解讀資料

共變異數（Covariance）

共變異數是觀察兩變數同時偏離平均值方向與程度的指標。

它反映兩變數間是否有「同向」或「反向」變動的趨勢。

- 若兩變數同時上升或下降 → 共變異數為正
- 若一個上升、一個下降 → 共變異數為負
- 若變化無一致方向 → 共變異數接近 0

$$\text{共變異數} = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{n - 1}$$

x_i, y_i：第 i 筆觀察值

$\bar{x}, \bar{y}$：兩變數的平均值

n：樣本數

共變異數 為兩變數偏離各自平均值的乘積平均

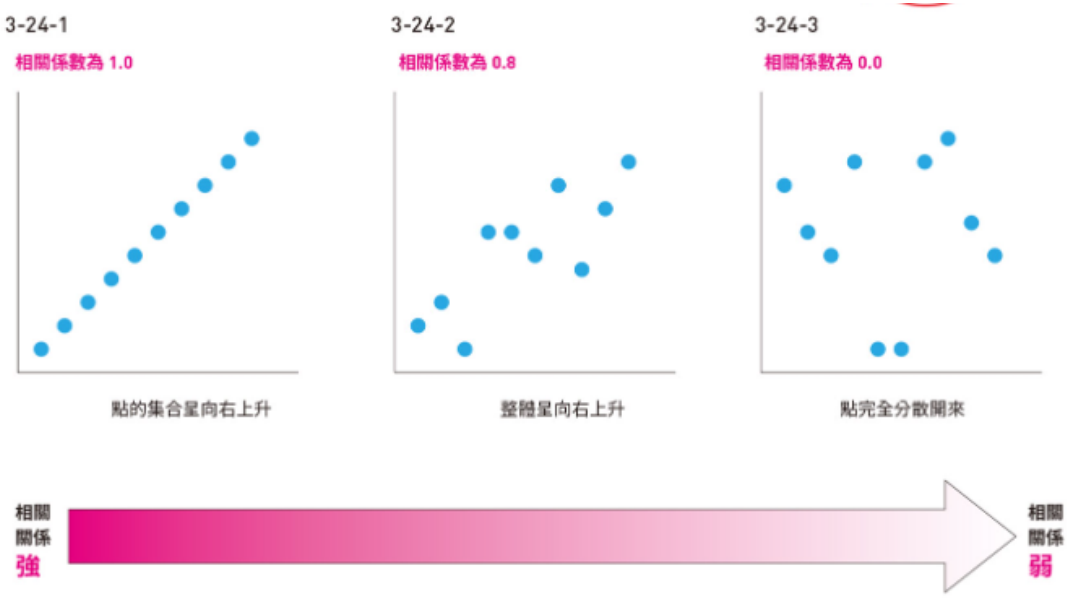
相關係數（Correlation Coefficient）

兩個變數之間的關係並不一定完全無關或完全相關，而是存在不同程度的線性關聯。

「相關係數」是一種指標，用來衡量兩個變數之間的線性關係

- 若一個變數上升時，另一個變數也隨之上升 → 正相關
- 若一個變數上升時，另一個變數下降 → 負相關
- 若兩者變化無明顯關聯 → 不相關

- 相關係數以 r 表示，數值範圍介於 −1 到 +1 之間。
- 其絕對值越接近 1，代表關聯程度越強



$$\text{共變異數} = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{n-1}$$

$$\text{相關係數} = \frac{\text{共變異數}}{s_x s_y}$$

s_x, s_y 為 X、Y 的標準差

週次	商品B	商品D
第1週	52	210
第2週	41	200
第3週	55	400
第4週	46	170
第5週	42	160

▼圖 3-25　商品 B 與商品 D 的相關係數計算方法

相關係數的算法

$$\text{商品 B 與商品 D 的相關係數} = \frac{\text{商品 B 與商品 D 的共變異數}}{\text{商品 B 的標準差} \times \text{商品 D 的標準差}}$$

$$= \frac{-1002.9}{11.2 \times 112.4} = -0.79\cdots$$

商品B與商品D有相當強的負相關關係。

共變異數的算法

商品 B 的星期日銷售額與算術平均數的差

商品 D 的星期日銷售額與算術平均數的差

①商品 B 與商品 D 各天的銷售額偏差（銷售額與算術平均數的差）相乘

②合計星期日到星期六的商品 B 偏差 × 商品 D 偏差

$$\frac{(52-40)(210-400)+(21-40)(520-416.7)+\cdots+(55-40)(300-416.7)}{7} = -1002.9\cdots$$

