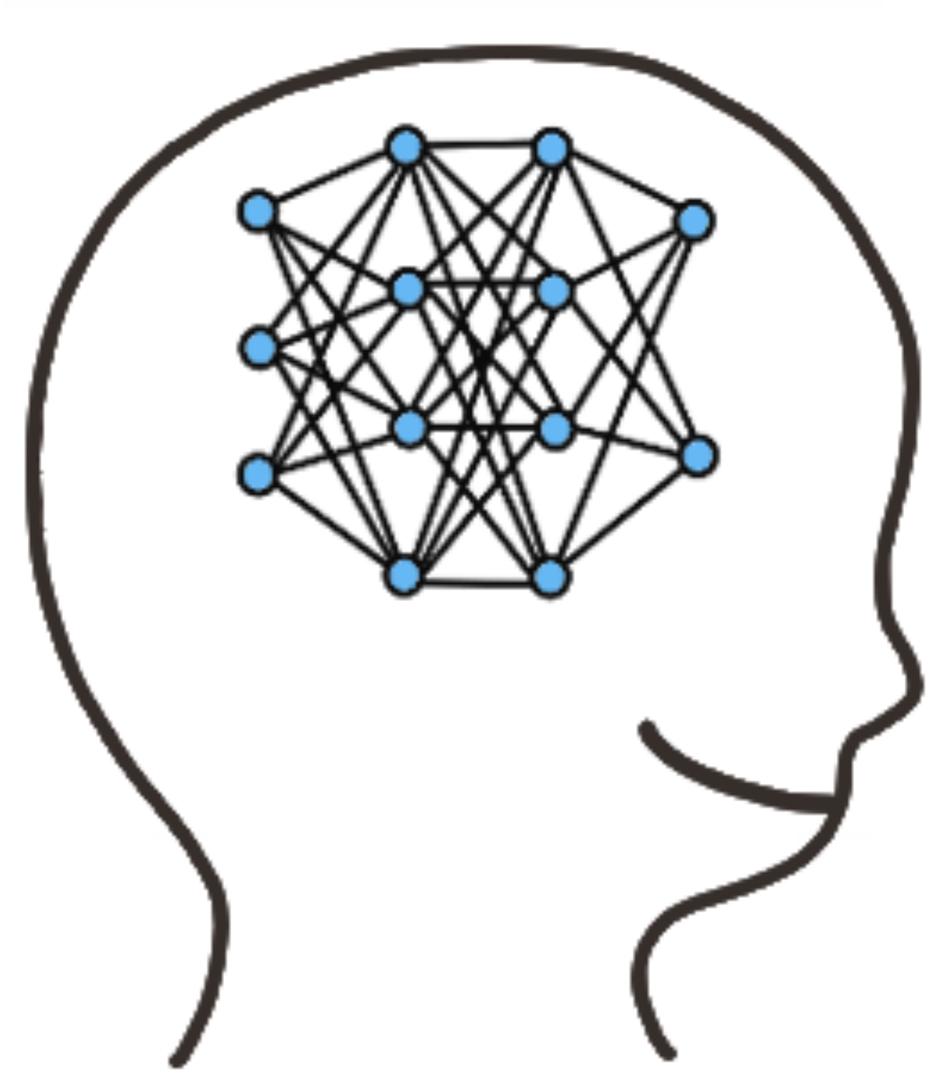


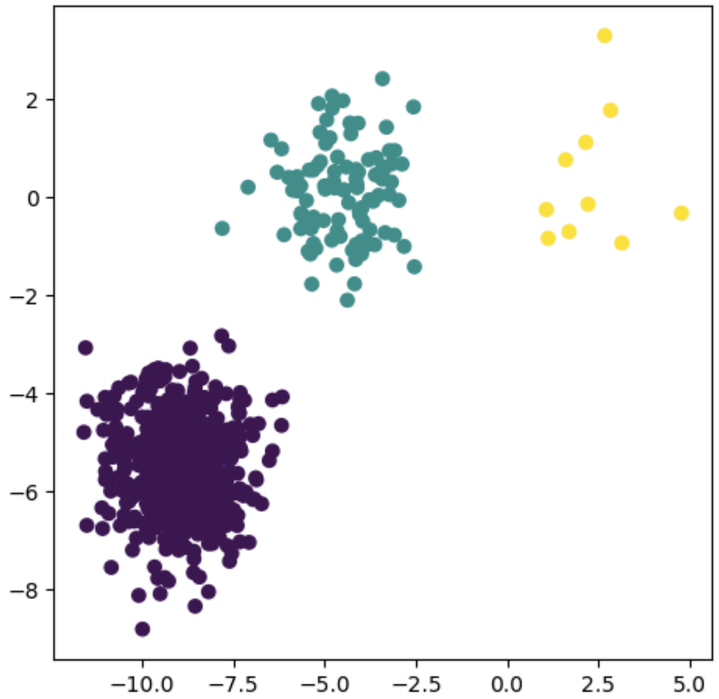
非監督式學習-分群

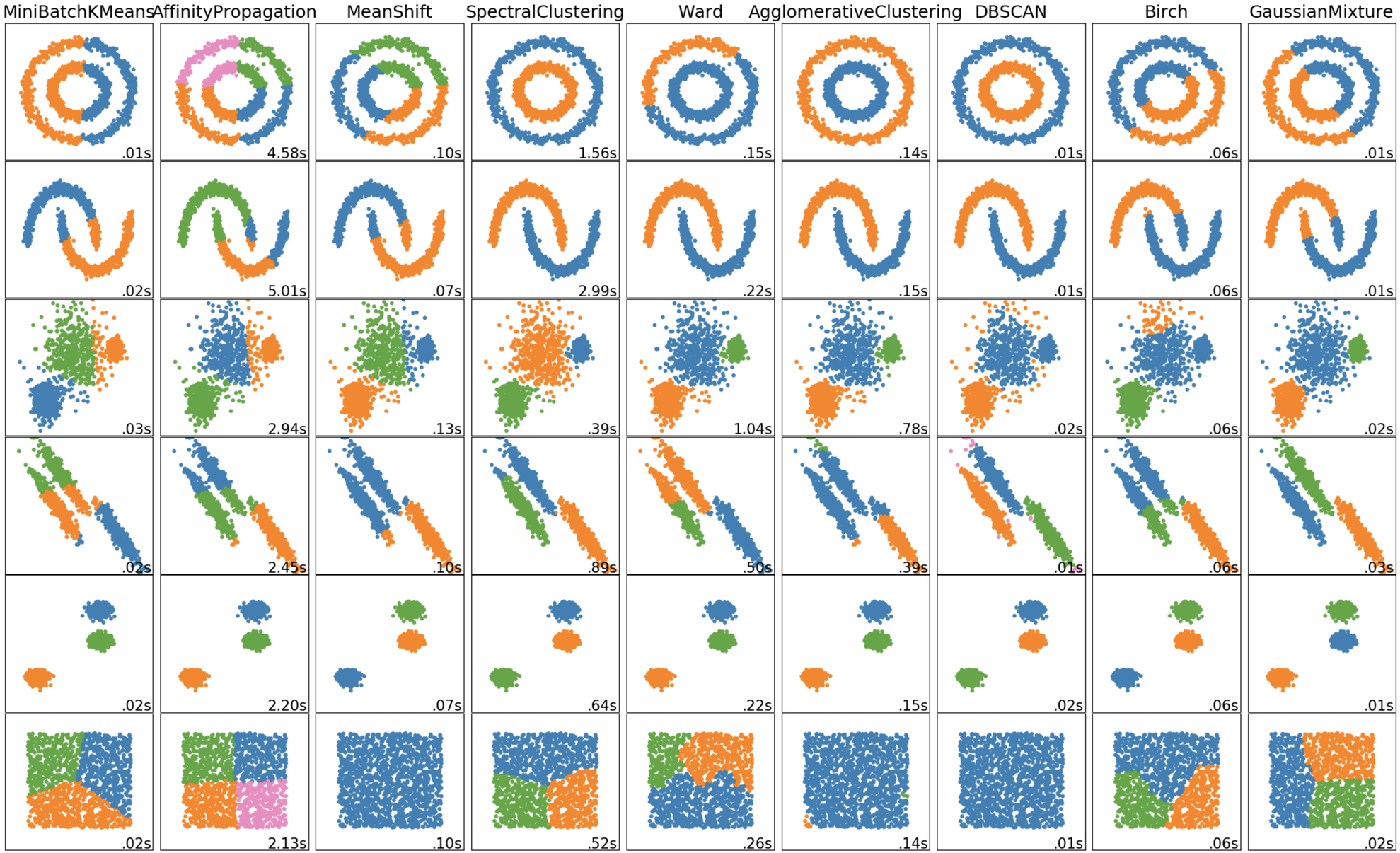
Unsupervised Learning-Clustering

分群 (Clustering)

- 非監督式：沒有y，只有X(features)

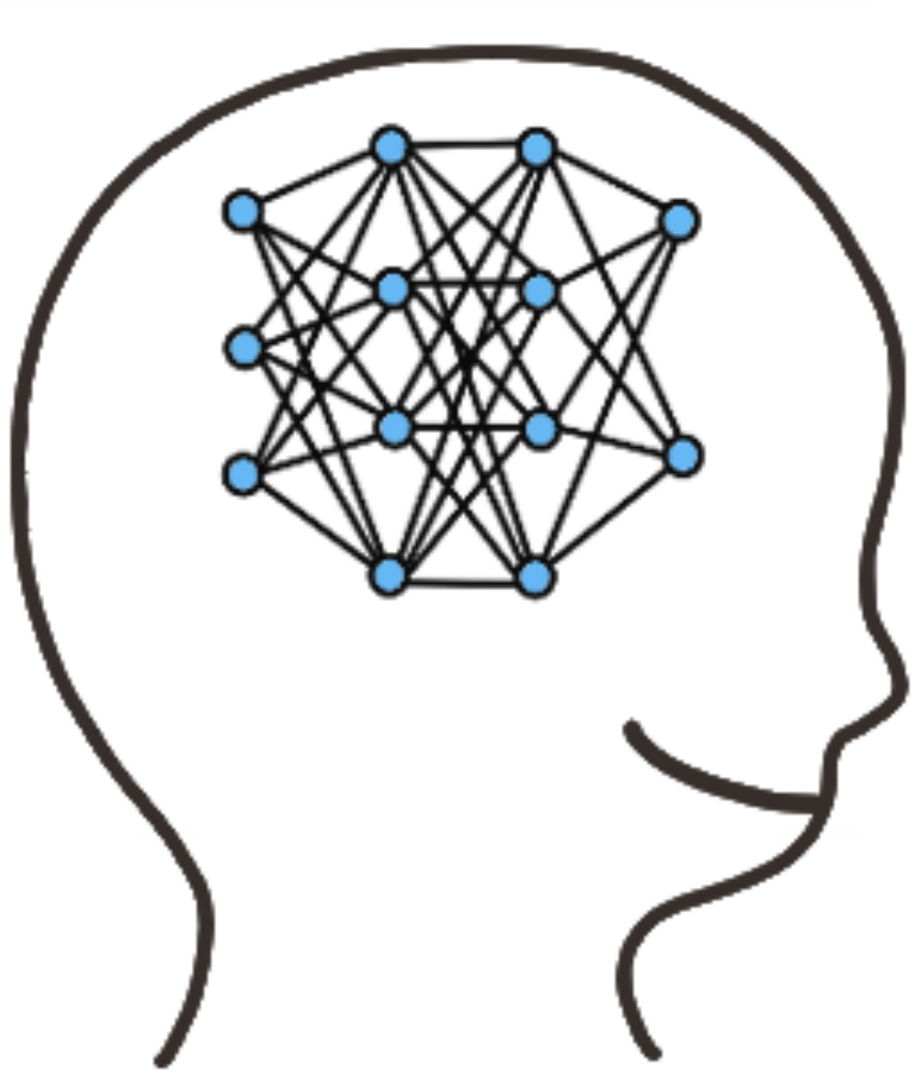
- 透過資料彼此的距離分群




MiniBatchKMeans
AffinityPropagation
MeanShift
SpectralClustering
Ward
AgglomerativeClustering
DBSCAN
Birch
GaussianMixture
.01s 4.58s .10s 1.56s .15s .14s .01s .06s .01s
.02s 5.01s .07s 2.99s .22s .15s .01s .06s .01s
.03s 2.94s .13s .39s 1.04s .78s .02s .06s .02s
.02s 2.45s .10s .89s .50s .39s .01s .06s .03s
.02s 2.20s .07s .64s .22s .15s .02s .06s .01s
.02s 2.13s .10s .52s .26s .14s .01s .06s .02s

Clustering with sklearn

Method name	Parameters	Scalability	Usecase	Geometry (metric used)
K-Means	number of clusters	Very large `n_samples`, medium `n_clusters` withMiniBatch code	General-purpose, even cluster size, flat geometry, not too many clusters	Distances between points
Affinity propagation	damping, sample preference	Not scalable with n_samples	Many clusters, uneven cluster size, non-flat geometry	Graph distance (e.g. nearest-neighbor graph)
Mean-shift	bandwidth	Not scalable with `n_samples`	Many clusters, uneven cluster size, non-flat geometry	Distances between points
Spectral clustering	number of clusters	Medium `n_samples`, small `n_clusters`	Few clusters, even cluster size, non-flat geometry	Graph distance (e.g. nearest-neighbor graph)
Ward hierarchical clustering	number of clusters	Large `n_samples`and `n_clusters`	Many clusters, possibly connectivity constraints	Distances between points
Agglomerative clustering	number of clusters, linkage type, distance	Large `n_samples`and `n_clusters`	Many clusters, possibly connectivity constraints, non Euclidean distances	Any pairwise distance
DBSCAN	neighborhood size	Very large `n_samples`, medium `n_clusters`	Non-flat geometry, uneven cluster sizes	Distances between nearest points
Gaussian mixtures	many	Not scalable	Flat geometry, good for density estimation	Mahalanobis distances to centers
Birch	branching factor, threshold, optional global clusterer.	Large `n_clusters`and `n_samples`	Large dataset, outlier removal, data reduction.	Euclidean distance between points



K-means

Kmeans

- 設K = 3，起始**隨機挑選**3個點作為集群**中心點**(centroid)
- repeat：將附近的點根據與這3個中心點的距離分配到這三群，並重新計算中心點，直到收斂為止
- 收斂：得到與各集群中心點距離和最小值
  - Kmeans 使用歐幾里德距離(Euclidean Distance)

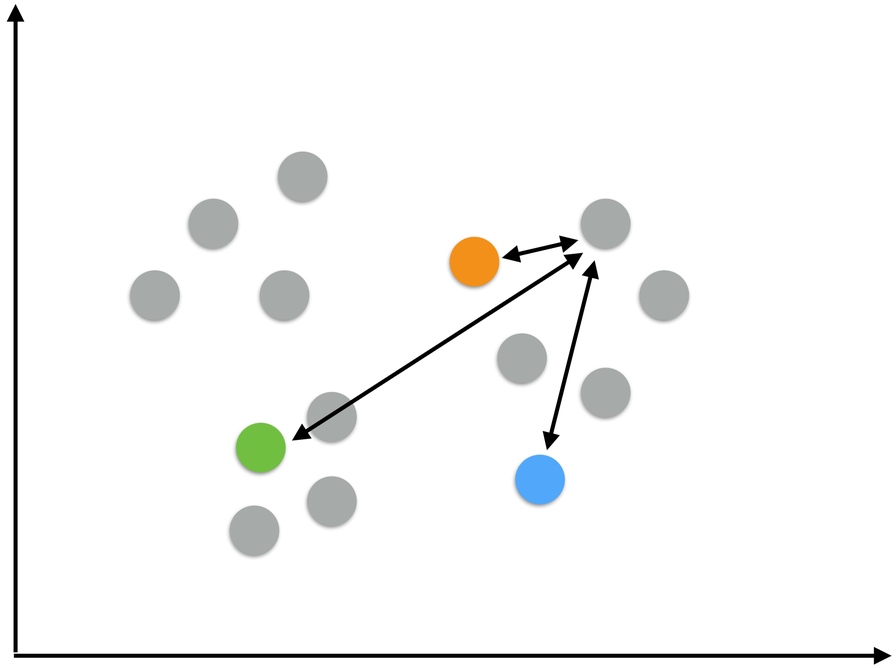
$$d(x,y) := \sqrt{(x_1-y_1)^2+(x_2-y_2)^2+\cdots+(x_n-y_n)^2} = \sqrt{\sum_{i=1}^{n}(x_i-y_i)^2}$$

