Clustering

Machine Learning in Computer Vision

This report is what I've learn about clustering in Machine Learning and Computer Vision.

It is not a document about clustering, I just write down what I found out and try to prove it through examples, tests with skicit-learn so please feel free to correct me if something goes wrong.

A lot of images in this report are borrowed from Wiki and this online courses.

Machine Learning is Fun!

More information: please email me at 14520040@gm.uit.edu.vn

Overview

Let's start with some numbers from KDnuggest.

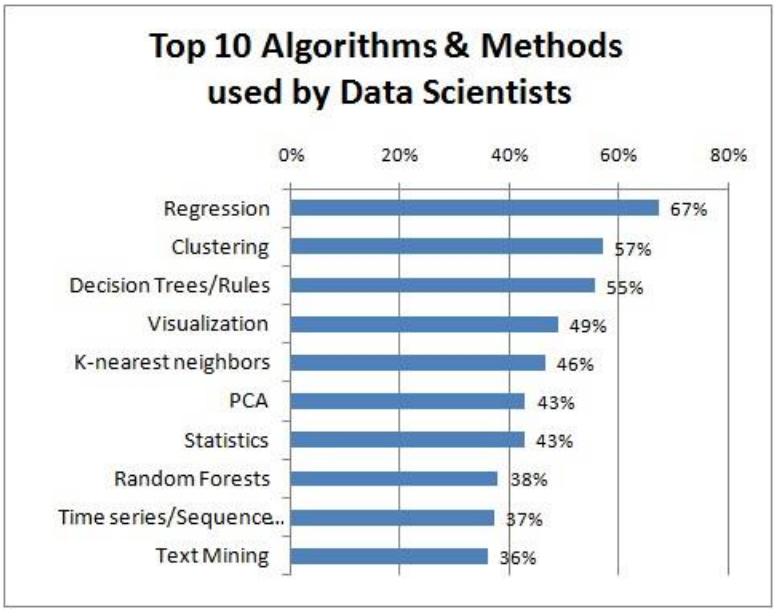


Figure 0. Top 10 algorithms and methods used by data Scientists

Hm… Clustering seems be an interesting stuff. Let's find out why.

What is clustering?

Clustering is the task of dividing data (samples) into a number of groups (clusters). Samples in the same group are more similar to each other than to those in other groups.

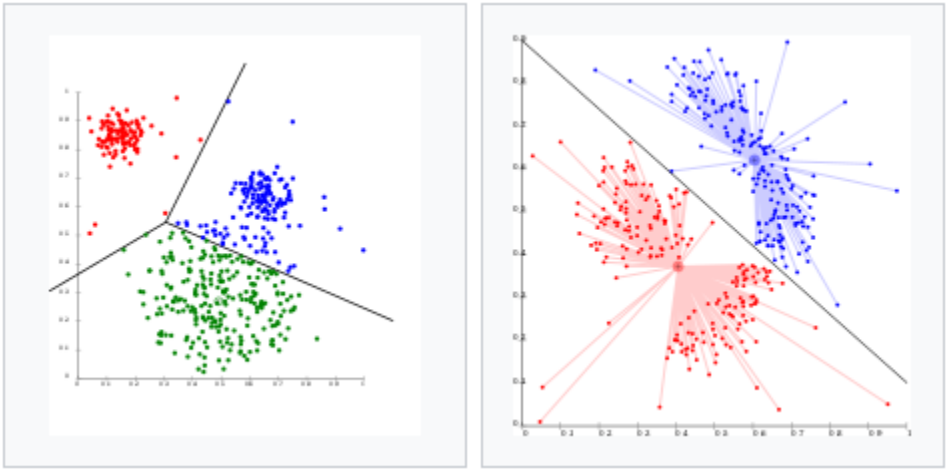


Figure 1. Clustering examples

Note: Since the task of clustering is subjective. It can be used in different ways to achieving different goals.

In the example below, we can use clustering for either grouping by gender or by relationship between people.

