15BCE0517

M.S.SANJAY

L7+L8

MACHINE LEARNING

EXPT 9- GAUSSIAM MIXTURE MODEL USING EM ALGORITHM

**DATASET:**

<https://raw.githubusercontent.com/BlackArbsCEO/Mixture_Models/K-Means%2C-E-M%2C-Mixture-Models/Class_heights.csv>

Gender,Height (in)

Male,72

Male,72

Female,63

Female,62

Female,62

Male,73

Female,64

Female,63

Female,67

Male,71

Male,72

Female,63

Male,71

Female,67

Female,62

Female,63

Male,66

Female,60

Female,68

Female,65

Female,64

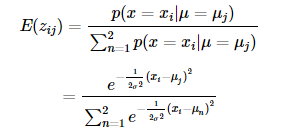
**ALGORITHM:**

To start the algorithm, we choose two random means.

From there we repeat the following until convergence.

**The expectation step:**

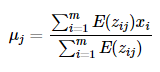
We calculate the expected values E(zij), which is the probability that x was drawn from the jth distribution.



The formula simply states that the expected value for zij is the probability xi given μj divided by the sum of the probabilities that xi belonged to each μ

**The maximization step:**

After calculating all E(zij) values we can calculate (update) new μ values.



This formula generates the maximum likelihood estimate.

By repeating the E-step and M-step we are guaranteed to find a local maximum giving us a maximum likelihood estimation of our hypothesis.

**CODE:**

import warnings

warnings.filterwarnings("ignore")

from IPython.core.display import display, HTML

import time

import pandas as pd

#import pandas\_datareader.data as web

import numpy as np

import scipy.stats as scs

from scipy.stats import multivariate\_normal as mvn

import sklearn.mixture as mix

import matplotlib as mpl

import matplotlib.pyplot as plt

import seaborn as sns

# import class heights

f = 'https://raw.githubusercontent.com/BlackArbsCEO/Mixture\_Models/K-Means%2C-E-M%2C-Mixture-Models/Class\_heights.csv'

data = pd.read\_csv(f)

# data.info()

height = data['Height (in)']

data

def em\_gmm\_orig(xs, pis, mus, sigmas, tol=0.01, max\_iter=100):

n, p = xs.shape

k = len(pis)

ll\_old = 0

for i in range(max\_iter):

print('\nIteration: ', i)

print()

exp\_A = []

exp\_B = []

ll\_new = 0

# E-step

ws = np.zeros((k, n))

for j in range(len(mus)):

for i in range(n):

ws[j, i] = pis[j] \* mvn(mus[j], sigmas[j]).pdf(xs[i])

ws /= ws.sum(0)

# M-step

pis = np.zeros(k)

for j in range(len(mus)):

for i in range(n):

pis[j] += ws[j, i]

pis /= n

mus = np.zeros((k, p))

for j in range(k):

for i in range(n):

mus[j] += ws[j, i] \* xs[i]

mus[j] /= ws[j, :].sum()

sigmas = np.zeros((k, p, p))

for j in range(k):

for i in range(n):

ys = np.reshape(xs[i]- mus[j], (2,1))

sigmas[j] += ws[j, i] \* np.dot(ys, ys.T)

sigmas[j] /= ws[j,:].sum()

new\_mus = (np.diag(mus)[0], np.diag(mus)[1])

new\_sigs = (np.unique(np.diag(sigmas[0]))[0], np.unique(np.diag(sigmas[1]))[0])

df = (pd.DataFrame(index=[1, 2]).assign(mus = new\_mus).assign(sigs = new\_sigs))

print(df.T)

# update complete log likelihoood

ll\_new = 0.0

for i in range(n):

s = 0

for j in range(k):

s += pis[j] \* mvn(mus[j], sigmas[j]).pdf(xs[i])

ll\_new += np.log(s)

print(f'log\_likelihood: {ll\_new:3.4f}')

if np.abs(ll\_new - ll\_old) < tol:

break

ll\_old = ll\_new

return ll\_new, pis, mus, sigmas

height = data['Height (in)']

n = len(height)

# Ground truthish

\_mus = np.array([[0, data.groupby('Gender').mean().iat[0, 0]],

[data.groupby('Gender').mean().iat[1, 0], 0]])

\_sigmas = np.array([[[5, 0], [0, 5]],

[[5, 0],[0, 5]]])

\_pis = np.array([0.5, 0.5]) # priors

# initial random guesses for parameters

np.random.seed(0)

pis = np.random.random(2)

pis /= pis.sum()

mus = np.random.random((2,2))

sigmas = np.array([np.eye(2)] \* 2) \* height.std()

# generate our noisy x values

xs = np.concatenate([np.random.multivariate\_normal(mu, sigma, int(pi\*n))

for pi, mu, sigma in zip(\_pis, \_mus, \_sigmas)])

ll, pis, mus, sigmas = em\_gmm\_orig(xs, pis, mus, sigmas)

**OUTPUT:**

Iteration: 0

1 2

mus 61.362928 59.659685

sigs 469.240750 244.382352

log\_likelihood: -141.8092

Iteration: 1

1 2

mus 68.73773 63.620554

sigs 109.85442 7.228183

log\_likelihood: -118.0520

Iteration: 2

1 2

mus 70.569842 63.688825

sigs 4.424452 3.139277

log\_likelihood: -100.2591

Iteration: 3

1 2

mus 70.569842 63.688825

sigs 4.424427 3.139278

log\_likelihood: -100.2591

Screenshot:

