CHAPTER 10

進階群集分析

我們已經在第 9 章學習群集分析的基本原理，在本章節，我們探討群集分析的進階議題，更明確地，我們專研底下四個主要方面：

- **機率模型式分群法**：10.1 節介紹推導出機率式群集的一般化框架與方法，其中每個物件皆被指派一個屬於該群集的機率值，機率模型式分群法已廣泛使用在許多資料探勘應用問題中，例如文字探勘。
- **高維度資料分群**：當維度很高時，傳統的距離量測會被雜訊資料支配，10.2 節介紹對高維度資料進行群集分析的基本方法。
- **圖形與網路資料分群**：圖形與網路資料已日趨盛行，諸如社會網路、WWW 與數位圖書館，在 10.3 節，你將學會分群圖形與網路資料的關鍵議題，包含它們的相似度量測，以及分群的方法。
- **限制式分群**：迄今為止的探討中，我們都沒有假設分群法涉及任何限制，然而，在某些應用問題中，可能存在各種各樣的限制式，這些限制式可能是源於背景知識或是物件的空間分佈。在 10.4 節中，你將學習到

如何對不同類型的限制式進行群集分析。

在本章結束時，你將對進階群集分析的議題與技術有很好的理解。

10.1 機率模型式分群法

迄今為止，我們所介紹的群集分析方法，皆是將資料物件指派給數個群集中的其中一個群集，這樣的群集指派規則在某些應用是必要的，例如將顧客指派給銷售經理，然而，在其它應用中，這樣死板的限制可能並非我們所期望的。在本章節中，我們將論證模糊或靈活的群集指派在某些應用中是需要的，並介紹計算機率群集與指派的一般化方法。

「在什麼情況下，資料物件可能歸屬於一個以上的群集？」讓我們考慮以下範例 10.1。

範例 10.1 產品評論分群

AllElectronics 公司擁有線上商場，顧客不僅可以線上購買商品，而且可以為商品撰寫評論。並非所有商品都有顧客撰寫評論，有些商品可能有很多評論，而其它的商品可能沒有評論，或是評論很少，除此之外，一份顧客評論可能涉及多個產品。因此，身為 AllElectronics 公司的評論編輯者，你的任務是為這些評論分群。

理想的狀況下，一個群集對應到一個主題，例如，一組高度相關的產品、服務或議題。將一份評論互斥地指派到單一群集，並無法讓你的任務執行良好。假設有一個群集是對應“數位相機與攝影機”，而另一個群集是“個人電腦”，那麼如果有一份評論探討數位攝影機與個人電腦的相容性，則該如何呢？此評論與這兩個群集都有相關，而無法互斥地屬於其中一個群集。

你將願意使用一個能夠允許評論屬於多個群集的分群方法，如果此評論確實涉及多個主題，為了反映此評論屬於一個群集的強度，你想要對於評論到群集的指派上，對應一個權重值，來表示他的歸屬程度。

資料物件屬於多個群集的狀況，可能會發生在許多情境下，我們將在範例 10.2 中闡述。

範例 10.2 研究使用者搜尋意圖的分群法

AllElectronics 公司線上商場記錄每個顧客瀏覽與購買行為在 log 檔上，而一項重要的資料探勘任務，就是使用這些 log 資料來區分與理解使用者的搜尋意圖。舉例來說，考慮一次使用者 session (使用者與線上商場互動的短期間)，使用者是否在搜尋一個商品、比較不同的商品或是在尋找顧客的支持資訊？群集分析能夠幫助這項任務，因為要完全事先定義使用者的行為樣式，是一件很困難的事情。包含相似的使用者瀏覽軌跡，可能代表相似的使用者行為。

然而，並非每一個 session 都僅屬於唯一個群集，舉例來說，假設某一群集內的使用者 session 涉及購賣數位相機，而另一個群集內的使用者 session 為比較筆記型電腦。那麼如果一個使用者在 session 內購買數位相機，同時又比較數款筆記型電腦，那應該怎麼辦呢？這樣的 session 在某種程度上應該同時屬於兩個群集。

在此節，我們系統化的研究能夠允許一個物件同時屬於多個群集的分群方法，在 10.1.1 節，我們從模糊群集的概念開始，我們接著在 10.1.2 節把此概念推廣到機率模型式群集。在 10.1.3 節，我們介紹期望最大化 (expectation-maximization) 演算法，它是探勘這類群集的一般化框架。

10.1.1 模糊群集

給定一組資料物件 $X = \{x_1, \ldots, x_n\}$，模糊集合 (fuzzy set) S 是 X 的子集合，它允許每個物件 X 擁有 0 至 1 間的歸屬程度，正規地說，模糊集合 S 可以透過函數 $F_S : X \to [0, 1]$ 來模組化。

範例 10.3 模糊集合

數位相機的銷售量越大，代表它越流行，在 AllElectronics 公司，我們可以使用下述的公式，來計算數位相機 o 在給定銷售量 i 時的流行程度

$$pop(o)=\begin{cases}1 & \text{如果 } o \text{ 銷售了1000 部，或是更多}\\ \dfrac{i}{1000} & \text{如果 } o \text{ 銷售了 } i \text{ 部 } (i<1000)\end{cases} \quad \textbf{(10.1)}$$

函數 $pop()$ 定義一個流行的數位相機的模糊集合，舉例來說，假設 AllElectronics 公司的數位相機的銷售量顯示在表格 10.1，此流行的數位相機的模糊集合為 $\{A(0.05), B(1), C(0.86), D(0.27)\}$，其歸屬程度寫在括弧當中。

我們可以套用模糊集合的概念在群集上，也就是說，給定一組物件集合，群集是物件的模糊集合，此群集稱為模糊群集，因此，分群法包含數個模糊群集。

正規地說，給定一組物件集合 $o_1,\ldots,o_n$，模糊分群的 k 個模糊群集 $C_1,\ldots,C_k$ 可以使用分割矩陣 $M=[w_{ij}](1\le i\le n, 1\le j\le k)$ 來表示，其中 w_{ij} 是物件 o_i 屬於模糊群集 C_j 的歸屬程度，此分割矩陣應該滿足下列三項要求：

- 對每一個物件 o_i 與群集 C_j，我們有 $0\le w_{ij}\le 1$，這項要求確保模糊群集是模糊集合。

表 10.1 數位相機與它們在 AllElectronics 公司的銷售量

數位相機	銷售量（台）
A	50
B	1320
C	860
D	270

- 對每一個物件 o_i，我們有 $\sum_{j=1}^{k} w_{ij} = 1$，這項要求確保每個物件相等地參與群集。
- 對每一個群集 C_j，我們有 $0 < \sum_{i=1}^{n} w_{ij} < n$，這項要求確保對每一個模糊群集，至少有一個物件屬於他的歸屬函數值大於 0。

範例 10.4 模糊群集

假設 AllElectronics 公司線上商店有六份顧客評論，包含在這些評論內的關鍵字列在表格 10.2 中。

我們可以將這些評論群組成兩個模糊群集 C_1 與 C_2，其中 C_1 是針對“數位相機”與“鏡頭”，而 C_2 是針對“電腦”，其分割矩陣為

$$M = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ \frac{2}{3} & \frac{1}{3} \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

此處，我們使用關鍵字“數位相機”與“鏡頭”來做為群集 C_1 的特徵，而且關鍵字“電腦”為群集 C_2 的特徵，對於評論 R_i 與群集 $C_j (1 \le i \le 6, 1 \le j \le 2)$，$w_{ij}$ 定義為

$$w_{ij} = \frac{|R_i \cap C_j|}{|R_i \cap (C_1 \cup C_2)|} = \frac{|R_i \cap C_j|}{|R_i \cap \{\text{數位相機，鏡頭，電腦}\}|}$$

在模糊分群中，評論 R_4 屬於群集 C_1 與 C_2 的歸屬程度分別為 2/3 與 1/3。

表 10.2　評論集合與使用的關鍵字

評論 ID	關鍵字
R_1	數位相機、鏡頭
R_2	數位相機
R_3	鏡頭
R_4	數位相機、鏡頭、電腦
R_5	電腦、CPU
R_6	電腦、電腦遊戲

「我們該如何評估模糊群集描述資料集的優劣呢？」考慮一組物件集合 $o_1,\ldots,o_n$，與擁有 k 個群集 $C_1,\ldots,C_k$ 的模糊分群 C，令 $M=[w_{ij}](1\le i\le n,1\le j\le k)$ 為分割矩陣，以及 $c_1,\ldots,c_k$ 分別為模糊分群 $C_1,\ldots,C_k$ 的中心點，此處，群集中心點可以為平均值或中位數，或是根據應用問題的特色來定義。

如同第 9 章所探討的，資料物件與它所指派的群集之間的距離（或相似度），可以用來量測此物件屬於該群集的優劣，對任意物件 o_i 與群集 C_j，如果 $w_{ij}>0$，則 $dist(o_i,c_j)$ 量測物件 o_i 被 c_j 表達的程度，也就是它屬於群集 C_j 的程度。由於一個物件可能歸屬於多個群集，到那些群集中心點的距離與歸屬程度加權後的總合，可以代表此物件符合分群結果優劣程度。

更正式地說，對於物件 o_i，誤差平方和 (sum of squared error, SSE) 是根據下式計算

$$SSE(o_i)=\sum_{j=1}^{k} w_{ij}^{p}\,dist(o_i,c_j)^2 \qquad \textbf{(10.2)}$$

其中參數 $p(p\ge 1)$ 控制歸屬程度的影響力，p 值越大，歸屬程度的影響越大。而對於群集 C_j 的 SSE 為

$$SSE(C_j)=\sum_{i=1}^{n} w_{ij}^{p}\,dist(o_i,c_j)^2 \qquad \textbf{(10.3)}$$

最終，分群結果的 *SSE* 定義為

$$SSE(C) = \sum_{i=1}^{n}\sum_{j=1}^{k} w_{ij}^{p} dist(o_i, c_j)^2 \qquad \textbf{(10.4)}$$

此 *SSE* 可用來量測模糊分群契合資料集的程度。

模糊分群也稱為軟式分群 (soft clustering)，因為它允許一個物件歸屬於多個群集，我們可以輕易看出，強制每個物件互斥地歸屬於單一群集的傳統 (硬式) 分群法，是模糊分群的一個特例。我們將在 10.1.3 節介紹如何計算模糊分群。

10.1.2 機率模型式群集

「模糊群集 (10.1.1 節) 提供允許一個物件能參與多個群集的靈活彈性，是否存在機率式方法，來指定物件可以參與多個群集的一般化分群框架？」在此節，我們介紹機率模型式分群的一般概念，來回答此問題。

如同第 9 章所探討的，因為我們假定在資料集中的物件，實際上是屬於不同的內在類別，所以我們在資料集上執行群集分析。回顧 9.5.1 節介紹的群集趨勢分析，可用來檢測資料集中的物件是否存在有意義的群集結構。此處，隱藏在資料集的內部類別稱為潛在 (latent)，代表它們無法被直接觀察到，相對地，我們必須使用觀察到的資料來推論它們。舉例來說，隱藏在 AllElectronics 公司線上商店的商品評論中的主題，便是潛在的，因為我們無法直接看到這些主題。然而，我們可以從評論當中來推導出這些主題，因為每一份評論都涉及一個或多個主題。

因此，群集分析的目標是要找出隱藏的類別，做為群集分析目標的資料集，可視為隱藏類別中可能範例的樣本，但它們沒有包含任何類別標籤。從群集分析中推導出群集，是透過從資料集中推論，並設計來近似這些隱藏類別。

就統計上來說，我們可以假定隱藏類別為資料空間上的一個分佈，並可以使用機率密度函數 (或分佈函數) 來數學化的表示，我們稱這樣的

隱藏類別為機率式群集。對於一個機率式群集 C，以及他的機率密度函數 f 與資料空間上的一個資料點 o，$f(o)$ 是 C 中一個實例出現在 o 的相對概似率。

範例 10.5 **機率式群集**

假設 AllElectronics 公司販售的數位相機可以分成兩個類別：C_1 業餘型（例如，傻瓜相機）與 C_2 專業型（例如，單眼相機），它們對應的機率密度函數分別為 f_1 與 f_2（對應於變數“價格”），顯示在圖 10.1 中。

對於價格 $1000 美元為例子，$f_1$ (1000) 為業餘型數位相機價格為 $1000 的相對概似率，而，$f_2$ (1000) 則為專業型數位相機價格為 $1000 的相對概似率。

機率密度函數 f_1 與 f_2 不可以直接觀察到，相對地，AllElectronics 公司僅能夠透過分析這些數位相機的售價，來推論這些分佈。除此之外，數位相機通常不是來自良好定義的類別（例如，業餘型或專業型），相對地，這些類別通常是根據使用者的背景知識，而且可能因人而異，類單眼相機可能被某些顧客視為業餘型相機的高階機種，而其餘顧客可能將之視為專業相機的低階機種。

做為 AllElectronics 公司的分析師，你可能將每一個類別視為一個機率式群集，並對相機的價格執行群集分析，來近似那些類別。

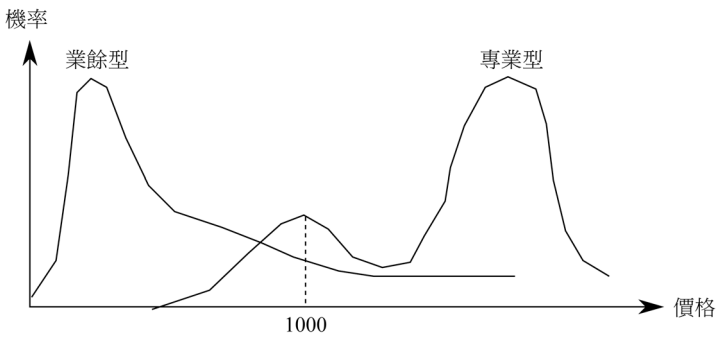

圖 10.1 兩個機率式群集的機率密度函數。

假設我們想要透過群集分析來找出 k 個機率式群集 $C_1,\ldots,C_k$，對於擁有 n 個物件的資料集合 D，我們可以將 D 視為群集可能實例的有限樣本集，概念上，我們可以假設 D 按如下方式形成，每一個群集 $C_j(1\le j\le k)$ 皆對應到一個機率值 w_j，代表某實例乃是由該群集取樣而來的機率，通常假設 $w_1,\ldots,w_k$ 值的給定為問題設定中的一部分，而且 $\sum_{j=1}^{k} w_j = 1$，這確保所有物件皆是由這 k 個群集所產生。此處，參數 w_j 捕捉了群集 C_j 佔總體相對比率的背景知識。

我們接著執行下列兩個步驟來產生 D 中的一個物件，這兩個步驟總共執行 n 次，來產生 D 中的 n 筆物件 $o_1,\ldots,o_n$。

1. 根據機率值 $w_1,\ldots,w_k$ 來選取群集 C_j。
2. 根據群集 C_j 的機率密度函數 f_j，來選取 C_j 的實例。

此資料產生程序是混合模型的基本假設，混合模型 (mixture models) 假設觀察物件集合為多個機率式群集所產生的實例之混合，概念上來說，每一個觀察物件是獨立地由下面兩個步驟產生：首先，根據群集的機率值來選取一個群集，接著根據選取到群集的機率密度函數來取樣。

給定資料集 D 以及所要求的群集數目 k，機率式群集分析的任務是，推論出最有可能使用上述資料產生程序來產生資料集 D 的 k 個機率式群集。而接下來重要的問題是，我們如何量測 k 個機率式群集以及它們對應的機率值，將會產生觀察資料集的概似率 (likelihood)。