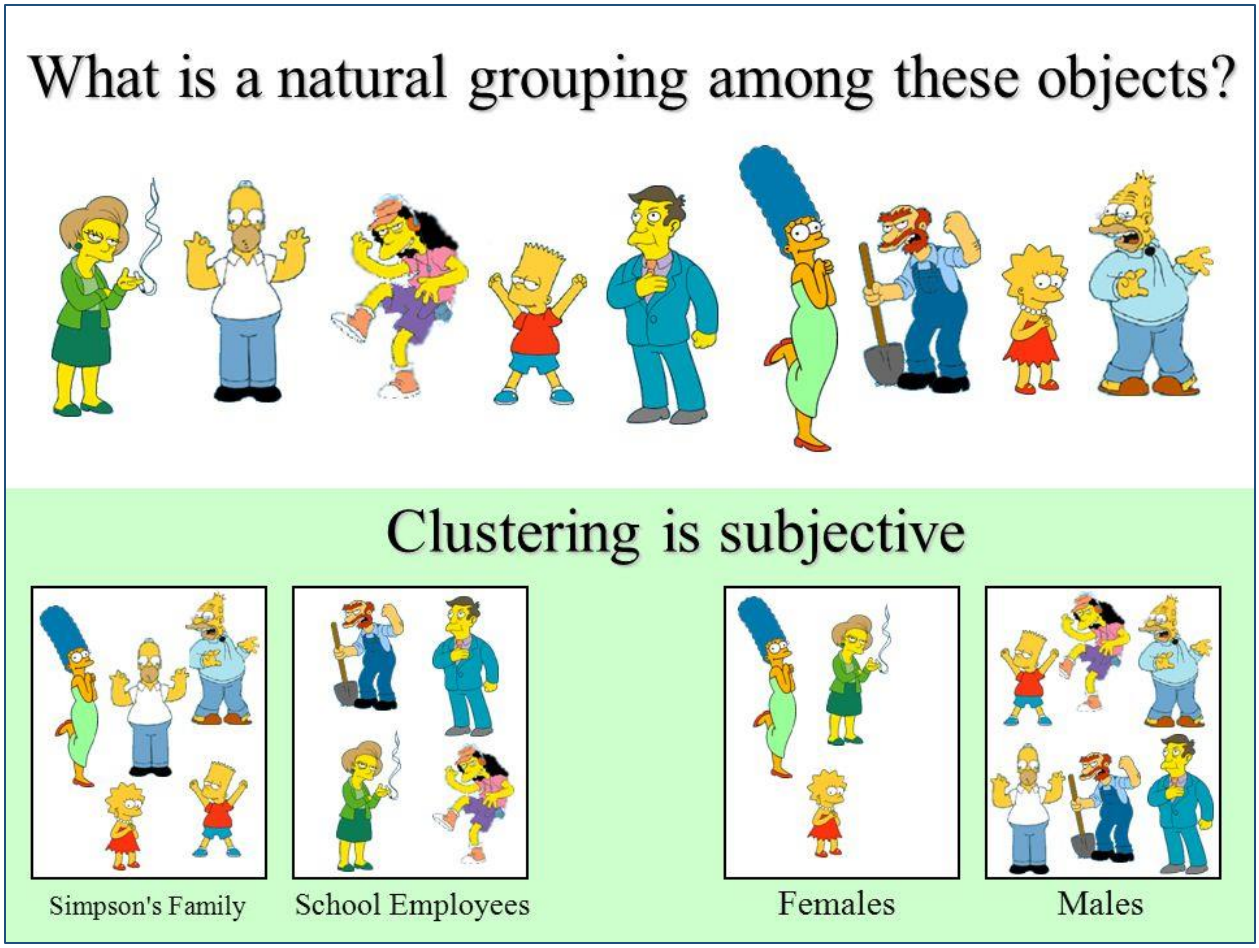


Figure 2. Clustering is subjective

Why do I need clustering in Machine Leanring?

In Machine learning, Clustering is used to find structure of unlabeled data. It is the most common form of unsupervised learning.

Given a dataset we don't know anything about, a clustering algorithm can discover groups of objects where the average distances between the members of each cluster smaller than to those in other cluster.

How to measure performance a clustering algorithm?

There are two common ways to measure performance a clustering algorithm: Internal measure and External measure.

Internal measure: measuring without any external information

In comon, Objective function value is used.

Ex. Minimal SSD (Sum of square distance) or SSE (Sum of square distance error) between every data point in a cluster C to the center point c_i.

External measure: measuring agreement between two partions – the ground-truth labels and the result of clustering task. (used when we have data with ground-truth labels)

There are a lot of external functions can be used. It all calculates using different bettwen the structure of ground-truth and clustering reulst. Below is some of those.

- Clustering accuracy (CA): Find a maximum bipartite matching (using Hungarian Algorithm or Munkres Algorithm) between ground-truth labels and assigned labels.

$$\mathrm{CA} = \sum_{i=1}^{n} \delta(g_i, map(l_i))/n$$

- Normalized mutual information (NMI): Compare the distributions defined by ground-truth partitioning and clustering result.

$$\mathrm{NMI} = \frac{\sum_{p,q} n_{pq} \log \frac{n_{pq} n}{n_p n_q}}{\max\{\sum_p n_p \log \frac{n_p}{n}, \sum_q n_q \log \frac{n_q}{n}\}}$$

  n_p: Number of data points with p-th ground-truth label

  n_q: Number of data points with q-th assigned label

  n_{pq}: Number of data points with p-th ground-truth label and q-th assignedlabel

- Purity: $P = \frac{1}{n} \sum_q \max_p n_{pq}$

- Entropy: $E = \frac{1}{n \log_2 k} \sum_{p,q} n_{pq} \log_2 \frac{n_{pq}}{n_q}$

Note: Clustering task is just grouping so the labels from clustering result is just the name of the group, it will not be the same with the labels in ground truth and External functions above also do not use directly the labels value for calculating.

Ex. Labels_true = [1, 1, 2, 2, 3] // the ground-truth

Labels_pred = [3, 3, 1, 1, 2] // the clustering result

NMI (labels_true, labels_pred) = 1 //it means clustering absolutely right

As we can see, the clustering result is just a group name, not really a labels. The really important thing we can get from result is structure of data.

One more thing

I don't really know how those external functions were created. For now, I just use it as a measure.

The formulas are copied from here.

Which clustering algorithm should I use?

The answer is It depends on what data we have, and what we are going to do with the data.

There are several different clustering algorithms we can easy to find with just a name such as Kmeans, Spectral clustering, DBSCAN, etc.

In this project, I will go on detail about K-means and just an over view about other algorithms. The reason of choosing K-means will be metioned in below.

Then, what I can really do with clustering?

Hm… I can't even count. There are a lot of cool stuffs we can do with clustering.

We just need to remember that clustering is used to find out the struture of data. Therefore, whenever we have data and want to do something cool with it but we don't know what we are really doing, then just using clustering to have a better view.

Ex. Clustering can be used to group all the shopping items available on the web into a set of unique products (like eBay)

Or in the study of social networks, clustering may be used to recognize communities within large groups of people

Or even cluster analysis can be used to identify areas where there are greater incidences of particular types of crime.

Or at least, we can used it for school project (Ex. Image Segmentation)

K-means Clustering

Kmeans is a common clustering algorithm used in Computer Vision. I have no number for this one but just see how many Computer Vision courses have "Kmean" as an basic method for segmentation and orther tasks.

Basic idea

Minimize SSD between all points and their nearest clustering center. (This is an internal measure above).

Algorithm

1. Randomly initializes K cluster centers $c_1, c_2 \ldots c_k$
2. Determine points in each cluster.
   For each point p in dataset, find closest c_i, put p into cluster i
3. Given points in each cluster, calculate center c_i again.
4. If any c_i has changed, repeat step 2.

Ex. Implement Kmeans would be something like this (from a to f)

