VE414 HW4

徐远洋

(a) suppose the missing variables are y_a and y_b, $y_a + y_b = y_1 \Rightarrow y_a = \frac{1}{2},\ y_b = \frac{\theta}{4}$

$$Q(\theta|\theta^{(t)}) = E[\log P(y_{obs}, y_{mis}|\theta)|y_{obs}, \theta^{(t)}]$$

$$= E[\log(\frac{(y_1+y_2+y_3+y_4)!}{y_1!\,y_2!\,y_3!\,y_4!} \cdot \frac{1}{2}^{y_a} \cdot (\frac{\theta}{4})^{y_b} \cdot (\frac{1-\theta}{4})^{y_2+y_3} \cdot (\frac{\theta}{4})^{y_4})]$$

$$= E[\ C + y_a \log\frac{1}{2} + y_b \log\frac{\theta}{4} + (y_2+y_3)\log(\frac{1-\theta}{4}) + y_4 \log(\frac{\theta}{4})]$$

$$E[y_a] = \frac{\frac{1}{2}}{\frac{1}{2}+\frac{\theta^{(t)}}{4}} y_1 \qquad E[y_b] = \frac{\frac{\theta^{(t)}}{4}}{\frac{1}{2}+\frac{\theta^{(t)}}{4}} \cdot y_1$$

$$\Rightarrow C + \frac{\frac{\theta^{(t)}}{4}}{\frac{1}{2}+\frac{\theta^{(t)}}{4}} y_1 \cdot \log\theta + (y_2+y_3)\log(1-\theta) + y_4\log\theta$$

$$\frac{\partial Q}{\partial \theta} = \frac{\theta^{\tau}}{2+\theta^{\tau}} y_1 \cdot \frac{1}{\theta} + (y_2+y_3)\frac{-1}{1-\theta} + \frac{y_4}{\theta} = 0$$

$$\underset{\theta}{\arg\max}\, Q(\theta|\theta^{(t)}) \Rightarrow \theta = \frac{\frac{\theta^{\tau}}{2+\theta^{\tau}} y_1 + y_4}{\frac{\theta^{\tau}}{2+\theta^{\tau}} y_1 + y_4 + y_2 + y_3}$$

In [2]:
```python
def EM(theta0):
    itr = 10000
    theta1 = 0.5
    for i in range(itr):
        theta0 = theta1
        theta1 = (125*theta1/(2+theta1)+ 34)/(125*theta1/(2+theta1)+18+20
    return theta1
```

In [3]:
```python
EM(0.5)
```

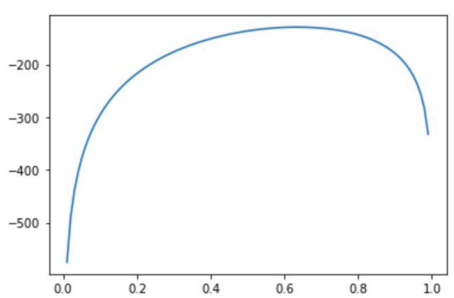
Out[3]: 0.6268214978709824

(b)

In [4]:
```python
x = np.linspace(0,1,100)
y = np.log(np.power(x,125) * np.power((1-x),72))
plt.plot(x,y)
plt.show
```

```
<ipython-input-4-d0c101e43c50>:2: RuntimeWarning: divide by zero encounte
red in log
  y = np.log(np.power(x,125) * np.power((1-x),72))
```