Notes

- **需做資料標準化，讓Kmenas計算距離時使用相同的尺規。**

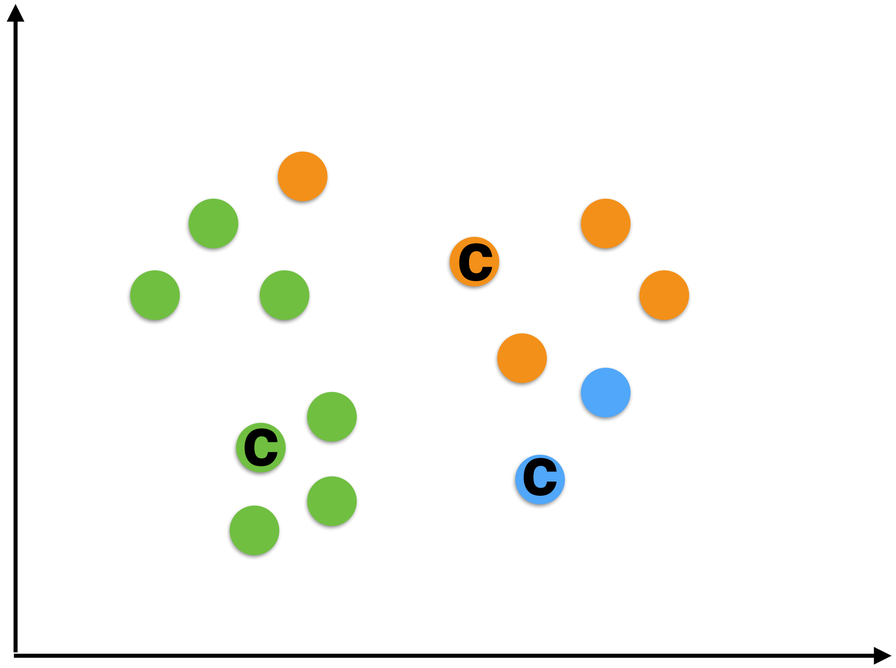
Kmeans Algorithm

- choose k centroids



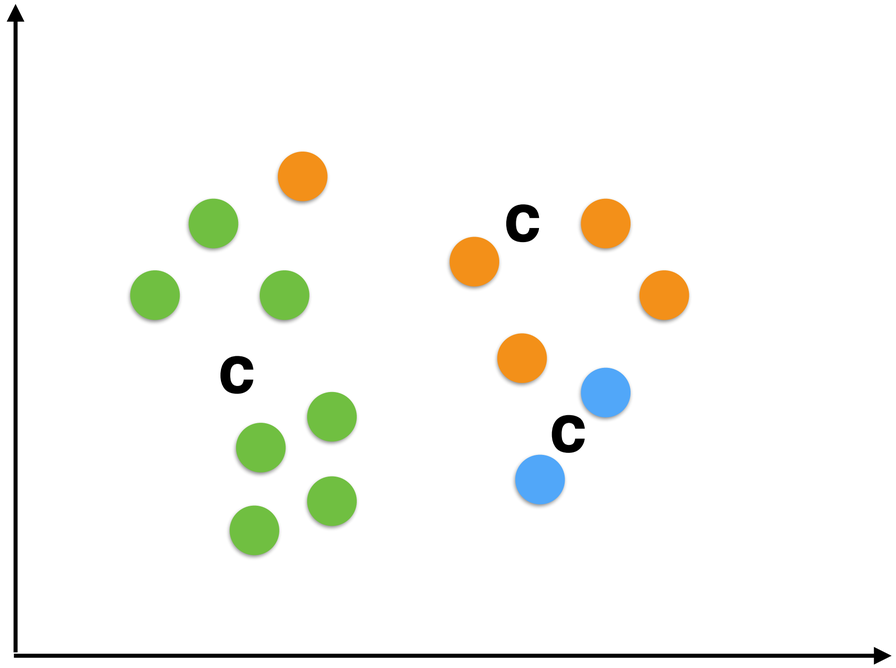
Kmeans Algorithm

- 將資料點指定給距離最近的集群中心點



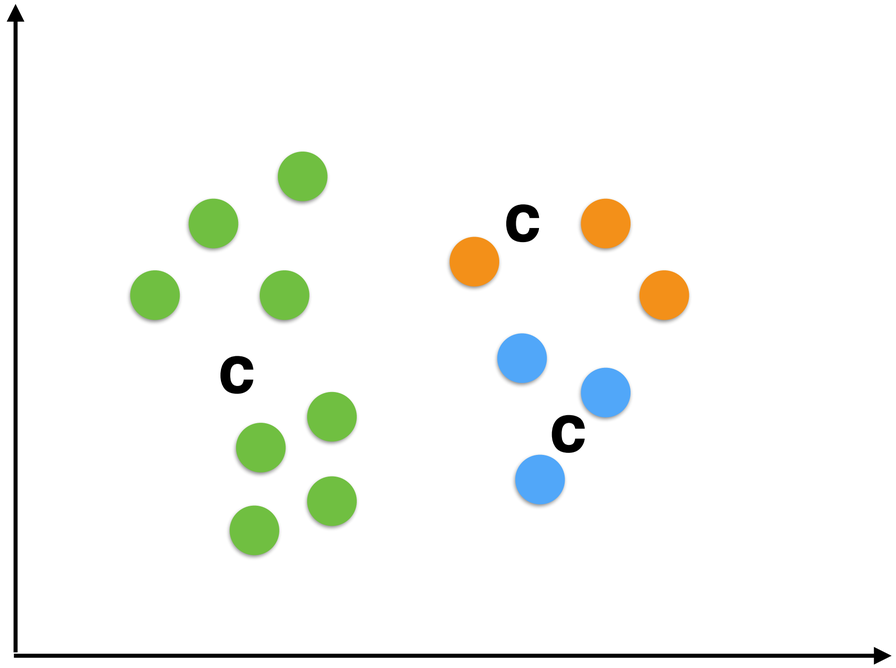
Kmeans Algorithm

- 重新計算並設定新的集群中心點



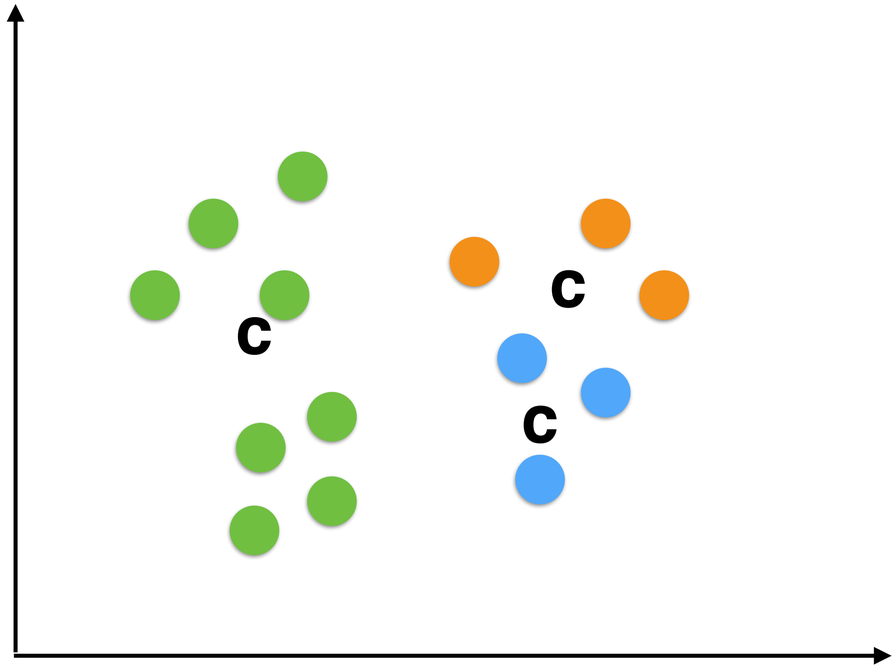
Kmeans Algorithm

- 重新指定每個點到距離最近的新集群中心



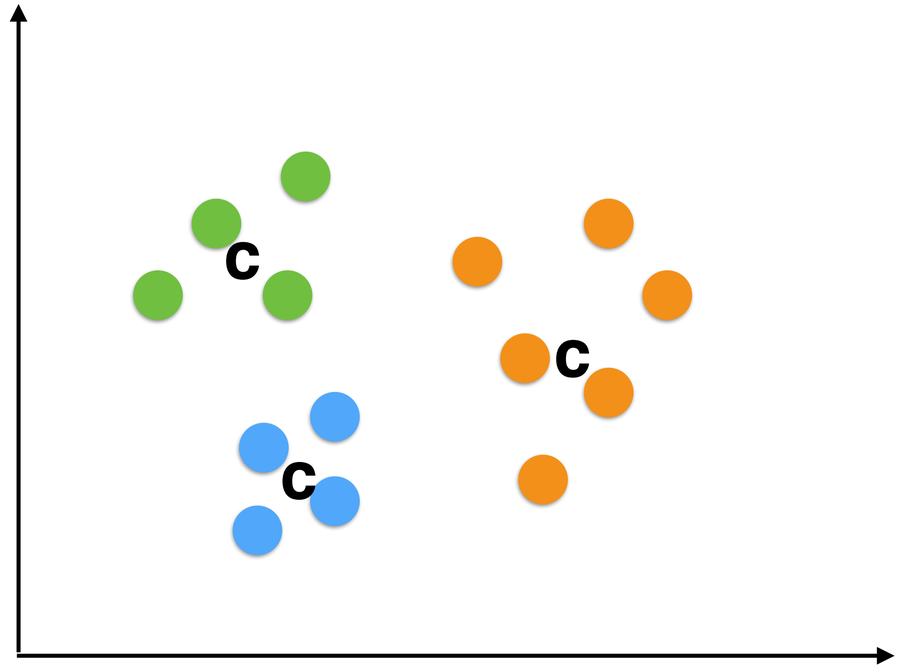
Kmeans Algorithm

- 重新計算新的集群中心點



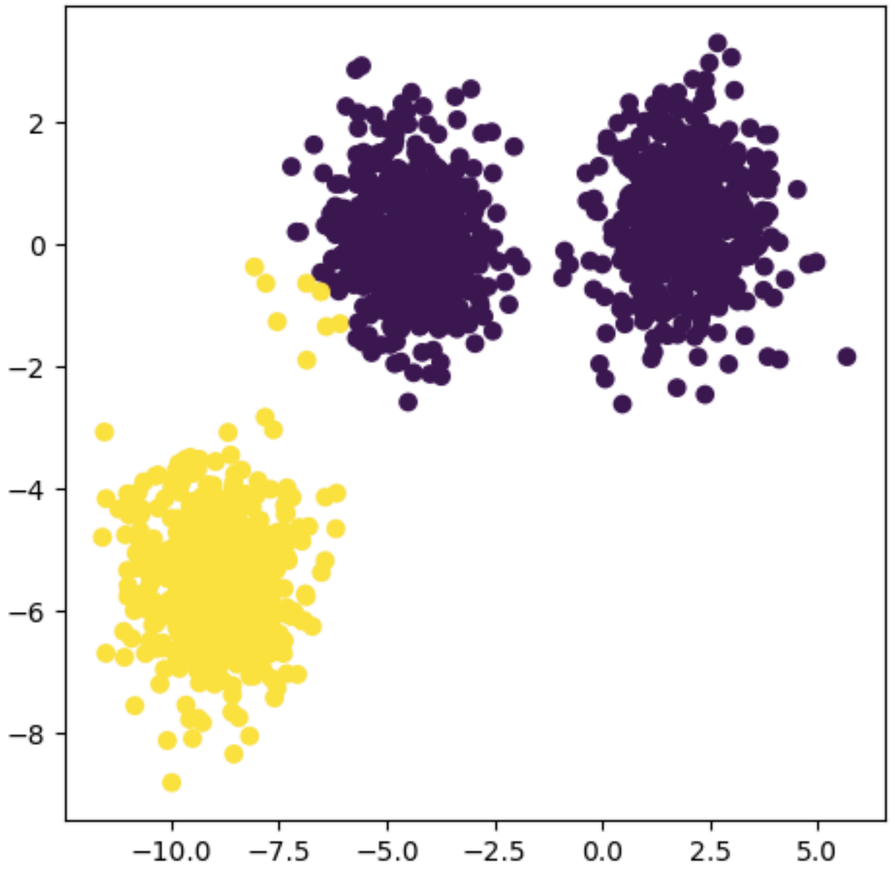
Kmeans Algorithm

- 重複計算集群中心點、分群直達收斂(各點與集群中心距離和最小)



Problem 1 - k值

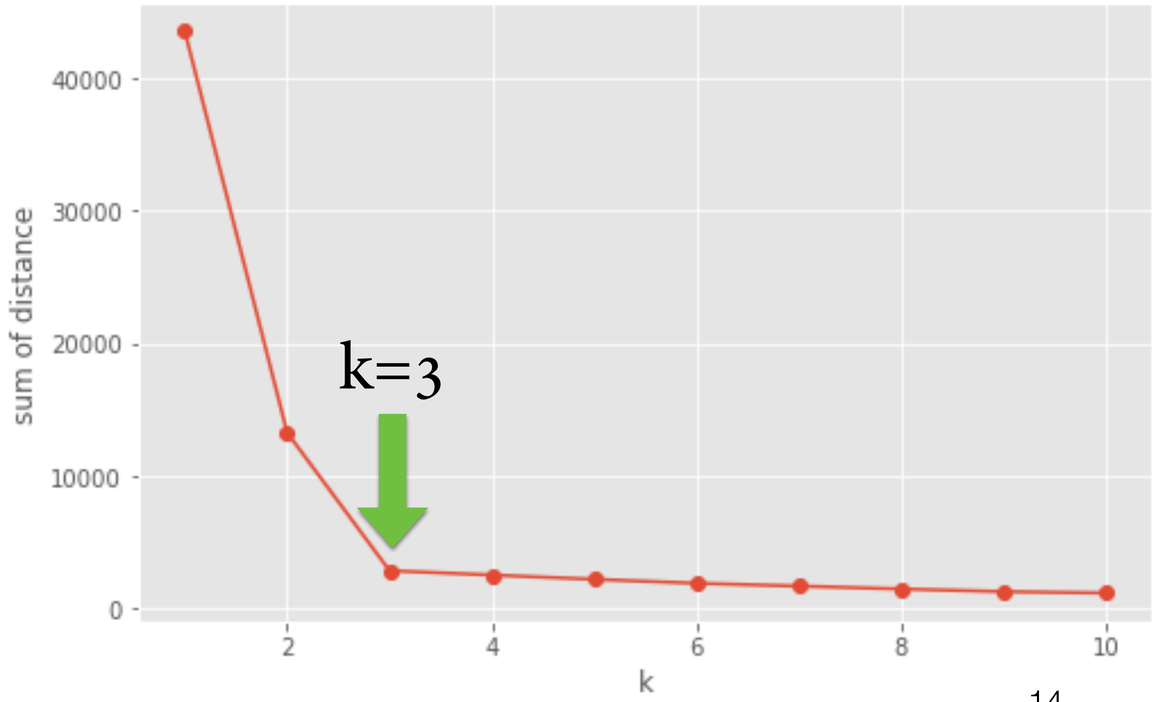
- 如何選擇好的k值?



Tuning k

- 若已經確定要分幾群(e.g. L、M、S)，可指定k值

- 計算不同k值之各集群總距離和



Problem 2 - 起始點與Kmeans++

- 隨機起始點若挑不好，可能導致不同的結果、花更多的時間才達到收斂
