Expectation_Maximization

January 20, 2021

[ ]:
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn import mixture
```

Estimating Parameters

[ ]:
```
#Define means and covariances of the three classes
mean1 = [1,1]
mean2 = [3,3]
mean3 = [2,6]
mean = [mean1,mean2, mean3]         #used for error calculation later
cov1 = 0.1*np.identity(2)
cov2 = 0.2*np.identity(2)
cov3 = 0.3*np.identity(2)
cov = [0.1, 0.2, 0.3]
weights = [0.4, 0.4, 0.2]           #used for error calculation later
n_samples = 600

#Generate the training data
Train1 = np.random.multivariate_normal(mean1,cov1,int(n_samples*0.4))
Train2 = np.random.multivariate_normal(mean2,cov2,int(n_samples*0.4))
Train3 = np.random.multivariate_normal(mean3,cov3,int(n_samples*0.2))
Train = np.concatenate((Train1,Train2,Train3), axis=0)
```

[ ]:
```
#Use EM algorithm to determine the parameters of the gaussian mixture␣
 ↪distribution generated above
gmm = mixture.GaussianMixture(n_components=3,␣
 ↪covariance_type='spherical',random_state=5)
gmm.fit(Train)

#plot the data and the estimates for the means
plt.scatter(Train[:,0], Train[:,1], label='samples')
plt.scatter(gmm.means_[:,0], gmm.means_[:,1], label='mean_est')
plt.title('Samples and EM estimates')
plt.axis('equal')
plt.legend()
plt.show()
```

```
#calculate and print the errors for the different parameters in each class
for k in [0,1,2]:
    weight_error = abs(weights[k] - gmm.weights_[k])
    mean_error = np.linalg.norm(mean[k] - gmm.means_[k])
 ↪    #error in l2 norm
    covariance_error = abs(cov[k] - gmm.covariances_[k])

    print('\nResults for class ' + str(k+1) + ':')
    print('weight_error = ' + str(weight_error))
    print('mean_error = ' + str(mean_error))
    print('covariance_error = ' + str(covariance_error))
```

Varying Initial Parameters

[ ]:
```
#Setting the initial parameters of the two cases
K = [3,2]
weights_init = [[0.34, 0.33, 0.33], [0.5, 0.5]]
means_init = [[[0,2], [5,2], [5,5]], [[1.6,1.4], [1.4,1.6]]]
precision_init = [[6.66, 3.70, 2.50], [5, 2.5]]
titles = ['init_params set 1', 'init_params set 2']

for i in [0,1]:
    gmm = mixture.GaussianMixture(n_components=K[i],␣
 ↪covariance_type='spherical', weights_init=weights_init[i],␣
 ↪means_init=means_init[i], precisions_init=precision_init[i] ,random_state=50)
    gmm.fit(Train)

    tmp = np.array(means_init[i])   #constructing an array from the list, to␣
 ↪plot it more easily

    #plot the samples and the estimated means for each set
    plt.scatter(Train[:,0], Train[:,1], label='samples')
    plt.scatter(gmm.means_[:,0], gmm.means_[:,1], label='mean_est')
    plt.scatter(tmp[:,0], tmp[:,1], label='mean_init')
    plt.legend()
    plt.axis('equal')
    plt.title(titles[i])
    plt.show()

    #print all the estimated parameters
    print("weights:")
    print(gmm.weights_)
    print("means:")
    print(gmm.means_)
    print("covariances:")
    print(gmm.covariances_)
```