Chapter 15 - Exercise 2: Pov

Cho dữ liệu như pov_12.csv

1. Chuẩn hóa dữ liệu X chứa cột 0, 1
2. Tìm số cụm k phù hợp
3. Áp dụng thuật toán GMM để giải bài toán phân cụm với số cụm = k ở câu 2
4. Cho X_test = np.array([[0.1, 0.1], [0.8,0.8], [0.5, 0.5]]), cho biết những phần tử này thuộc cụm nào?
5. Vẽ hình, xem kết quả

In [1]:

```
from sklearn import metrics
from scipy.spatial.distance import cdist
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

In [2]:

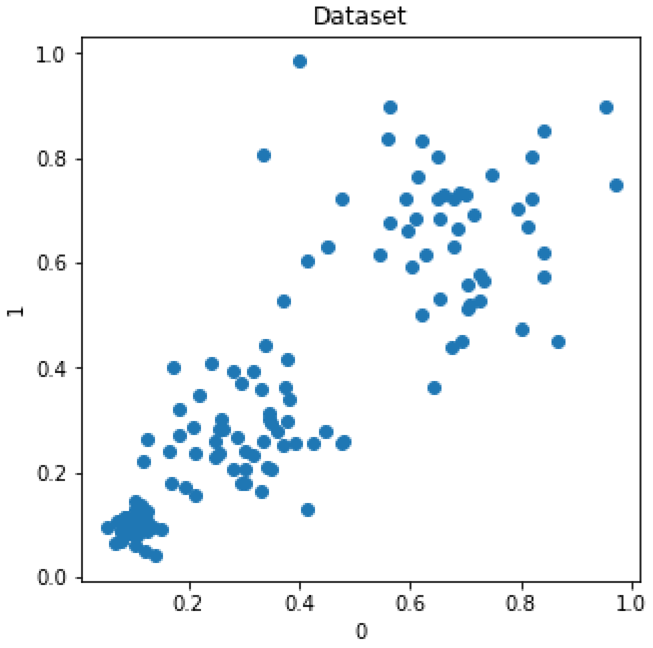
```
data = pd.read_csv("pov_12.csv", sep=" ", header=None)
print(data.shape)
data.head()
```

```
(150, 3)
```

Out[2]:

	0	1	2
0	0.109393	0.085409	Cluster1
1	0.082571	0.101796	Cluster1
2	0.084990	0.113641	Cluster1
3	0.114611	0.115524	Cluster1
4	0.097356	0.095484	Cluster1

In [3]:
```
plt.figure(figsize=(5,5))
plt.scatter(data[0], data[1])
plt.title('Dataset')
plt.xlabel("0")
plt.ylabel("1")
plt.show()
```



In [4]:
```
X_train = data[[0,1]]
```

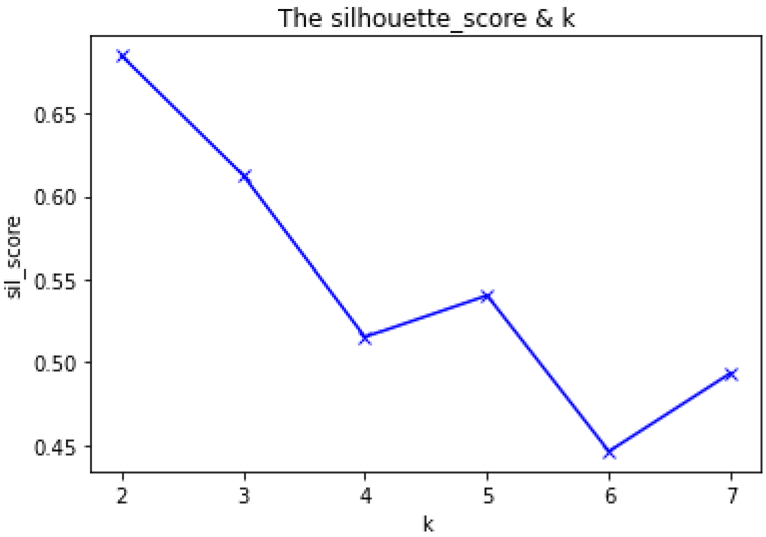
In [5]:
```
from sklearn.mixture import GaussianMixture
```

In [6]:

```
from sklearn import metrics
list_sil = [] # chua danh sach cac gia tri sil
K = range(2,8) # chua danh sach cac k
for k in K:
    gmm = GaussianMixture(n_components=k) # 2, 3, 4...
    gmm.fit(X_train)
    labels = gmm.predict(X_train)
    # k = 2 => 0,1
    # k = 3 => 0, 1, 2
    sil = metrics.silhouette_score(X_train, labels, metric='euclidean')
    list_sil.append(sil)
```

In [7]:

```
# Plot
plt.plot(K, list_sil, 'bx-')
plt.xlabel('k')
plt.ylabel('sil_score')
plt.title('The silhouette_score & k')
plt.show()
```



In [8]:

```
# Select k = 2
```

In [9]:

```
import numpy as np
from sklearn.mixture import GaussianMixture
gmm = GaussianMixture(n_components=2)
gmm.fit(X_train)
```

Out[9]:

```
GaussianMixture(covariance_type='full', init_params='kmeans', max_iter=100,
                means_init=None, n_components=2, n_init=1, precisions_init=Non
e,
                random_state=None, reg_covar=1e-06, tol=0.001, verbose=0,
                verbose_interval=10, warm_start=False, weights_init=None)
```

In [10]:

```
print(gmm.weights_)
```

```
[0.66606181 0.33393819]
```

In [11]:
```
print(gmm.means_)
```

```
[[0.20020692 0.18647424]
 [0.66682326 0.66125669]]
```

In [12]:
```
print(gmm.covariances_)
```

```
[[[ 0.01339299  0.00880714]
  [ 0.00880714  0.01067513]]

 [[ 0.01872054 -0.00121377]
  [-0.00121377  0.01794757]]]
```

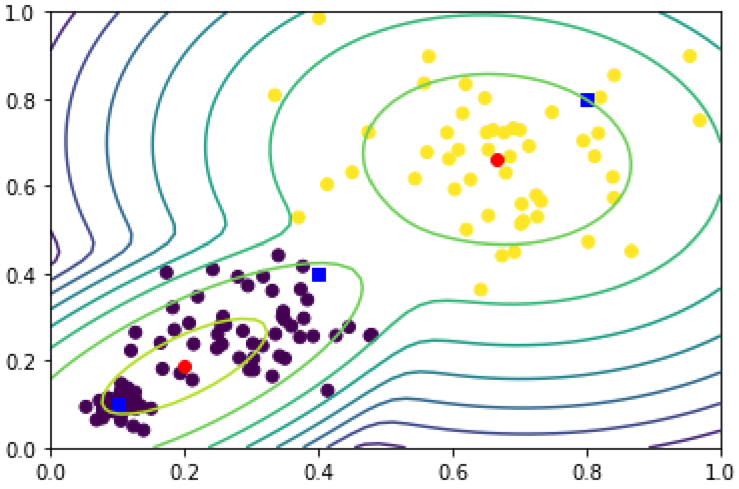
In [13]:
```
X_test = np.array([[0.1, 0.1], [0.8,0.8], [0.4, 0.4]])
pred = gmm.predict(X_test)
pred
```

Out[13]:
```
array([0, 1, 0], dtype=int64)
```

In [14]:
```
types = gmm.predict(X_train)
```

In [16]:
```
# plot mixture of Gaussians
plt.figure(figsize=(6,4))
X, Y = np.meshgrid(np.linspace(0,1), np.linspace(0,1))
XX = np.array([X.ravel(), Y.ravel()]).T
Z = gmm.score_samples(XX)
Z = Z.reshape((50,50))

plt.contour(X, Y, Z)
plt.scatter(X_train[0], X_train[1], c=types)
plt.scatter(X_test[:,0], X_test[:,1], marker="s", c='b')
plt.scatter(gmm.means_[:,0], gmm.means_[:,1], color="red")
plt.show()
```


1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0