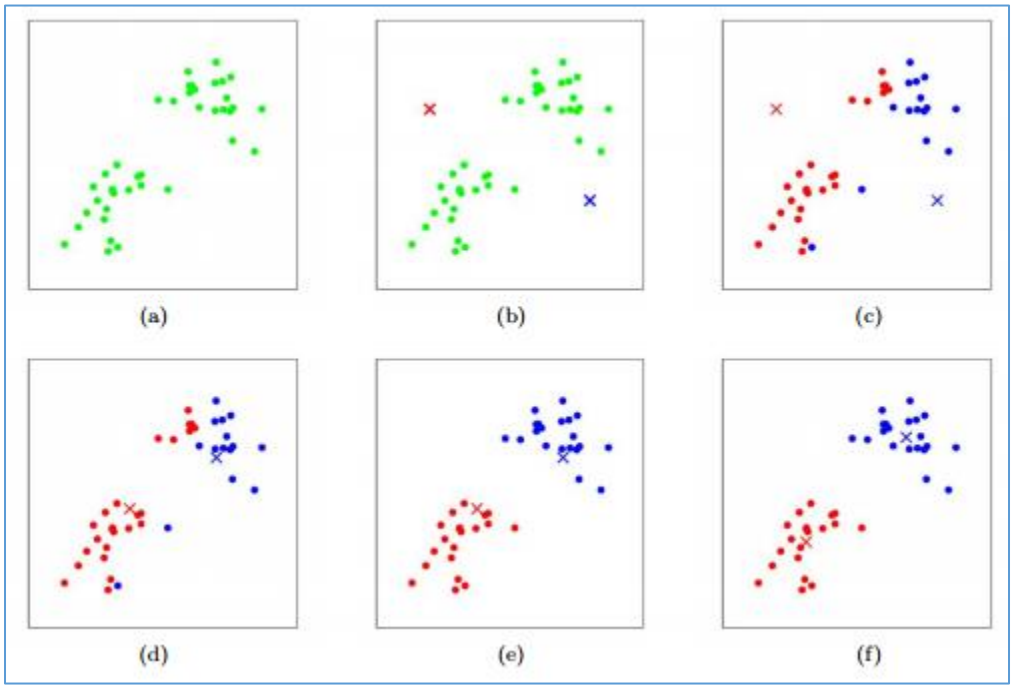


Figure 3. An example of Kmean clustering

Pros and Cons

Pros

1. Very simple algorthm, easy to implement, widely usage.
2. Converges to a local minimun of error function (that how the basic idea of Kmean effects).

Cons

1. Need to pick K.

➔ *It can be easy improved by picking diffrerent K and see which is good (depend on our goal of clustering).*

2. Sensitive to initialization

To see how initialization effects on Kmeans, I implement k-mean few times with a simple dataset, here is some result

i	time
1	7.20s
2	10.88s
3	6.67s
4	6.95s
5	9.36s
6	10.17s
7	9.20s
8	7.76s
9	4.80s

Figure 4. Execution time has changed by initialization with the same data.

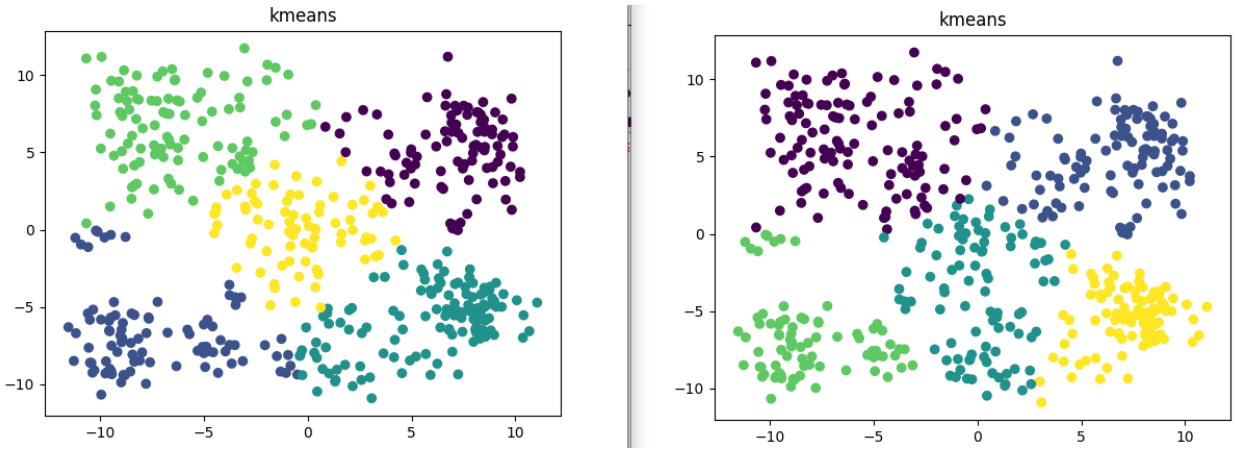
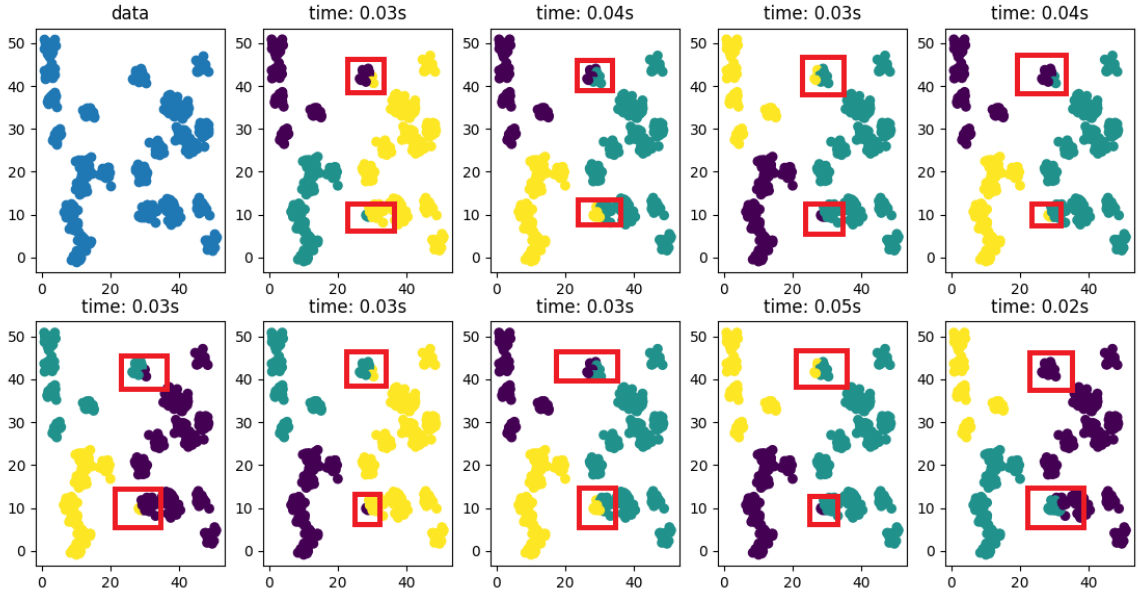
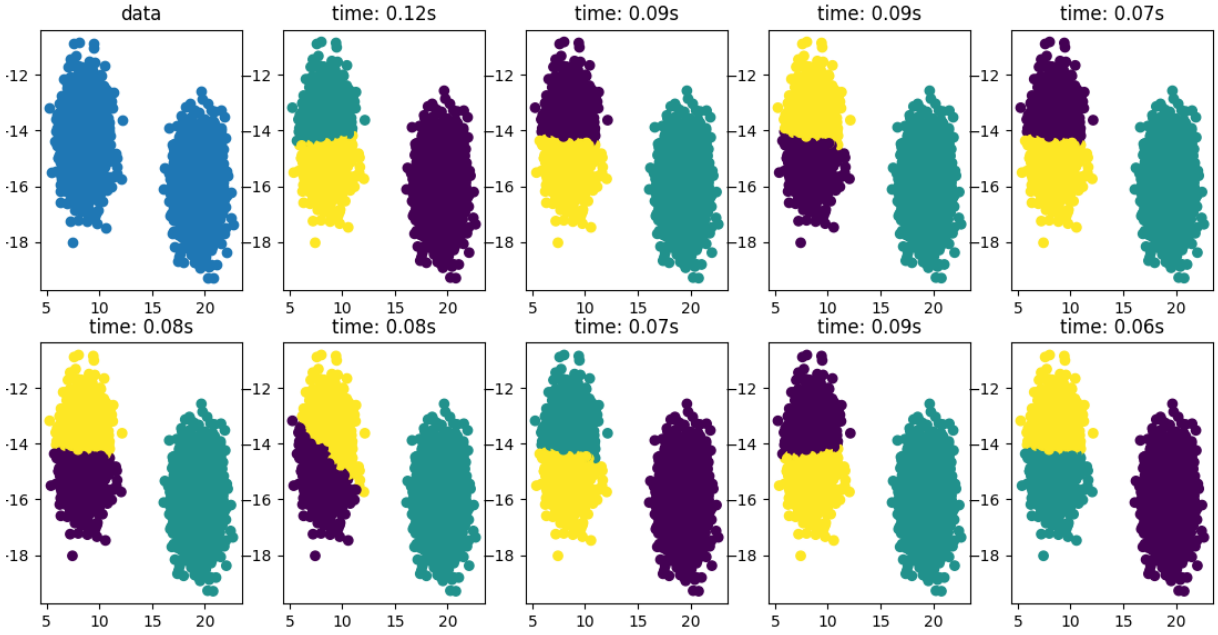


