TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA KHOA HỌC MÁY TÍNH
MÔN MÁY HỌC TRONG THỊ GIÁC MÁY TÍNH

Bài Tập Thực Hành 1
Clustering

Lớp: KHTN2014

GVLT: Lê Đình Duy

GVHDTH: Mai Tiến Dũng

SVTH: Hoàng Ngọc Thạch - 14520811

TP HCM, ngày 20 tháng 10 năm 2017

Mục Lục

Link Github: https://github.com/thachhoang1909/homework_1

Bài tập 1: KMeans trên bộ random 2 Gaussian

- Import hàm KMean từ thư viện scikit-learn.
- Hàm make_blobs tạo bộ dữ liệu ngẫu nhiên.

In [203]:

```
# import library
from sklearn.cluster import KMeans
from sklearn.datasets import make_blobs
import matplotlib.pyplot as plt
#plt.gray()
%matplotlib inline
```

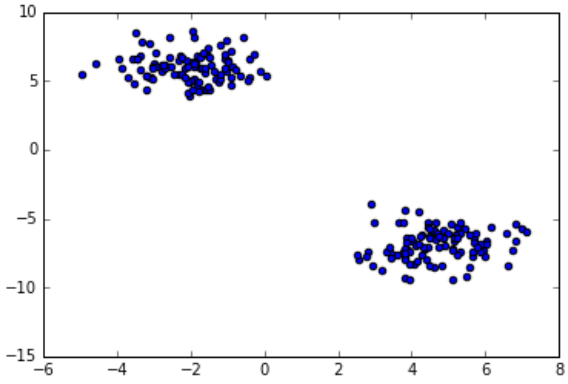
- Phát sinh ngẫu nhiên bộ dữ liệu gồm 2 chiều (feature) và có 2 tâm (2 vùng).
- X gồm 2 cột (feature), 150 dòng (sample).
- Y là ID vùng của mỗi điểm mà nó thuộc về.

In [204]:

```
X,Y = make_blobs(n_samples=200, n_features=2, centers=2)
```

In [205]:

```
plt.scatter(X[:,0], X[:,1])
plt.show()
```

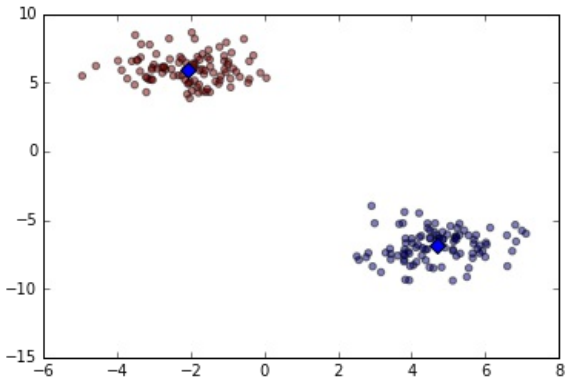


- Bên trên là visualize bộ dữ liệu X, gồm 2 vùng phân biệt.
- Tạo KMean model với k=2
- Hàm KMeans.fit_predict chạy thuật toán KMean clustering trên bộ dữ liệu X và trả về label vùng của mỗi điểm.

In [206]:

```
model = KMeans(n_clusters=2)
label = model.fit_predict(X)
```

Visualize kết quả clustering



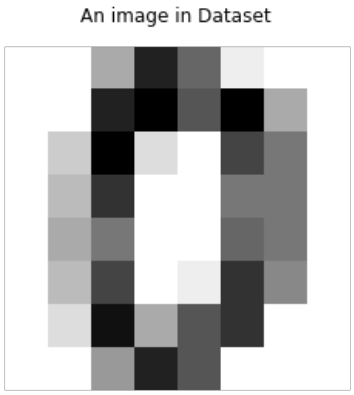
2. Bài tập 2: Hand-writen digits

In [209]:

```
# Load dataset
digits = datasets.load_digits()
```

```
Dataset info:  (1797, 8, 8)
```

```
(-0.5, 7.5, 7.5, -0.5)
```



- Dataset gồm:
  - 1797 hình 8x8 đã được gán nhãn (digits.target).
  - 10 lớp.

2.1 KMean Clustering

In [211]:

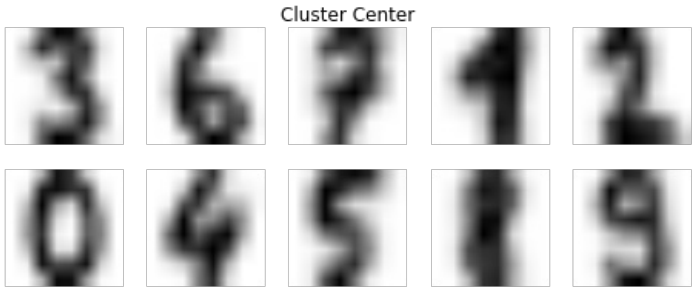
```
# Apply KMeans clustering to data
number_clusters = 10
#create KMeans model
model = KMeans(n_clusters=number_clusters)

# fit model to data and predict on this data
label_kmean = model.fit_predict(digits.data)
```

Truth_labels	0	1	2	3	4	5	6	7	8	9
labels										
0	0	1	13	156	0	1	0	0	2	6
1	0	2	0	0	0	1	177	0	2	0
2	0	0	3	6	10	0	0	170	3	8
3	0	54	2	0	3	0	0	2	6	20
4	0	24	148	0	0	0	0	0	3	0
5	177	0	1	0	0	0	1	0	0	0
6	1	0	0	0	166	2	0	0	0	0
7	0	1	0	2	0	136	0	5	7	7
8	0	100	8	7	2	0	3	2	99	0
9	0	0	2	12	0	42	0	0	52	139

- Dựa trên cross table có thể thấy:

- Có sự nhập nhằng khó xác định cluster của hình digit 1 và hình digit 8.
- Thông qua cluster center, có thể nhận ra 99 hình digit 1 được cho vào cluster của digit 8.
- Không có hình noise.

- Clustering Center



2.2 Spectral Clustering

In [214]:

```
# import library
from sklearn.cluster import spectral_clustering
from sklearn.metrics.pairwise import cosine_similarity
```

- Tính ma trận tương đồng giữa các sample
- Áp dụng thuật toán spectral_clustering trên ma trận tương đồng vừa tính.

In [215]:

```
graph = cosine_similarity(digits.data)
label_spectral = spectral_clustering(graph, n_clusters=10)
```

Cross-table

Truth_labels	0	1	2	3	4	5	6	7	8	9
labels										
0	1	0	0	0	163	2	0	0	0	0
1	0	0	0	4	0	157	0	0	3	3
2	0	86	53	6	5	0	6	10	101	1
3	0	2	0	1	0	2	172	0	13	0
4	177	0	1	0	1	1	0	0	0	3
5	0	0	1	145	0	0	0	0	6	2
6	0	0	0	16	0	20	3	0	7	134
7	0	58	5	5	1	0	0	15	40	35
8	0	0	2	2	11	0	0	154	3	2
9	0	36	115	4	0	0	0	0	1	0

- Dựa trên cross table có thể thấy:

- Có sự nhập nhằng khó xác định cluster của hình digit 1 và hình digit 8.
- Thông qua cluster center, có thể nhận ra phần lớn hình digit 1 được cho vào cluster của digit 8.
- Không có hình noise.

2.3 DBSCAN [3]

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu (1996). [3]"A density-based algorithm for discovering clusters in large spatial databases with noise". Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD- 96). AAAI Press. pp. 226-231

- DBSCAN là thuật toán gom tụm dựa trên mật độ, hiệu quả với cơ sở dữ liệu lớn, có khả năng xử lý nhiễu.
- Ý tưởng chính của thuật toán là vùng lân cận mỗi đối tượng trong một cụm có số đối tượng lớn hơn ngưỡng tối thiểu. Hình dạng vùng lân cận phụ thuộc vào hàm khoảng cách giữa các đối tượng (khoảng cách Manhattan, khoảng cách Euclidean).
- Độ phức tạp thuật toán: tính toán (N*logN), dữ liệu (N^2).
- Thuật toán cơ bản:
  - Gồm 2 tham số: eps và minPts
  - Từ một mẫu (nút) chưa được chọn, kiểm tra các điểm gần nhất, nếu số lượng các điểm này lớn hơn giá trị minPts thfi bắt đầu một nhóm mới. Nếu không sẽ đánh dấu là điểm nhiễu. Điểm nhiễu này vẫn có thể thuộc một nhóm khác, khi đó sẽ bỏ đánh dấu điểm nhiễu.

- Cứ thế mở rộng ra đến khi không thể tìm thêm điểm mới cho nhóm.

In [217]:

```
eps, min_samples = 0.0595 , 10

#import DBSCAN
from sklearn.cluster import DBSCAN
dbscan = DBSCAN(eps=eps, min_samples=min_samples, metric='cosine', algorithm='brute')

label_dbscan = dbscan.fit_predict(digits.data)
```

Cross table

Truth labels	0	1	2	3	4	5	6	7	8	9
labels										
-1	7	13	41	49	35	68	5	52	81	78
0	171	0	0	0	0	0	0	0	0	0
1	0	143	0	0	0	0	1	0	93	1
2	0	0	0	0	0	0	175	0	0	0
3	0	0	0	134	0	1	0	0	0	101
4	0	0	0	0	146	0	0	0	0	0
5	0	0	136	0	0	0	0	0	0	0
6	0	0	0	0	0	65	0	0	0	0
7	0	0	0	0	0	0	0	127	0	0
8	0	0	0	0	0	48	0	0	0	0
9	0	26	0	0	0	0	0	0	0	0

- Dựa trên cross table có thể thấy:

- Có sự nhập nhằng khó xác định cluster của hình digit 3 và hình digit 9.
- Thông qua cluster center, có thể nhận ra phần lớn hình digit 3 được cho vào cluster của digit 9.
- Có khá là nhiều hình không bị gán noise, không tìm được cluster.

2.4 Agglomerative Clustering

In [219]:

```
# import library
from sklearn.cluster import AgglomerativeClustering

aggModel = AgglomerativeClustering(n_clusters=10)

label_agglomerative = aggModel.fit_predict(digits.data)
```

Cross table

Truth labels	0	1	2	3	4	5	6	7	8	9
labels										
0	0	0	0	0	0	179	0	0	0	2
1	0	0	0	169	0	2	0	0	1	145
2	0	27	166	0	0	0	0	0	4	0
3	0	0	1	1	3	0	0	179	1	11
4	0	0	10	13	0	0	1	0	165	2
5	0	0	0	0	178	0	0	0	0	0
6	0	0	0	0	0	1	180	0	0	0
7	178	0	0	0	0	0	0	0	0	0
8	0	59	0	0	0	0	0	0	1	20
9	0	96	0	0	0	0	0	0	2	0

- Dựa trên cross table có thể thấy:

- Có sự nhập nhằng khó xác định cluster của hình digit 3 và hình digit 9.
- Thông qua cluster center, có thể nhận ra phần lớn hình digit 3 được cho vào cluster của digit 9.
- Không có hình noise.

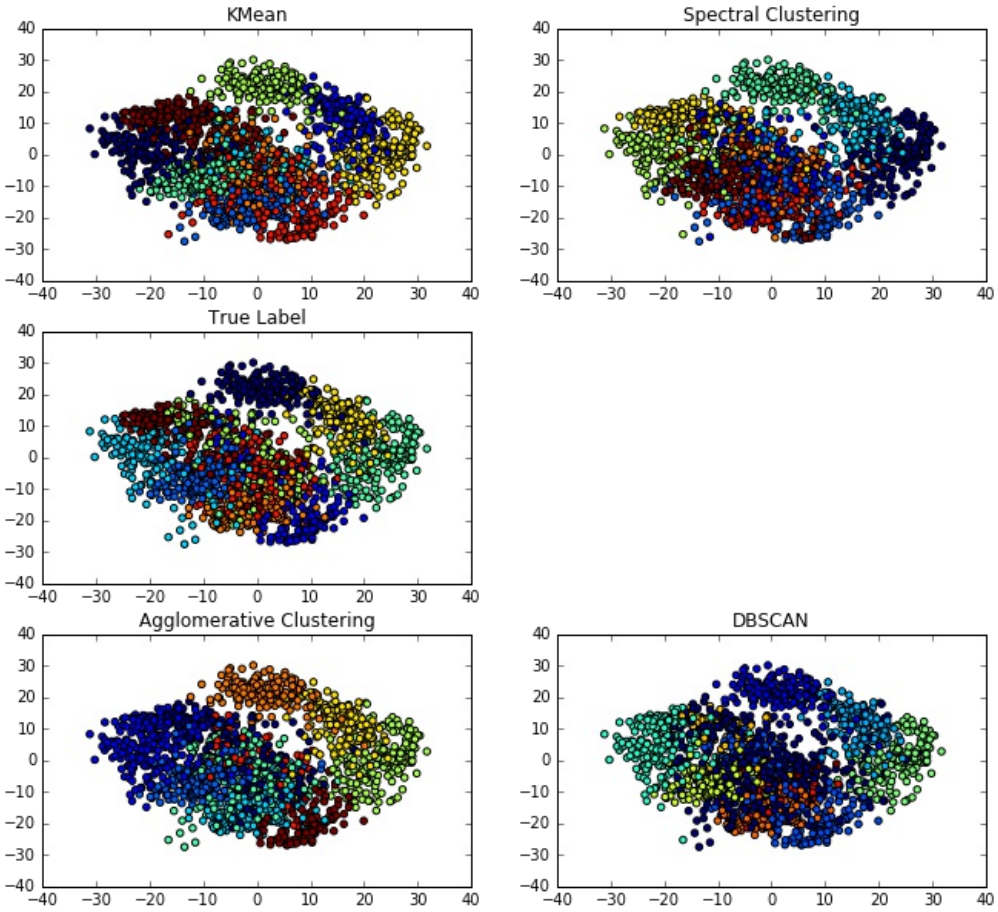
2.5 Visualize kết quả của thuật toán phân lớp

2.5.1 PCA

- Sử dụng thuật toán PCA để giảm số chiều features.

In [221]:

```
from sklearn.decomposition import PCA

pca = PCA(n_components=2).fit_transform(digits.data)
```

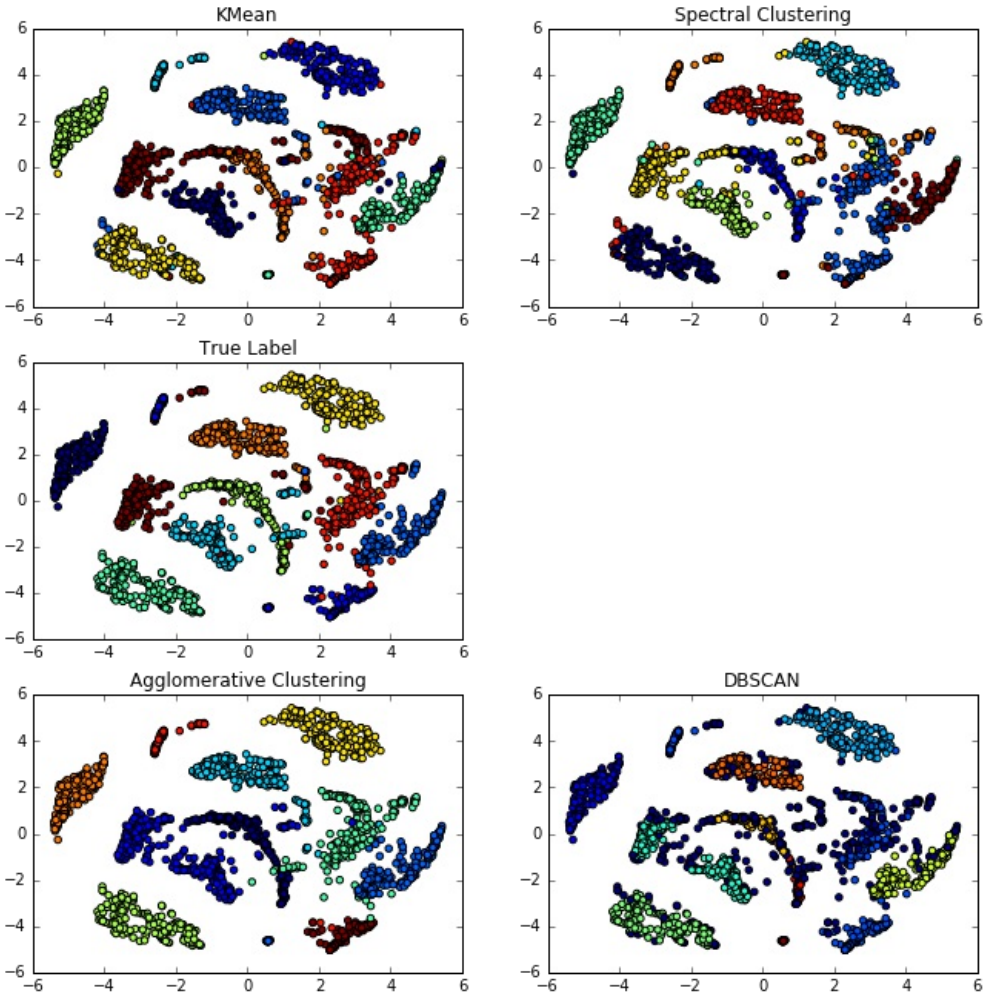


2.5.2 Visualize by T-SNE

- T-SNE giảm số chiều của data về số liệu 2 chiều.

In [225]:

```
from sklearn.manifold import TSNE

TSNE_model = TSNE(learning_rate=100)

tnse = TSNE_model.fit_transform(digits.data)
```



2.6 Evaluate clustering algorithm

- Sử dụng các hệ đo lường để đánh giá thuật toán: Homogeneity, Completeness, V-measure, Adjusted Random, Adjusted Mutual Information.

In [228]:

```
compareAlgorithm({'KMean':label_kmean, 'Spectral':label_spectral,'DBSCAN':label_dbscan,'Agglomerative':label
_agglomerative},\
                 digits.target, digits.data)
```

```
#Sample: 1797 #Class: 10 #feature: 64
_____________________________________________________________________________________
init  homo compl v-meas ARI AMI silhouette
Spectral  0.711 0.715 0.713 0.624 0.708 0.160
KMean     0.739 0.747 0.743 0.666 0.736 0.188
DBSCAN    0.703 0.739 0.721 0.495 0.700 0.136
Agglomerative 0.858 0.879 0.868 0.794 0.856 0.190
_____________________________________________________________________________________
```

Nhận xét:

- Dựa trên bảng số liệu. Có thể thấy thuật toán Agglomerative cho ra kết quả chính xác hơn DBSCAN, Spectral, KMean với raw digit data.

3. Bài tập 3: Face Dataset

In [230]:

```
faces = fetch_lfw_people(min_faces_per_person=70)
```

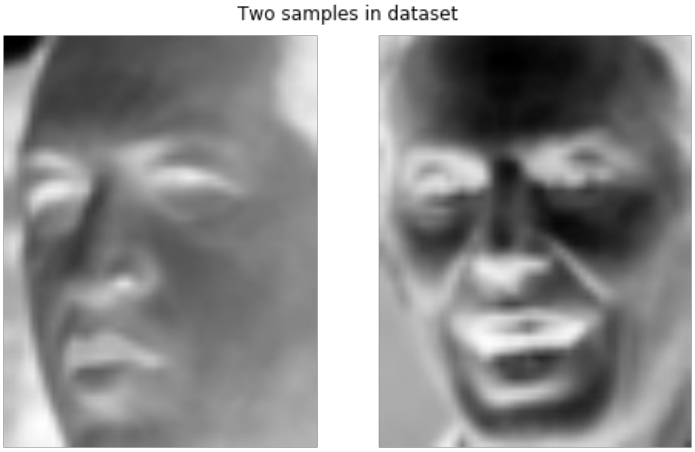
In [231]:

```
print("Number of images: ", faces.images.shape)
print("Number of classes: ",len(set(faces.target)))
```

```
Number of images:  (1288, 62, 47)
Number of classes:  7
```

Dataset

- Dataset gồm 1288 ảnh 62 x 47.
- Đã được label với target gồm 5749 class.
- Mỗi class gồm ảnh của một người.
- Mỗi class gồm ít nhất 70 ảnh.



3.1 Trích xuất đặc trưng

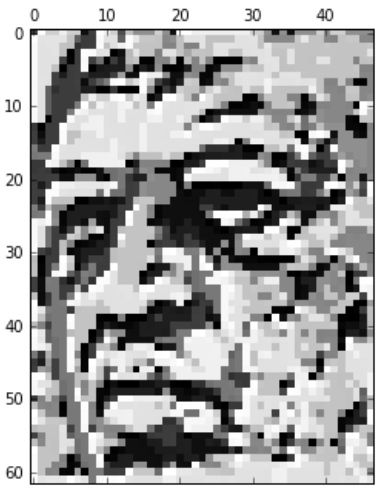
- Sử dụng Local Binary Pattern.

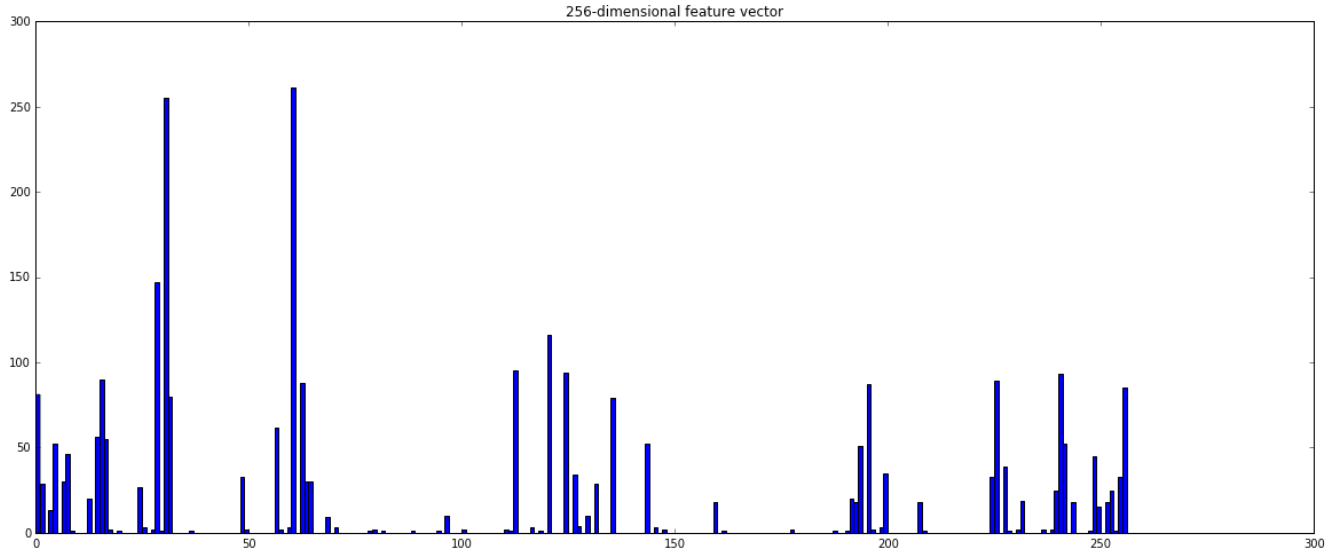
In [233]:

```
# import feature detector & descriptor library
from skimage.feature import local_binary_pattern

feature = local_binary_pattern(faces.images[0], P=8, R=0.5)
plt.matshow(feature, cmap=plt.cm.binary)
```