- 修正方法：Kmeans ++
  - 以距離較遠優先挑選初始中心點
    1. 隨機挑選一個中心點
    2. 計算所有資料對這些中心點的距離
    3. 將距離較遠的點加權(提高被挑中的機率)
    4. 重複以上步驟直到挑選完k個中心點再開始

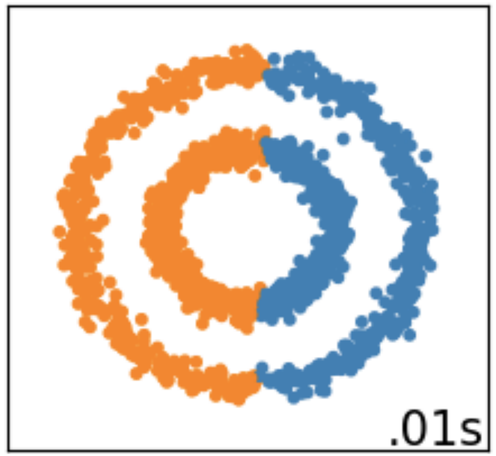
Kmeans with sklearn

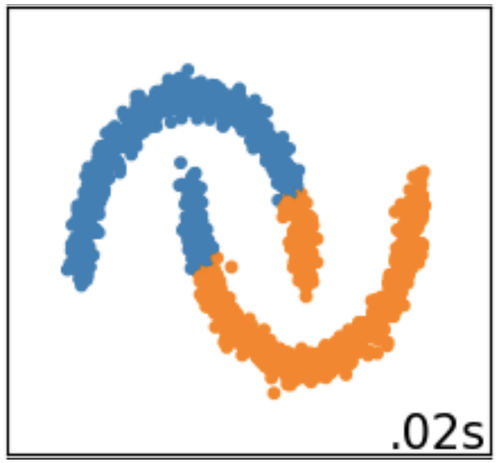
- *class* `sklearn.cluster.`**`KMeans`**(*n_clusters=8, init='k-means++', n_init=10, max_iter=300, tol=0.0001, precompute_distances='auto', verbose=0, random_state=None, copy_x=True, n_jobs=1, algorithm='auto'*)
  - *n_clusters: k*值*(*分群數*)*
  - *init: 'k-means++' , 'random'(*原始隨機挑選初始中心點方法*)*
  - *max_iter:* 最多迭代次數*(*可避免出現無法收斂的情況*)*

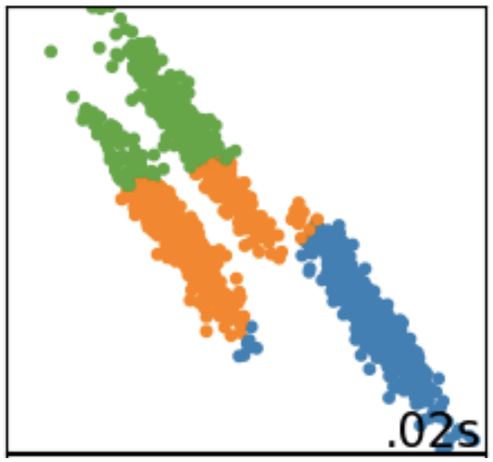
- *attributes:*
  - *cluster_centers_:* 各集群中心點
  - *labels_:* 各資料點屬於的集群標注
  - *inertia_:* 資料與各集群距離總和