Figure 5. The result has also changed by initialization


data
time: 0.03s
time: 0.04s
time: 0.03s
time: 0.04s
time: 0.03s
time: 0.03s
time: 0.03s
time: 0.05s
time: 0.02s


data
time: 0.12s
time: 0.09s
time: 0.09s
time: 0.07s
time: 0.08s
time: 0.08s
time: 0.07s
time: 0.09s
time: 0.06s

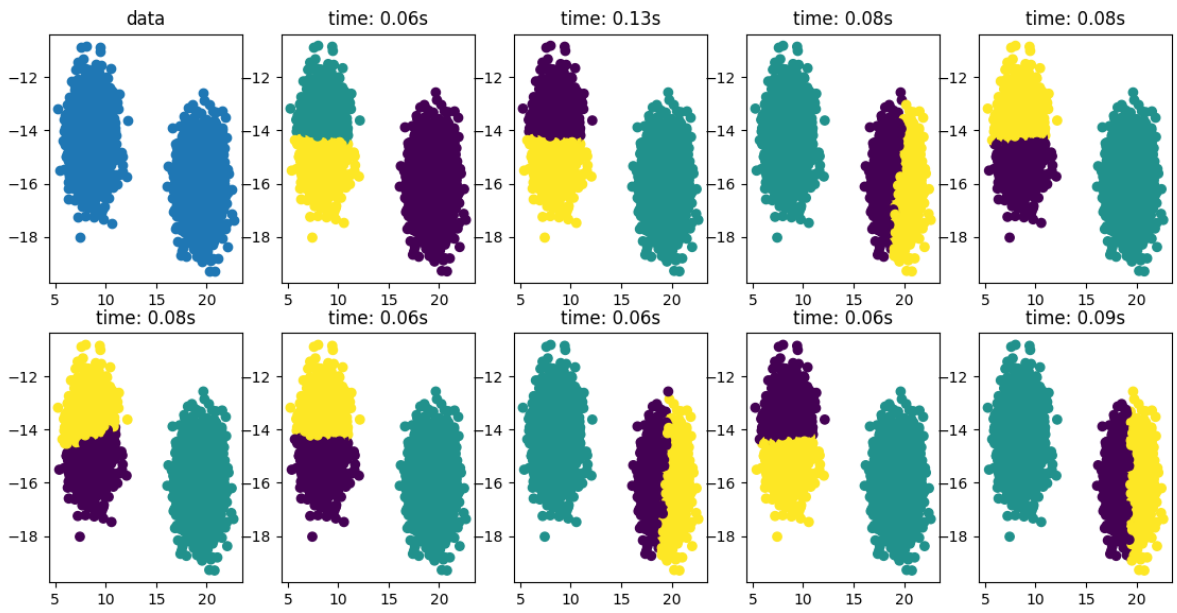


Figure 6. More tests for kmeans

To simplest way to avoid this problem I've found is re-runing Kmeans and "hope" the random initialization will be better in next time (just run it again if we feel the result is not good enough).

3. Only finds "spherical" clusters

   Let's see how is it

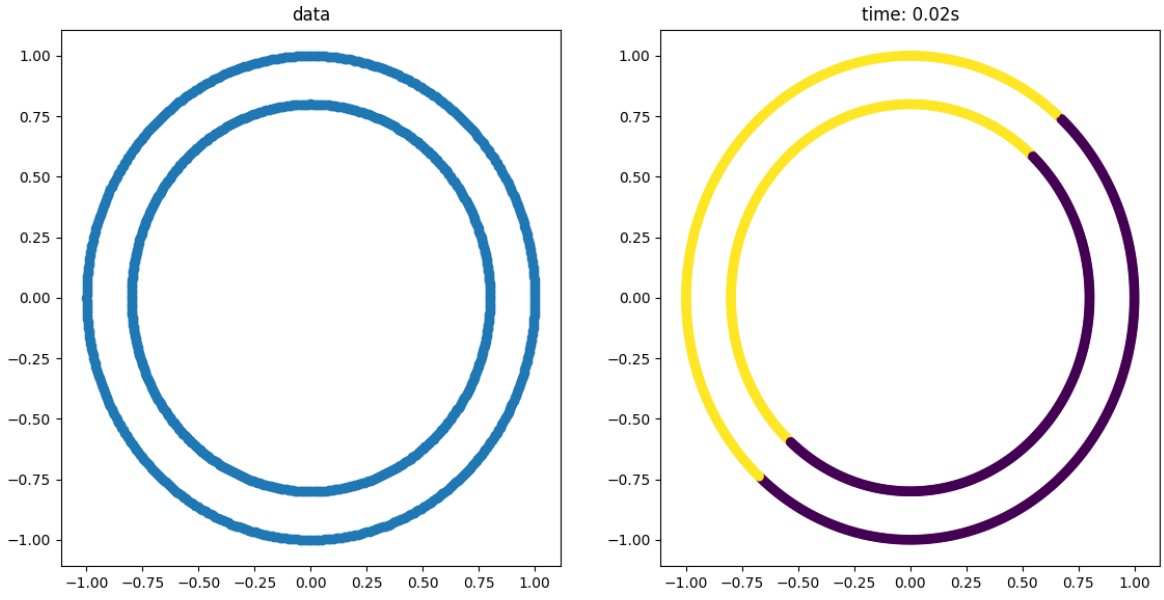


Figure 7. Kmeans only finds spherical clusters

We expects the result to be 2 cluster of 2 circles but Kmeans can not find them.