考慮具有 k 個機率式群集 $C_1,\ldots,C_k$ 的分群 C，與它們分別對應的機率密度函數 $f_1,\ldots,f_k$，以及它們的機率值 $w_1,\ldots,w_k$，對於物件 o，它是由群集 $C_j(1\le j\le k)$ 所產生的機率為 $P(o\mid C_j)=w_j f_j(o)$。因此，物件 o 是由分群 C 產生的機率為

$$P(o\mid C)=\sum_{j=1}^{k} w_j f_j(o) \qquad \textbf{(10.5)}$$

由於我們假設物件是獨立地產生，對於擁有 n 筆物件 $\{o_1,\ldots,o_n\}$ 的資料集合 D，我們有

$$P(D|C)=\prod_{i=1}^{n}\sum_{j=1}^{k}w_j f_j(o_i) \tag{10.6}$$

現在，機率模型式群集分析在資料集 D 上面的任務已經很清楚了，他要找出具有 k 個機率群集的分群 C，使得 $P(D|C)$ 能夠最大化。最大化 $P(D|C)$ 通常是棘手的工作，因為一般來說，群集的機率密度函數可能是任意複雜形式的函數。為了使機率模型式分群是計算可行的，我們通常採折衷方案，假設機率密度函數是參數式分佈。

正規地說，假設 $o_1,\ldots,o_n$ 為 n 個觀察到物件，並令 $\Theta_1,\ldots,\Theta_k$ 為 k 個分佈的參數，並表示為 $O=\{o_1,\ldots,o_n\}$ 與 $\Theta=\{\Theta_1,\ldots,\Theta_k\}$，則對任意物件 $o_i\in O$，公式 (10.5) 可改寫為

$$P(o_i|\Theta)=\sum_{j=1}^{k}w_j P_j(o_i|\Theta_j) \tag{10.7}$$

其中 $P_j(o_i|\Theta_j)$ 是物件 o_i 由第 j 個分佈使用參數 Θ_j 來產生的機率，因此，公式 (10.6) 可改寫為

$$P(O|\Theta)=\prod_{i=1}^{n}\sum_{j=1}^{k}w_j P_j(o_i|\Theta_j) \tag{10.8}$$

使用參數化機率分佈模型，機率式群集分析的任務便是推論能最大化公式 (10.8) 的參數集合 Θ。

範例 10.6 單變量高斯混合模型

讓我們以單變量高斯分佈為範例，也就是說，我們假設每一個群集的機率密度函數為 1-D 高斯分佈。假設有 k 個群集，每一個群集的機率密度函數使用兩個參數，中心 u_j 與標準差 σ_j，我們令 $\Theta_j=(u_j,\sigma_j)$ 且 $\Theta=\{\Theta_1,\ldots,\Theta_k\}$，令資料集為 $O=\{o_1,\ldots,o_n\}$，其中 $o_i(1\le i\le n)$ 為實數，我們有

$$P(o_i|\Theta_j)=\frac{1}{\sqrt{2\pi}\sigma_j}e^{-\frac{(o_i-\mu_j)^2}{2\sigma^2}} \tag{10.9}$$

假設每個群集具有相同的機率值，亦即 $w_1 = w_2 = \cdots w_k = 1/k$ ，並將公式 (10.9) 代入 (10.7)，我們得到

$$P(o_i \mid \Theta) = \frac{1}{k}\sum_{j=1}^{k}\frac{1}{\sqrt{2\pi}\sigma_j}e^{-\frac{(o_i-\mu_j)^2}{2\sigma^2}} \quad \textbf{(10.10)}$$

代入公式 (10.8)，我們有

$$P(O \mid \Theta) = \frac{1}{k}\prod_{i=1}^{n}\sum_{j=1}^{k}\frac{1}{\sqrt{2\pi}\sigma_j}e^{-\frac{(o_i-\mu_j)^2}{2\sigma^2}} \quad \textbf{(10.11)}$$

使用單變量高斯混合模型的機率模型式群集分析的任務便是，推論出能夠最大化公式 (10.11) 的參數集 Θ。

10.1.3 期望最大化演算法

「我們該如何計算模糊分群與機率模型式分群？」在此章節，我們介紹一個原理式方法，讓我們回顧第 9 章介紹的 *k*-means 分群問題與 *k*-means 演算法。

可以容易證明 *k*-means 分群問題是模糊分群的一個特例 (習題 10.1)，*k*-means 分群演算法將持續疊代，直至分群結果無法改善為止，每一次疊代包含底下兩步驟：

- **期望步驟** (expectation step, E-step)：給定目前群集中心，每一個物件被指派給與群集中心距離最近的群集，此處，期望資料物件是屬於最接近的群集。
- **最大化步驟** (maximization step, M-step)：給定此群集指派，對每一個群集，此演算法調整每一個群集的中心，使得物件與所指派群集的新的群集中心點的距離總合，是最小化的。也就是，物件與指派到群集之間的相似度，為最大化的。

我們可以這兩步驟方法來處理模糊分群與機率模型式分群，一般來

說，期望最大化 (expectation-maximization, EM) 演算法是能夠逼近統計模型的最大化概然率或後驗參數估計的一般化框架。在模糊或機率模型式分群的情況下，EM 演算法從初始參數集合開始，並持續疊代，直至分群結果沒有改善為止，也就是說，直到分群收斂，或是改變足夠小為止 (小於預先設定的門檻值)。每一次疊代包含以下兩個步驟：

- **期望步驟**：根據目前的模糊分群或機率群集的參數，將物件指派至群集中。
- **最大化階段**：找出新的分群或是參數，使得能夠最大化模糊分群中的 SSE (公式 10.4)，或是最大化機率模型式分群中的期望概似率。

範例 10.7 模糊分群：使用 EM 演算法

考慮圖 10.2 中的 6 個資料點，該點的座標亦顯示在圖上。讓我們使用 EM 演算法來計算兩個模糊群集。

我們隨機選取兩個資料點做為群集的初始中心，例如 $c_1 = a$ 與 $c_2 = b$，第一次疊代中的期望與最大化步驟執行如下：

在 E-步驟中，我們計算每一個資料點屬於各個群集的歸屬程度，對於任意點 o，我們指派 o 至群集 c_1 與 c_2 的歸屬程度分別為

$$\frac{\frac{1}{dist(o,c_1)^2}}{\frac{1}{dist(o,c_1)^2}+\frac{1}{dist(o,c_2)^2}} = \frac{dist(o,c_2)^2}{dist(o,c_1)^2 + dist(o,c_2)^2}$$

與

$$\frac{dist(o,c_1)^2}{dist(o,c_1)^2 + dist(o,c_2)^2}$$

其中 $dist(,)$ 為歐基里德距離，其理由是如果 o 接近於 c_1，則 $dist(o, c_1)$ 值較小，而 o 歸屬為 c_1 的歸屬程度應當較高，我們同時正規化歸屬程度，使得物件屬於各個群集的歸屬程度總合等於 1。

對於資料點 a，我們有 $w_{a,c_1} = 0$ 與 $w_{a,c_2} = 1$，亦即 a 互斥地歸屬於 c_1，

對於資料點 b，我們有 $w_{b,c_1}=1$ 與 $w_{b,c_2}=0$，亦即 b 互斥地歸屬於 c_2，對於資料點 c，我們有 $w_{c,c_1}=41/(45+41)=0.48$ 與 $w_{c,c2}=45/(45+41)=0.52$，其它點的歸屬程度顯示在表格 10.3 的分割矩陣。

在 M-步驟，我們根據分割矩陣重新計算群集中心，為了最小化公式 (10.4) 中的 SSE，新的群集中心應該調整為

$$c_j=\frac{\sum_{\text{each point } o} w_{o,c_j}^2 o}{\sum_{\text{each point } o} w_{o,c_j}^2} \tag{10.12}$$

其中 $j=1,2$，在此範例中

$$\begin{aligned} c_1 &= \left(\frac{1^2\times 3+0^2\times 4+0.48^2\times 9+0.42^2\times 14+0.41^2\times 18+0.47^2\times 21}{1^2+0^2+0.48^2+0.42^2+0.41^2+0.47^2},\right.\\ &\quad \left.\frac{1^2\times 3+0^2\times 10+0.48^2\times 6+0.42^2\times 8+0.41^2\times 11+0.47^2\times 7}{1^2+0^2+0.48^2+0.42^2+0.41^2+0.47^2}\right)\\ &= (8.47, 5.12) \end{aligned}$$

與

$$\begin{aligned} c_2 &= \left(\frac{0^2\times 3+1^2\times 4+0.52^2\times 9+0.58^2\times 14+0.59^2\times 18+0.53^2\times 21}{0^2+1^2+0.52^2+0.58^2+0.59^2+0.53^2},\right.\\ &\quad \left.\frac{0^2\times 3+1^2\times 10+0.52^2\times 6+0.58^2\times 8+0.59^2\times 11+0.53^2\times 7}{0^2+1^2+0.52^2+0.58^2+0.59^2+0.53^2}\right)\\ &= (10.42, 8.99) \end{aligned}$$

我們重複此疊代，每一次疊代包含 E-步驟與 M-步驟，表格 10.3 顯示前 3 次疊代的結果，當群集中心點收斂或改變足夠小時，則此演算法便停止。

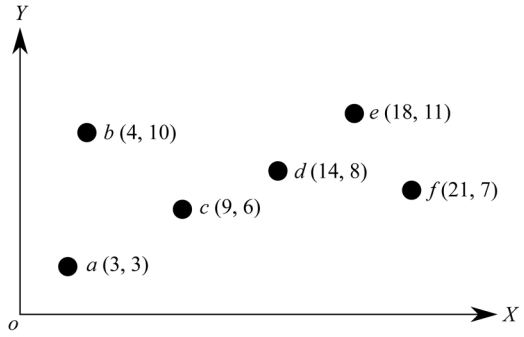

圖 10.2 模糊分群的資料集。

表 10.3 範例 10.7 中 EM 演算法前三次疊代執行的中繼結果

疊代	E-步驟	M-步驟
1	$M^T = \begin{bmatrix} 1 & 0 & 0.48 & 0.42 & 0.41 & 0.47 \\ 0 & 1 & 0.52 & 0.58 & 0.59 & 0.53 \end{bmatrix}$	$c_1 = (8.47, 5.12)$ $c_2 = (10.42, 8.99)$
2	$M^T = \begin{bmatrix} 0.73 & 0.49 & 0.91 & 0.26 & 0.33 & 0.42 \\ 0.27 & 0.51 & 0.09 & 0.74 & 0.67 & 0.58 \end{bmatrix}$	$c_1 = (8.51, 6.11)$ $c_2 = (14.42, 8.69)$
3	$M^T = \begin{bmatrix} 0.80 & 0.76 & 0.99 & 0.02 & 0.14 & 0.23 \\ 0.20 & 0.24 & 0.01 & 0.98 & 0.86 & 0.77 \end{bmatrix}$	$c_1 = (6.40, 6.24)$ $c_2 = (16.55, 8.64)$

「如何套用 EM 演算法來計算機率模型式分群？」，讓我們使用單變量高斯混合模型 (範例 10.6) 來闡述。

範例 10.8 使用 EM 演算法於混合模型

給定一組物件 $O = \{o_1, \ldots, o_n\}$ 集合，我們想要探勘參數集合 $\Theta = \{\Theta_1, \ldots, \Theta_k\}$，使得公式 (10.11) 中的 $P(O|\Theta)$ 能夠最大化，其中 $\Theta_j = (u_j, \sigma_j)$ 分別為第 j 個單變量高斯分佈 $(1 \le j \le k)$ 的平均值與標準差。

我們可以套用 EM 演算法，隨機指派參數 Θ 的初始值，接著疊代地執行以下的 E-步驟與 M-步驟，直至參數收斂或是改變足夠小時才停止。

在 E-步驟中，對每一個物件 $o_i \in O(1 \le i \le n)$，我們計算物件 o_i 屬於

每一個分佈的機率值，也就是

$$P(\Theta_j \mid o_i, \Theta) = \frac{P(o_i \mid \Theta_j)}{\sum_{l=1}^{k} P(o_i \mid \Theta_l)} \quad \textbf{(10.13)}$$

在 M-步驟中，我們調整參數 Θ 的值，使得公式 (10.11) 中的期望概似率 $P(O|\Theta)$ 能夠最大化，這可透過設定

$$\mu_j = \frac{1}{k}\sum_{i=1}^{n} o_i \frac{P(\Theta_j \mid o_i, \Theta)}{\sum_{l=1}^{n} P(\Theta_j \mid o_l, \Theta)} = \frac{1}{k}\frac{\sum_{i=1}^{n} o_i P(\Theta_j \mid o_i, \Theta)}{\sum_{i=1}^{n} P(\Theta_j \mid o_i, \Theta)} \quad \textbf{(10.14)}$$

與

$$\sigma_j = \sqrt{\frac{\sum_{i=1}^{n} P(\Theta_j \mid o_i, \Theta)(o_i - \mu_j)^2}{\sum_{i=1}^{n} P(\Theta_j \mid o_i, \Theta)}} \quad \textbf{(10.15)}$$

來達成。

在許多應用當中，機率模型式分群演算法已表現出它的有效性，因為它比分割式與模糊分群方法更通用，它具有一個明顯的優點，便是它可以使用適當的統計模型來捕獲潛在的群集，因為 EM 演算法的計算簡單，使得它廣泛用於處理資料探勘與統計學中的許多學習問題。請注意，一般來說，EM 演算法可能不會收斂至全域最佳解，而可能僅收斂至區域最大值，有發展出許多啟發式方法來避免此問題，舉例來說，我們可以執行 EM 演算法多次，每一次使用不同的隨機初始值。除此之外，當分佈的數目很大，或是資料集包含的觀察值極少，則 EM 演算法的成本可能會非常大。

10.2 高維度資料分群

迄今為止，我們所探討的分群方法在維度不高的資料集上（亦即，少於 10 個屬性）都能執行良好，然而，某些重要的應用問題擁有高維度資料，「如何能在高維度資料集上執行群集分析呢？」

在此章節中，我們研究高維度資料分群的方法，10.2.1 節從介紹高維度分群主要的挑戰與所使用的方法開始，高維度資料分群的方法可以分成兩類：子空間分群方法（10.2.2 節）與維度精簡方法（10.2.4 節）。

10.2.1 高維度資料分群：問題、挑戰與主要方法

在我們介紹高維度資料分群的具體方法之前，讓我們先使用範例來說明高維度分群的必要性，檢視新方法所需要的挑戰，接著，我們根據是否在原來空間上的子空間上來搜尋群集，或是建立一個較低維度的新空間並且在那裡搜尋群集，而將主要方法加以分類。

在某些應用中，資料物件可能是由 10 個以上的屬性所描述，我們稱這些物件在高維度資料空間內。

範例 10.9 高維度資料與分群

AllElectronics 公司記錄每一位顧客所購買的產品，做為一位顧客關係管理經理，你想要根據顧客從 AllElectronics 公司購買哪些產品，而對顧客進行分群。

顧客購買資料的維度非常高，AllElectronics 公司銷售數萬種產品，因此，顧客的購物簡況是由公司銷售產品所組成的向量，它的維度是數萬的。

「經常用於低維度群集分析的傳統距離量測，是否在高維度資料中依然有效？」考慮表格 10.4 中的顧客資料，其中我們使用 10 個

產品 $(P_1, \ldots, P_{10})$ 來說明，如果顧客購買了某個產品，則對應位置的位元被設置為 1，否則該位元設置為 0，讓我們來計算 Ada, Bob 與 Cathy 之間的歐基里德距離 (公式 2.16)，很容易可以看出

$$dist(\text{Ada}, \text{Bob}) = dist(\text{Bob}, \text{Cathy}) = dist(\text{Ada}, \text{Cathy}) = \sqrt{2}$$