③除以一週天數（資料個數）

④共變異數

多變量資料分析（Multivariate Analysis）

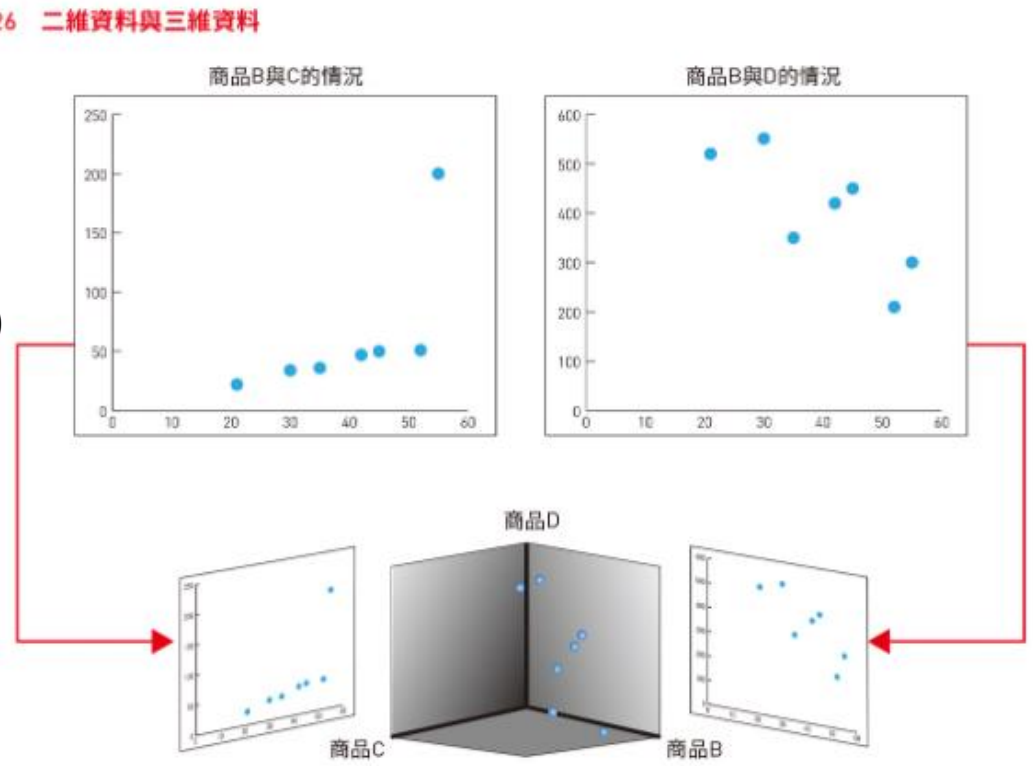
實際的資料分析中，我們常同時觀察三個以上的變數

例如:

- 學生成績資料（國文、英文、數學三科）
- 商品銷售資料（價格、廣告費用、銷售量）
- 醫療數據（年齡、血壓、膽固醇值）

當資料中包含多個變數時，便形成了多維資料集。

分析這類資料的方法稱為多變量分析（Multivariate Analysis）



- 銷售預測（價格、廣告費、庫存 → 銷量）
- 醫療診斷（年齡、血壓、血糖 → 疾病風險分類）
- 市場分析（性別、年齡、興趣 → 消費群組分類）

結論一般化（Generalization）

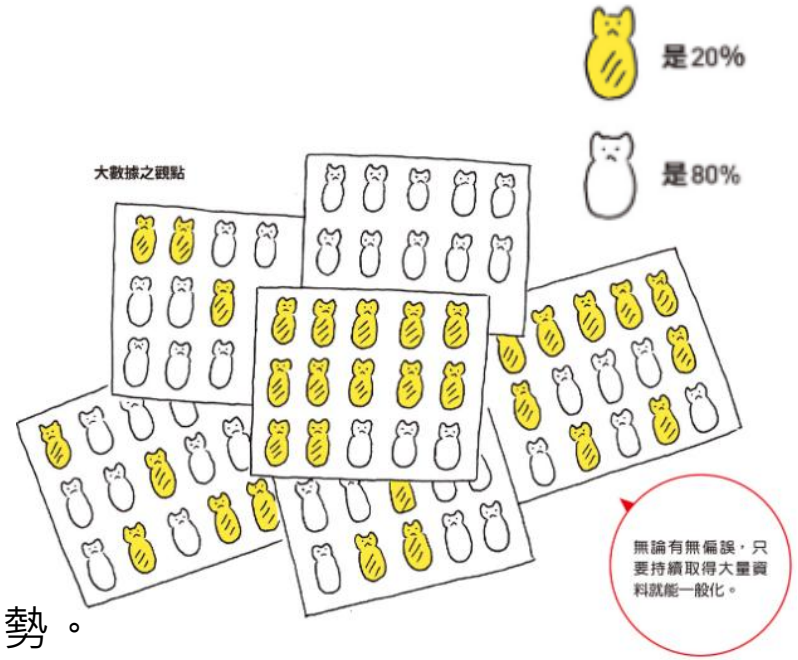
資料分析中，最終的目的並不僅是「描述資料」，而是希望根據樣本資料，歸納出可推論至整體的結論。
這種將分析結果「擴展」至更廣泛情境的過程，稱為 一般化（Generalization）。

