Coin head/tail probability estimation using EM algorithm

Selection of features:

The initial data is in terms of ones and zeros. Let's consider a sequence - [1,0,0,1,1,0,1,0,1,0,0,1,0,0,0,1,0,1,1,0] which contains 9 heads. Another sequence - [1,0,1,0,1,0,0,1,0,0,0,1,0,1,1,0,1,0,0,1] – 9 heads. Despite both of these sequences being the same, they have the same probability of head that is 45%. Thus the sequence by itself doesn't give up much information. Since every flip in a draw is an independent event, we can trim the data to contain actual probabilities of heads.

So I had two options, either to just select the probability of heads as a single feature or select heads and P(1 – heads) = P(tails) as two features. Apparently, both methods on convergence yield the same result just represented differently.

Observations:

In the 2nd case, for both coins, the value of the 2nd dimension for mean is always 1 – a value of 1st dimension. The covariance is just a constant multiplied into [[1, -1] ; [-1, 1]]. This basically shows the correlation between head and tails. Head is inversely proportional to tails

Algorithm:

EM requires initial estimates to work with. I randomly initialized the Theta parameters from the uniform normal distribution. Weights can be randomly assigned or one weight can be given a value total probability of heads from all flips of all draws with the other weight being equal to the remaining probability

I used the Gaussian pdf function to model the normal distribution assuming that the probabilities follow a normal distribution

There are two steps in the algorithm:

E step

Expectation step where you find the p(x|y) -> gamma, where x is a sample and y is the particular class

M step

Maximization step. Once we know the class membership distribution over the classes, we start to find the new mean, deviations/covariances, and class contributions (weights)

We do these steps repeatedly until the algorithm converges.

To check convergence, we check the difference between maximum likelihood estimation values from the current and previous steps. If the difference is below a certain value, we stop the algorithm

Results:

Overrunning the EM algorithm 300 times, Theta parameters on average are as follows:

Means = [0.32, 0.67] -> chances of ‘head’, Sigmas = [0.014, 0.014] -> deviation from probability of yielding a ‘head’. The weights cannot be averaged over multiple runs since the number of draws from a certain coin is much more volatile than the other theta parameters. Although, the weights do range from 0.2 to 0.7 for one coin and inversely for the other coin

References:

1) https://haipeng-luo.net/courses/CSCI567/2021_fall/lec8.pdf
2) https://towardsdatascience.com/implement-expectation-maximization-em-algorithm-in-python-from-scratch-f1278d1b9137