其他問題

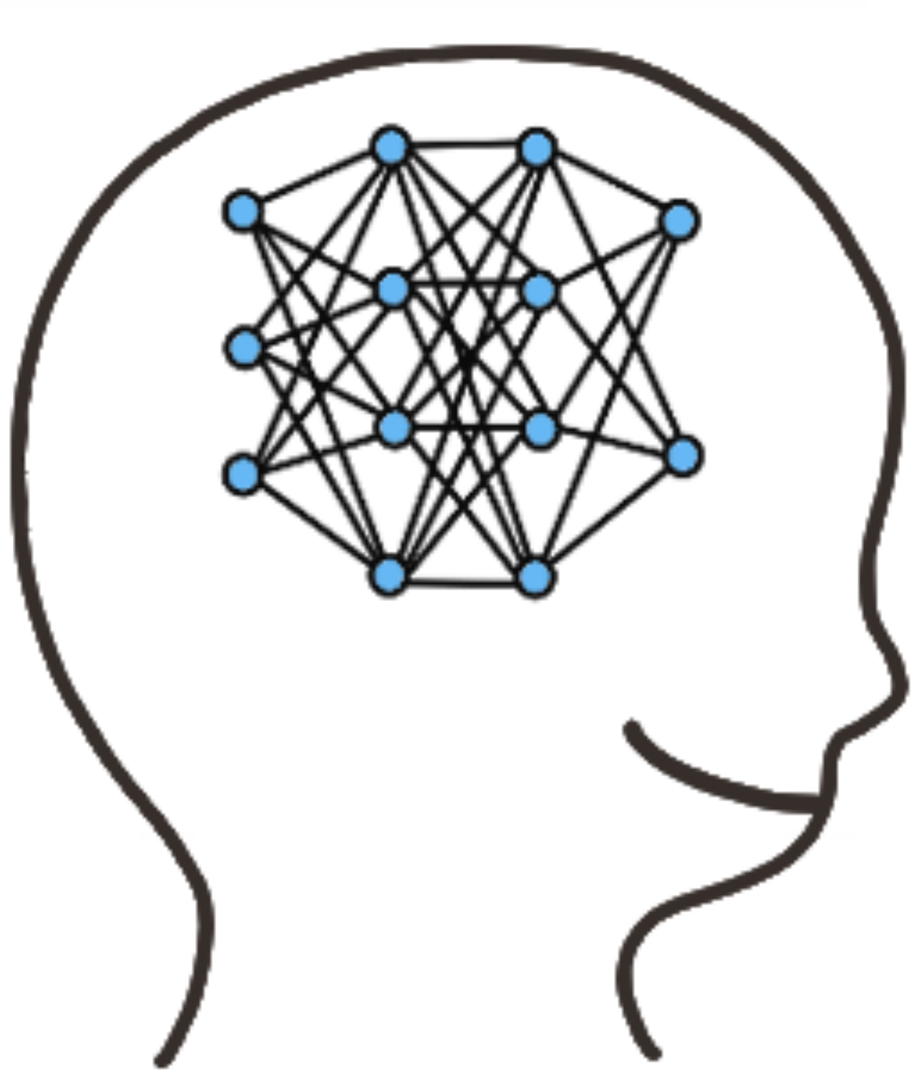
- 無法正確分群非圓狀的資料散佈，還可能無法成功收斂







- 容易受極端值影響



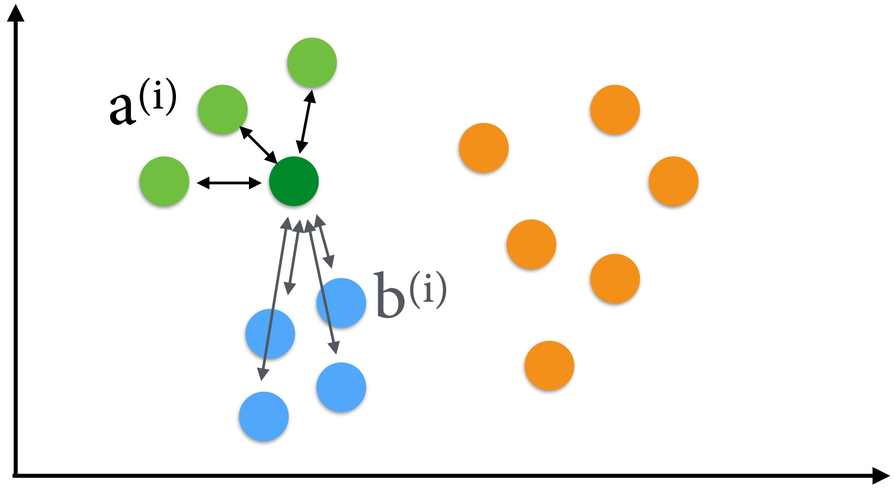
輪廓圖分析

Silhouette Analysis

輪廓係數 (silhoutte coefficient)

- 集群內聚(對內) $a^{(i)}$: 該資料與**同集群**中其他樣本的平均距離

- 集群分離(對外) $b^{(i)}$: 該資料與其他**最近集群**中所有樣本的平均距離



正常情況: $b^{(i)} > a^{(i)}$

輪廓係數 (silhoutte coefficient)