根據歐基里德距離，這三個物件的相似性 (或相異性) 是彼此相等的，然而，進一步觀察告訴我們，Ada 與 Cathy 應該比 Ada 與 Bob 更為接近，因為 Ada 與 Cathy 皆有購買產品 P_1。

表 10.4　顧客購物資料

顧客	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}
Ada	1	0	0	0	0	0	0	0	0	0
Bob	0	0	0	0	0	0	0	0	0	1
Cathy	1	0	0	0	1	0	0	0	0	1

如同範例 10.9 所顯示，傳統的距離量測在高維度資料上可能是無效的，這些距離量測可能被其他維度上的誤差所主導，因此在高維度上的群集可能是不可靠的，找出這樣的群集是沒有意義的。

「那麼，什麼類型的群集在高維度上是有意義的呢？」對於高維度空間上的資料分群，我們依舊是要將相似的物件群組在一起，然而，由於資料空間通常太大太混亂了，沿伸而出的挑戰就是我們不只要找出群集，還要對每一個群集，找出顯示它們為群集的屬性集合。換句話說，高維度資料空間上的群集通常是使用較小的屬性集合所定義，而不是整個資料空間。本質上，高維度資料分群應該傳回物件的群集 (如同傳統群集分析一般)，此外，對於每一個群集，還要傳回刻劃該群集特徵的屬性集合。舉例來說，在表格 10.4 中，刻劃 Ada 與 Cathy 之間相似性的屬性 P_1 應該回傳，因為 Ada 與 Cathy 皆有購買 P_1。

高維度資料分群是在尋找群集與它們所存在的空間，有兩種主要的方法：

- 子空間分群方法尋找在給定高維度資料空間中，存在有群集的子空間，其中子空間是由完整空間中屬性的子集合所定義的，子空間分群方法將在 10.2.2 節介紹。
- 維度精簡方法嘗試建構一個更低維度的空間，並且在該空間上搜尋群集，通常，該方法是藉由合併原始資料中某些維度成為新的維度，維度精簡方法是 10.2.4 節的主題。

一般來說，高維度資料分群所帶來全新的挑戰有：

一個主要議題是如何建構在高維度空間上合適的群集模型，不同於在低維度空間中的群集，隱藏在高維度空間中的群集通常是明顯地更小。舉例來說，當為顧客購物資料進行分群，我們並不期望有許多顧客擁有相同的購物樣式，搜尋這些小型又有意義的群集，就有如在乾草堆中尋找一根細針般困難。如同之前的範例所顯示，傳統的距離量測在此處可能不適用的，相反地，我們必須考慮各種更精密複雜的技術，來模組化在子空間中資料物件之間的相關性與一致性。

通常會有指數數目的可能的子空間或維度精簡的選項，因此要找出最佳解，其計算成本通常是高到無法負荷，舉例來說，如果原始資料空間有 1000 維，而我們想要找出維度為 10 的群集，則總共有 $\binom{1000}{10} = 2.63 \times 10^{23}$ 種可能的子空間。

10.2.2 子空間分群方法

「我們如何從高維度資料中發掘子空間上的群集？」已經有許多方法提出，它們通常可以分成三個主要的類別：子空間搜尋方法、基於相互關係式的分群方法、與雙分群方法。

子空間搜尋方法

子空間搜尋方法為群集尋找多個子空間 (subspace)，此處，群集是在

一個子空間上面彼此相似的物件集合。相似度可用傳統的方法量測，例如距離或密度。舉例來說，10.5.2 節介紹的 CLIQUE 演算法便是子空間分群演算法，它以維度遞增的順序來枚舉出子空間，以及找出子空間上的群集，並且套用反遞增性來修剪不可能存在群集的子空間。

子空間搜尋方法所面對的挑戰是如何有效與有效率的搜尋一系列的子空間，一般來說有兩種策略：

- **由下而上**方法從低維度子空間開始，並且只有當更高維度的子空間有可能存在群集時，才會搜尋那些更高維度的子空間，已經探索出許多種修剪的技術，來減少需要蒐尋的更高維度的子空間的數目，CLIQUE 是由下而上方法的範例。
- **由上而下**方法從完整的空間開始，遞迴地搜尋越來越小的子空間，由上而下方法只有當區域性假設成立時，才會顯得有效，該假設需要子空間的群集是可以由它的區域鄰居來決定。

範例 10.10 PROCLUS，由上而下的方法

PROCLUS 是一種類似 *k*-medoid 的方法，它首先對資料集合採樣，並從中產生高維度資料空間中 *k* 個潛在的群集中心，它接著疊代式地精煉子空間的群集，在每一次疊代中，對每一個目前的 *k*-medoid，PROCLUS 考慮此 medoid 在整個資料集中的區域鄰居，並藉由在每一個維度上最小化區域鄰居點至 medoid 的距離的標準差，來判別出子空間，一但對此 medoid 的所有子空間皆決定了，資料集中所有的樣本點皆根據對應的子空間被指派給最近的 medoid，群集與離群值被判別出來。在下一次疊代時，新的 medoid 取代現有的，如果這樣做能更提升分群的品質。

基於相互關係式的分群方法

當子空間搜尋方法尋找群集所用的相似度量測是傳統的尺度 (距離或

密度)，基於相互關係式的分群方法能定義進階的相互關係模型，來進一步的發掘出群集。

範例 10.11 使用 PCA 的相互關係式方法

舉例來說，以 PCA (主成份分析，見課本第 3 章) 為基礎的方法，首先套用 PCA 來推導出一組新的、沒有相關的維度，接著在此新的空間與他的子空間上面探勘群集，除了 PCA 之外，也可以使用其它的空間轉換技術，例如 Hough 轉換或是碎形維度 (fractal dimensions)。

雙分群方法

在某些應用中，我們想要同時的對物件與屬性進行分群，所得到的群集稱為雙群集 (biclusters)，並且滿足下列四項要求：(1) 只有少數的物件集合參與在群集內；(2) 一個群集只涉及少數的屬性；(3) 一個資料物件可能參與在多個群集內，或是不屬於任何群集；(4) 一個屬性可能涉及到多個群集，或是不涉及任何群集。我們將在 10.2.3 節詳細介紹雙分群技術。

10.2.3 雙分群方法

在迄今所探討的群集分析中，我們根據資料物件的屬性值來群集它們，物件與屬性不是使用相同的方式來對待，然而，在某些應用中，物件與屬性是以對稱的方式定義，而資料分析涉及從資料矩陣中找出顯示獨特樣式的子矩陣做為群集，這類型的分群技術屬於雙分群 (biclustering) 方法。

應用範例

雙分群技術最早是為了滿足分析基因表達的需求而提出的，基因是有機生命體向其後代傳遞遺傳特徵的單元，一般來說，基因駐留在 DNA 片段上，對於所有的生物，基因都是至關重要的，因為他指定了所有的蛋白

質與功能 RNA 鏈，它們保留建構與維護有機生命體細胞的訊息，以及傳遞給後代的遺傳特徵。合成的功能性基因成品（蛋白質或 RNA）都是依賴基因表達 (gene expression)，基因表達是研究遺傳學的基礎。

使用 DNA 晶片（也稱為 DNA 微陣列），我們可以量測在不同實驗條件下，大量有機生命體的基因的表達程度，這些實驗條件可能對應到實驗中不同的時間點，或是不同器官的樣本，粗略地說，基因表達資料（或 DNA 微陣列資料）為基因－樣本 / 條件的矩陣，其中每一列對應到一個基因，而每一行對應到一個條件或樣本，矩陣中每一個元素是一個實數值，它記錄基因在特定條件下的表達程度，圖 10.3 顯示基因表達資料的範例。

從分群的觀點來看，有趣的是基因表達資料能夠從兩個維度來分析—基因維度與樣本 / 條件維度。

- 當從基因維度分析時，我們將每個基因視為物件對待，而將樣本/條件視為屬性對待，藉由對基因維度探勘，我們可以發掘出多個基因共享的樣式，或是將基因群組成群集，例如我們可能想找出一組表達彼此相似的基因群集，這對於生物資訊學是高度感興趣的，例如可幫助找出生物路徑。

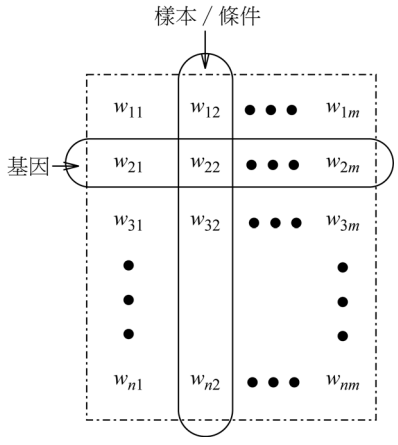

圖 10.3 微陣列資料矩陣。

- 當從樣本 / 條件維度分析時，我們將每一個樣本 / 條件視為物件對待，而將基因視為屬性對待，使用此方式，我們可以找出樣本 / 條件中的樣式，或是將樣本 / 條件群組成群集。舉例來說，我們可能想要藉由比較腫瘤與非腫瘤樣本間，來發掘它們基因表達之間的差異。

範例 10.12 基因表達

基因表達矩陣在生物資訊研究與開發中非常盛行，例如，我們可能想要使用一個新基因的表達與其他已知類別的基因表達資料，來對新基因進行分類。對稱地，我們可能想要使用一個新樣本 (例如，新的病患) 的表達與其它已知類別 (例如，腫瘤或非腫瘤) 的樣本的表達資料，來對新樣本進行分類，這樣的任務對於理解疾病的機制與醫療處置是極富價值的。

如我們所見，基因表達資料探勘與群集分析是高度相關的，然而，它的挑戰在於，並非僅對單一維度 (基因或樣本 / 條件) 進行分群，在許多情況下，我們需要同時對兩個維度進行分群 (即，同時對基因與樣本 / 條件)。此外，不同於我們之前所探討的群集模型，在基因表達資料矩陣中的群集是一個子矩陣，而且通常具有下列特徵。

- 只有少數的基因集合參與在群集內。
- 一個群集只涉及少數的樣本 / 條件集合。
- 一個基因可能參與在多個群集內，或是不屬於任何群集。
- 一個樣本/條件可能涉及到多個群集，或是不涉及任何群集。

要找出基因－樣本 / 條件矩陣中的群集，我們需要符合下述需求的新雙分群技術：

- 基因群集是由少數的樣本 / 條件子集合所定義的。
- 樣本 / 條件群集是由少數的基因子集合所定義的。
- 群集既非是互斥的 (例如，一個基因可能參與多個群集中)，也非是徹底的 (例如，一個基因可能沒有參與任何群集)。

雙分群不只對於生物資訊是很有用的，它對於其它的應用也是一樣有用，我們考慮推薦系統為範例。

範例 10.13 對推薦系統使用雙分群法

AllElectronic 公司收集顧客對產品的評價資料，並使用這些資料來對顧客推薦產品，這些資料可以模組成顧客－產品矩陣，其中每一列代表一個顧客，每一行代表一個產品，矩陣中每一個元素代表顧客對該產品的評價，它們可能是評分 (例如，喜歡、有點喜歡、不喜歡) 或是購買行為 (例如，買或不買)，圖 10.4 闡述該結構。

顧客－產品矩陣可以使用兩個維度來分析：顧客維度或產品維度。將顧客視為物件與產品視為屬性對待，AllElectronic 公司能找出具有相似偏好或購買樣式的顧客群組。將產品視為物件與顧客視為屬性對待，AllElectronic 公司能找出具有相似客戶感興趣的產品群組。

此外，AllElectronic 公司能同時對顧客與產品探勘出群集，這樣的群集包含一部分的顧客子集合，也僅涉及到一部分的產品子集合。例如，AllElectronic 公司可能對於找出喜愛相同產品群組的顧客群組十分感興趣，這樣的群集是顧客－產品矩陣中的一個子矩陣，而且子矩陣中的元素值都非常高。使用這樣的群集，AllElectronic 公司能夠使用兩個方向來進行推薦，首先，公司可以對新顧客推薦產品，它是透過分析新顧客與群集內哪些顧客相似而進行推薦的。其次，它也可對顧客推薦新產品，這是根據新產品與群集內哪些產品相似而進行推薦的。

	產　品			
顧　客	w_{11}	w_{12}	…	w_{1m}
	w_{21}	w_{22}	…	w_{2m}
	…	…	…	…
	w_{n1}	w_{n2}	…	w_{nm}

圖 10.4 顧客－產品矩陣。

如同基因表達資料矩陣中的雙群集一般，在顧客－產品矩陣中的雙群集通常有以下特徵：

- 只有少數的顧客集合參與在群集內。
- 一個群集只涉及少數的產品集合。
- 一個顧客可能參與在多個群集內，或是不屬於任何群集。
- 一個產品可能涉及到多個群集，或是不涉及任何群集。

我們可以套用雙分群技術於顧客－產品矩陣上，來探勘出滿足這些需求的群集。

雙群集的類型

「我們該如何模組化雙群集，並且探勘它們呢？」，讓我們從基本概念開始，為了簡化問題，在以下討論中，我們使用“基因”與“條件”做為兩個維度，這可以輕易的延伸到其它應用問題中，例如，我們可以簡單地將“顧客”與“產品”取代“基因”與“條件”，來處理顧客－產品的雙分群問題。

令 $A=\{a_1,\ldots,a_n\}$ 為基因集合，$B=\{b_1,\ldots,b_m\}$ 為條件集合，令 $E=[e_{ij}]$ 為基因表達矩陣，也就是基因－條件矩陣，其中 $1\le i\le n$ 與 $1\le j\le m$。一個子矩陣 $I\times J$ 是由基因子集合 $I\subseteq A$ 與條件子集合 $J\subseteq B$ 定義而成的，舉例來說，圖 10.5 中的 $\{a_1,a_{33},a_{86}\}\times\{b_6,b_{12},b_{36},b_{99}\}$ 便是一個子矩陣。

	...	b_6	...	b_{12}	...	b_{36}	...	b_{99}...
a_1	...	60...	...	60...	...	60...	...	60...
...	...	...	...	...	...	...	...	... ...
a_{33}	...	60...	...	60...	...	60...	...	60 ...
...	...	...	...	...	...	...	...	... ...
a_{86}	...	60...	...	60...	...	60...	...	60 ...
...	...	...	...	...	...	...	...	... ...