資料完全從母體隨機取得之觀點

- 若我們的資料樣本是從整體母體中隨機抽樣而來，
- 可以推論樣本的特性能代表整體母體的特性。

大數據採樣之觀點

- 在現代資料分析中，樣本並非總是隨機取得。
- 有時資料是從網路紀錄、交易系統、感測器等自動收集而來，
- 未必能保證「隨機性」。
- 當資料量極大時，即使存在某些偏差，也常可呈現出某種穩定的整體趨勢。



練習：X = 每週自學時數（hr），Y = 小測驗分數（分）

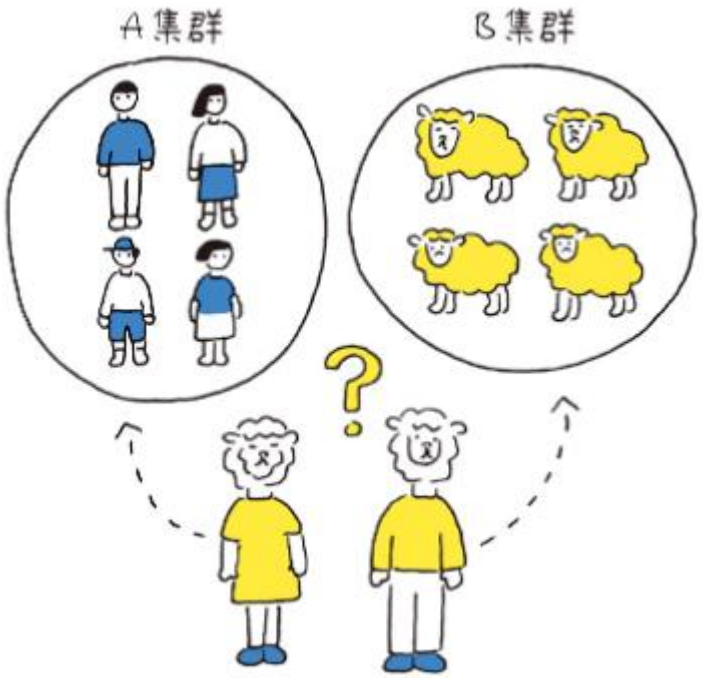
序	X	Y	序	X	Y	序	X	Y
1	78.4	162.5	11	53.1	105.3	21	88.3	171.9
2	41.5	67.9	12	70.3	137.7	22	81.9	168.8
3	56.5	86.8	13	41.6	84.3	23	60.4	118.3
4	53.4	100.7	14	51.9	112	24	49.3	122
5	84.2	159.2	15	79	164.4	25	97.4	186.6
6	80.6	170	16	72.7	148.9	26	60.2	109.4
7	93.5	193.6	17	53.2	112.9	27	45.6	98.9
8	45.2	78.2	18	75.4	155.6	28	45.8	105.8
9	65.3	139.1	19	88.6	170.9	29	90.8	186.7
10	41.8	73.6	20	40.4	73.6	30	76.2	160.8

