[Lab] Gaussian Mixture Model (GMM)

Jae Yun JUN KIM[*]

October 13, 2021

Due: Before the end of today lab session
Evaluation: Code and explanation about the code in groups of only two or three people
Remark:

- Only groups of two or three people accepted (preferably three).
- Before you leave today lab session, you must show the lab task results to the professor.
- No plagiarism. If plagiarism happens, both the "lender" and the "borrower" will have a zero.
- Code yourself from scratch. No lab/homework will be considered if any ML library is used.
- Do thoroughly all the demanded tasks.
- Study the theory for the questions.

Sources: Scikit-learn

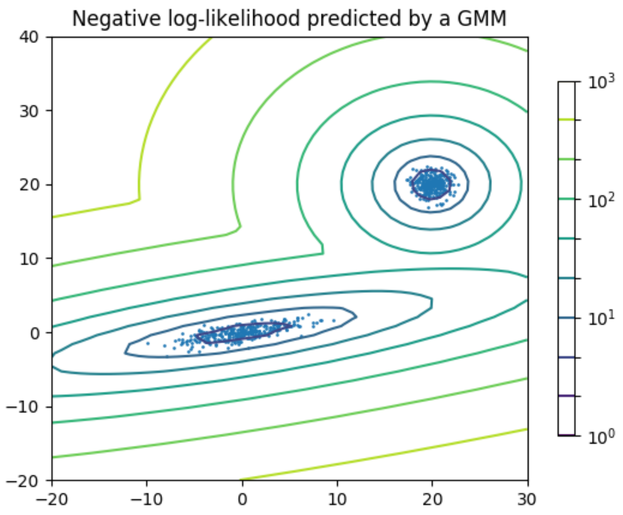

Figure 1: Density estimation

1 Example (with Scikit-learn): Density estimation for a Gaussian mixture

Plot the density estimation of a mixture of two Gaussians. Data is generated from two Gaussians with different centers and covariance matrices.

[*]ECE Paris Graduate School of Engineering, 37 quai de Grenelle 75015 Paris, France; jae-yun.jun-kim@ece.fr

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import LogNorm
from sklearn import mixture

n_samples = 300

# generate random sample, two components
np.random.seed(0)

# generate spherical data centered on (20, 20)
shifted_gaussian = np.random.randn(n_samples, 2) + np.array([20, 20])

# generate zero centered stretched Gaussian data
C = np.array([[0., -0.7], [3.5, .7]])
stretched_gaussian = np.dot(np.random.randn(n_samples, 2), C)

# concatenate the two datasets into the final training set
X_train = np.vstack([shifted_gaussian, stretched_gaussian])

# fit a Gaussian Mixture Model with two components
clf = mixture.GaussianMixture(n_components=2, covariance_type='full')
clf.fit(X_train)

# display predicted scores by the model as a contour plot
x = np.linspace(-20., 30.)
y = np.linspace(-20., 40.)
X, Y = np.meshgrid(x, y)
XX = np.array([X.ravel(), Y.ravel()]).T
Z = -clf.score_samples(XX)
Z = Z.reshape(X.shape)

CS = plt.contour(X, Y, Z, norm=LogNorm(vmin=1.0, vmax=1000.0),
                 levels=np.logspace(0, 3, 10))
CB = plt.colorbar(CS, shrink=0.8, extend='both')
plt.scatter(X_train[:, 0], X_train[:, 1], .8)

plt.title('Negative log-likelihood predicted by a GMM')
plt.axis('tight')
plt.show()
```

2 GMM from scratch

Consider the example problem (and the data) considered in the previous section. Implement one iteration of

a. E-step

b. M-step

using the theory learned in class.