The only way to advoid this problem is using another clustering algorithm because this is exactly how K-means works.

4. Sensitive to outliers
   There are usually some outliers in data, it makes kmeans work worse. It would be something like the below example.

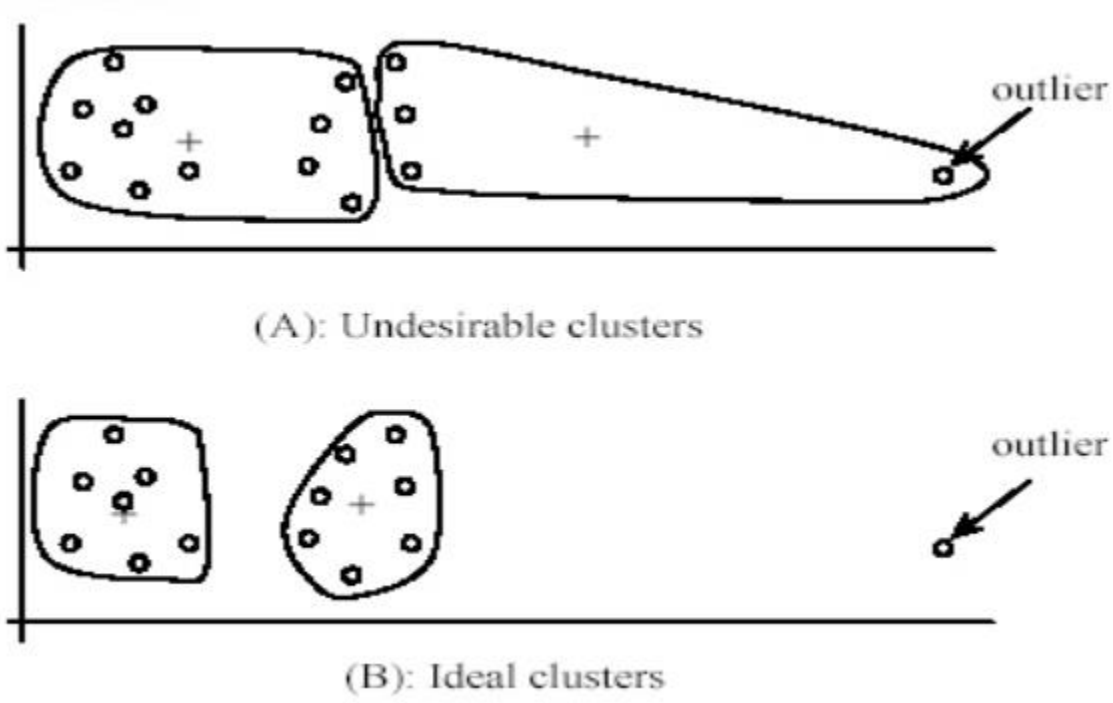


Figure 8. Outliers and Kmeans

The common way to deal with outlier is "Outlier Removal"- All we need to do is handle with outlier (remove them) in data first, then we can apply kmeans later.
Here is some blogs and articles for finding and dealing with outlier:
https://conversionxl.com/blog/outliers/
https://www.umiacs.umd.edu/~kmitra/files/talksPosters/HandlingOutliersMissingDataISI.pdf
http://cs.joensuu.fi/~villeh/35400978.pdf

I haven't read all of them, but in case I get trouble with outlier, they will be really helpful.

For now, I just need to know there are some way to deal with outliers if needed.

Some other clustering algorithms

Spectral Clustering

Introduction

The goal of spectral clustering is clustering data that is connected but not necessarily compact or clustered within convex boundaries (like Kmean).

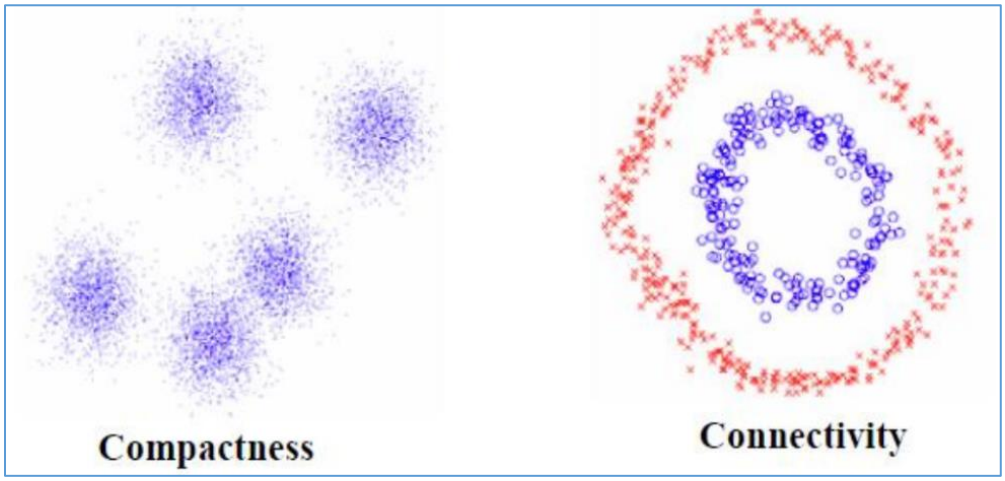


Figure 9. Compactness vs Connectivity

Where can I use it

Whenever we need clustering connectivity data and Compactness clustering (ex. Kmeans) can't do it well.

Segmenting objects from a noisy background using spectral clustering (Compactness Clustering like K-means can not do it).

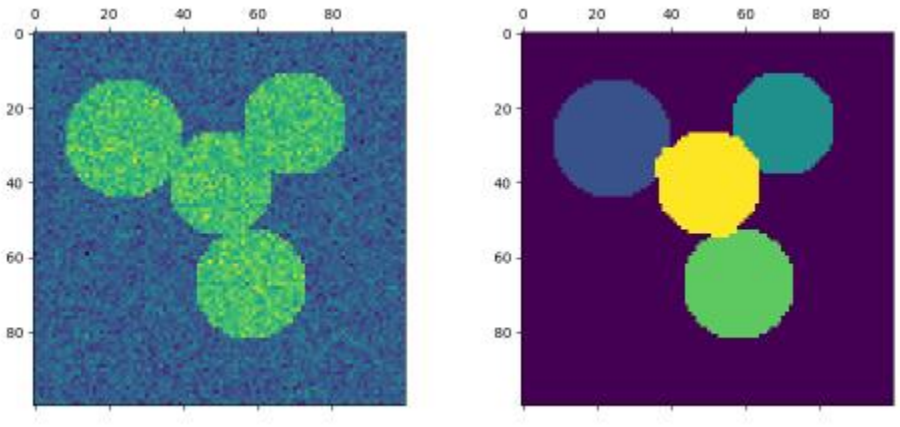


Figure 10. Noise data with spectral clustering

DBSCAN

Introduction

The DBSCAN algorithm views clusters as areas of high density separated by areas of low density.