平均值X：
平均值Y：

共變異數：

標準差X：
標準差Y：

相關係數：



分類相似者

A. 群集分析 (Cluster analysis)

B. 主成分分析 (Principal components analysis, PCA)

C. 對應分析 (Correspondence analysis)

A. 群集分析 (Cluster analysis)

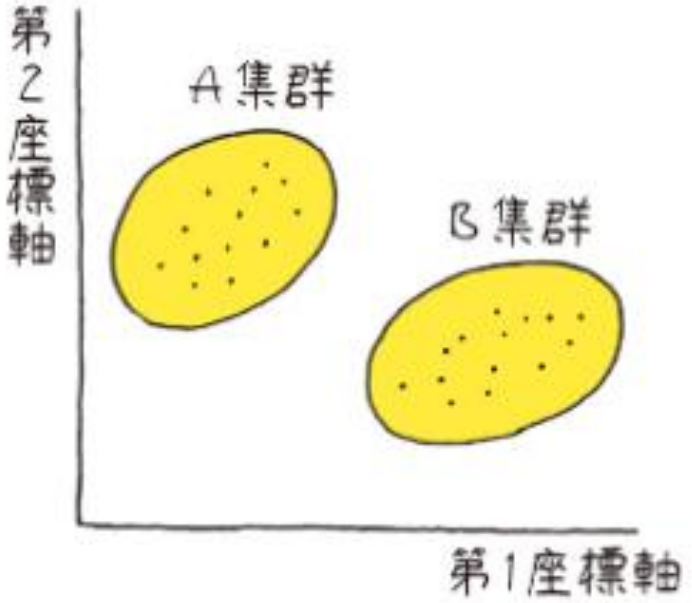
找出相似群組：

- 將資料劃分成數個群組，使同群內的資料彼此相似、不同群之間差異較大

➔ 廣泛應用於市場區隔、顧客行為分析、生物分類、影像辨識

根據「相似度」進行分類：

- 資料點越接近 → 相似程度越高
- 資料點距離越遠 → 差異越大



非階層式分群（K-means Clustering）

- 將資料依據特徵相似性分成 K 個群集，使群內距離最小、群間距離最大

Step 1：初始設定 (Initialization)

隨機選取 K 個「中心點」（Centroids），圖中以 三角形 表示。

每個資料點（圓點）依與中心點的距離，先分配到最近的群。

Step 2：重新計算中心 (Update Centroids)

對每個群組計算平均位置，作為新的中心。

圖中三角形位置更新（有箭頭顯示移動方向）。

Step 3：重新分配資料點 (Reassignment)

根據新的中心，重新劃分資料點的群歸屬。

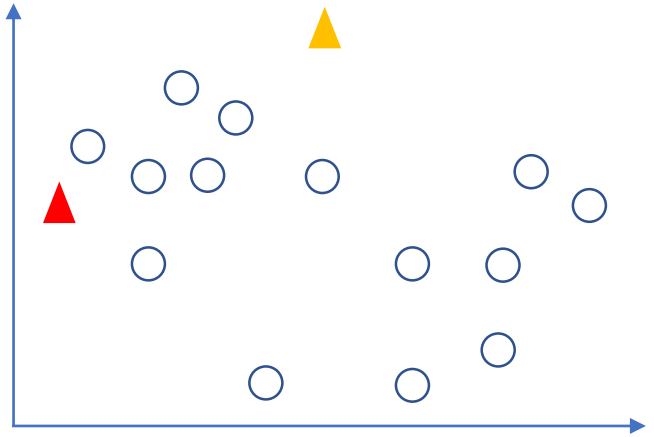
部分資料點可能改變群。

Step 4：持續迭代直到收斂 (Repeat until convergence)