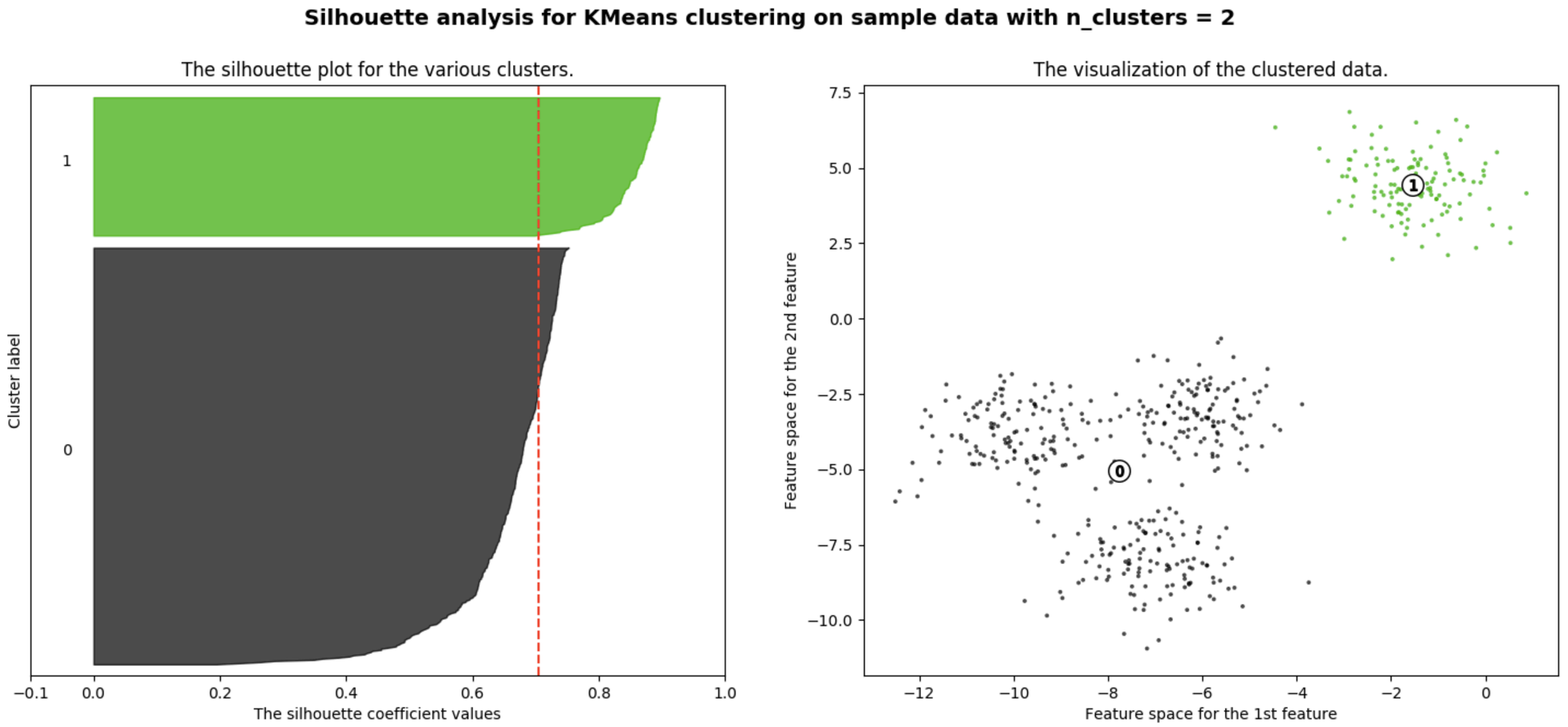
- 輪廓係數

$$s^{(i)} = \frac{b^{(i)} - a^{(i)}}{\max\{b^{(i)}, a^{(i)}\}}$$

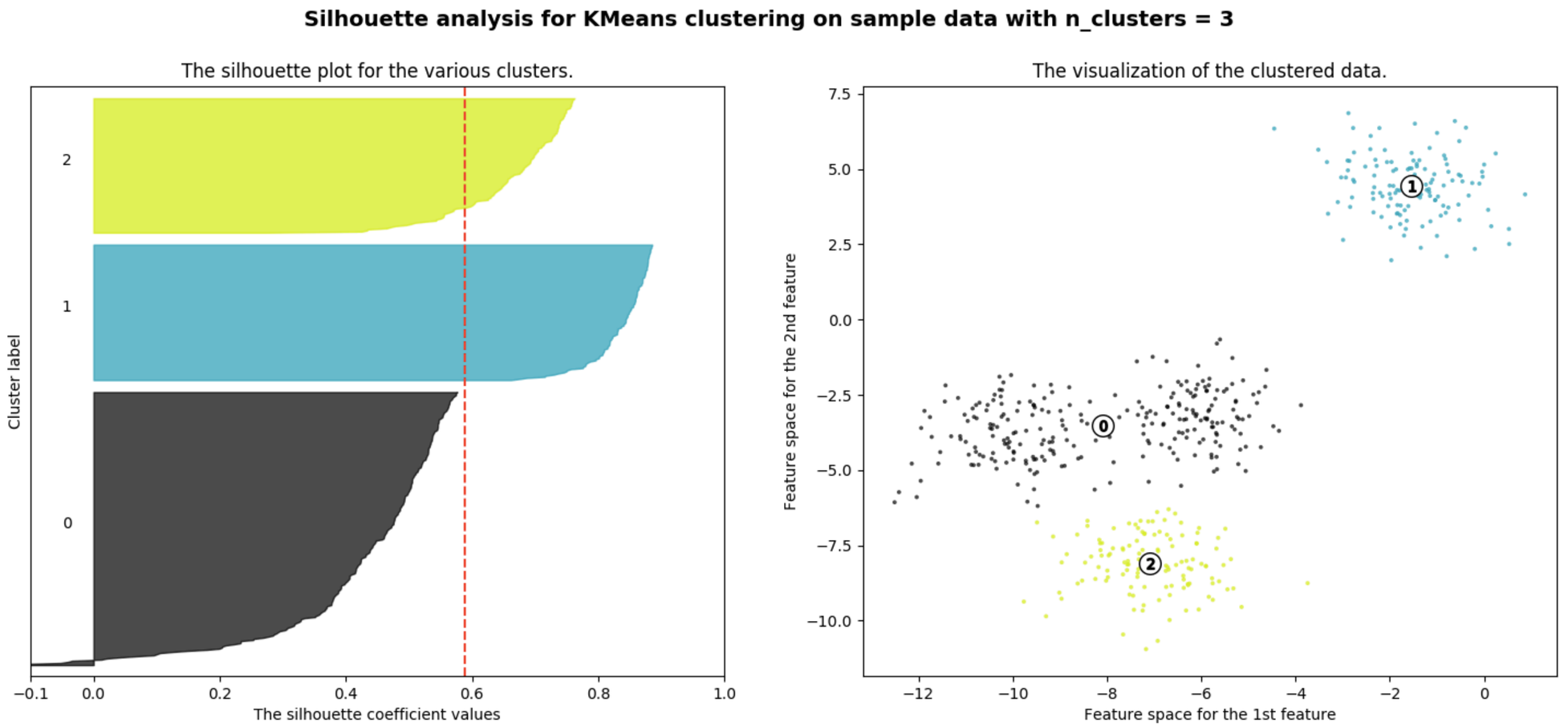
$$s(i) = \begin{cases} 1 - a(i)/b(i), & \text{if } a(i) < b(i) \\ 0, & \text{if } a(i) = b(i) \\ b(i)/a(i) - 1, & \text{if } a(i) > b(i) \end{cases}$$

  - $s^{(i)}$ 範圍：-1 ~ +1
  - $s^{(i)}$=0: 內聚程度=分離程度
  - $s^{(i)}$=1: 理想值(分離程度>>內聚程度)

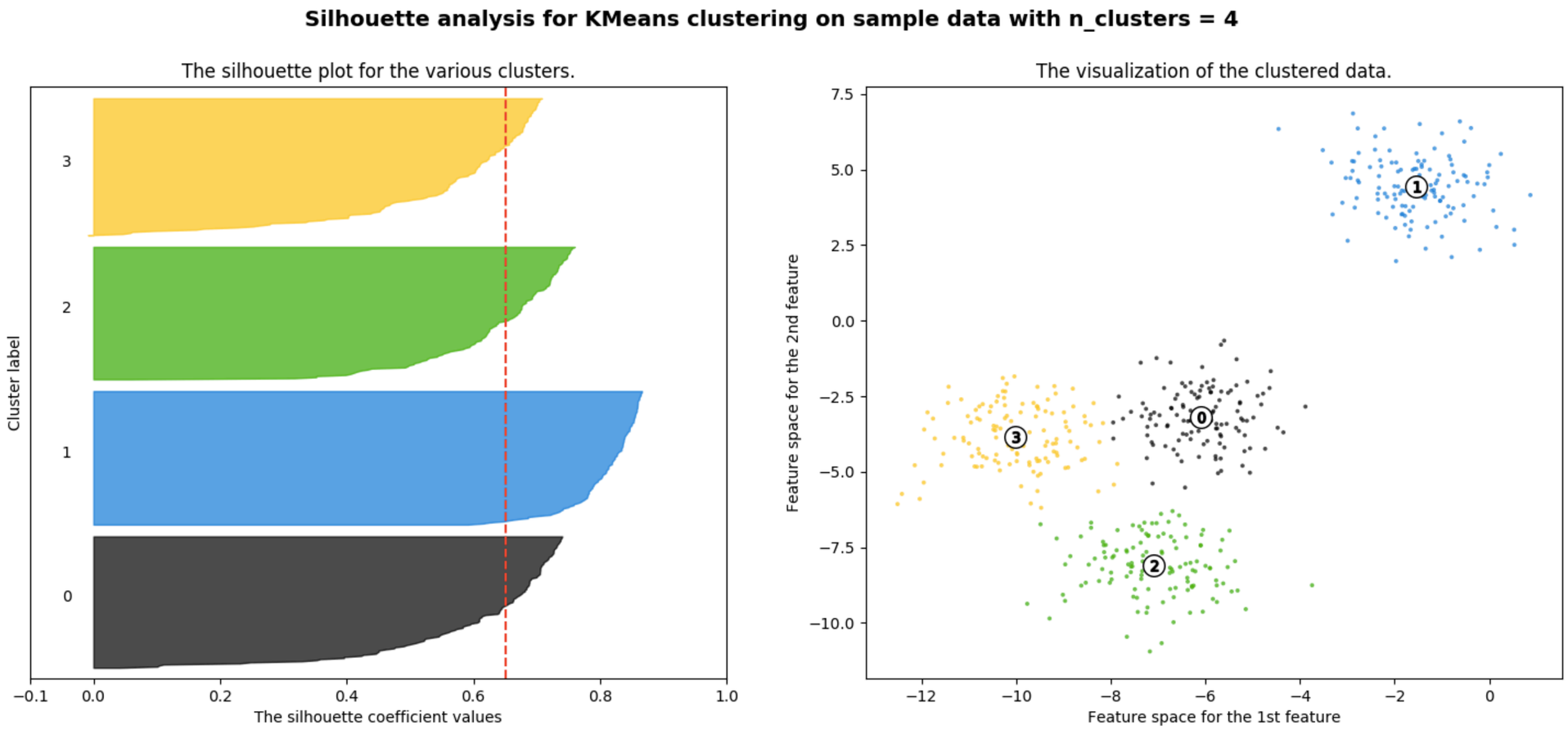
輪廓圖


Silhouette analysis for KMeans clustering on sample data with n_clusters = 2
The silhouette plot for the various clusters.
Cluster label
1
0
−0.1
0.0
0.2
0.4
0.6
0.8
1.0
The silhouette coefficient values
The visualization of the clustered data.
Feature space for the 2nd feature
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−12
−10
−8
−6
−4
−2
0
Feature space for the 1st feature