圖 10.5 基因－條件矩陣，一個子矩陣與一個雙群集。

雙群集是遵循一致樣式的子矩陣 (由基因與條件組成)，我們可以根據這些樣式定義不同類型的雙群集。

- 最簡單的例子是，子矩陣 $I \times J (I \subseteq A, J \subseteq B)$ 是**具有固定常數值的雙群集** (bicluster with constant values)，若對任何 $i \in I$ 與 $j \in J$，我們有 $e_{ij} = c$，其中 c 是一個常數。舉例來說，圖 10.5 中的 $\{a_1, a_{33}, a_{86}\} \times \{b_6, b_{12}, b_{36}, b_{99}\}$ 便是一個具有固定常數值的雙群集。
- 如果每一列都是固定的常數值，但是不同的行可以有不同的數值，這樣的雙群集也是讓人感興趣的。**列上具有固定常數值的雙群集** (bicluster with constant values on rows) 是一個子矩陣 $I \times J$，滿足對任何 $i \in I$ 與 $j \in J$，我們有 $e_{ij} = c + \alpha_i$，其中 α_i 是列 i 的調整量。舉例來說，圖 10.6 顯示一個列上具有固定常數值的雙群集。

  對稱地，**行上具有固定常數值的雙群集** (bicluster with constant values on columns) 是一個子矩陣 $I \times J$，滿足對任何 $i \in I$ 與 $j \in J$，我們有 $e_{ij} = c + \beta_j$，其中 β_j 是列 j 的調整量。
- 如果列的改變同步於行的改變，而且反之亦然，這樣的雙群集也是很讓人感興趣的，以數學化的定義來說，**具有一致值的雙群集** (bicluster with coherent values) (也稱**樣式基礎的群集** (pattern-based cluster)) 是一個子矩陣 $I \times J$，滿足對任何 $i \in I$ 與 $j \in J$，我們有 $e_{ij} = c + \alpha_i + \beta_j$，其中 α_i 是列 i 的調整量，β_j 是列 j 的調整量。舉例來說，圖 10.7 顯示一個具有一致值的雙群集。

  我們可以看出，如果 $I \times J$ 是一個具有一致值的雙群集，若且唯若對任何 $i_1, i_2 \in I$ 與 $j_1, j_2 \in J$，我們有 $e_{i_1,j_1} - e_{i_2,j_1} = e_{i_1,j_2} - e_{i_2,j_2}$。此外，除了使用加法，我們可以定義使用乘法的具有一致值雙群集，也就是

10	10	10	10	10
20	20	20	20	20
50	50	50	50	50
0	0	0	0	0

圖 10.6 列上具有固定常數值的雙群集。

10	50	30	70	20
20	60	40	80	30
50	90	70	110	60
0	40	20	60	10

圖 10.7 具有一致值的雙群集。

$e_{ij} = c \cdot (\alpha_i \cdot \beta_j)$。明顯地，行上或列上具有固定常數值的雙群集，是具有一致值的雙群集的特例。

- 在某些應用中，我們可能只對沿著基因或條件方向往上或往下相關的變動感到興趣，而不用限制其精確數值，**列上具有一致值演變的雙群集** (bicluster with coherent evolutions on rows) 是一個子矩陣 $I \times J$，滿足對任何 $i_1, i_2 \in I$ 與 $j_1, j_2 \in J$，我們有 $(e_{i_1 j_1} - e_{i_2 j_1})(e_{i_1 j_2} - e_{i_2 j_2}) \geq 0$。舉例來說，圖 10.8 顯示一個列上具有一致值演變的雙群集，對稱地，我們也可以定義行上具有一致值演變的雙群集。

  接下來，我們研究如何探勘雙群集。

雙分群方法

之前指明的雙群集類型皆僅考慮理想的狀況，在現實資料集中，如此完美的雙群集是非常罕見的，當它們真的存在時，它們通常非常小型。並且，隨機誤差可能影響 e_{ij} 的讀數，而使得雙群集無法以完美的型狀出現。

有兩種主要的方式，能夠從具有雜訊的資料中發掘出雙群集。**基於最佳化** (optimization-based methods) 的方法執行疊代式的搜尋，在每一次疊

10	50	30	70	20
20	100	50	1000	30
50	100	90	120	80
0	80	20	100	10

圖 10.8 列上具有一致值演變的雙群集。

代中，具有最高顯著分數的子矩陣被判別為雙群集，此程序持續進行，直到符合使用者指定的條件才終止。基於計算成本的考量，通常會採用貪婪式搜尋策略來找出區域最佳的雙群集。**枚舉方法** (enumeration methods) 使用容忍門檻值來指定在探勘雙群集時允許的雜訊程度，並試圖枚舉雙群集中所有符合此要求的子矩陣。我們使用 δ-群集與 MaPle 演算法做為闡述這些概念的範例。

使用 δ-群集演算法來最佳化

對於子矩陣 $I \times J$ ，它第 i 列的平均值為

$$e_{iJ} = \frac{1}{|J|}\sum_{j \in J} e_{ij} \tag{10.16}$$

對稱地，它第 j 行的平均值為

$$e_{Ij} = \frac{1}{|I|}\sum_{i \in I} e_{ij} \tag{10.17}$$

子矩陣中所有元素的平均值為

$$e_{IJ} = \frac{1}{|I||J|}\sum_{i \in I, j \in J} e_{ij} = \frac{1}{|I|}\sum_{i \in I,} e_{iJ} = \frac{1}{|J|}\sum_{j \in J} e_{Ij} \tag{10.18}$$

作為雙群集的子矩陣的品質可以由均方殘差來量測

$$H(I \times J) = \frac{1}{|I||J|}\sum_{i \in I, j \in J} (e_{ij} - e_{Ij} - e_{iJ} + e_{IJ})^2 \tag{10.19}$$

子矩陣 $I \times J$ 是一個 δ-群集 (δ-bicluster)，若它滿足 $H(I \times J) \le \delta$ ，其中 $\delta \ge 0$ 為一個門檻值。當 $\delta = 0$ ，子矩陣 $I \times J$ 是一個具有一致值的完美雙群集，藉由設定 $\delta > 0$ ，使用者可以指定相對於完美雙群集中元素的平均誤差的容忍值，因為在公式 (10.19) 中，每一個元素的殘差為

$$residue(e_{ij}) = e_{ij} - e_{Ij} - e_{iJ} + e_{IJ} \tag{10.20}$$

一個最大 δ-群集 (maximal δ-cluster) 是一個 δ-群集 $I \times J$ ，使得不存在另一個 δ-群集 $I' \times J'$ ，其中 $I \subseteq I'$ 與 $J \subseteq J'$ ，而且至少一個不等式成立。找

出具有最大尺寸的最大 δ-群集的計算成本是龐大的，因此我們使用啟發式貪婪搜尋方法來得到區域最佳的群集，此演算法執行分為兩個階段。

- 在刪除階段，我們從整個矩陣開始，當此矩陣的均方殘差值超過 δ 時，我們疊代地移除直行與橫列，在每一次疊代，我們計算每一列 i 的均方殘差為

$$d(i)=\frac{1}{|J|}\sum_{j\in J}(e_{ij}-e_{Ij}-e_{iJ}+e_{IJ})^2 \qquad \textbf{(10.21)}$$

  此外，我們計算每一行 j 的均方殘差為

$$d(j)=\frac{1}{|I|}\sum_{i\in I}(e_{ij}-e_{Ij}-e_{iJ}+e_{IJ})^2 \qquad \textbf{(10.22)}$$

  我們移除具有最大均方殘差的行或列，在此階段結束時，我們得到子矩陣 $I\times J$，它是 δ-群集。然而，此子矩陣可能不是最大 δ-群集。
- 在增加階段，我們疊代地擴充在刪除階段得到的 δ-群集 $I\times J$，只要它能滿足 δ-群集的要求，在每一次疊代，我們考慮沒有涉及在目前 δ-群集 $I\times J$ 的行與列，藉由計算這些行與列的均方殘差，擁有最小均方殘差的行或列被增加入目前的 δ-群集。

此貪婪式演算法僅能找到一個 δ-群集，要找出沒有重疊的數個雙群集，我們可以執行此演算法多次，在每一次執行結束輸出 δ-群集時，我們將在輸出雙群集中的元素用隨機數值取代。雖然此貪婪式演算法可能既找不到最佳的雙群集，也可能找不到所有的雙群集，它的執行是相當快速的，即便在大型矩陣上亦然。

使用 MaPle 來枚舉所有的雙群集

如前所述，子矩陣 $I\times J$ 是具有一致值的雙群集，若且唯若對任何 $i_1,i_2\in I$ 與 $j_1,j_2\in J$，我們有 $e_{i_1j_1}-e_{i_2j_1}=e_{i_1j_2}-e_{i_2j_2}$。對任何 2×2 子矩陣 $I\times J$，我們可以定義 p-分數為

$$p-score\begin{pmatrix}e_{i_1j_1} & e_{i_1j_2}\\ e_{i_2j_1} & e_{i_2j_2}\end{pmatrix}=|(e_{i_1j_1}-e_{i_2j_1})-(e_{i_1j_2}-e_{i_2j_2})| \qquad \textbf{(10.23)}$$

子矩陣 $I \times J$ 是一個 δ-p 群集 (樣式基礎群集)，如果 $I \times J$ 中所有的 2×2 子矩陣的 p- 分數皆至多為 δ，其中 $\delta \geq 0$ 是一個使用者指定的門檻值，指明以完美雙群集為標準，使用者對於雜訊的容忍值。此處， p- 分數控制雙群集中每一個元素的雜訊，而均方殘差捕捉平均雜訊值。

δ-p 群集有一個有趣的性質，如果 $I \times J$ 是一個 δ-p 群集，則 $I \times J$ 中所有的子矩陣 $x \times y (x, y \geq 2)$ 皆是 δ-p 群集，此單調性質可以幫助我們得到非冗餘 δ-p 群集的簡潔表示，此 δ-p 群集是最大的，如果沒有別的行或列能增加入此群集中，而同時能保持 δ-p 群集的性質。為了避免冗餘性，我們僅需要計算所有的最大 δ-p 群集，而不用找出所有的 δ-p 群集。

MaPle 是一個能枚舉出所有最大 δ-p 群集的演算法，它系統地使用集合枚舉樹與深先搜尋，來枚舉出每一個條件的組合，這種枚舉的框架與第 6 章介紹的頻繁樣式探勘中的樣式增長方法相同。考慮基因表達資料，對每一個條件組合 J，MaPle 演算法找出基因的最大子集合 I，使得 $I \times J$ 是一個 δ-p 群集。

可能有龐大數量的條件組合，MaPle 使用 δ-p 群集的單調性來刪除許多無效果的組合。對於一個條件組合 J，如果不存在一組基因集合 I 使得 $I \times J$ 是一個 δ-p 群集，則我們不用考慮 J 的任何超集合。此外，只有當 J 中每一個 $(|J|-1)$- 子集合 J'，$I \times J'$ 是一個 δ-p 群集，我們應該考慮 $I \times J'$ 是 δ-p 群集的一個候選者。MaPle 也會套用許多刪除技術來加速搜尋，同時保持回傳 δ-p 群集的完整性。舉例來說，當在檢測目前的 δ-p 群集 $I \times J$ 時，MaPle 蒐集所有可能添加來擴充群集的基因或條件，如果基因或條件與 I 和 J 在一起組成 δ-p 群集的子矩陣已經被發掘過，則 $I \times J$ 與任何 J 的超集合的搜尋都可以被修剪掉。

我們可以觀察到一個有趣的現象，MaPle 演算法在搜尋最大 δ-p 群集時，與探勘頻繁封閉項目集是有點類似的。因此 MaPle 演算法借用了頻繁樣式探勘中深先搜尋框架與樣式增長方法中修剪技術的概念，這是頻繁樣式探勘與群集分析共享相似概念與技術的例子。

MaPle 與其它枚舉雙群集式演算法的優點是，它們能確保找出雙群集的完整性，不會遺漏任何重疊的群集。然而，這些枚舉式演算法的挑戰在

於，當矩陣非常大時，例如顧客－產品矩陣有數十萬個顧客與數百萬項產品時，執行會非常的耗時。

10.2.4 維度精簡方法與譜分群

子空間分群方法試圖找出在原始資料空間中的子空間內的群集，在某些情況下，建構一個新空間，會比使用原始資料的子空間更有效，這便是使用維度精簡方式來對高維度資料分群的背後動機。

範例 10.14 在推導出的空間中分群

考慮圖 10.9 中三個群集的資料點，要在原始空間 $X \times Y$ 中任何的子空間下對那些資料點分群，是不可能的事情，因為那三個群集不論是投影至 x 軸或是 y 軸上，都會與其他的群集重疊。但是，如果我們轉而建立一個新的維度 $-(\sqrt{2}/2)x+(\sqrt{2}/2)y$（在圖中用虛線表示）會如何呢？藉由將那些資料點投影至新維度，這三個群集變得顯而易見的。

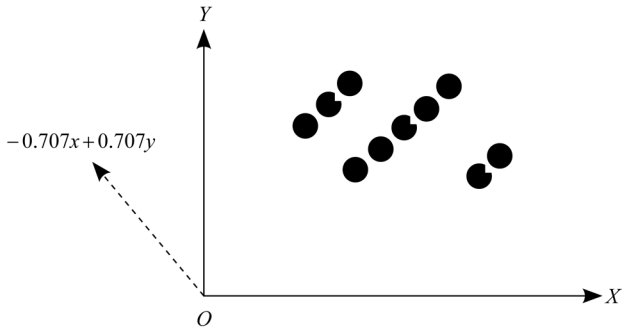

圖 10.9 在推導出的空間進行分群，可能效果會更好。

雖然範例 10.14 僅牽涉 2 個維度，然而建構新的空間（使得隱藏在資料中的群集結構變得清楚明顯）的概念可以延伸到高維度資料分群上，更理想的是，新建構的空間應當會有較少的維度。

有許多維度精簡的方法，最直接的做法是對資料套用第 3 章所介紹的特徵選取與特徵萃取方法，然而，那些方法可能不適用於偵測群集結構，因此，結合特徵萃取與分群的方法是更為吸引人的。在本章節中，我們介紹譜分群 (spectral clustering) 方法，它是一組能有效應用於高維度資料的方法。

圖 10.10 顯示譜分群方法的一般化框架，Ng-Jordan-Weiss 演算法是一種譜分群方法，讓我們仔細看此框架中的每一個步驟，在這麼做的同時，我們也會以 Ng-Jordan-Weiss 演算法為範例，並看到套用 Ng-Jordan-Weiss 演算法的特殊條件。

給定一組物件集合 $o_1,\ldots,o_n$，每一對物件之間的距離為 $dist(o_i,o_j)$，其中 $1\le i,j\le n$，而且期望的群集數目為 k，則譜分群方法執行步驟如下：

1. 使用距離量測，我們計算相似矩陣 (affinity matrix) W，使得

$$W_{ij}=e^{-\frac{dist(o_i,o_j)}{\sigma^2}}$$

其中 σ 是一個縮放參數，它控制著相似度 W_{ij} 隨著 $dist(o_i,o_j)$ 上升而下降的速度，在 Ng-Jordan-Weiss 演算法中，W_{ii} 設定為 0。

2. 使用相似矩陣 W，我們推導出 $A=f(W)$，導出的方法可能不同，在 Ng-Jordan-Weiss 演算法中，我們定義矩陣 D 為一個對角線矩陣，使得 D_{ii} 的值為 W 第 i 列的總和，也就是

$$D_{ii}=\sum_{j=1}^{n}W_{ij} \qquad \textbf{(10.24)}$$

接著，設置 A 為

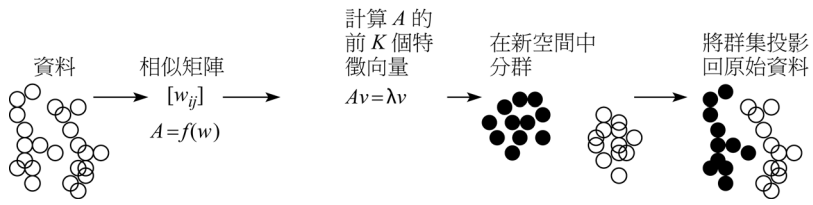

圖 10.10 譜分群方法的框架，來源：http://videolectures.net/micued08_azran_mcl/ 上的投影片 8。

$$A = D^{-\frac{1}{2}} W D^{-\frac{1}{2}} \quad \textbf{(10.25)}$$

3. 求出矩陣 A 中前 k 個特徵向量，回顧一下，方陣的特徵向量為非零向量，使得特徵向量與方陣相乘後所得的向量，與原特徵向量成比率。數學上來說，向量 v 是矩陣 A 的特徵向量，如果 $Av = \lambda v$，其中 λ 稱為對應的特徵值。此步驟從 A 推導出 k 個新維度 (以相似矩陣 W 為基礎)，通常，k 會遠小於原始資料的維度。