不斷重複步驟 2、3，直到群中心不再變化，

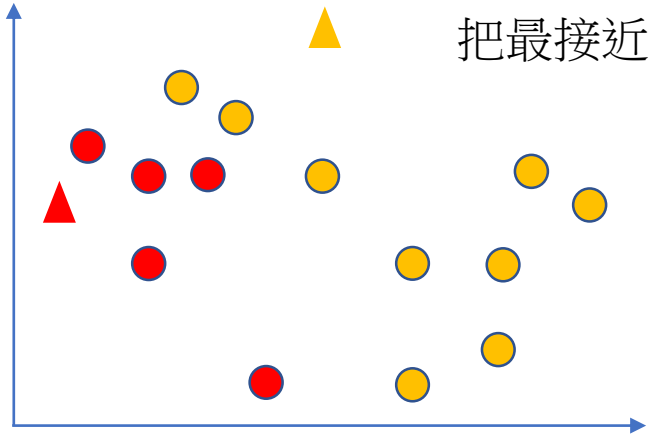
或資料點分群結果穩定為止。

隨機放2個中心



把最接近▲的點塗上黃色

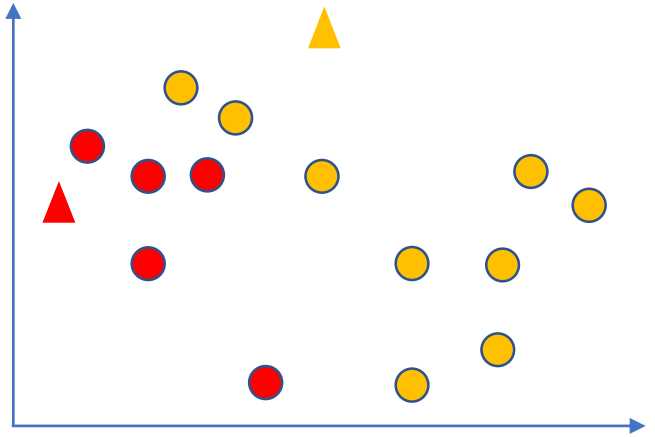
把最接近▲的點塗上紅色



https://www.youtube.com/watch?v=_aWzGGNrcic

非階層式分群（K-means Clustering）

- 將資料依據特徵相似性分成 K 個群集，使群內距離最小、群間距離最大



Step 1：初始設定 (Initialization)

隨機選取 K 個「中心點」（Centroids），圖中以 三角形 表示。

每個資料點（圓點）依與中心點的距離，先分配到最近的群。

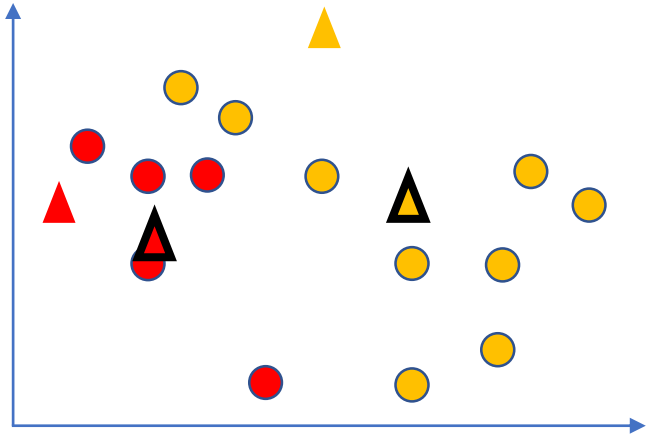
Step 2：重新計算中心 (Update Centroids)

對每個群組計算平均位置，作為新的中心。

圖中三角形位置更新（有箭頭顯示移動方向）。

計算紅色點的平均位置，找出新的▲

計算黃色點的平均位置，找出新的▲



Step 3：重新分配資料點 (Reassignment)

根據新的中心，重新劃分資料點的群歸屬。

部分資料點可能改變群。

Step 4：持續迭代直到收斂 (Repeat until convergence)

不斷重複步驟 2、3，直到群中心不再變化，

或資料點分群結果穩定為止。

https://www.youtube.com/watch?v=_aWzGGNrcic

非階層式分群（K-means Clustering）

- 將資料依據特徵相似性分成 K 個群集，使群內距離最小、群間距離最大

Step 1：初始設定 (Initialization)

隨機選取 K 個「中心點」（Centroids），圖中以 三角形 表示。

每個資料點（圓點）依與中心點的距離，先分配到最近的群。

Step 2：重新計算中心 (Update Centroids)

對每個群組計算平均位置，作為新的中心。

圖中三角形位置更新（有箭頭顯示移動方向）。

Step 3：重新分配資料點 (Reassignment)