輪廓圖

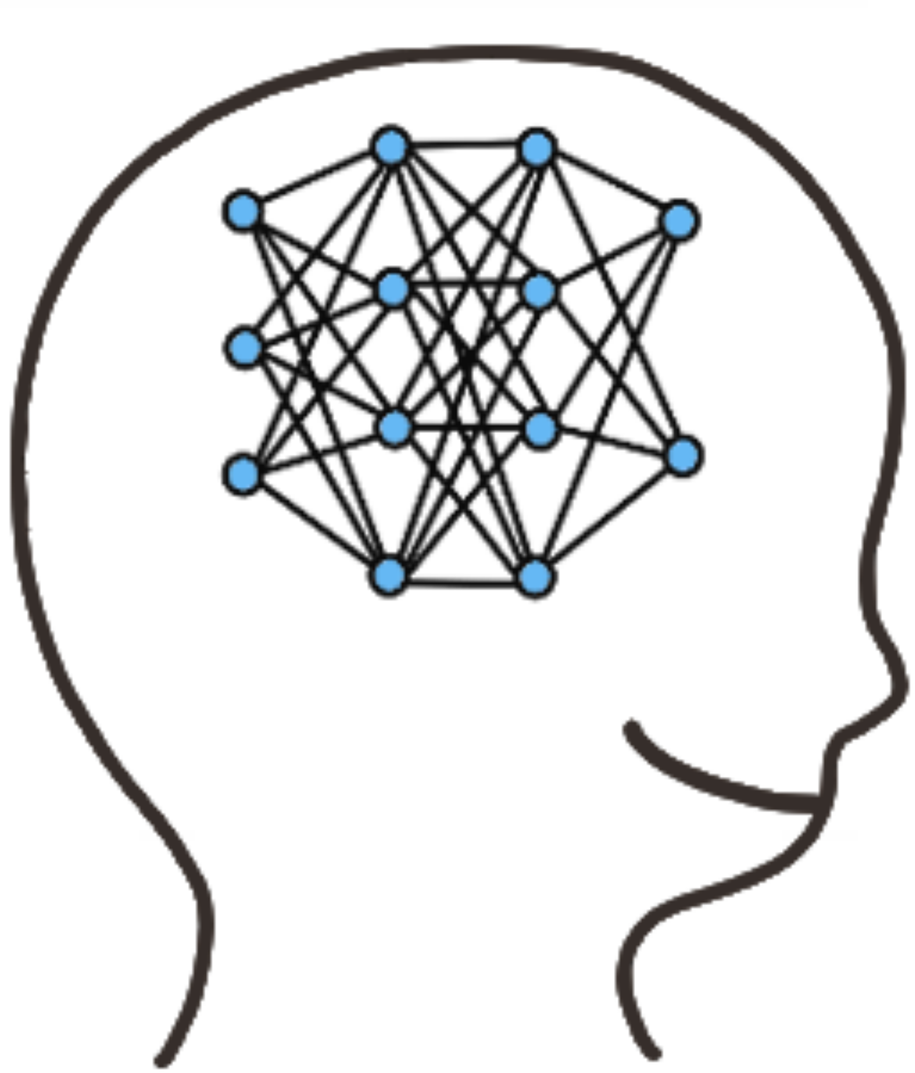

Silhouette analysis for KMeans clustering on sample data with n_clusters = 3
The silhouette plot for the various clusters.
Cluster label
2
1
0
−0.1
0.0
0.2
0.4
0.6
0.8
1.0
The silhouette coefficient values
The visualization of the clustered data.
Feature space for the 2nd feature
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−12
−10
−8
−6
−4
−2
0
Feature space for the 1st feature

輪廓圖


Silhouette analysis for KMeans clustering on sample data with n_clusters = 4
The silhouette plot for the various clusters.
Cluster label
3
2
1
0
−0.1
0.0
0.2
0.4
0.6
0.8
1.0
The silhouette coefficient values
The visualization of the clustered data.
Feature space for the 2nd feature
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−12
−10
−8
−6
−4
−2
0
Feature space for the 1st feature

Silhoutte coefficient with sklearn

- `sklearn.metrics.`**`silhouette_samples`**(*X*, *labels*, *metric='euclidean'*, ***kwds*)
  - 計算每筆樣本的輪廓係數(Silhouette Coefficient)
- `sklearn.metrics.`**`silhouette_score`**`(X, labels, metric='euclidean', sample_size=None, random_state=None, **kwds)`
  - 計算所有樣本的輪廓係數平均`(Mean of Silhouette Coefficient)`

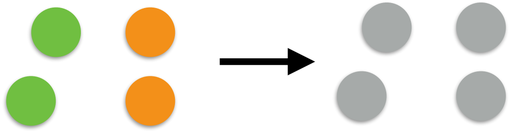


階層式分群

Hierarchical Clustering

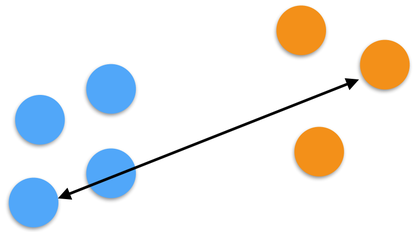
階層式分群

- 兩種方式：
  - Top-down/群由大變小(分裂)
    - 分離分層集群(Divisive Hierarchical Clustering)
  - Bottom-up/群由小變大(合併)
    - 凝聚分層集群(Agglomerative Hierarchical Clustering)



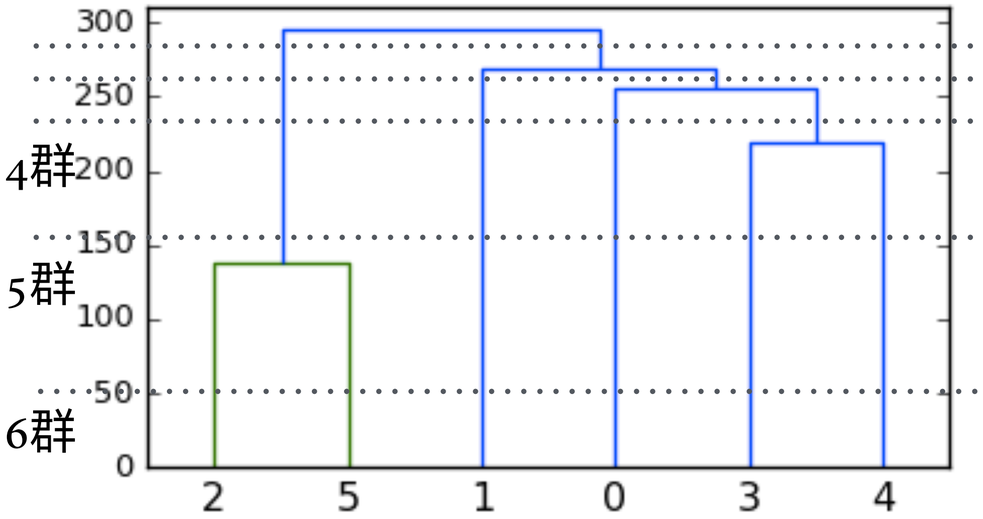
凝聚分層

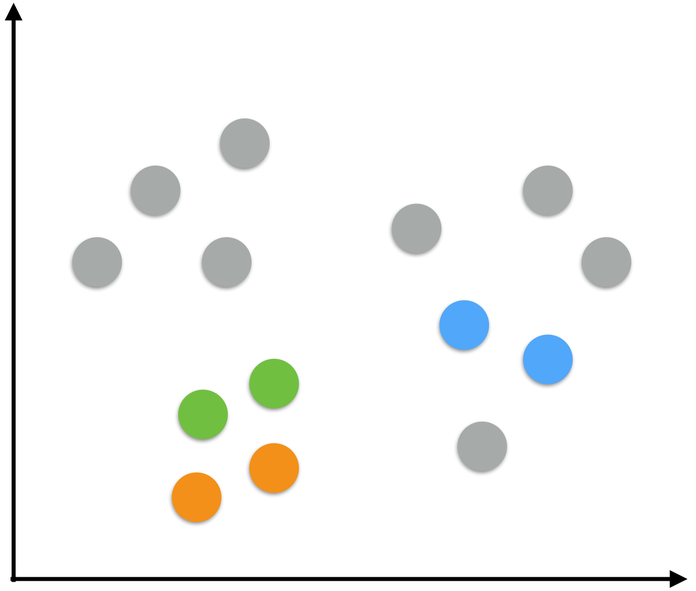
- 每個資料點都當成獨自一群
- 從彼此距離最近的群開始合併，直到產生一定量的集群
  - 計算距離方式
    - Ward: 集群聚合後的變異(variance)/距離平方和最小
    - Average: 集群中各點與各點資料點的平均距離
    - Complete: 集群中兩點最遠的距離