Ng-Jordan-Weiss 演算法計算 A 中具有最大特徵值的前 k 個特徵向量 $x_1, \ldots, x_k$ 。

4. 使用此 k 個特徵向量，將原始資料投影至由此前 k 個特徵向量所定義的空間，並執行諸如 k-means 分群演算法來找出 k 個群集。

Ng-Jordan-Weiss 演算法將 k 個最大特徵向量堆積成一個矩陣 $X = [x_1 x_2 \ldots x_k] \in \mathbb{R}^{n \times k}$，該演算法對矩陣 X 的每一列正規化成具有單位長度，形成矩陣 Y，也就是

$$Y_{ij} = \frac{X_{ij}}{\sqrt{\sum_{j=1}^{k} X_{ij}^2}} \quad \textbf{(10.26)}$$

此演算法接著將 Y 中每一列視為 k-維空間 $\mathbb{R}^k$ 中的一個點，並執行 k-means 分群演算法 (或其它能作分割式用途的分群演算法) 來將這些資料點分群成 k 個群集。

5. 根據轉換後的資料點是如何指派給步驟 4 所得到的群集，將原始資料點指派給群集。

在 Ng-Jordan-Weiss 演算法中，原始資料 o_i 被指派給第 j 個群集，若且唯若矩陣 Y 中的第 i 列在步驟四中被指派給第 j 個群集。

在譜分群方法中，新空間的維度數目被設置為期望的群集數目，這樣的設置是期望每一個新的維度應該能夠表露一個群集。

範例 10.15 Ng-Jordan-Weiss 演算法

考慮圖 10.11 中的資料點集合，它顯示資料集、相似矩陣、前 3 個最大的特徵向量與正規化後的向量，請注意使用這 3 個新的維度 (前 3 個最大特徵向量所組成)，這些群集可以輕鬆地被偵測出來。

譜分群方法在高維度應用中 (例如圖形處理) 是很有效的，理論上來說，在符合特定條件時，它可以執行的很好。然而，可擴充性是它的一大挑戰，計算大型矩陣的特徵向量是成本耗大的。譜分群方法可以與其他的分群演算法相結合，例如雙分群法。

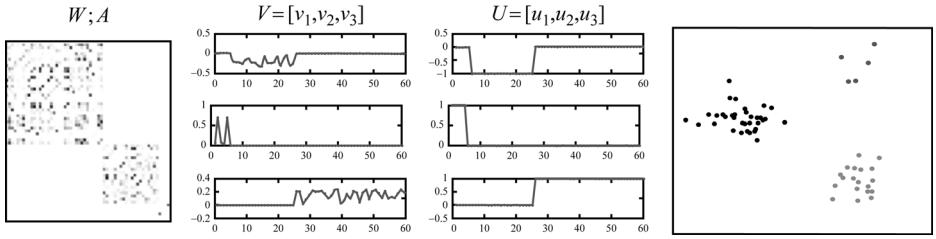

圖 10.11 Ng-Jordan-Weiss 演算法的新維度與分群結果。(資料來源：http://videolectures.net/micued08_azran_mcl/ 上的投影片 9)

10.3 圖形與網路資料分群

在圖形與網路資料上執行群集分析，能夠萃取出富有價值的知識與資訊，這類型的資料在許多應用中已日趨普遍，我們在 10.3.1 節探討圖形與網路資料分群的應用與挑戰，對於這類型分群的相似度量測將在 10.3.2 節介紹，我們將在 10.3.3 節學習圖形分群的方法。

一般來說，“圖形”與“網路”這兩個術語可以交換地使用，在往後的章節，我們主要使用“圖形”這個名詞。

10.3.1 應用與挑戰

做為 AllElectronics 公司的顧客管理經理，你注意到許多與顧客以及它們購買行為有關的資料，使用圖形來模組會更加理想。

範例 10.16 二部圖

顧客在 AllElectronics 公司的購買行為可以表示為二部圖 (亦稱偶圖)，在二部圖中，節點被分成兩個不相交的集合，使得每一條邊都連接集合中一個節點與另一個集合中的節點。對於 AllElectronics 公司的顧客購買資料，其中一組節點集合代表顧客，一個節點代表一位顧客。另一組節點集合代表商品，一個節點代表一項商品。一條邊連結一位顧客與一項商品，代表該商品被這位顧客所購買，圖 10.12 闡述此範例。

「對於顧客－商品的二部圖進行群集分析，我們可以獲得到哪些類型的知識呢？」通過對顧客分群，把購買相似商品的顧客放在同一個群組，AllElectronics 公司的顧客關係管理經理能藉此進行商品推薦。舉例來說，假設艾達所屬於的顧客群組，其中大部分的顧客都在最近 12 個月內購買了數位相機，然而艾達至今仍未購買數位相機，身為經理，你決定向她推薦數位相機。

相對地，我們可以對商品分群，使得被相似顧客購買的商品被群組在一起，此分群資訊也可以用做商品推薦。舉例來說，如果數位相機與快閃記憶卡屬於相同的商品群集，則當某位顧客購買了一個數位相機時，我們也可以向他推薦快閃記憶卡。

二部圖可廣泛使用於各項應用問題，考慮另一個範例。

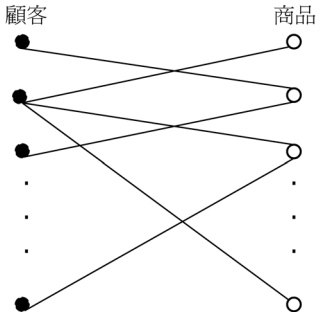

圖 10.12 代表顧客－商品的二部圖。

範例 10.17 網站搜尋引擎

在網站搜尋引擎中，搜尋日誌記錄使用者的查詢與相關的點擊資訊 (點擊資訊告訴我們，在搜尋引擎回傳的結果中，使用者點選了哪些網頁)。查詢與點擊資訊可以表達成二部圖，其中兩組節點集合分別對應到查詢與網頁，一條邊連結查詢與網頁，代表有使用者在該查詢結果中點擊了該網頁。我們能夠從查詢－網頁二部圖中獲得極具價值的資訊，例如，我們可以判別使用不同語言提出的查詢，但意指相同的事物，只要對這些查詢的點擊資訊是相似的。

另一個例子是，在網際網路中的所有網頁組成一個有向圖，也稱為 Web 圖，其中每一個節點代表一個網頁，而每一個超鏈結代表一條從來源網頁指向目標網頁的有向邊，在 Web 圖型上執行群集分析，可以揭露出網路社群、找出中心與權威網頁、並偵測出垃圾網頁。

除了二部圖外，群集分析也可套用到其他類型的圖形，包含一般化圖形，如下面範例 10.18 所示。

範例 10.18 社會網路

社會網路 (social network) 是一個社會的結構，它可用圖形表示，其中一個節點代表個人或組織，而邊代表節點之間的相互依賴，可表示朋友關係、共同興趣、或合作活動。AllElectronic 公司的顧客也可組成社會網路，其中每一個節點代表一位顧客，而兩個顧客之間有邊相連結，代表他們彼此認識。

做為顧客關係管理經理，你對於透過群集分析從 AllElectronic 公司的社會網路發掘出的有用資訊感到興趣，你從社會網路所得到的群集，其中在同一群集的顧客，代表她們彼此認識，或是他們有共同的朋友。在同一群集的顧客可能會去影響其他同群集顧客的購買決策制訂。此外，可以設計溝通管道來通知群集的“領頭”(即與群集中其他顧客連結最佳的人)，使得促銷訊息可以快速地傳播。因此，你可以使用顧客群集來提升 AllElectronic 公司的銷售量。

另一個例子是，作者的科學著作組成一個社會網路，其中一個節點代表一位作者，而兩個作者之間有邊相連結，如果他們有共同合作一篇科學著作。一般來說是，此網路圖是一個加權圖形 (weighted graph)，而邊上所賦與的權重，代表兩個作者之間合作的強度，例如共同發表的著作數目。對此共同作者網路執行群集分析，可以對於作者的研究社群與合作樣式提供洞悉與理解。

「對於圖形與網路資料執行群集分析，有什麼特殊的挑戰嗎？」迄今為止所介紹的絕大多數分群方法，資料物件都是由一組屬性的集合來表示。而圖形與網路資料的獨特特徵是，只有給出物件 (即，節點) 以及物件之間的關係 (即，邊)，沒有明確定義的維度或是屬性。要在圖形與網路資料上執行群集分析，有兩項主要的挑戰。

- 「我們如何量測圖形中兩個物件的相似性？」我們不可能使用傳統的距離度量方法，諸如歐基里德距離，我們必須轉而開發新的量測方式

來量化其相似度。這些量測方法通常不是尺度，因此為開發有效的分群方法提出嚴峻的挑戰，對於圖形資料的相似量測將在 10.3.2 節探討。

- 「我們如何設計對圖形與網路資料有效的分群模型與方法？」圖形與網路資料通常都是相當複雜，它所攜帶的拓樸結構比傳統分群應用更為精細複雜。許多圖形資料集合是相當大型，例如 web 圖形中包含至少數十億個網頁，而且圖形經常也可能非常稀疏，在平均情況下，一個節點只會與圖形中極少數的節點相連結。要發掘出深藏在此資料中正確與有用的知識，需要能顧及到這些因素的好的分群方法，對於圖形與網路資料分群的方法，將在課本 10.3.3 節介紹。

10.3.2 相似度量測

「我們該如何量測圖形中兩個節點之間的相似度或是距離呢？」在底下的探討中，我們審視兩種量測方式：測地距離 (geodesic distance) 與以隨機遊走為基礎的距離 (distance based on random walk)。

測地距離

量測圖形中兩個節點之間的距離，其中一種簡單的方式是根據兩節點之間的最短路徑，正式地說，圖形中兩節點之間的測地距離 (geodesic distance)，是她們之間最短路徑所通過的邊的數目，如果兩個節點沒有連通，則測地距離設定為無窮大。

使用測地距離，我們可以為圖形分析與分群定義許多種有用的量測方法，給定一個圖形 $G=(V,E)$，其中 V 是節點的集合，E 是邊的集合，我們可以有以下定義：

- 對於節點 $v \in V$，節點 v 的**偏心距離** (eccentricity) 標記為 $eccen(v)$，它是 v 與其他任意節點 $u \in V-\{v\}$ 之間最大的測地距離，節點 v 的偏心距離捕捉了節點 v 與圖中最遠節點之間的偏離程度。
- 圖形 G 的**半徑** (radius) 定義為所有節點的最小偏心距離，也就是說

$$r=\min_{v \in V} eccen(v) \tag{10.27}$$

此半徑捕捉了“最中心的節點”與“最遠邊界的節點”之間的距離。

- 圖形 G 的直徑定義為所有節點的最大偏心距離，也就是說

$$d = \max_{v \in V} eccen(v) \tag{10.28}$$

此直徑表示任意配對節點之間的最長距離。

- **圓周節點** (peripheral vertex) 是實現直徑的節點。

範例 10.19 以測地距離為基礎的量測

考慮圖 10.13 中所示的圖形 G，節點 a 的偏心距離為 2，亦即 $eccen(a) = 2$，而 $eccen(b) = 2$ 與 $eccen(c) = eccen(d) = eccen(e) = 3$，因此，圖形 G 的半徑為 2，直徑為 3，請注意，$d = 2 \times r$ 不是必要的，其中節點 c, d 與 e 是圓周節點。

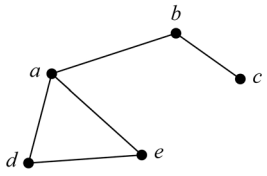

圖 10.13 圖形 G，其中節點 c, d 與 e 是圓周節點。

SimRank：基於隨機遊走與結構情境的相似度

對於某些應用問題，測地距離可能不適用於量測圖形中兩節點之間的相似度，此處，我們介紹 SimRank，它是以隨機遊走 (random walk) 與圖形中結構情境 (structure context) 為基礎的相似度量測。就數學的意義來說，隨機遊走是由相繼的隨機步伐所組成的軌跡。

範例 10.20 在社會網路中兩個人的相似度

讓我們考慮量測在範例 10.18 中 AllElectronics 公司的顧客社會網路中，兩個節點的相似度。此處，相似度可以解釋為兩者參與網路的接近

程度，也就是說，以社會網路所表示的關係角度來看，這兩個人有多麼接近。

「使用測地距離量測這樣的網路中的相似度與接近程度，效果如何呢？」假設艾達與鮑伯是網路中的兩個顧客，而且此網路是無向的，他們的測地距離（即艾達與鮑伯之間最短路徑的長度）是訊息從艾達傳遞至鮑伯的最短路徑，且反之亦然。然而，此訊息對於 AllElectronics 公司的顧客關係管理經理不是這麼有用的，因為，公司通常不會要求一個顧客傳遞特定訊息給另一位顧客。因此，測地距離不適用於此應用問題。

「在社會網路中的相似度所代表的意義為何？」讓我們考慮以下兩種方式來定義相似度。

- 兩個顧客被認為是彼此相似的，如果它們在社會網路中有相似的鄰居，此直覺的判斷是因為，在實際中，從許多共同朋友那邊得到推薦的兩個顧客，經常會做出相似的決定。這種類型的相似度是根據節點的區域結構（亦即，它的鄰居），因此也稱為以結構情境為基礎的相似度。
- 假設 AllElectronics 公司將促銷訊息同時傳送給社會網路中的艾達與鮑伯，艾達與鮑伯可能隨機將訊息傳遞給他們的朋友（亦即，網路中的鄰居），艾達與鮑伯的緊密度，可以根據其他顧客同時接受到原始傳遞給艾達與鮑伯的概似率來量測。此類型的相似度是根據在網路上隨機遊走的可達性，因此稱為以隨機遊走為基礎的相似度。

讓我們更進一步檢視以結構情境與隨機遊走為基礎的相似度。基於結構情境相似性背後直覺的概念是，圖形中兩個節點是相似的，如果它們與相似的節點相連結。要量測這類的相似度，我們需要定義個體鄰近區域 (individual neighborhood) 的概念。在一個有向圖 $G=(V,E)$ 中，其中 V 是節點的集合，$E \in V \times V$ 是邊的集合。對於任意節點 $v \in V$，節點 v 的個體入－鄰近區域 (individual in-neighborhood) 定義為

$$I(v) = \{u \mid (u,v) \in E\} \tag{10.29}$$

對稱地，節點 v 的個體出－鄰近區域 (individual out-neighborhood) 定義為

$$O(v) = \{w \mid (v,w) \in E\} \tag{10.30}$$

依循著範例 10.20 闡述的概念，我們定義對任意配對的節點之間的結構情境式相似度 SimRank，其值為 0 至 1 之間，對任意節點 $v \in V$，它與自身的相似度為 $s(v,v)=1$，因為它與自身的鄰居是一模一樣的，對任意節點 $u,v \in V$，使得 $u \neq v$，我們可以定義

$$s(u,v) = \frac{C}{|I(u)||I(v)|} \sum_{x \in I(u)} \sum_{y \in I(v)} s(x,y) \tag{10.31}$$

其中 C 是介於 0 與 1 之間的常數，一個節點可能沒有任何入－鄰近區域，因此如果 $I(v)$ 或 $I(u)$ 為 $\varnothing$，則我們定義公式 (10.31) 為 0，參數 C 指定了相似度隨著邊而傳遞時的衰減率。