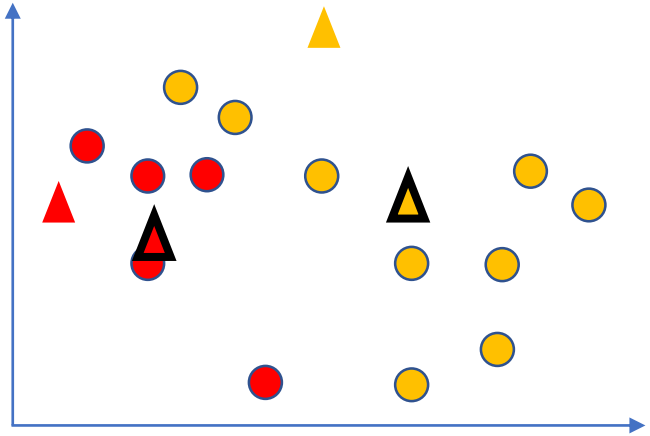
根據新的中心，重新劃分資料點的群歸屬。

部分資料點可能改變群。

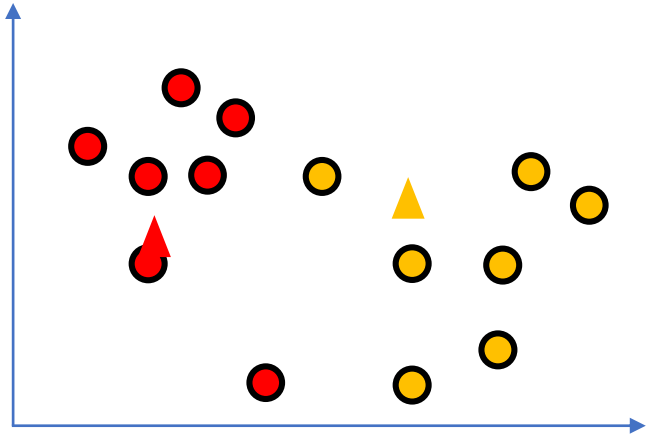
Step 4：持續迭代直到收斂 (Repeat until convergence)