Out[4]: <function matplotlib.pyplot.show(close=None, block=None)>



In [5]:
```python
np.argmax(y)/100
```

Out[5]: 0.63

It reaches maximum at $x = 0.63$

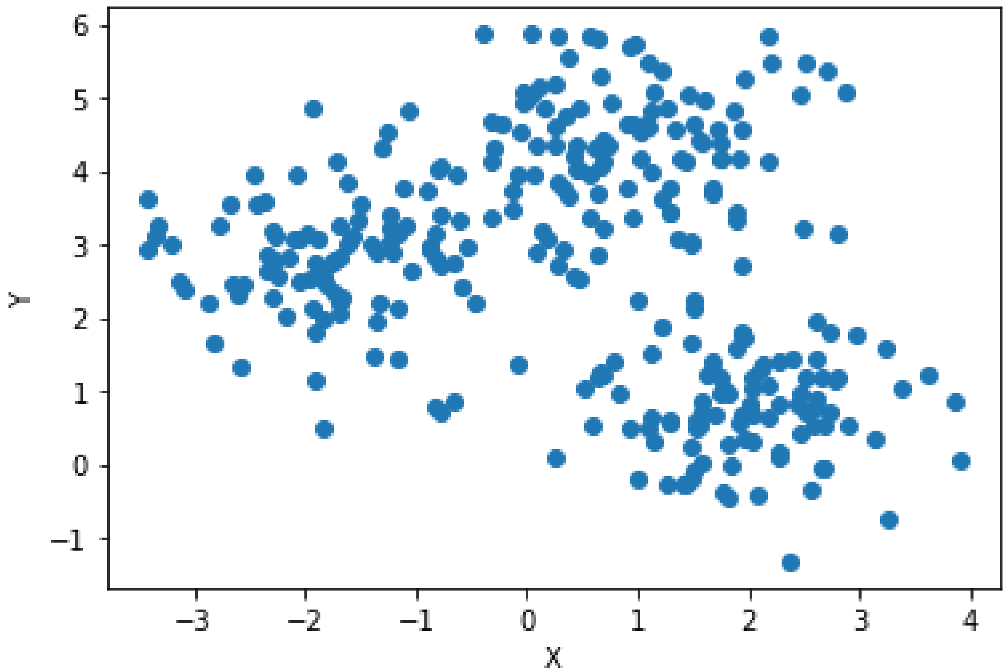
VE414_HW4

November 30, 2021

[1]:
```
#Your turn
#The samples are comming from 3 gaussian distributions,
#please find the source of each cluster & the corresponding parameter
#mu & sigma
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.stats import multivariate_normal as mvn
from numpy.core.umath_tests import matrix_multiply as mm


df = pd.read_csv('raw_data.csv',index_col=0)
plt.scatter(df['X'],df['Y'])
plt.xlabel('X');
plt.ylabel('Y');
```

```
<ipython-input-1-306fdb4aa462>:9: DeprecationWarning: numpy.core.umath_tests is
an internal NumPy module and should not be imported. It will be removed in a
future NumPy release.
  from numpy.core.umath_tests import matrix_multiply as mm
```



[2]:

```
# The initial value of parameters
mus = np.array([[-2,3], [1, 5], [2, 1]])
sigmas = np.array([[[1,0],[0,1]], [[1,0],[0,1]], [[1,0],[0,1]]])
pis = np.array([1/3, 1/3, 1/3])
xs = df.values

def em_gmm_vect(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):
        ws = np.zeros((k,n))
        for j in range(k):
            ws[j, :] = pis[j] * mvn(mus[j], sigmas[j]).pdf(xs)
        ws /= ws.sum(0)

        pis = ws.sum(axis = 1)
        pis /= n
        mus = np.dot(ws, xs)
        mus /= ws.sum(1)[:, None]
        sigmas = np.zeros((k, p, p))
        for j in range(k):
            ys = xs - mus[j,:]
            sigmas[j] = (ws[j,:, None, None] * mm(ys[:,:,None],ys[:,None,:])).
 ↪sum(axis = 0)
```

```python
        sigmas /= ws.sum(axis=1)[:,None,None]
        ll_new = 0
        for pi, mu, sigma in zip(pis, mus, sigmas):
            ll_new += pi*mvn(mu,sigma).pdf(xs)
        ll_new = np.log(ll_new).sum()
        if np.abs(ll_new-ll_old) < tol:
            break
        ll_old = ll_new

    print("Total iteration: %d" %(i+1))
    print(" mu_1:",mus[0], "\n mu_2: ", mus[1], "\n mu_3: ", mus[2])
    print("\n sigma_1:", sigmas[0], "\n sigma_2: ", sigmas[1], "\n sigma_3: ",␣
 ↪sigmas[2])

em_gmm_vect(xs, pis, mus, sigmas, tol = 0, max_iter = 100)
```

```
Total iteration: 100
 mu_1: [-1.74427754  2.75730959]
 mu_2:  [0.81315808 4.27595904]
 mu_3:  [1.93899422 0.7726094 ]

 sigma_1: [[ 0.62294001 -0.02641632]
 [-0.02641632  0.63883231]]
 sigma_2:  [[0.7917954  0.0651846 ]
 [0.0651846  0.74009291]]
 sigma_3:  [[ 0.59709477 -0.01084667]
 [-0.01084667  0.48586406]]
```