「我們該如何計算 SimRank？」，最直覺的做法是疊代式地計算公式 (10.31)，直至到達固定點，令 $s_i(u,v)$ 為第 i 次疊代計算的 SimRank，在開始時，我們設定

$$s_0(u,v) = \begin{cases} 0 & if\ u \neq v \\ 1 & if\ u = v \end{cases} \tag{10.32}$$

我們使用公式 (10.31)，從 $s_i(u,v)$ 計算 $s_{i+1}(u,v)$

$$s_{i+1}(u,v) = \frac{C}{|I(u)||I(v)|} \sum_{x \in I(u)} \sum_{y \in I(v)} s_i(x,y) \tag{10.33}$$

我們可以證明 $\lim_{i \to \infty} s_i(u,v) = s(u,v)$。

現在，讓我們考慮以隨機遊走為基礎的相似度，一個有向圖形稱為是強連通 (strongly connected) 的，如果對任意兩個節點 u 與 v，均存在一條路徑從 u 走至 v，對一個強連通有向圖 $G=(V,E)$ 中，對任意兩節點 $u,v \in V$，我們可以定義從 u 至 v 的期望距離為

$$d(u,v) = \sum_{t:u \sim > v} P[t]l(t) \tag{10.34}$$

其中 u ~> v 為從 u 走至 v 的路徑，它可能包含循環，但直至結束才到達節點 v ，對於一條漫遊 $t = w_1 \to w_2 \to \ldots \to w_k$ ，其長度為 $l(t) = k - 1$ ，此漫遊的機率定義為

$$P[t] = \begin{cases} \prod_{i=1}^{k-1} \frac{1}{|O(w_i)|} & if\ l(t) > 0 \\ 0 & if\ l(t) = 0 \end{cases} \quad \textbf{(10.35)}$$

要量測節點 w 收到同時源於 u 與 v 傳送訊息的機率，我們將期望距離延伸至期望相遇距離 (expected meeting distance) 的概念，也就是說

$$m(u,v) = \sum_{t:(u,v)\sim>(x,x)} P[t]l(t) \quad \textbf{(10.36)}$$

其中 (u,v) ~> (x,x) 為一對長度相同的 u ~> x 與 v ~> x 的漫遊路徑，使用介於 0 與 1 之間的常數 C ，我們定義期望相遇機率 (expected meeting distance) 為

$$p(u,v) = \sum_{t:(u,v)\sim>(x,x)} P[t]C^{l(t)} \quad \textbf{(10.37)}$$

它即為以隨機遊走為基礎的相似度量測，此處，參數 C 指定在軌跡的每一步繼續遊走的機率。

已經證明出對任意節點 u 與 v ，我們有 $s(u,v) = p(u,v)$ ，也就是說，SimRank 是同時以結構情境與隨機遊走為基礎。

10.3.3 圖形分群的方法

讓我們考慮如何在圖形上執行分群，我們首先描述圖形分群背後的直覺概念，接著介紹兩種一般的圖形分群方法。

為了找出圖形中的群集，想像將圖形切割成若干片段，每一個片段為一個群集，使得同一群集內的節點有很好的相互連結，而不同群集內的節點則連結的較差。對於圖形 $G = (V, E)$ ，切割 $C = (S, T)$ 是圖形 G 中節點 V 的分割，也就是 $V = S \cup T$ 與 $S \cap T = \varnothing$ ，切割內的切割集 (cut set) 是一組邊的集合，使得 $\{(u,v) \in E \mid u \in S, v \in T\}$ ，切割的尺寸是切割集內邊的

數目。對於加權圖形，切割的尺寸是切割集內邊的權重的總和。

「哪些類型的切割能有效的推導出圖形中的群集呢？」在圖形理論與某些網路應用中，最小化切割是很重要的。一個切割是最小的，如果此切割的尺寸不大於其他切割的尺寸。有能夠在多項式時間內計算圖形中最小切割的演算法，我們可以使用這些演算法來進行圖形分群嗎？

範例 10.21 切割與群集

考慮圖 10.14 中的圖形 G，該圖形中有兩個群集：$\{a,b,c,d,e,f\}$ 與 $\{g,h,i,j,k\}$，以及一個離群點 l。

考慮切割 $C_1=(\{a,b,c,d,e,f,g,h,i,j,k\},\{l\})$，只有一個邊 (e,l) 橫跨 C_1 創建的兩個分割，所以，C_1 的切割集是 $\{(e,l)\}$，而 C_1 的尺寸為 1（注意，在連通圖形中，任意切割的尺寸不可能少於 1）。做為最小切割，C_1 並不能推導出好的群集，因為它只將離群點 l 與圖形其餘的節點區隔開來。

切割 $C_2=(\{a,b,c,d,e,f,l\},\{g,h,i,j,k\})$ 產生的分群比 C_1 好得多，C_2 內的切割集是那些連結圖形中兩個“自然群集”的邊。明確地說，對於切割集內的邊 (d,h) 與 (e,k)，絕大多數連結節點 d,h,e 或 k 的邊均屬於同一個群集。

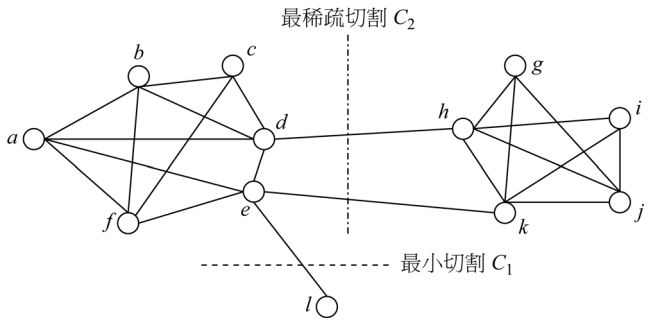

圖 10.14 圖形 G 與兩個切割。

範例 10.21 表明使用最小切割可能不會導致較佳的分群結果，我們所選擇的切割最好是要能夠，對每一個涉及在切割集內的邊上的節點 u，連接到節點 u 的邊絕大多數皆屬於同一個群集。正規地說，令 $deg(u)$ 為 u 的度數 (degree)，即連接到 u 的邊的數目，切割 $C=(S,T)$ 的稀疏性定義為

$$\Phi = \frac{\text{cut size}}{\min\{|S|,|T|\}} \tag{10.38}$$

一個切割是最稀疏的，如果它的稀疏性不大於任何其它切割的稀疏性，最稀疏的切割可能會有多個。

在範例 10.21 與圖 10.14 中，切割 C_2 是最稀疏的，使用稀疏性做為目標函數，最稀疏的切割嘗試最小化橫跨在分割之間的邊數，同時讓不同分割的尺寸能夠平衡。

考慮圖形 $G=(V,E)$ 中的分群，它將圖形分割成 k 個群集，分群的模組性 (modularity) 評估分群的品質，並且定義為

$$Q=\sum_{i=1}^{k}\left(\frac{l_i}{|E|}-\left(\frac{d_i}{2|E|}\right)^2\right) \tag{10.39}$$

其中 l_i 是第 i 個群集內節點之間的邊的數目，d_i 是第 i 個群集內節點的度數的總和，圖形中分群的模組性為落入群集的邊數佔全部邊數的比率，與圖形中節點隨機連接的邊數佔全部邊數的比率，兩者之間的差值，對所有群集的總合。圖形中最佳的分群能最大化模組性。

理論上，許多分群問題可以視為找出最佳的切割來對待，例如圖形中最稀疏的切割，然而，實際上，仍有許多挑戰存在。

- **高額的計算成本**：許多圖形切割問題是計算昂貴的，舉例來說，找出最稀疏切割的問題是 NP-hard，因此，在大型圖形上找出最佳解經常是不可能的，必須要在有效性 / 可擴充性與品質之間找出良好的折衷方案。
- **複雜的圖形**：圖形可能比此處討論的更為複雜，包含權重與 / 或循環。
- **高維度**：圖形可能有許多節點，在相似度矩陣中，一個節點代表一個向量 (表格中一個橫列)，其維度為圖形中節點的數目，因此，圖形分群方法必須能夠處理高維度資料。

■ **稀疏性**：大型圖形通常是稀疏的，代表平均而言，每一個節點僅與其中一小部分的節點相連接，在大型稀疏圖形中的相似矩陣也會是稀疏的。

有兩種類型的圖形分群方法可以處理這些挑戰，其中一類使用對高維度資料分群的方式，另一類是特別為圖形分群設計的。

第一種方法是基於一般的高維度資料分群方法，它們使用 10.3.2 節介紹的相似度量測，並從圖形中萃取出相似矩陣，便可套用一般的分群演算法來發掘出群集，通常會採用高維度資料分群演算法。舉例來說，在許多情況下，一但得到相似矩陣，可以套用譜分群方法 (10.2.4 節)，譜分群能夠逼近最佳圖形切割之解。

第二類方法是特別為圖形設計的，它們搜尋圖形以找出良好的連通成份 (connected component) 做為群集，讓我們以 SCAN (Structural Clustering Algorithm for Network) 演算法為範例。

給定一個結構化圖形 $G=(V,E)$，對任意節點 $u\in V$，u 的鄰近區域定義為 $\Gamma(u)$，其中 $\Gamma(u)=\{v\,|\,(u,v)\in E\,|\}\cup\{u\}$，使用結構情境的相似度概念，對任意的節點 $u,v\in E$ 配對，SCAN 藉由正規化它們鄰近區域的尺寸，來量測 u 與 v 之間的相似度，也就是說

$$\sigma(u,v)=\frac{|\Gamma(u)\cap\Gamma(v)|}{\sqrt{|\Gamma(u)||\Gamma(v)|}} \qquad \textbf{(10.40)}$$

公式 (10.40) 的值越大，代表此兩節點越相似，SCAN 使用相似度門檻值 ε 來定義群集的歸屬程度，對一個節點 $u\in V$，u 的 ε-鄰近區域定義為 $N_\varepsilon(u)=\{v\in\Gamma\,|\,\sigma(u,v)\geq\varepsilon\}$，$u$ 的 ε-鄰近區域包含 u 所有的鄰居，而且與 u 的結構情境相似度大於 ε。

在 SCAN 中，核心點 (core vertex) 是在群集內的一點，亦即，如果 $|N_\varepsilon(u)|\geq\mu$，則 $u\in V$ 是一個核心點，其中 μ 是一個樣本數門檻值。SCAN 從核心點來讓群集增長，如果節點 v 是在核心點 u 的 ε-鄰近區域內，則節點 u 與 v 指派給同一個群集。此群集增長的程序不斷執行，直至沒有群集增長為止，此方法與第 9 章介紹的 DBSCAN 類似。

如果節點 $v\in N_\varepsilon(u)$，則節點 v 是可從核心點 u 直接可到達的 (directly

reached)，而節點 v 是從核心點 u 可到達的，如果存在 $w_1, \ldots, w_n$，使得 w_1 是從 u 可到達的，w_i 是從 w_{i-1} 可到達的 $(1 < i \le n)$，而且 v 是從 w_n 可到達的。此外，對兩個節點 $u, v \in V$，不論它們是否為核心點，u 與 v 被稱為是相連的，如果存在一個核心點 w，使得 u 與 v 皆是從 w 可到達的。所有在同一個群集內的節點都是相連的，亦即，群集是最大的節點集合，使得該集合任意配對的節點都是相連的。

有些節點可能不屬於任何群集（例如節點 u），如果節點 u 的鄰近區域 $\Gamma(u)$ 包含來自超過一個群集的節點，則稱節點 u 為中心 (hub)。如果節點 u 不屬於任何群集，而且也不是中心，則節點 u 是離群值 (outlier)。

SCAN 的演算法顯示在圖 10.15 中，它的搜尋方式與 DBSCAN 找出群集的步驟雷同。SCAN 找出圖形的切割，使得每一個群集內的節點，根據結構情境的相似度，是彼此相連的。

SCAN 的優點是他的時間複雜度，與圖形的邊數是線性的關係，在大型與稀疏的圖形上，圖形的邊數與節點的數目是在同一量級上，因此，SCAN 可期望在大型圖形資料分群上，有很好的可擴充性。

演算法：針對圖形資料分群的 SCAN 演算法
輸入：圖形 $G = (V, E)$，相似度門檻值 ε，樣本數門檻值 μ
輸出：群集的集合
方法：設定 V 中所有節點為未標記的
(1) **For all** 未標記節點 u **do**
(2) **If** u 為核心點 **then**
(3) 產生新的群集 ID c
(4) 將所有 $v \in N_\varepsilon(u)$ 的節點插入至佇列 Q 中
(5) **While** $Q \neq \varnothing$ **do**
(6) $w \leftarrow Q$ 中第一個節點
(7) $R \leftarrow$ 可從 w 直接可到達的節點集合
(8) **For all** $s \in R$ **do**
(9) **If** s 不是“未標記”或是標記為“沒有群組” **then**

圖 10.15 針對圖形資料分群的 SCAN 演算法。

```
(10)                    將目前的群集 ID 指派給 s
(11)                Endif
(12)                If s 是“未標記” then
(13)                    將 s 插入佇列 Q 中
(14)                Endif
(15)            Endfor
(16)            將 w 從佇列 Q 中移除
(17)        Endwhile
(18)    Else
(19)        將 u 標記為“沒有群組”
(20)    Endif
(21) Endfor
(22) For all 標記為“沒有群組”的節點 u do
(23)    If ∃x, y ∈ Γ(u) ，x 與 y 有不同的群集 ID then
(24)        將 u 標記為“中心”
(25)    Else
(26)        將 u 標記為“離群值”
(27)    Endif
(28) Endfor
```

圖 10.15 （續）

10.4 限制式分群

使用者通常擁有他們想要整合入群集分析的背景知識，應用問題也可能會有一些特定的要求，這些資訊可以模組成群集的限制，我們使用兩個步驟來接近限制式分群的議題，我們在 10.4.1 節對圖形資料分群的限制式進行分類，10.4.2 節介紹使用限制式分群的方法。

10.4.1 限制式的類型

本節研討如何對群集分析所使用的限制式進行分類，明確地說，我們

可以根據限制式設定的主題，或是限制的約束程度而加以分類。

同第 9 章所介紹的，群集分析主要涉及三個方面：物件是群集的實例，群集是物件的群組，與物件之間的相似度。因此，我們第一個探討的方法是根據限制是套用在何者上，而將限制式予以分類。我們有三種類型：實例上的限制、群集上的限制與相似度量測的限制。

1. **實例上的限制**：對於實例的限制指明在群集分析中，一對或一組實例集合應該如何被群組，這類的限制有兩種常見的類型，包含：
   - **必須連結限制** (must-link constraint)：如果將“必須連結限制”套用在兩個物件 x 與 y 的上面，則 x 與 y 應當要被群組在同一個群集內，這些“必須連結限制”是遞移的，亦即，如果 must-link(x, y) 與 must-link(y, z)，則 must-link(x, z)。
   - **不能連結限制**(cannot-link constraint)：“不能連結限制”是“必須連結限制”的相反，如果將“不能連結限制”套用在兩個物件 x 與 y 的上面，則 x 與 y 應當要屬於不同的群集內，“不能連結限制”是可繼承的，亦即，如果 cannot-link(x, y)，而且 cannot-link(x, x') 與 cannot-link(y, y')，則 cannot-link(x', y')。

   實例上的限制可以對特定的實例來定義，此外，它也可使用實例變數或實例的屬性來定義，例如以下的限制

   $$\text{Constraint } (x, y)\text{: must-link } (x, y) \quad \text{如果} \quad \text{dist } (x, y) \leq \epsilon$$