Out[233]:

```
<matplotlib.image.AxesImage at 0xe972dd8>
```





Trích xuất feature từ mỗi ảnh

In [235]:

```
def getLBP_feature(image):
    feature = local_binary_pattern(image, P=8, R=0.5)
    return np.histogram(feature, bins=list(range(257)))[0]
```

In [236]:

```
feature_LBP = list(map(getLBP_feature, faces.images))
feature_LBP = np.array(feature_LBP)
```

3.2 KMean Clustering

In [237]:

```
model_kmean = KMeans(n_clusters=7)
label_kmean = model_kmean.fit_predict(feature_LBP)

df = pd.DataFrame({'label':label_kmean, 'True Label':faces.target})
ct = pd.crosstab(df['label'], df['True Label'])
print(ct.tail(10))
```

```
True Label   0   1   2    3   4   5   6
label
0           26  67  30  107  17   6  29
1            9   6   6  108   5  10  22
2            7  16  28   87  22  12  22
3           11  39   7   39  15  18  20
4            5  25   2   42   8   8  13
5           14  53  26   67  26   8  22
6            5  30  22   80  16   9  16
```

- Nhận xét hiện tại:
  - Kết quả trả về của KMean không hiệu quả. Khó phân biệt label đúng cho mỗi vùng.

3.3 Spectral Clustering

In [238]:

```
# import library
from sklearn.cluster import spectral_clustering
from sklearn.metrics.pairwise import cosine_similarity

graph = cosine_similarity(feature_LBP)
label_spectral = spectral_clustering(graph, n_clusters=7)
```

- Cross-table

```
Truth_labels   0   1   2    3   4   5   6
labels
0             26  59  36  108   6   3  12
1             10  46  22   58  19  17  18
2              9  42  10   44  16  11  31
3              9  32   9   80  12  10  30
4              8   8   9   97  11   8  14
5             13  19  24   88  14  13  19
6              2  30  11   55  31   9  20
```

- Dựa trên Cross Table:
  - Các hình thuộc cùng một class được cluster dàn trải vào các vùng. Nên không xác định được vùng nào của class nào.
  - Kết quả cluster cho ra thấp.

3.4 DBSCAN

In [240]:

```
eps, min_samples = 0.015065,10

#import DBSCAN
from sklearn.cluster import DBSCAN
dbscan = DBSCAN(eps=eps, min_samples=min_samples, metric='cosine', algorithm='brute')

label_dbscan = dbscan.fit_predict(feature_LBP)
```

```
Truth labels   0    1   2    3   4   5    6
labels
-1            59  166  93  393  90  60  105
 0             9   49  24  119  13   7   27
 1             2    2   1    8   2   0    3
 2             0    1   2    2   1   1    8
 3             1    6   0    0   1   0    0
 4             2    4   0    6   2   2    0
 5             1    5   0    0   0   0    0
 6             3    3   1    2   0   1    1
```

- Dựa trên Cross Table:
  - Có thể thấy số lượng điểm noise rất nhiều.
  - Phần nhiều hình được cluster vào chung một vùng dù khác class.

3.5 Agglomerative Clustering

In [242]:

```
# import library
from sklearn.cluster import AgglomerativeClustering

aggModel = AgglomerativeClustering(n_clusters=10)

label_agglomerative = aggModel.fit_predict(feature_LBP)
```

Cross table

```
Truth labels   0   1   2   3   4   5   6
labels
0              1  14   7  48   5   4  10
1             19  39  12  91   9   8  28
2             13  60  21  69  21   9  20
3             16  16  31  71   8   4  10
4              2  12  17  60  17   5  18
5              3  21   4  39   8   7   7
6              4   7  12  79  10   9  20
7              4  24  11  40  14   9  12
8              4  20   2  19   7  11   8
9             11  23   4  14  10   5  11
```

- Dựa trên Cross table:

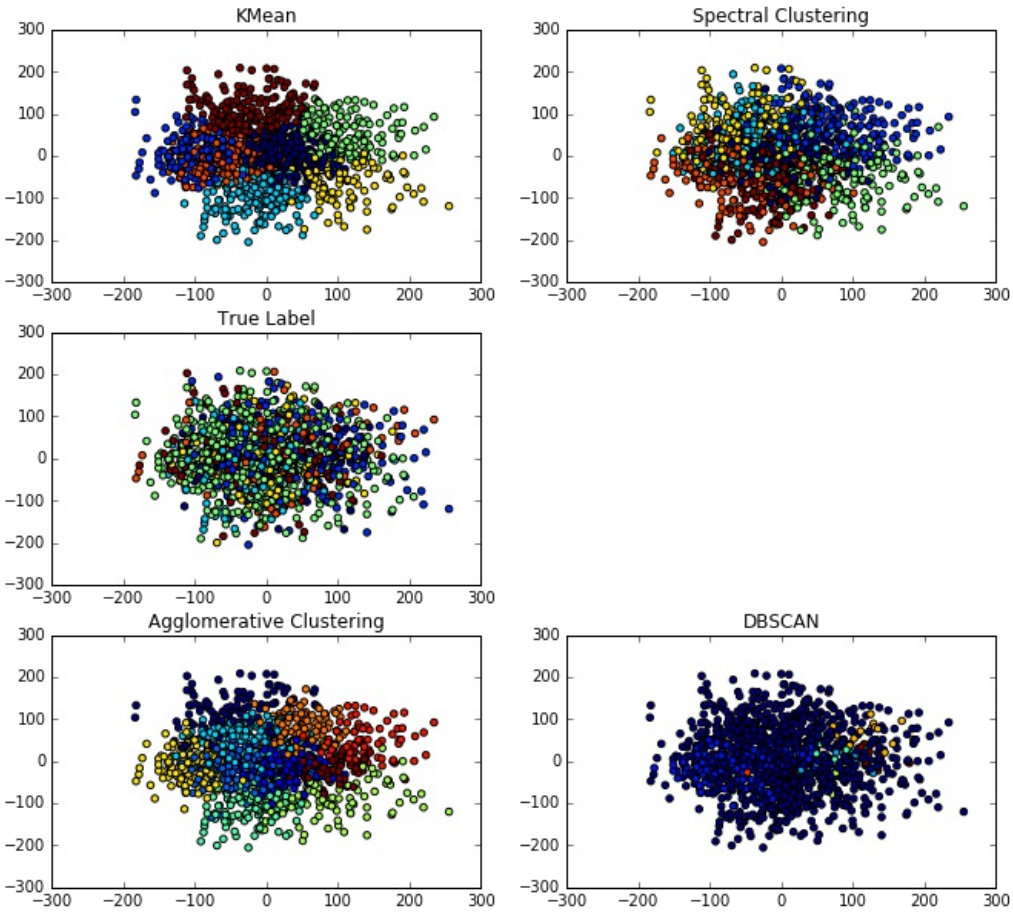
    - Nhận thấy kết quả cluster không hiệu quả.
    - Các hình được cluster đồng đều các vùng. Không xác định được đâu là cluster đúng của một class.
- Dựa vào data visualization phần nào đó (không chắc chắn, do giảm số chiều để visualize) cho thấy dữ liệu hiện tại được gom tụ thành một nhóm, mật độ cao ở trung tâm và hơi rời rạc ở rìa. Nên do đặc thù lan của DBSCAN không thể cluster tốt.

3.6 Visualize kết quả của thuật toán phân lớp

3.6.1 PCA

In [244]:

```
pca = PCA(n_components=2).fit_transform(feature_LBP)
```



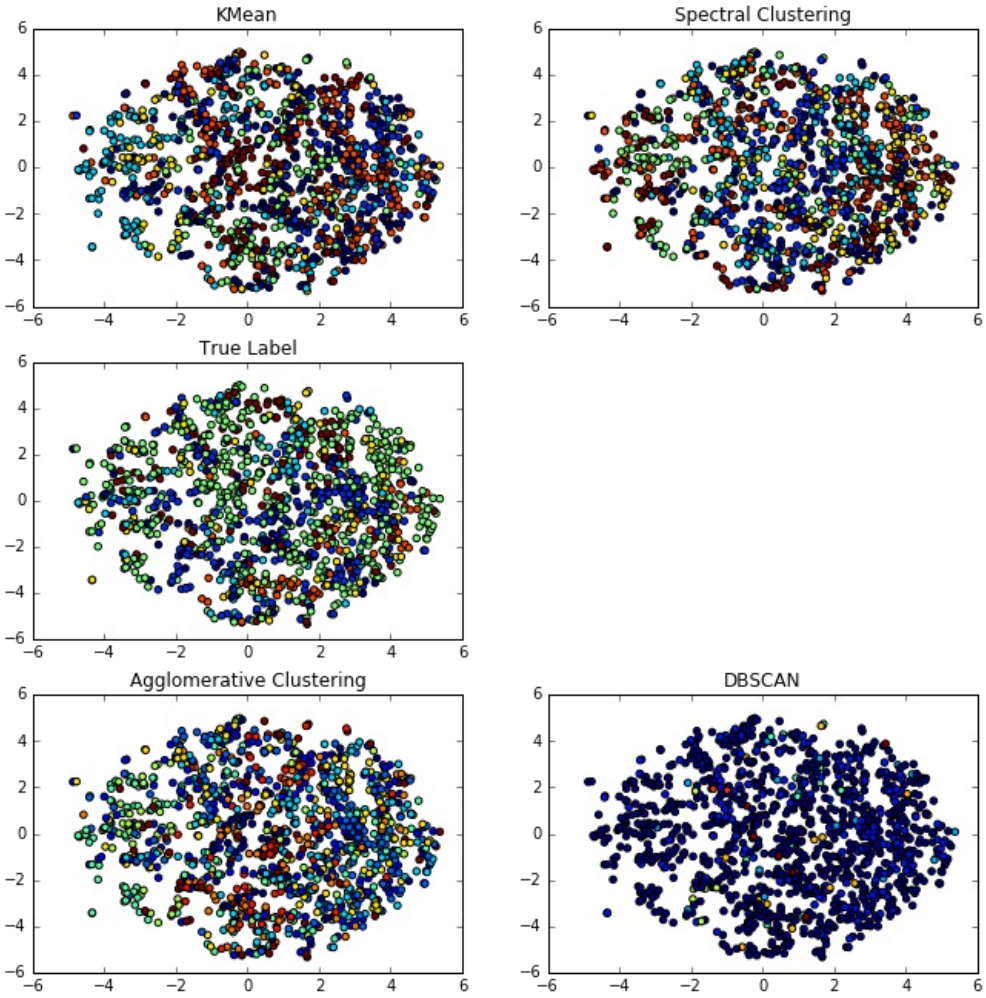
- Từ hình trên cho thấy, khi sử dụng thuật toán PCA để giảm số chiều và visualize data gây khó khăn cho việc xác định vùng bằng mắt (True Label).
- Biểu đồ True Label cho thấy các class được trộn lẫn vào nhau, trong khi đó KMean và Agglomerative Clustering lại cho ra các vùng riêng biệt. Spectral clustering cho kết quả khả quan hơn, khi các vùng có trộn lẫn, có vẻ giống với True Label.

