Expectation Maximization Algorithm

1.Theory

Maximum likelihood estimation is an approach to density estimation for a dataset by searching across probability distributions and their parameters. It is a general and effective approach that underlies many machine learning algorithms like , although it requires that the training dataset is complete, e.g. all relevant interacting random variables are present. Maximum likelihood becomes intractable if there are variables that interact with those in the dataset but were hidden or not observed, so-called latent variables. The expectation maximization algorithm enables parameter estimation in probabilistic models with incomplete data i.e. data with latent variables. It does this in two steps:

E-step: Estimating the values for the latent variables by guessing a probability distribution over completions of missing data given the current model.

M-step: Re-estimating the model parameters using these completions because model re-estimation can be thought of as 'maximization' of the expected log-likelihood of the data.

Then repeating these two steps until convergence. It is an effective and general approach and is most commonly used for density estimation with missing data. Following figure illustrate one iteration of EM algorithm.

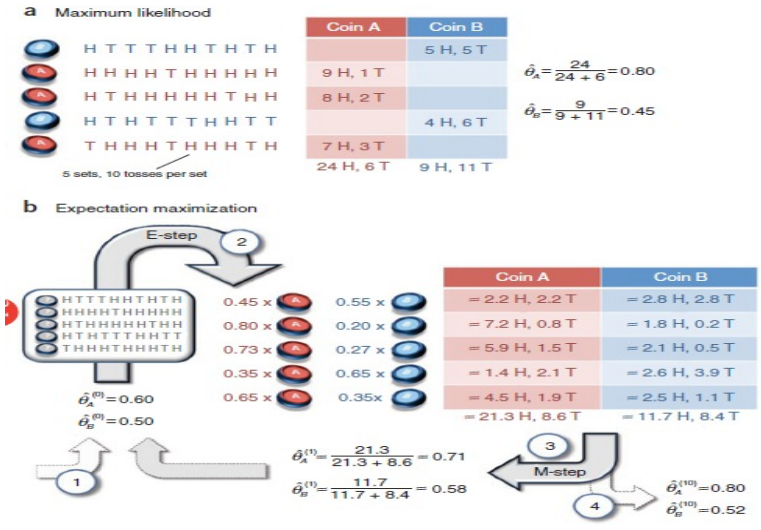

Figure 1 Parameter estimation for complete and incomplete data. (a) Maximum likelihood estimation. For each set of ten tosses, the maximum likelihood procedure accumulates the counts of heads and tails for coins A and B separately. These counts are then used to estimate the coin biases. (b) Expectation maximization. 1. EM starts with an initial guess of the parameters. 2. In the E-step, a probability distribution over possible completions is computed using the current parameters. The counts shown in the table are the expected numbers of heads and tails according to this distribution. 3. In the M-step, new parameters are determined using the current completions. 4. After several repetitions of the E-step and M-step, the algorithm converges.

2.Implementation

Implementation of the EM is challenging, especially when developing from scratch without using any built-in libraries.

Initialization of model parameter is crucial here. We can choose any value in the range of 0 to 1 except the case when θ_a equals θ_b. When both the model parameter is equal, then

it's a perfect symmetry and EM algorithms can't break the symmetry and will be lost in some local minima or flat error surface.

Laplace smoothing is carried out if somehow, the number of heads or tails happened to be 0. But here, I didn't apply the smoothing because the data coming from API is reliable and after many experiment, I didn't find any such case. So, to keep the implementation clean, I shunned smoothing.

The stopping point of algorithms i.e. when algorithm converges, the model parameter does not show variation. After running the experiment for than 100 times with different number of epochs experiment shows model parameter, converges in 10 epochs.

3. Results

After the running algorithms more than 100 times, the model parameter converges as follows: the probability of head when flipping coin A, $\theta\_a = 0.71$ and the probability of head when flipping coin B, $\theta\_b = 0.30$. In the implementation, I avoided taking average of the model parameter to reduce execution time. Following figure displays the convergence of EM Algorithms.