不斷重複步驟 2、3，直到群中心不再變化，

或資料點分群結果穩定為止。

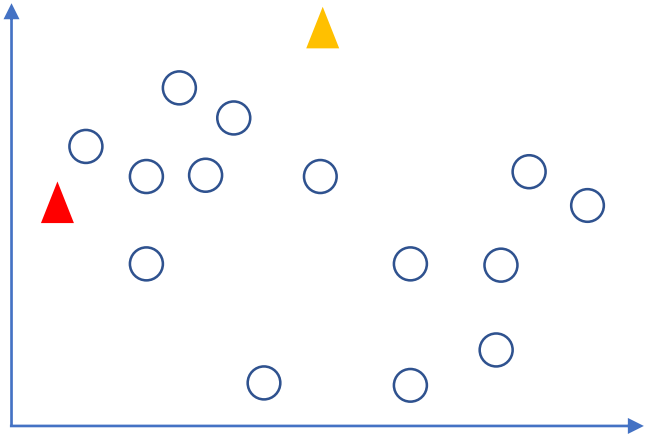


根據新的中心點，重新計算距離最近三角形， 塗上紅色、黃色

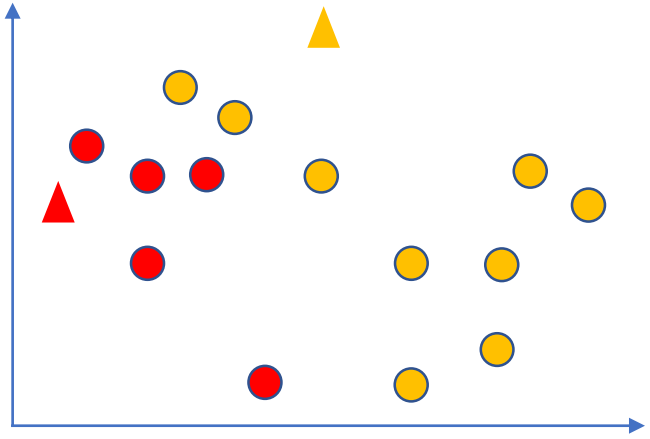


https://www.youtube.com/watch?v=_aWzGGNrcic

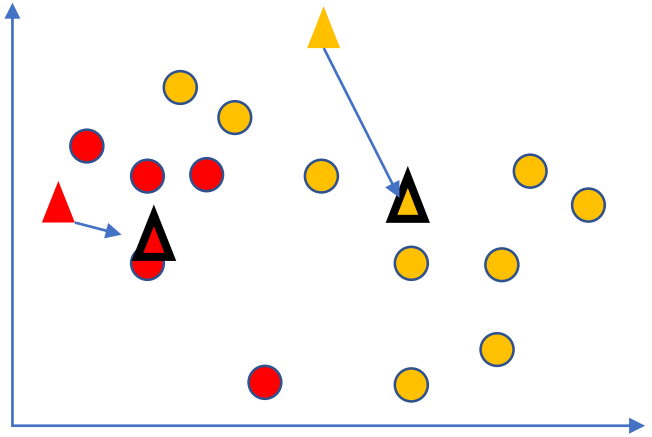
隨機放2個中心



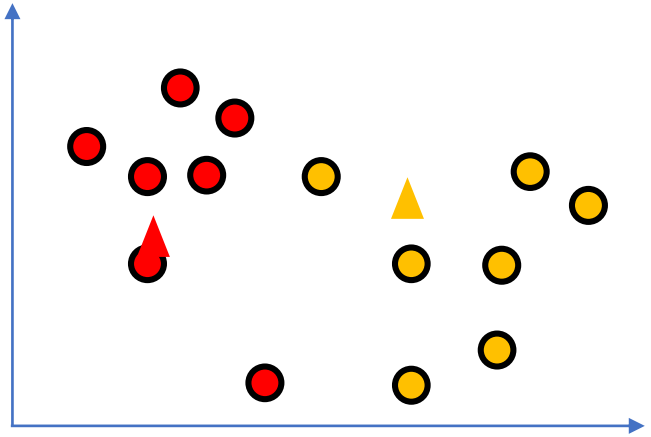
(1) 把最接近▲的點塗上黃色
把最接近▲的點塗上紅色



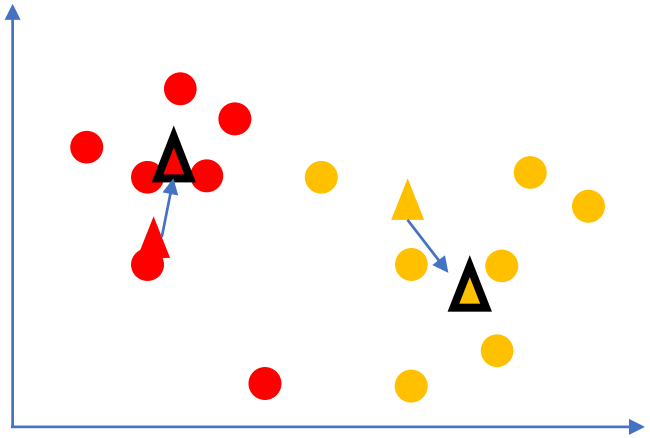
(2) 計算紅色點的平均位置，找出新的▲
計算黃色點的平均位置，找出新的▲



(3)根據新的中心點，重新根據距離最近的三角形，塗上紅色、黃色



計算紅色點的平均位置，找出新的▲
計算黃色點的平均位置，找出新的▲

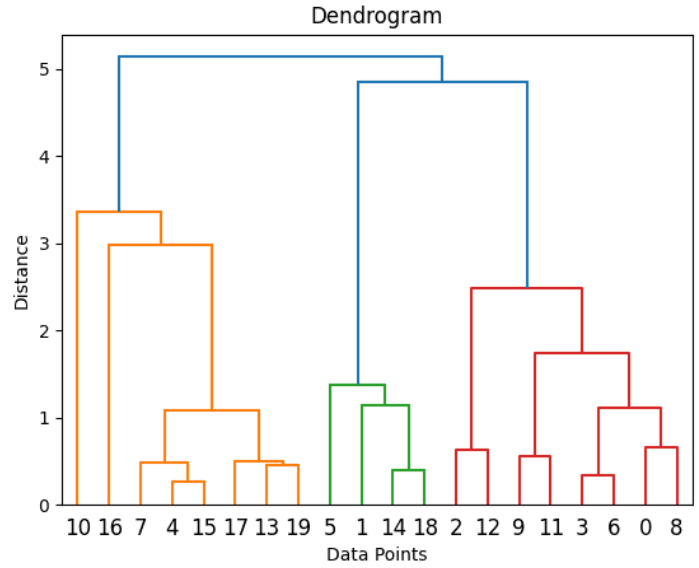


不斷重複步驟 2、3，直到群中心不再變化，或資料點分群結果穩定為止。

階層式分群（Hierarchical Clustering）

- **逐步建立層級結構（Hierarchy）**不需要事先指定群數 K，
- 由個別資料點逐步合併或分裂，形成一棵「樹狀結構圖（Dendrogram

分群方式	說明	適用情境
Agglomerative（凝聚式）	從每個資料點作為一個獨立群開始，逐步合併最相近的兩群，直到只剩一群。	常用（Bottom-up Approach）
Divisive（分裂式）	從全部資料為一群開始，逐步將群分裂為更小的子群。	較少用（Top-down Approach）



階層式分群（Hierarchical Clustering）

Step 1：每個點都是一個群

Step 2：計算 群與群的距離 (Distance Matrix)
根據距離公式（如 Euclidean distance）計算群間距離。
例如資料 A、B、C、D，計算 A-B、A-C、B-C 等距離。

Step 3：合併最近的兩個群
找出距離最小的兩群合併為一群
➔ 繪製樹狀圖中的第一次連線。

Step 4：重新計算群與群之間的距離
根據所選的「連結方式 (Linkage Method)」，重新計算群間距離。

Step 5：重複合併，直到所有點成為一群

不斷重複步驟 3、4，直到所有資料點歸為同一群。
樹狀圖完成後，可依距離（高度）切割出所需的群數。

連結方式 (Linkage Method)