   使用物件之間的距離來指定“必須連結限制”。
2. **群集上的限制**：群集上的限制指明對於群集的要求，有可能使用群集的屬性，舉例來說，限制式可能指明群集內物件的最少數量要求、群集的最大直徑要求、或是群集的形狀 (例如、凸形)。分割式分群方法中指定群集的數目，也可視為對於群集的限制。
3. **相似度量測的限制**：通常，相似度量測 (例如歐基里德距離) 可用來量測物件之間的相似程度，然而在某些應用當中，可能會有意外，相似度量測的限制指明在計算相似度時必須遵從的要求。舉例而言，要將市場中的人們當作移動物件而予以分群，可使用歐基里德距離當

作兩點之間步行的距離，但相似度量測的限制是，兩點之間最短距離的步行軌道不可以穿越牆壁。

表達限制的方法可能不只一種，依據它所屬得的類別，舉例來說，我們可指定群集上的限制如

$$\text{constraint}_1：\text{群集的直徑不可超過 } d$$

這項要求也可以由實例上的限制來表達，如

$$\text{constraint}_1'：\text{cannot-link}(x, y) \text{ 如果 } dist(x, y) > d \qquad \textbf{(10.41)}$$

範例 10.22 **實例、群集與相似度量測上的限制**

AllElectronics 公司對它們的顧客進行分群，使得每一群集的顧客指派給一位顧客關係管理經理負責，假設我們想要指明所有地址相同的顧客，均會放置在相同的群集內，這能為家庭提供更綜合性的服務，這項要求能用實例上的限制來表達

$$\text{constraint}_{家庭}(x, y)：\text{must-link}(x, y) \quad \text{如果} \quad x.\text{地址} = y.\text{地址}$$

AllElectronics 公司有 8 位顧客關係管理經理，為了確保每位經理有相似的工作負擔，我們套用群集上的限制，使得總共找出 8 個群集，而且每個群集包含的顧客數目在 10 ~ 15% 之間。我們可以使用開車距離來計算兩位顧客之間的空間距離，但如果兩位顧客是居住在不同的國家，我們可轉而使用飛行距離，這是在相似度量測上的限制。

另一種對分群限制的劃分，是考慮限制必須被遵守的穩固程度，如果違背此限制的分群是不被接受的，則此限制是**硬性** (hard) 的，如果當找不到更佳的解答時，違背此限制是不滿意，但是可以接受的，則此限制是**軟式** (soft) 的，軟式限制也稱為偏愛 (preference)。

範例 10.23 硬式與軟式限制

對 AllElectronic 公司而言，範例 10.22 中的 $\text{constraint}_{家庭}(x, y)$ 是硬式限制，因為將家族成員分割至不同的群集，將使得公司無法為家庭提供綜合性的服務，而導致顧客滿意度下降。對於群集數量的限制 (這對應到公司中有多少位顧客關係管理經理) 也是硬性的。範例 10.22 同時也對群集大小的平均性設定限制，雖然能夠滿足此限制是非常理想的，但公司有這份靈活彈性，它願意指派資深有能力的顧客關係經理來管理較大的群集，所以此限制是軟式的。

理想上，指定資料集與限制式的集合，所得到的分群皆能滿足這些限制。然而，資料集中有可能不存在能滿足所有限制的分群結果，顯而易見的，如果有兩個限制式是互相衝突的，則沒有分群能同時滿足這兩個限制式。

範例 10.24 限制式衝突

考慮這些限制

$$\begin{aligned} &\text{must-link}(x, y) \quad \text{如果} \quad dist(x, y) < 5 \\ &\text{cannot-link}(x, y) \quad \text{如果} \quad dist(x, y) > 3 \end{aligned}$$

如果資料集中有兩個物件 x 與 y，其 $dist(x, y) = 4$，則沒有分群結果可以同時滿足這兩個限制。考慮以下這兩個限制

$$\begin{aligned} &\text{must-link}(x, y) \quad \text{如果} \quad dist(x, y) < 5 \\ &\text{must-link}(x, y) \quad \text{如果} \quad dist(x, y) < 3 \end{aligned}$$

對第 1 條限制而言，第 2 條限制是冗餘的。此外，給定一資料集，其中任意兩個物件之間的距離至少為 5 的話，則所有可能的分群結果都滿足這些限制。

「我們如何量測這些限制式的品質與有用程度呢？」一般而言，我們考慮它們的資訊性與一致性，**資訊性** (informativeness) 是限制式所攜帶的資訊量，給定資料集 D，分群方法 A，與限制式集合 C，C 對 A 在 D 上的資訊性，是由使用分群法 A 在 D 上所計算出來的分群不滿足限制 C 的比率來度量的，資訊性越高，代表限制式攜帶越多的指定要求與背景知識。限制式的**一致性** (coherence) 代表限制式對自身之間的同意程度，這可透過量測限制式的冗餘度來得到。

10.4.2 使用限制式的分群方法

雖然我們對限制式的類型予以分類，然而應用問題可能有截然不同的限制式形態，因此，需要各種技術來處理這些特殊的限制式，在此章節，我們探討處理硬式與軟式限制的一般原理。

處理硬式限制式

處理硬式限制的一般策略是，在群集指派程序中嚴格遵守限制式的要求，為了闡明此概念，我們以分割式分群法為範例。

給定一個資料集與一組實例上的限制式 (亦即，必須連結與不能連結限制)，我們該如何延伸 *k*-means 方法來滿足這些限制呢？COP-*k*-means **演算法**執行如下。

1. **對於必須連結限制產生超實例** (superinstance)：對於必須連結限制計算其遞移閉包 (transitive closure)，此處，所有的 must-link 限制皆視為等價關係對待，此閉包給定一個或多個物件的子集合，在同一子集合的所有物件都必須指派到同一個群集中。為了表示這樣的子集合，我們將此子集合中所有的物件用其平均值取代，此超實例也有攜帶權重值，這是它所代表物件的數量。

   在此步驟之後，必須連結限制會必然滿足。
2. **執行修改式 *k*-means 分群演算法**：回顧一下，在 *k*-means 演算法中，資料物件被指派給最接近的群集中心，如果最接近的群集中心違反不能

連結限制，那該怎麼辦？為了遵守不能連結限制，我們修改 *k*-means 演算法的中心點指派程序為最近可行中心點指派，亦即，當物件依序指派給群集中心時，在每一個步驟，我們確保迄今為止限制沒有違反任何不能連結限制，資料物件被指派至最近的群集，而且該指派遵守所有不能連結限制。

由於 COP-*k*-means 演算法確保在每一步驟中沒有限制被違反，因此，它不需要回溯，它是貪婪的演算法，只要限制之間沒有衝突，它能夠產生滿足所有限制的分群。

處理軟式限制式

使用軟式限制的分群是一個最佳化問題，當分群結果違背了軟式限制，便對分群結果施加懲罰值。因此，分群的最佳化目標包含兩個部分：最佳化分群品質與最小化違背限制的懲罰值，整體的目標函數是結合分群品質分數與懲罰分數。

為了闡述此點，我們同樣以分割式分群為範例，給定一資料集與一組實例上的軟式限制式，CVQE (Constrained Vector Quantization Error) 演算法執行 *k*-means 演算法，同時賦予限制違背懲罰值，CVQE 演算法使用的目標函數，是 *k*-means 演算法所用的距離總和，並施加限制違背懲罰值來加以調整，其計算方式如下。

- 必須連結違反懲罰值：如果物件 x 與 y 上有“必須連結限制”，但是它們被指派到不同的群集 c_1 與 c_2，則此限制被違背了，結果便是，我們將群集 c_1 與 c_2 之間的距離 $dsit(c_1, c_2)$ 添加入目標函數中，做為懲罰值。
- 不能連結違反懲罰值：如果物件 x 與 y 上有“不能連結限制”，但是它們被指派到相同的群集 c，則此限制被違背了，結果便是，我們將群集 c 與 c' 之間的距離 $dist(c, c')$，添加入目標函數中，做為懲罰值。

加速限制式分群

諸如在相似度量測上的限制式，可能會導致分群時有沉重的計算負擔，考慮以下具有障礙物的分群問題：要將市集中的人群視為移動物件來分群，我們可用歐基里德距離來量測兩個點之間的行走距離，然而，在量測相似度時有一個限制，便是實現最短距離的行走軌跡不可以穿透牆壁，由於物件之間可能有障礙物，所以兩物件之間的距離可能要透過幾何計算來推導 (例如，涉及三角測量)，如果有大量的物件與障礙物涉及其中的話，其計算成本會非常的高。

具有障礙物的分群問題可以用圖形化概念來表達，首先，資料點 $\boldsymbol{p}$ 與 $\boldsymbol{q}$ 在區域 R 中是**可見的** (visible)，如果連結 $\boldsymbol{p}$ 與 $\boldsymbol{q}$ 的直線沒有和任何障礙物交集。圖形 $VG=(V,E)$ 是**可見圖形** (visible graph) 如果它能滿足障礙物中每一個頂點對應到 V 中的一個節點，而且 V 中任兩個節點 $\boldsymbol{v}_1$ 與 $\boldsymbol{v}_2$ 有 E 中的邊相連結，若且唯若它們所代表的頂點是彼此可見的。令 $VG'=(V',E')$ 是由 VG 擴充，它是透過在 V 中加入兩個節點 $\boldsymbol{p}$ 與 $\boldsymbol{q}$ 所建立的可見圖，E' 包含連結 V' 中兩個節點的邊，如果這兩個節點是彼此可見的。兩個節點 $\boldsymbol{p}$ 與 $\boldsymbol{q}$ 之間的最短路徑是 VG' 中的一個子路徑，如圖 10.16(a) 所示，我們看到從節點 $\boldsymbol{p}$ 開始，通過往 $\boldsymbol{v}_1$、$\boldsymbol{v}_2$ 或 $\boldsymbol{v}_3$ 的邊，透過 VG 中的一個路徑，然而經過 $\boldsymbol{v}_4$ 或 $\boldsymbol{v}_5$，最終到達節點 $\boldsymbol{q}$。

為了節省計算任意配對物件之間距離的成本，可使用數種預處理與最

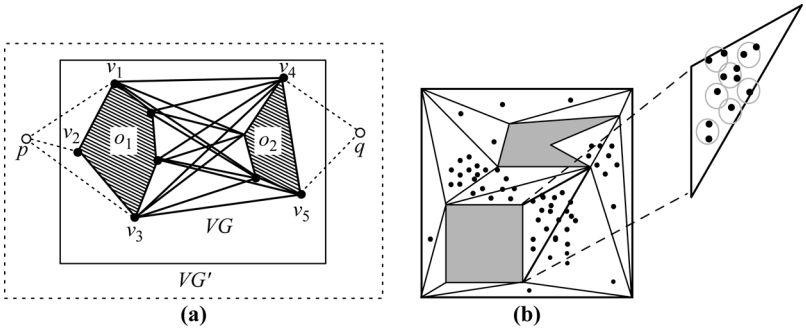

圖 10.16 含有障礙物 o_1 與 o_2 的分群。(a) 可見圖；(b) 含有微群集區域的三角形分割。

佳化技術，其中一種方法將彼此接近的點群組成微群集，它可首先將區域 R 劃分成數個三角形，然後使用諸如 BIRCH 與 DBSCAN 演算法，來將同一個三角形內相近的點群組成微群集，如同圖 10.16(b)。藉由對微群集做處理，而不是處理個別的資料點，能夠縮減整體的計算成本。在此之後，可以執行預先計算，根據最短路徑的計算，來建構兩種類型的連結索引：(1) VV 索引：針對任意配對的障礙物頂點 (2) MV 索引：針對任意配對的微群集與障礙物頂點，使用這些索引能幫忙更進一步最佳化整體效能。

透過這些預先計算與最佳化策略，任意配對的節點之間的距離 (在微群集的解析層級) 可以有效率地計算，所以，分群程序可以使用相似於 CLARANS 這類型有效率的 k-medoids 演算法，並能在大型資料集上達到良好的分群品質。

10.5 方格式分群演算法

目前所探討的分群演算法都是資料驅動 (data-driven) 的，它們將資料物件分割成數個集合，並且在空間上適應這些物件的分佈。相反地，**方格式分群演算法** (grid-based clustering method) 採用空間驅動 (space-driven) 的概念，不論輸入資料的分佈如何，它將空間切割成數個格子單元 (cell) 以進行群集分析。

方格式分群演算法使用多重解析度方格式資料結構 (multiresolution grid data structure)，它將物件空間量化成有限數目的格子單元 (cell)，並由這些格子單元來建構方格結構 (grid structure)，所有的分群操作都是在方格結構上執行。此方法最大的優點是它的執行速度十分快速，它的執行速度通常與資料物件的數目無關，而是僅與量化空間 (quantized space) 中每一個維度上格子單元的數目有關。

在本節，我們將介紹兩種典型的方格式分群演算法，STING 演算法 (10.5.1 節) 發掘方格內的統計資訊，CLIQUE 演算法 (10.5.2 節) 為方格式與密度式的子空間 (subspace) 分群演算法。

10.5.1 STING：依據統計訊息方格的分群法

STING (STatistical INformation Grid) 是一種方格式多重解析度分群技術，它將包含輸入資料的空間分割成數個長方形的格子，並且使用階層式與遞迴式的方式來分割空間，不同層級下的長方形格子對應到不同的解析度，這些格子形成一個階層的架構 (hierarchical Structure)，在上層的格子將會被再分割進而產生下一層級的格子。我們會計算並儲存每個方格 (grid cells) 的統計訊息，例如平均值、最大值最小值等，並將這些資訊視為方格的屬性，這些統計參數能有助於查詢與其他資料分析任務。

圖 10.17 顯示 STING 分群法的階層架構，透過下層格子內的參數，我們可以輕易地計算出上層格子內的參數，這些參數包含與屬性獨立 (attribute-independent) 的參數，例如個數 (count)；以及與屬性相依 (attribute-dependent) 的參數，例如平均值 (mean)、標準差 (std)、最大值 (max)、最小值 (min)，與方格內屬性值的分佈類型，例如常態 (normal) 分佈、均勻 (uniform) 分佈、冪次 (exponential) 分佈或 NONE (當分佈是未知時)。此處屬性是為分析所選取的量測，例如對房屋物件所用的價格屬性。當資料載入到資料庫時，最底層格子的參數，諸如平均值、標準差、最大值、最小值，可直接從資料計算，而分佈類型可由使用者指定 (如果分佈為事先已知)，或是透過假說檢定法來估測 (例如 χ^2 檢定)。高層格子的分佈類型可以由它下一層格子主要的分佈類型搭配門檻

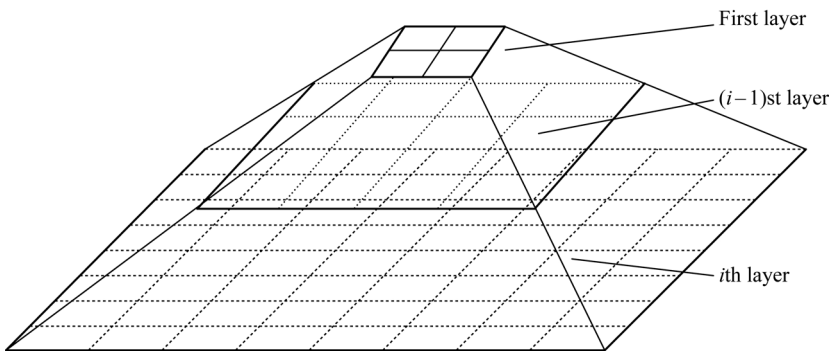

圖 10.17 STING 分群演算法所用的階層架構。

值過濾程序來得知，如果下一層格子的分佈類型彼此不一致，並且沒有通過門檻值測試，則把上一層格子的分佈類型設為 NONE。

「如何使用這些統計參數來查詢回覆？」這些統計參數可透過由上而下方格式的作法來進行查詢，首先我們從階層結構中的第一層開始，此層通常包含格子的數量最少，對目前層級下的所有方格，我們計算此方格對所查詢內容之相關性的信賴區間 (或估計機率範圍)，無關的格子便移除，不再考慮，僅對有關的格子往下一層繼續處理，此處理程序重複進行，直至達到最底層為止。在此時，如果符合查詢條件，滿足此查詢的格子將會回傳，否則，則取出相關格子內的資料，再做進一步處理，直到符合查詢條件為止。