3.6.2 Visualize by T-SNE

- T-SNE giảm số chiều của data về số liệu 2 chiều.

In [246]:

```
tnse = TSNE_model.fit_transform(faces.data)
```



3.7 Evaluate clustering algorithm

- Sử dụng các hệ đo lường để đánh giá thuật toán: Homogeneity, Completeness, V-measure, Adjusted Random, Adjusted Mutual Information.

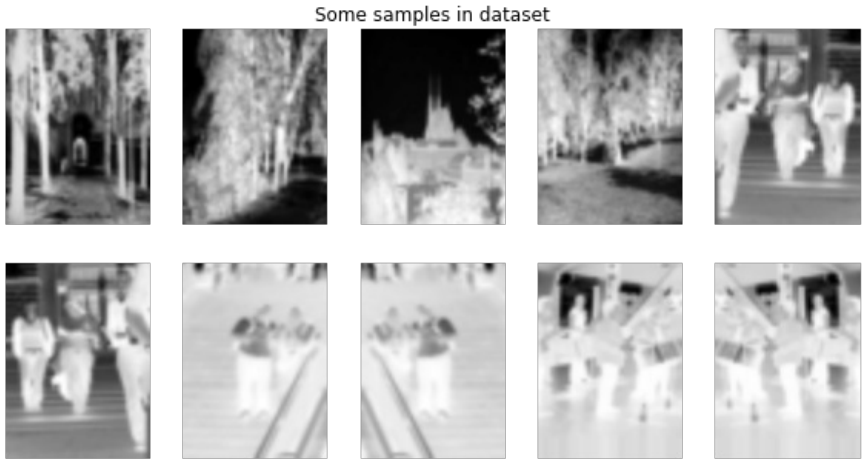
In [250]:

```
compareAlgorithm({'KMean':label_kmean, 'Spectral':label_spectral,'DBSCAN':label_dbscan,'Agglomerative':label
_agglomerative},\
                 faces.target, feature_LBP)
```

```
#Sample: 1288 #Class: 7 #feature: 256
_____________________________________________________________________________________
init  homo compl v-meas ARI AMI silhouette
Spectral  0.043 0.037 0.040 0.018 0.030 0.072
KMean     0.039 0.035 0.037 0.014 0.028 0.119
DBSCAN    0.022 0.046 0.029 -0.004 0.011 -0.237
Agglomerative 0.043 0.032 0.037 0.014 0.023 0.042
_____________________________________________________________________________________
```

4. Bài tập 4: Human Dataset

- Mục tiêu: Gom nhóm hình có sự xuất hiện của con người và không có con người.
- Dataset: INRIA Person Dataset.

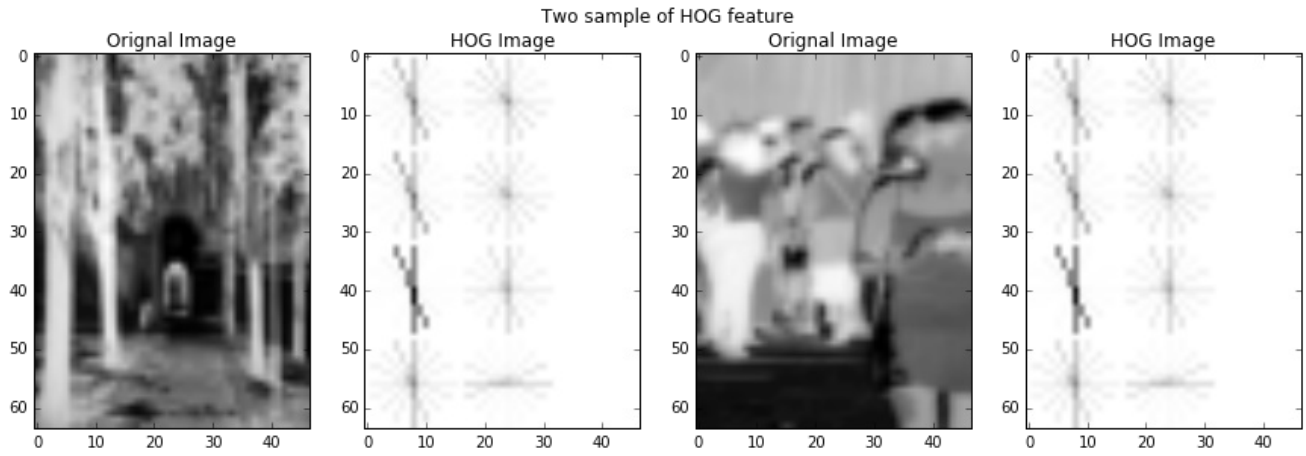


- Dataset gồm 1044 64x47. Gồm 2 class: 558 hình person và 486 hình non-person
- Tiến hành rút trích HOG feature từ mỗi hình và lưu lại.

4.1 Extract HoG Feature

- Sử dụng hàm HoG của thư viện Skimage để thực hiện rút trích đặc trưng HoG từ mỗi ảnh.

```
c:\python34\lib\site-packages\skimage\feature\_hog.py:119: skimage_deprecation: Default value of `block_norm
`=='L1` is deprecated and will be changed to `L2-Hys` in v0.15
  'be changed to `L2-Hys` in v0.15', skimage_deprecation)
```



4.2 Áp dụng thuật toán KMean, Spectral, Agglomerative Clustering

KMean

```
Cross Table
```

True Label	0	1
label		
0	219	396
1	267	162

Spectral Clustering

```
Cross Table
_____________________________________________________
True Label    0    1
label
0           210  367
1           276  191
```

Agglomerative Clustering

```
Cross Table
_____________________________________________________
True Label    0    1
label
0           152   62
1           334  496
```