連結法	計算公式	特性
Single Linkage	兩群間最小距離	易形成長鏈狀群
Complete Linkage	兩群間最大距離	群內緊密，群間距離大
Average Linkage	兩群間所有點距離平均	平衡效果佳
Ward’s Method	最小化群內平方誤差	常用於數值型資料

編號	中文成績	英文成績
A	90	60
B	85	65
C	60	90
D	62	85

計算每兩筆資料的Euclidean distance ，建立初始距離矩陣：

	A	B	C	D
A	0	7.07	42.43	37.95
B	7.07	0	35.36	30
C	42.43	35.36	0	7.07
D	37.95	30	7.07	0

Step 1：找最小距離 → 合併
- 最小距離 = 7.07
- 合併 (A, B) → 新群 **AB**

Step 2： 計算新群與其他群的距離
用 **Average Linkage**（平均連結法）

$$D(AB,C) = \frac{D(A,C) + D(B,C)}{2}$$

$$D(AB,D) = \frac{D(A,D) + D(B,D)}{2}$$

	AB	C	D
AB	0	38.9	33.98
C	38.9	0	7.07
D	33.98	7.07	0

$$D(AB,C) = \frac{42.43 + 35.36}{2} = 38.90$$

$$D(AB,D) = \frac{37.95 + 30.00}{2} = 33.98$$

	AB	CD
AB	0	36.44
CD	36.44	0

$$D(AB,CD) = \frac{D(A,C) + D(A,D) + D(B,C) + D(B,D)}{4} = 36.44$$

最終兩群（AB）與（CD），距離為 36.44
若設定「切割高度 = 30」，可得到2群

階層式分群（Hierarchical Clustering）

```python
import math

# 範例資料
data = {'A': (1, 2), 'B': (2, 3), 'C': (6, 8), 'D': (7, 9)}

def distance(p1, p2):
    return math.sqrt((p1[0]-p2[0])**2 + (p1[1]-p2[1])**2)

# Step 1: 建立距離矩陣
dist_matrix = {}
for i in data:
    for j in data:
        if i != j:
            dist_matrix[frozenset({i,j})] = distance(data[i], data[j])

clusters = [{k} for k in data]

while len(clusters) > 1:
    # 找出距離最小的兩群
    pair = min(dist_matrix, key=dist_matrix.get)
    c1, c2 = list(pair)
    new_cluster = set(c1) | set(c2)
    clusters = [c for c in clusters if c != set(c1) and c != set(c2)]
    clusters.append(new_cluster)
    print("合併:", c1, c2, "→", new_cluster)
    # 重新計算距離（使用 single linkage）
    dist_matrix = {}
    for a in clusters:
        for b in clusters:
            if a != b:
                dist_matrix[frozenset({tuple(a), tuple(b)})] = min(
                    distance(data[x], data[y]) for x in a for y in b
                )
```

```
合併: B A → {'B', 'A'}
合併: ('D',) ('C',) → {'D', 'C'}
合併: ('B', 'A') ('D', 'C') → {'D', 'C', 'B', 'A'}
```

項目	階層式分群法（Hierarchical Clustering）	非階層式分群法（K-means Clustering）
分析方式	由資料間距離建立「層級結構」，逐步合併或分割群組	先設定群數 K，反覆調整中心點，使群內距離最小化
需預先設定群數	不需要（會自動生成樹狀結構）	需要指定 K 值（群的數量）
結果呈現方式	樹狀圖（Dendrogram）	散佈圖（各群以不同顏色標示）
適用資料規模	資料量較少（100 筆以內較佳）	適合大量資料（上千筆也可）
優點	可視化群組形成過程、無需先指定群數	計算快速、適合大量資料
缺點	資料多時運算量大、不易修正	結果依初始值而異、需事先設定 K

作業

- 讀入 200 筆學生 chinese, english 成績資料
- 使用 K-means (K=3) 與 階層式分群 (自選 linkage) 各完成一支 Python 程式
- 輸出每位學生的群標籤與每群的成績平均
- 與真實群（true_cluster 欄）做對照，計算**分群準確度 (Accuracy)**

	A	B	C	D
1	student_id	chinese	english	true_cluster
2	S001	86	69	A
3	S002	100	71	A
4	S003	97	70	A
5	S004	100	48	A
6	S005	91	62	A

不得使用 numpy/scikit-learn，可用 csv, math, random 等標準庫
自行實作距離、中心點與合併邏輯。

繳交：
- 二支 Python 程式 (kmean.py, hierarchicalclustering.py)
- Powerpoint 檔：二個程式的執行畫面截圖、分群準確度
- **期限：2週**