, attached is Matlab source code of K-means and Gaussian mixture model and clustering result snapshots.

- Kmeans.m and GMM.m are the main scripts of the two algorithms and
  - k: component number
  - num: observation number in randomly generated .csv file.
  - dimension: valid value is 2 or 3
- Remaining problems
  - **Exit condition of GMM algorithm**: For now there is no exit condition to control the loop but using **a fix number of iteration**, regarding to some documents, they claim that it could be better if using a threshold. (**stability of mu or sigma? or estimate the posterior of all observations?**)
  - **Calculate precision**: During the implementation of multivariate GMM algorithm, I noticed the **posteriors of some observations are all zero**, that means the observation doesn't belong to any of the Gaussians which is abnormal. Then I found an interesting phenomenon that the Gaussian function returns 0 probability for those data observations since the probability is too small (saying the locations of those observations are too far away from all the components). In this case, the expression: probability = amplitude*(**exp(-exponent)**) returns o because exponent is too large that the exp() cannot handle. I consulted some documents but cannot find a perfect way to solve this issue ,could you give me some guide?
  - **Observation set**: Test data (observations) should be real data which will be better than random data.

Thank you,
Shuai Fu