Comp 543 HW4-EM

Kai-Po Lin

【Description】In this assignment, you are asked to actually implement the EM algorithm derived in class. Your implementation will be using Python. Recall that the basic setup was that we imagine that there are two coins in a bag. Repeatedly, we pick one out and flip it 10 times, then put it back in. We derived an algorithm to look at all of the sequences of 10 flips, and figure out the probability that each coin comes up heads.

【Code】

```
import numpy as np
import scipy.stats

def EstimateCoinHeadCnt(flips):
    # One coin has a probability of coming up heads of 0.2, the other 0.6
    coinProbs = np.zeros(2)
    coinProbs[0] = 0.2
    coinProbs[1] = 0.6

    # Reach in and pull out a coin numTimes times
    numTimes = 100

    # Flip it numFlips times when you do
    numFlips = flips

    # Flips will have the number of heads we observed in 10 flips for each coin
    flips = np.zeros(numTimes)

    for coin in range(numTimes):
        # which is a one-dimensional numpy array
        which = np.random.binomial(1, 0.5, 1);
        # How many times did the unknown coin show head when I flipped it 10 times
        flips[coin] = np.random.binomial(numFlips, coinProbs[which], 1);


    # Initialize the EM algorithm
    coinProbs[0] = 0.79
    coinProbs[1] = 0.51

    # Run the EM algorithm
```

```
    for iters in range (20):
        estimateHeadCntA = 0
        estimateHeadCntB = 0
        estimateTailCntA = 0
        estimateTailCntB = 0

        for flipIdx in range(numTimes):
            headCnt = flips[flipIdx]
            tailCnt = numFlips - headCnt
            coinA = (coinProbs[0] ** headCnt) * ((1 - coinProbs[0]) ** tailCnt)
            coinB = (coinProbs[1] ** headCnt) * ((1 - coinProbs[1]) ** tailCnt)
            coinAProb = coinA / (coinA + coinB)
            #coinBProb[flipIdx] = coinB / (coinA + coinB)
            coinBProb = 1 - coinAProb

            estimateHeadCntA += coinAProb * headCnt
            estimateTailCntA += coinAProb * tailCnt
            estimateHeadCntB += coinBProb * headCnt
            estimateTailCntB += coinBProb * tailCnt

        coinProbs[0] = estimateHeadCntA / (estimateHeadCntA + estimateTailCntA)
        coinProbs[1] = estimateHeadCntB / (estimateHeadCntB + estimateTailCntB)
        print(coinProbs)


if __name__ == '__main__':
    print("numFlips: 10")
    EstimateCoinHeadCnt(10)
    print("\nnumFlips: 2")
    EstimateCoinHeadCnt(2)
```

【Result】

```
numFlips: 10
[0.71175162 0.30662754]
[0.66245393 0.24313347]
[0.63328033 0.21678924]
[0.61752958 0.20463265]
[0.60925177 0.19871663]
[0.60494866 0.19575105]
[0.60272153 0.19424209]
[0.6015708  0.19346871]
```

[0.60097661 0.19307092]
[0.60066987 0.19286597]
[0.60051155 0.19276028]
[0.60042983 0.19270576]
[0.60038765 0.19267762]
[0.60036588 0.1926631]
[0.60035464 0.19265561]
[0.60034884 0.19265174]
[0.60034585 0.19264975]
[0.6003443 0.19264872]
[0.60034351 0.19264819]
[0.60034309 0.19264791]

numFlips: 2
[0.58794019 0.26713453]
[0.55125664 0.22732985]
[0.5485186 0.21568838]
[0.55155909 0.20974973]
[0.5550571 0.20559843]
[0.55806394 0.20239927]
[0.56048267 0.19988775]
[0.56238327 0.19791953]
[0.56385991 0.19638593]
[0.56499932 0.19519783]
[0.56587445 0.1942818]
[0.56654436 0.19357828]
[0.56705591 0.19303958]
[0.56744584 0.19262807]
[0.56774265 0.19231427]
[0.56796836 0.19207531]
[0.56813987 0.19189354]
[0.56827012 0.19175539]
[0.568369   0.19165045]
[0.56844403 0.19157077]

```
PS C:\Users\KB\Desktop\Rice\Courses\Tools & Models for DS\HW\HW4-EM> python .\HW4_EM.py
numFlips: 10
[0.71175162 0.30662754]
[0.66245393 0.24313347]
[0.63328033 0.21678924]
[0.61752958 0.20463265]
[0.60925177 0.19871663]
[0.60494866 0.19575105]
[0.60272153 0.19424209]
[0.6015708  0.19346871]
[0.60097661 0.19307092]
[0.60066987 0.19286597]
[0.60051155 0.19276028]
[0.60042983 0.19270576]
[0.60038765 0.19267762]
[0.60036588 0.1926631 ]
[0.60035464 0.19265561]
[0.60034884 0.19265174]
[0.60034585 0.19264975]
[0.6003443  0.19264872]
[0.60034351 0.19264819]
[0.60034309 0.19264791]

numFlips: 2
[0.58794019 0.26713453]
[0.55125664 0.22732985]
[0.5485186  0.21568838]
[0.55155909 0.20974973]
[0.5550571  0.20559843]
[0.55806394 0.20239927]
[0.56048267 0.19988775]
[0.56238327 0.19791953]
[0.56385991 0.19638593]
[0.56499932 0.19519783]
[0.56587445 0.1942818 ]
[0.56654436 0.19357828]
[0.56705591 0.19303958]
[0.56744584 0.19262807]
[0.56774265 0.19231427]
[0.56796836 0.19207531]
[0.56813987 0.19189354]
[0.56827012 0.19175539]
[0.568369   0.19165045]
[0.56844403 0.19157077]
```