凝聚分層

- 從每個點都是獨立集群開始依距離——合併集群

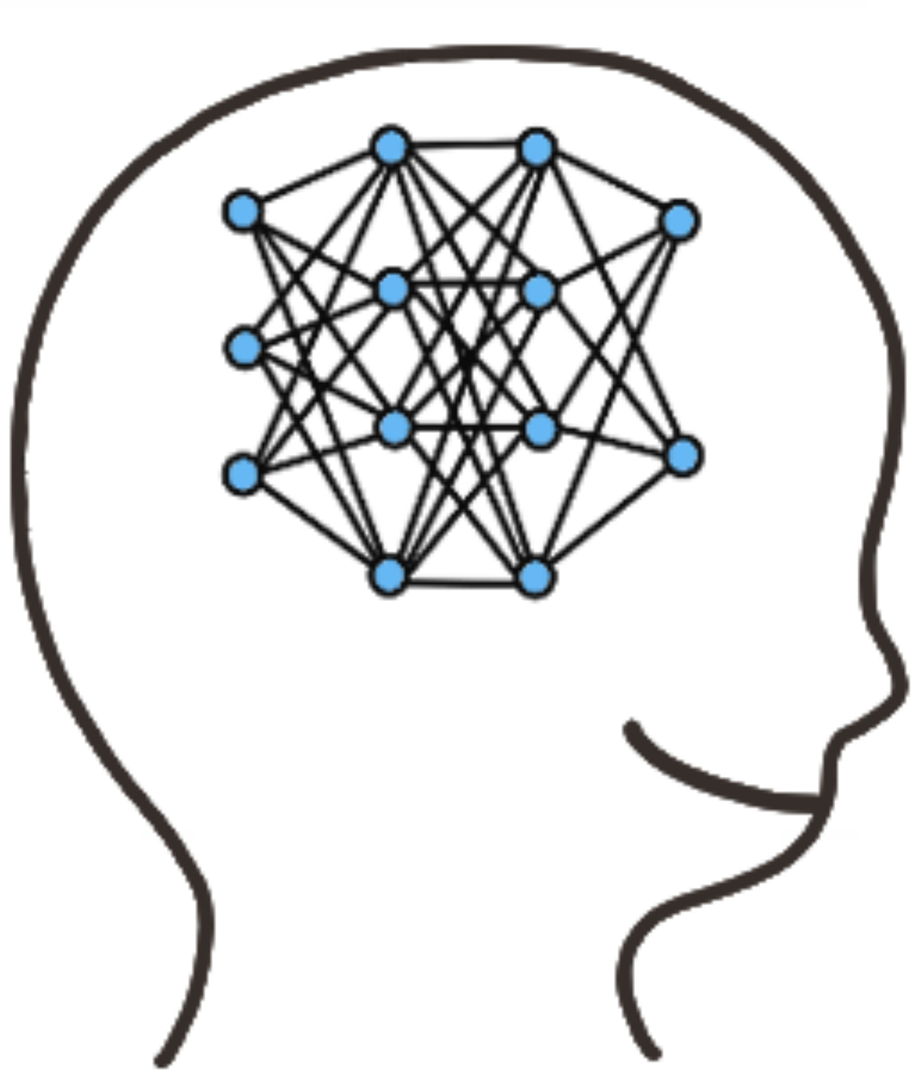




Agglomerative Clustering with sklearn

- *class* `sklearn.cluster.`**`AgglomerativeClustering`**(*n_clusters=2*, *affinity='euclidean'*, *memory=None*, *connectivity=None*, *compute_full_tree='auto'*, *linkage='ward'*, *pooling_func=<function mean>*)
  - n_clusters: 集群數
  - affinity: 距離計算方法 euclidean(歐幾里德距離)
  - linkage: 'ward','average', 'complete'

- `scipy.cluster.hierarchy.dendrogram`(*Z*, *p=30*, *truncate_mode=None*, *color_threshold=None*, *get_leaves=True*, *orientation='top'*, *labels=None*, *count_sort=False*, *distance_sort=False*, *show_leaf_counts=True*, *no_plot=False*, *no_labels=False*, *leaf_font_size=None*, *leaf_rotation=None*, *leaf_label_func=None*, *show_contracted=False*, *link_color_func=None*, *ax=None*, *above_threshold_color='b'*)
  - 畫階層樹狀圖

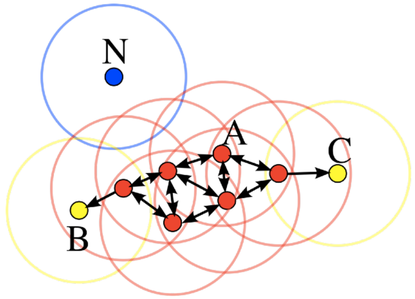


DBSCAN

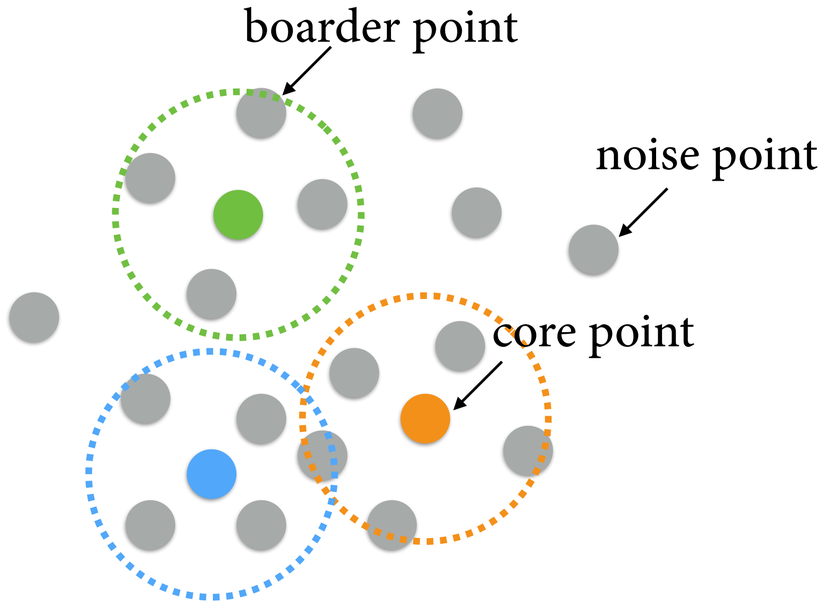
Density-based Spatial Clustering of Applications with Noise

DBSCAN

- 根據**密度**(指定半徑內)形成集群
  - 核心點(core point): 半徑內包含超過密度域值(minPts)的點
  - 邊緣點(border point): 核心點半徑內包含的點
  - 雜訊點(noise point)

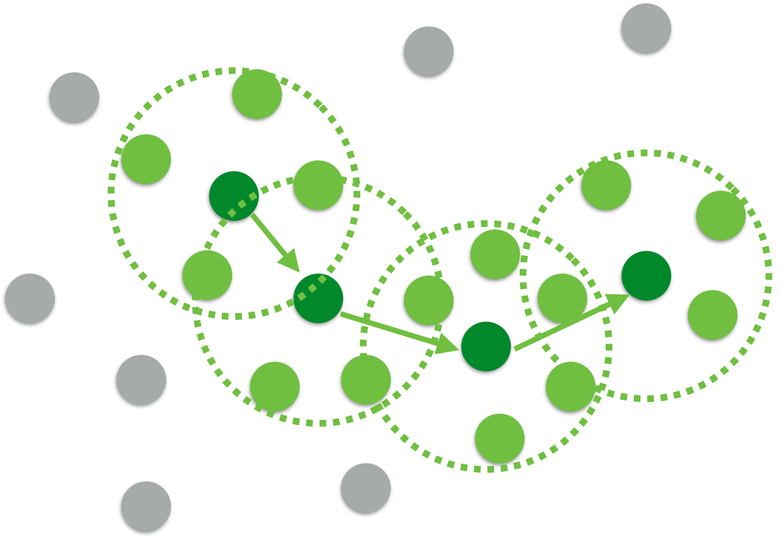


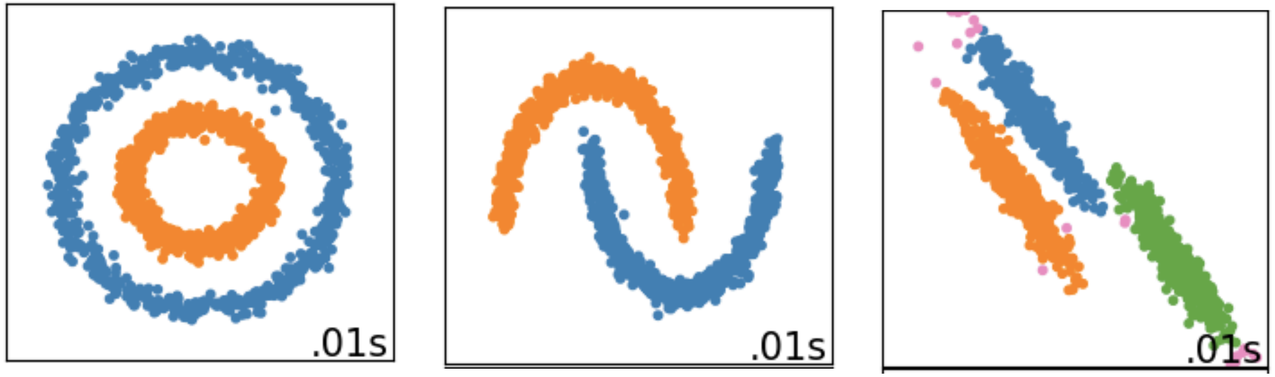
- 集群方法
  - 每個核心點或兩核心點距離小於半徑，形成一集群
  - 指定邊緣點到核心點集群



特點

- 可判別雜訊點(noise)，使分群不受雜訊影響
- 不用預先設定分群數
- 可順利將特殊形狀分群(如: 半月形、甜甜圈形)





DBSCAN with sklearn

- *class* `sklearn.cluster.`**`DBSCAN`**(*eps=0.5*, *min_samples=5*, *metric='euclidean'*, *metric_params=None*, *algorithm='auto'*, *leaf_size=30*, *p=None*, *n_jobs=1*)
  - eps: 半徑
  - min_samples: minPts 半徑內至少要有多少點才能被判定為core point

- attributes:
  - labels_: 分群標籤(若為noise則回傳-1)