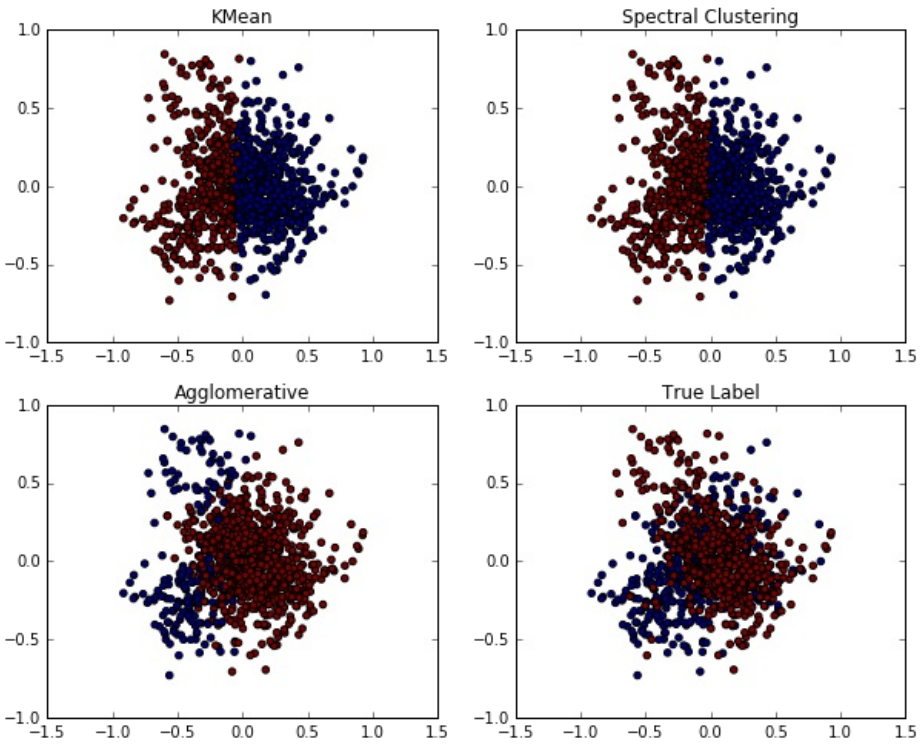
- Nhận thấy cả 3 thuật toán đề không cho ra kết quả tốt. Khó xác định giữa 2 class.

Visualize kết quả

PCA

In [263]:

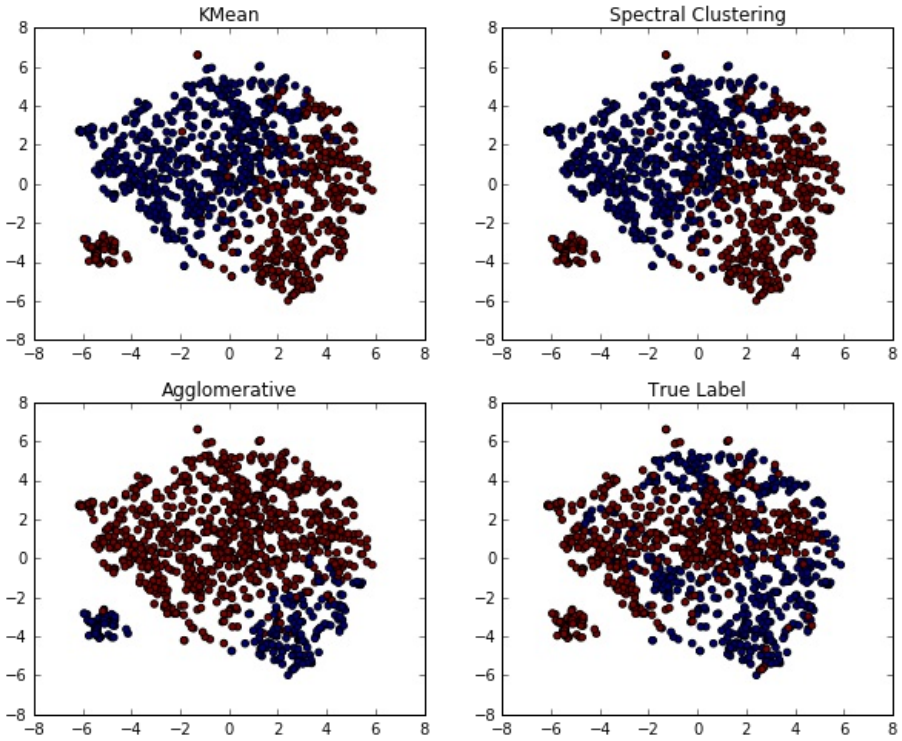
```
pca = PCA(n_components=2).fit_transform(feature_HOG)
```



T-SNE

In [265]:

```
tnse = TSNE_model.fit_transform(feature_HOG)
```



Evaluate

In [267]:

```
compareAlgorithm({'KMean':label_kmean, 'Spectral':label_spectral,'Agglomerative':label_agglomerative},\
                 target, feature_HOG)
```

```
#Sample: 1044 #Class: 2 #feature: 64
_____________________________________________________________________________________
init  homo compl v-meas ARI AMI silhouette
Spectral  0.037 0.038 0.037 0.053 0.037 0.110
KMean     0.050 0.051 0.051 0.072 0.050 0.107
Agglomerative 0.046 0.062 0.053 0.056 0.045 0.116
_____________________________________________________________________________________
```

Nhận xét:

- Kết quả không tốt. Với dữ liệu khó có thể cluster thành 2 nhóm phân biệt person và non-person.
- Nguyên nhân: có thể do input đầu vào là hình grayscale, một số hình bị quá sáng, mất mát thông tin trong quá trình chuyển ảnh màu sang ảnh grayscale.

5. Tham Khảo

1. http://scikit-learn.org/stable/modules/clustering.html
2. INRIA Person Dataset
3. Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu (1996). "A density-based algorithm for discovering clusters in large spatial databases with noise".
4. A demo of K-Means clustering on the handwritten digits data