Two Coin Flipping With EM Algorithm - Technical Report

Nguyen Cong Dat
ndat@usc.edu

1 Introduction

This document provides a detail report on an implementation using EM algorithm to estimate the theta parameters of the two coins flipping problem. The rest of this paper is organized as follows. Section 2 provides background and pseudo-code of EM algorithm. Section 3 describes my options when implementing the algorithm and Section 4 shows results of tuning those options. The Main pipeline and final result are given in Section 5. An estimation of the two thetas is: **[0.7, 0.3]**.

2 EM Algorithm

From description in (Do and Batzoglou, 2008), a pseudo-code of the algorithm can be written as below:

Algorithm 1 Simple EM algorithm

Require: dataset X
Initialize $\theta = \{\theta_A, \theta_B\}$
while not converged **do**
  for X_i in X **do** ▷ E-Step
    Count num_head, num_tail
    $weight_A \leftarrow \frac{P(X_i|Coin=A,\theta)}{P(X_i|\theta)}$
    $weight_B \leftarrow 1 - weight_A$
    $head_A += weight_A \times num\_head$
    $tail_A += weight_A \times num\_tail$
    $head_B += weight_B \times num\_head$
    $tail_B += weight_B \times num\_tail$
  end for
  Update θ: ▷ M-Step
    $\theta_A \leftarrow \frac{head_A}{head_A + tail_A}$
    $\theta_B \leftarrow \frac{head_B}{head_B + tail_B}$
end while

3 Extended Options

To use the above algorithm to solve our estimation, there are some hidden criterion that need to be clarified. Two of them are the initialization of θ and the conditions to coverage.
For initialization, we can test methods including:

1. Random initialization: randomly choose $\{\theta_A, \theta_B\}$.
2. Greedy initialization: choose $\{\theta_A, \theta_B\}$ from complete assignments in a small batch of data.
3. Polar initialization: choose $\{\theta_A, \theta_B\}$ = {0.9, 0.1}
4. Equal initialization: choose $\{\theta_A, \theta_B\}$ = {0.5, 0.5}

For converging conditions, we need to consider:

1. Converging threshold.
2. Number of unchanged steps before determining convergence.

Besides, to accelerate the running time, I picked Numpy to make use of vectorized implementation instead of running the for loop step-by-step.

4 Experiment

The experiment was performed to make hyper-parameter tuning for above options.
Dataset: Because the API provides only non-labeled data, I created a simple dataset including 30 draws and 20 flips in each draws for each trial set of the parameters.
Method: A comparison was made based on the averaged error of 10 trials in each set of the options.
Result: The setting {*greedy method,* e^{-10} *converging threshold, 20 converging steps*} is often ranked among the best three options. Therefore, this setting is chosen in the main pipeline.

5 Main Pipeline and Estimated Thetas

Main pipeline: With tuned options, the process is formed into 3 steps: 1. Crawling data from API, 2. Estimating $\{\theta_A, \theta_B\}$, 3. Return the value. Detail implementation is coded in *main.py*.
Final Value: Although the estimation of θs is dependant on each run due to the data difference, the commonly observed value is approximated to **[0.7, 0.3]**. It is my final estimation in this project.

References

Chuong B Do and Serafim Batzoglou. 2008. What is the expectation maximization algorithm? *Nature biotechnology*, 26(8):897–899.