Cluster found by DBSCAN can be any shape, as opposed to k-means which assumes that clusters are convex shaped.

DBSCAN always generating the same clusters when given the same data in the same order. However, the results can differ when data is provided in a different order.

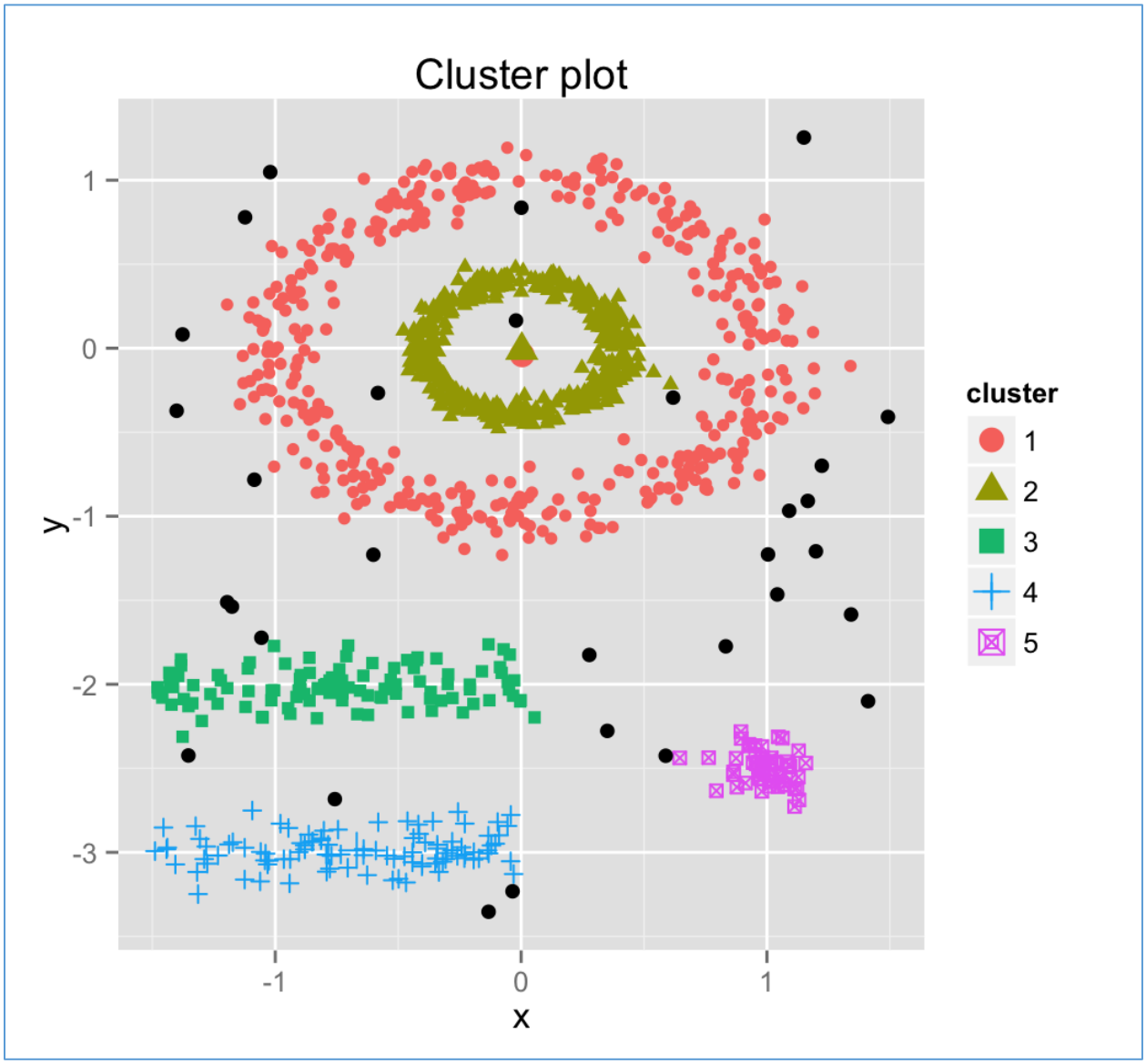


Figure 11. DBSCAN example

Where can I use it

DBSCAN is usually used in datamining.

Note

One important thing when using DBSC is choosing a suitable parameter value. (the clustering result is really sensitive with those value)

Agglomerative Clustering

Introduction

Agglomerative is one of clustering algorithms in Hierarchial clustering family.

It performs a hierarchical clustering using a bottom up approach: each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy.

In the general case, the complexity of this algorithm is $O(n^3)$

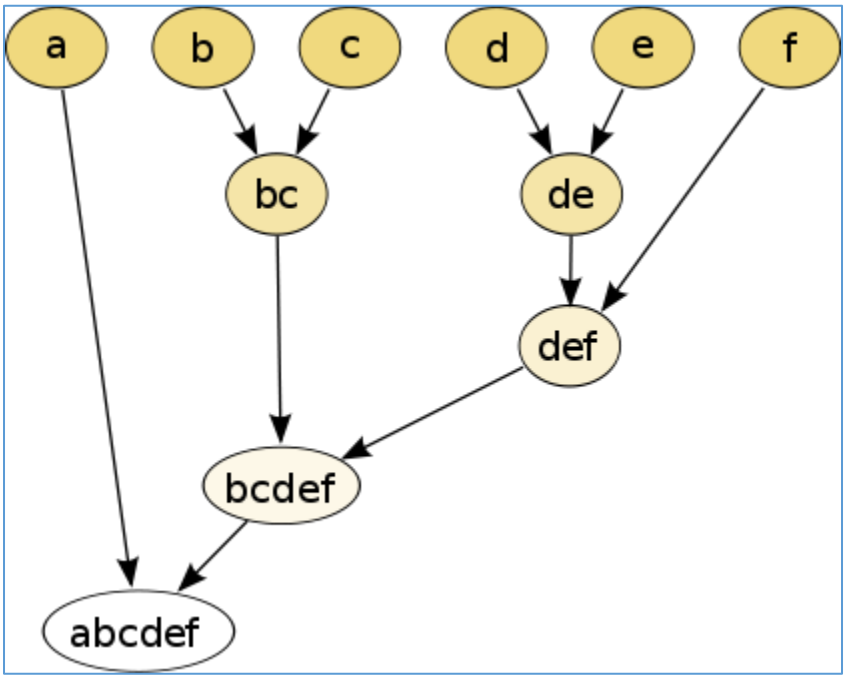


Figure 12. Agglomerative Clustering example

Where can I use it

Agglomerative clustering is used in data mining and statistics.

