Assignment 8

EMアルゴリズムによって，2つのガウス混合分布を推定した結果，表1のパラメータが得られた．また，この分布を利用して，データのクラス分けを行った結果，図1の結果が得られた．



図 1 得られた推定値によるクラス分類

|  |  |  |
| --- | --- | --- |
|  | Class 1 | Class 2 |
| ** | 0.355 | 0.645 |
| ****** | (-1.273, -0.666) | (1.273, 0.666) |
| **** |  |  |

表1 EMアルゴリズムで得られた推定値

パラメータ推定に使用したソースコードの一部メソッド(fit, calc\_probability, E\_step, M\_step)を図2に示す．なお，ソースコード全体は，添付する”Assignment8\_code.py”に記載する．パラメータ推定は，fitメソッドに間欠泉のデータを与えて行った．

fitメソッドでは，平均値の更新量が小さくなるまで，E-stepとM-stepを繰り返している．calc\_probabilityメソッドでは，入力xを与えたときの各クラスに属する確率を戻り値としている．E\_stepメソッドは，calc\_probabilityメソッドの同じ働きをしている．M\_stepメソッドでは，入力xとxが各クラスに属する確率prを入力として受け取り，各種の推定値の更新を行っている．

図 1 パラメータ推定に使用したソースコード

|  |  |
| --- | --- |
| 1 | def fit(self, x): |
| 2 | for t in range(self.max\_itr): |
| 3 | old\_mu = self.mu.copy() |
| 4 | pr = self.E\_step(x) |
| 5 | if np.abs(old\_mu - self.mu).mean() < self.eps: |
| 6 | return |
| 7 |  |
| 8 | def calc\_probability(self, x): |
| 9 | prob = self.pi[:, None] \* two\_dim\_gaussian(x, self.mu, self.sigma) |
| 10 | return prob / prob.sum(axis = 0) |
| 11 |  |
| 12 | def E\_step(self, x): |
| 13 | return self.calc\_probability(x) |
| 14 |  |
| 15 | def M\_step(self, x, pr): |
| 16 | nk = pr.sum(axis = 1) |
| 17 | self.mu = (x[None, :, :]\*pr[:, :, None]).sum(axis = 1)/ nk |
| 18 | for i in range(self.n\_class): |
| 19 | square = (x - self.mu[i])[:, :, None] \* \  (x - self.mu[i])[:, None, :] |
| 20 | prod = pr[i][:, None, None] |
| 21 | self.sigma[i] = (square \* prod).sum(axis = 0) / nk[i] |
| 22 | self.pi = pr.mean(axis = 1) |