Note

I don't really know how this algorithm work, above information is from wiki.

Simple Comparison

Firstly, let's see how those clustering algorithms work on different data set with [scikit-learn]

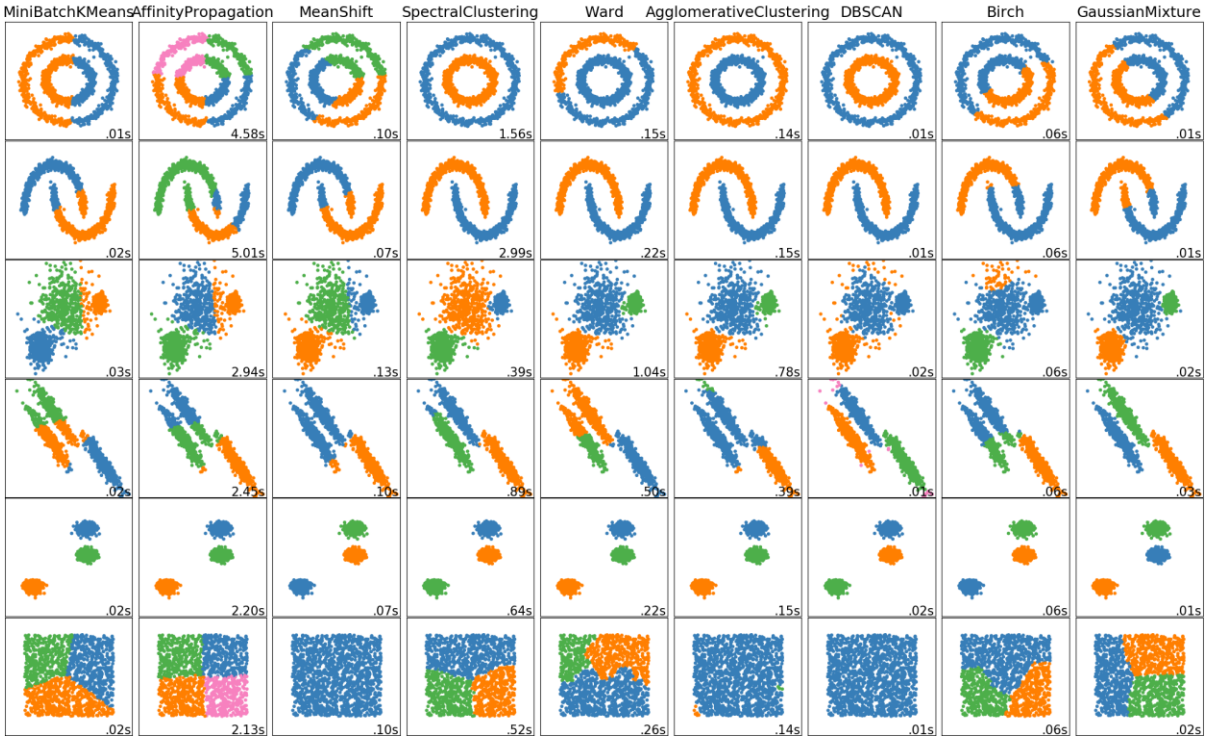


Figure 13. Clustering algorithm comparison

We can easily choose a algorithms which is suitable with our problem by looking at above example and finding some different attributes of those.

For practice, I use scikit-learn to implement those algorithm with different dataset (digits, faces) and some External measure for evaluation. Here is some result

Hand-written digits data

```
n_samples       : 1797
n_cluster       : 10
feature space   : 64
--------------------------------------------------------------------------------
cluster         time    homo    compl   v-meas  ARI     AMI
kmeans          0.35s   0.735   0.742   0.739   0.667   0.732
spectral        1.95s   0.713   0.717   0.715   0.625   0.710
DBSCAN          0.08s   0.703   0.739   0.721   0.495   0.700
Agglomerative   2.72s   0.858   0.879   0.868   0.794   0.856
--------------------------------------------------------------------------------
```

Figure 14. External measure for clustering on hand-written digits data

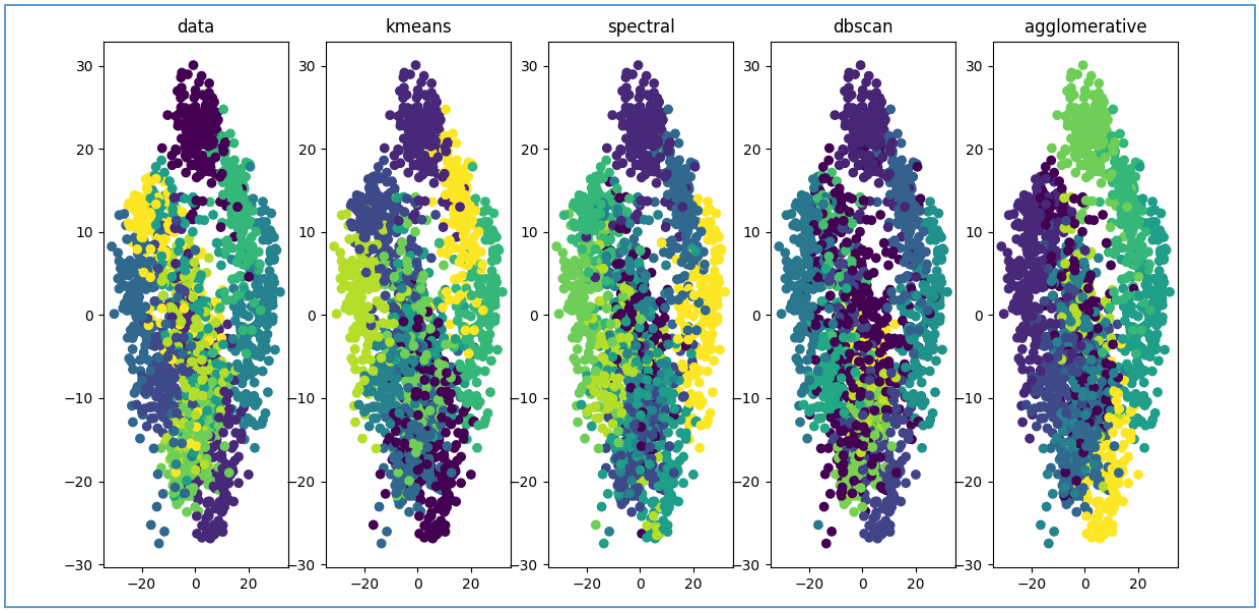
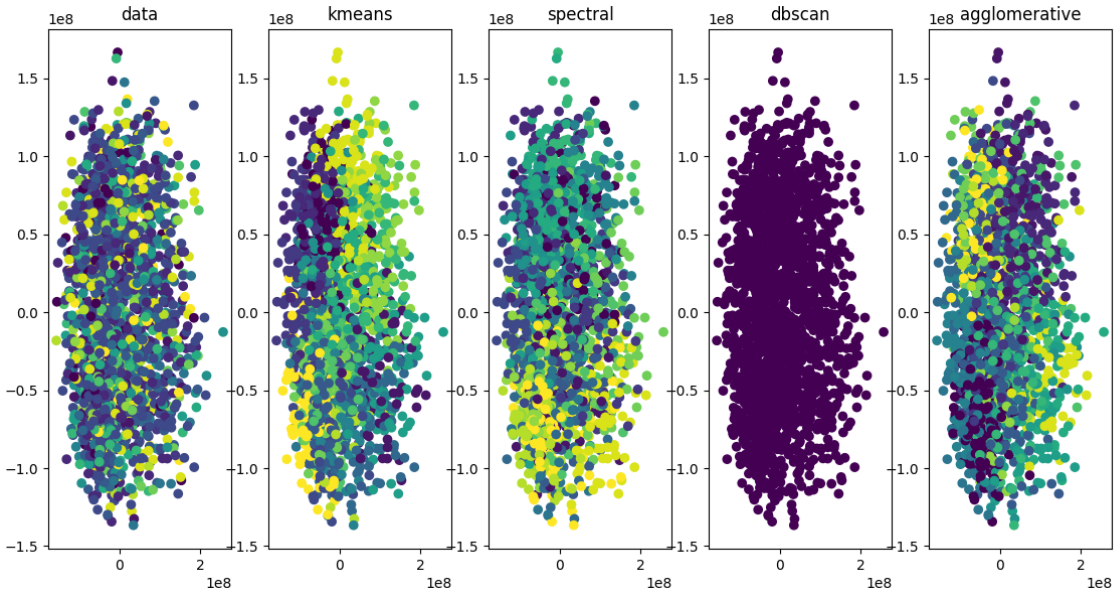


Figure 15. Clustering result after compressing with PCA (convert to 2D space)

Dataset face and LBP feature

```
n_samples       : 1867
n_cluster       : 19
feature space   : 2914
-------------------------------------------------------------------------------
cluster         time    homo     compl    v-meas   ARI      AMI
kmeans          16.34s  0.040    0.035    0.037    0.000    0.002
spectral        3.63s   0.040    0.035    0.037    0.001    0.002
DBSCAN          1.28s   -0.000   1.000    -0.000   0.000    0.000
Agglomerative   8.41s   0.044    0.039    0.042    -0.001   0.006
-------------------------------------------------------------------------------
```

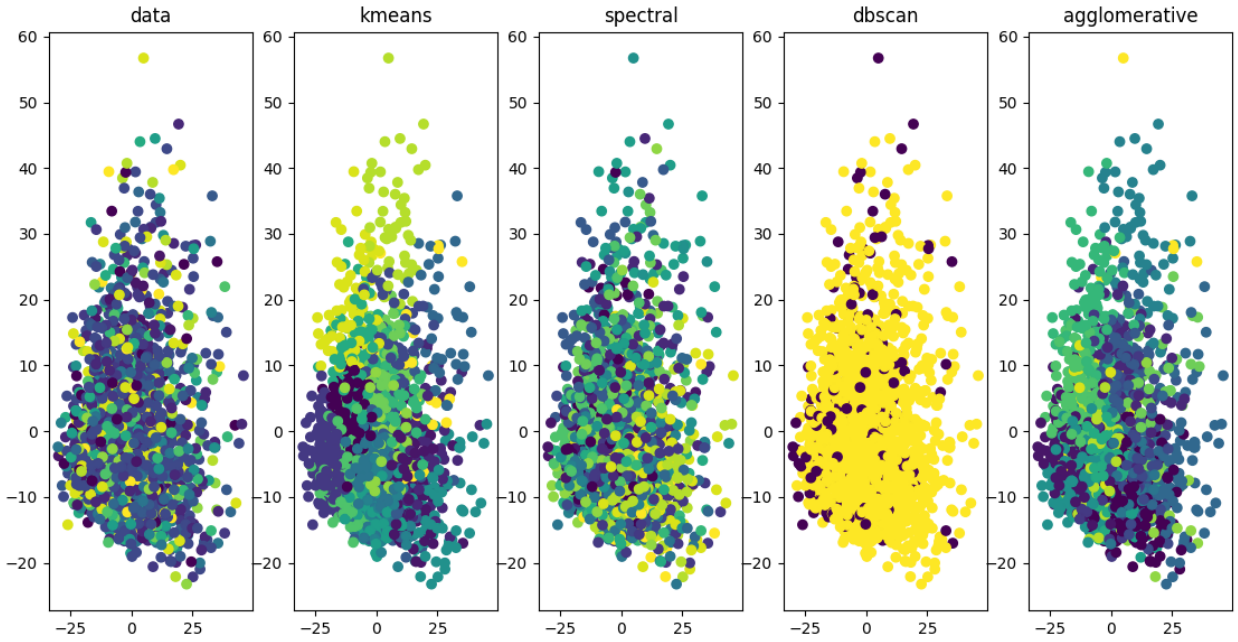


As the result, DBSCAN when I use default parameter value doen't work. That's how important paramter value is.

Kmeans is too slow on a large dataset.

Dataset face and HOG feature

```
n_samples       : 1867
n_cluster       : 19
feature space   : 2914
--------------------------------------------------------------------------------
cluster         time    homo    compl   v-meas  ARI     AMI
kmeans          14.19s  0.039   0.035   0.037   0.002   0.002
spectral        3.51s   0.036   0.032   0.034   0.000   -0.001
DBSCAN          1.32s   0.003   0.020   0.005   0.001   0.001
Agglomerative   8.62s   0.038   0.035   0.036   0.000   0.002
--------------------------------------------------------------------------------
```



Now, DBSCAN works but the result is still really bad.

Conclusion

To remember

Clustering is grouping similar data.

Use it whenever we need to know the structure of data.

When using clustering, pay attention on: **parameter value**, **feature space** and **distance function**

Different algorithms have different advantages and disadvantages, consider to choose a suitable one for different problems.