STING 有一個有趣的性質，是當精細度 (granularity) 逼近零時 (亦即，在非常低的層級下)，它能逼近 DBSCAB 分群的結果，換句話說，使用格子內的物件數量與格子尺寸等資訊，STING 能夠辨識出緊密的群集，因此，STING 可視為密度式分群演算法。

「相較於其他分群演算法，STING 有什麼優點？」STING 有許多優點：(1) 方格式計算是與查詢內容無關的，因為格子內的統計資訊是格子內資料物件的資訊摘要，這與查詢無關的；(2) 方格式結構有助於平行處理與遞增式更新 (incremental updating)；(3) 本方法最重要的優點是它執行的效率，STING 存取資料庫一次來計算每一個方格的統計資訊，因此產生群集的時間複雜度為 $O(n)$，其中 n 是資料物件的數目。在建立好階層架構後，查詢的時間複雜度為 $O(g)$，其中 g 是最低層下的方格數目，通常 g 是遠小於 n 的。

由於 STING 使用多重解析度來進行群集分析，STING 分群的品質取決於最低層方格架構的精細度 (granularity)，如果精細度十分細緻，會造成處理成本的增加，然而，如果最底層格子的精細度太粗糙，它可能會降低群集分析的品質。除此之外，STING 並不考慮子方格與他周遭方格之間的空間關係，結果，它所找到的群集形狀為 isothetic，亦即所有群集的邊界均為水平或垂直的，沒有對角線邊界的存在，這可能會降低分群的品質與正確性，即便它處理程序十分快速。

10.5.2 CLIQUE：近似 Apriori 演算法的子空間分群法

資料物件通常包含數十個屬性，其中有許多屬性是不相關的，而且這些屬性值的數目是非常可觀的，這種種因素使得要找出能延伸在整個空間的群集是十分困難的，相反地，發掘出在不同子空間 (subspace)下的群集會更有意義。舉例來說，考慮一個健康資訊應用系統為病患記錄了許多屬性，包含描述個人資訊的眾多屬性、以及許多症狀、狀態與家族病史等等屬性。想要找出一個使得所有的屬性 (或絕大多數的屬性) 是強烈一致的重要群集是不可能的事情。舉例來說，對於禽流感的病患，他們的「年齡」、「性別」與「職業」等屬性，可能在廣泛的範圍下彼此不同，因此，要在整體空間下找出這樣的群集是不容易的。相反地，藉由搜尋子空間，我們可以在較低維度的空間找出彼此相似的病患群集，例如彼此病症相似，有高燒、咳嗽、無流鼻水與年齡在 3 至 16 歲的病患。

CLIQUE (Clustering In QUEst) 是一個簡單的方格式分群演算法，它能找出子空間中的密度式群集 (density-based cluster)，CLIQUE 將每一個維度分割成不重疊的區間，從而將整個空間的資料物件分割到格子單元內，它使用一個密度門檻值來區分格子是密集或稀疏的，如果格子內的物件數目超過密度門檻值，則他就是一個密集格子 (dense cell)。

CLIQUE 背後的主要策略，是使用密集格子對維度的單調性，來辨認出候選搜尋空間，它是以找出頻繁項目集與關聯規則的 Apriori 性質 (第 5 章) 為基礎，在子空間分群的應用上，單調性質代表的是：一個 k- 維度空間 ($k>1$) 的方格 c 包含至少 l 個物件，除非將方格 c 投影到每一個 $k-1$ 維度下所得到的方格 (也就是 $k-1$ 維度的方格) 裡面皆至少包含 l 個物件。考慮圖 10.18 為範例，包含資料物件的空間有 3 個維度：年齡、薪資與假期，在一個由年齡與薪資所組成的子空間下的 2D 方格，其中包含至少 l 個物件，除非當此方格投影至年齡與薪資維度上，所得到的方格內都至少包含 l 個物件。

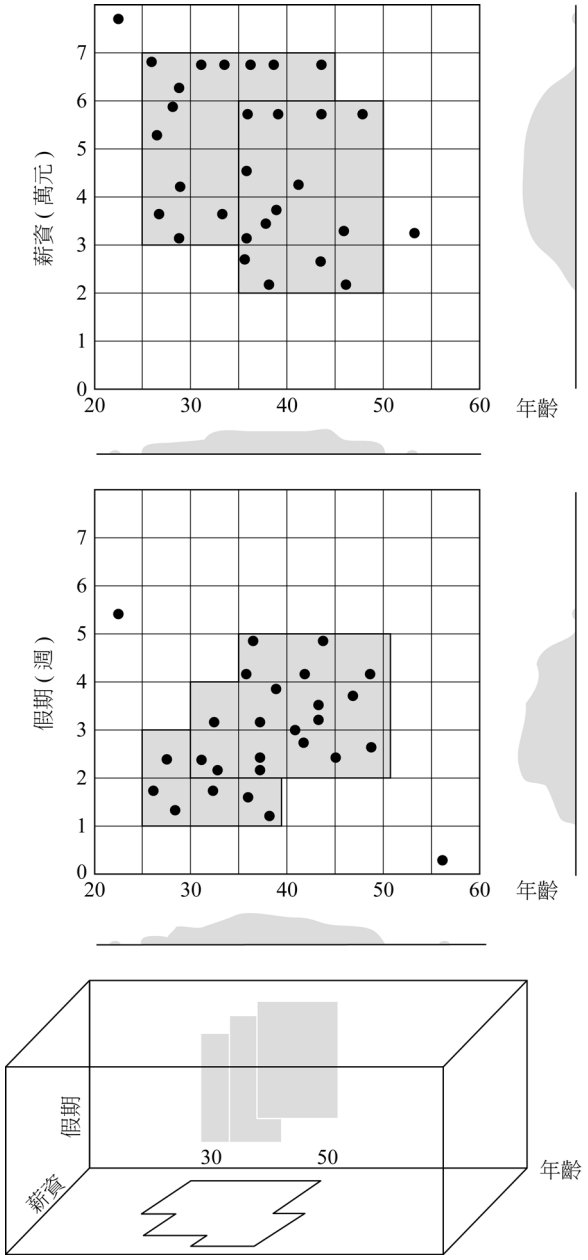

圖 10.18 將由維度年齡與薪資所構成的子空間下找到的密集區域，與年齡與假期的子空間下的密集區域進行結合，產生更高維度下搜尋密集區域的候選者。

CLIQUE 透過兩個主要步驟來執行群集分析，在第一步驟中，CLIQUE 將 d 維度空間分割成數過不重疊的矩形單元 (rectangular units)，並辨別出其中密集的格子。CLIQUE 要找出所有子空間中的密集格子，為了達成此

目的，CLIQUE 將每一個維度分割成數個區間 (interval)，並找出那些包含至少 l 個物件的區間，其中 l 是密度門檻值。CLIQUE 接著疊代地結合 (join) 兩個 k-維密集格子 c_1 與 c_2，其中 c_1 與 c_2 分別在子空間 $(D_{i_1},\ldots,D_{i_k})$ 與 $(D_{j_1},\ldots,D_{j_k})$ 中，如果 $D_{i_1}=D_{j_1},\ldots,D_{i_{k-1}}=D_{j_{k-1}}$，而且 c_1 與 c_2 在這些維度中共享相同的區間，則結合操作產生一個新的 $k+1$ 維子空間 $(D_{i_1},\ldots,D_{i_k},D_{j_k})$ 下的候選格子 c，CLIQUE 接著檢查侯選格子 c 中的物件數目是否超過密度門檻值，此疊代步驟到沒有候選格子可以產生，或沒有候選格子是密集時，便停止了。

在第二個步驟中，CLIQUE 使用所有子空間的密集格子召集成為群集，這些群集的形狀可以是任意的。它的概念是應用最小描述長度 (Minimum Description Length, MDL) 原則，來找出能夠覆蓋彼此連接的密集格子的最大區域 (maximal region)，此最大區域為一個超矩形 (hyperrectangle)，使得所有坐落在此區域中的格子皆是密集的，而且此區域不能再往任何維度下繼續擴展。找出最好的群集描述是 NP-hard 問題，因此，CLIQUE 使用一個貪婪式方法，它從任一個密集格子開始，找出覆蓋 (cover) 此格子的最大區域，並接著對那些尚未被覆蓋的密集格子進行上一步驟，此貪婪式法則持續進行，直到所有的格子都被覆蓋為止。

「CLIQUE 分群演算法的效能如何？」CLIQUE 自動地找出高維度空間下，具有高密度群集存在的子空間，它對輸入資料的順序並不敏感，而且不用擅自假定資料的分佈，它的執行成本與輸入資料數目呈線性關係，而且當維度增加時，它的擴充性 (scalability) 也很良好。然而，要得到有意義的群集，必須取決於適當地調整方格的尺寸與密度門檻值，這是非常困難的，因為這兩個數值對於所有不同維度組合的子空間上都會用到，因此，企圖簡化方法的代價是讓分群的正確性降低，而且，給定一個密集的區域 (dense region)，將它投影到較低子空間所得的區域也會是密集的，結果便是回傳的密集區域有大量重疊，更進一步，要找出在不同子空間上具有不同密度的群集也是很困難的一件事。

有許多延伸 CLIQUE 概念的方法被提出來，例如，我們可以將格子視為一堆容器 (bin) 的集合，我們可以不用在每一個維度上使用固定數量的

箱子，而是使用可適應式 (adaptive) 資料驅動 (data-driven) 的策略，根據資料在不同維度上的統計分佈來動態地決定箱子的數目。除此之外，我們也可以不要使用密度門檻值，而是使用熵 (entropy) 來量測子空間中群集的品質。

10.6 總結

- 在傳統群集分析中，一個物件是互斥地指派到一個群集，然而，在某些應用中，有可能需要使用模糊或機率方法，來將一個物件指派到多個群集中。**模糊分群** (fuzzy clustering) 與**機率模型式分群** (probabilistic model-based clustering) 允許一個物件屬於一個或多個群集，由一個**分割矩陣** (partition matrix) 記錄物件屬於多個群集的歸屬程度。
- **機率模型式分群**假設一個群集是一個參數式分佈，將要被分群的資料視為觀察到的樣本，我們可以估計出群集的參數。
- **混合模型** (mixture model) 假設觀察到的物件是來自多個機率群集的實例之混合，概念上來說，每一個觀察到的物件是獨立地產生，首先根據群集的機率，來選取一個機率式群集，接著根據所選取群集的機率密度函數，來選取一個樣本。
- **期望最大化** (expectation-maximization) 演算法是一種能逼近最大化概似率，或是最大化統計模型後驗參數估計的框架，期望最大化演算法可用來計算模糊分群與機率模型式分群。
- **高維度資料** (high-dimensional data) 對群集分析提出數個挑戰，包含如何模組化高維度群集，以及如何搜尋這樣的群集。
- 為高維度資料分群的方法有兩種主要的類型：子空間分群方法與維度精簡方法，**子空間分群方法** (subspace clustering method) 搜尋原始空間的子空間內的群集，範例包含**子空間搜尋方法**、**基於相互關係式的分群方法**、與**雙分群方法**。**維度精簡** (dimensionality reduction methods) 方法建構一個具有較低維度的空間，並且在那裏搜尋群集。

- **雙分群方法** (biclustering method) 同時對物件與屬性分群，雙群集的類型包含具有固定常數值的雙群集、列 (行) 上具有固定常數值的雙群集、列 (行) 上具有一致值的雙群集、列 (行) 上具有一致值演變的雙群集。雙分群方法的主要類型有兩種：**基於最佳化的方法**與**枚舉式方法**。
- **譜分群方法** (spectral clustering) 是**維度精簡方法**，其一般化概念是使用相似矩陣來建構新的維度。
- **圖形與網路分群** (clustering graph and network data) 有許多應用，例如社會網路分析，它所面對的挑戰包含：如何計算圖形中兩個物件之間的相似度量測，以及如何為圖形與網路資料設計分群模型與方法。
- **測地距離** (Geodesic distance) 是圖形中兩節點之間最短路徑所通過的邊的數目，它可用來量測相似度。此外，諸如社會網路等圖形中的相似度，可以使用結構情境或隨機遊走來量測，SimRank 是同時根據結構情境與隨機遊走的相似度量測。
- 圖形分群可由計算**圖形切割** (graph cut) 來模組化，**稀疏切割** (sparsest) 能導致好的分群結果，並可使用**模組性** (modularity) 來量測群集品質。
- SCAN 是一種圖形分群演算法，它搜尋圖形來判別出連結良好的連通成份做為群集。
- **限制式** (constraints) 可以用來表達群集分析的特定應用要求或是背景知識，分群的限制式可以劃分為：在實例上、在群集上、或是在相似度量測上的限制。在實例上的限制包含**必須連結** (must-link) 與**不能連結** (cannot-link) 限制，限制式可以是**硬式** (hard) 或是**軟式** (soft) 的。
- **硬式限制分群** (hard constraints for clustering) 能夠強制在群集指派程序中必須要嚴格遵守限制，**軟式限制分群** (clustering with soft constraints) 可考慮為最佳化問題，可使用啟發式策略來加速限制式分群。
- **方格式分群法** (grid-based method) 首先將物件空間量化成有限數目的格子單元來建構方格結構，接著在方格結構上執行分群操作。STING 是一個典型的方格式演算法，它根據儲存在方格上的統計資訊來進行分群，CLIQUE 是一個方格式與密度式的演算法。

10.1 傳統分群方法是僵硬的，它們要求每一個物件僅能互斥地屬於唯一一個群集，請解釋為何它是模糊分群的特例，你可使用 *k*-means 演算法為範例。

10.2 AllElectronics 公司銷售 1000 種產品 $p_1, \ldots, p_{1000}$，考慮顧客艾達、鮑伯、與凱絲，其中艾達與鮑伯購買 3 個相同的產品 p_1, p_2, p_3，對於其餘的 997 件產品，艾達與鮑伯獨立地隨機購買其中 7 件產品，而凱絲則是隨機地從 1000 件產品中購買了 10 項產品，以歐基里德距離而言，dist (艾達，鮑伯) > dist (艾達，凱絲) 的機率為何？如果使用 Jaccard 相似度 (第 2 章節) 又會如何？從這個範例你學習到什麼？

10.3 證明 $I \times J$ 是一個具有一致值的雙群集，若且唯若對任意 $i_1, i_2 \in I$ 與 $j_1, j_2 \in J$，我們有 $e_{i_1 j_1} - e_{i_2 j_1} = e_{i_1 j_2} - e_{i_2 j_2}$。

10.4 請比較 MaPle 演算法 (10.2.3 節) 與頻繁封閉項目集探勘中的 CLOSET 演算法，它們主要的相似處與差異處在哪裡？

10.5 SimRank 是為圖形與網路資料分群所使用的相似度量測，

(a) 請證明對於 SimRank 的計算，我們有 $\lim_{i \to \infty} s_i(u,v) = s(u,v)$。

(b) 請證明對於 SimRank，我們有 $s(u,v) = p(u,v)$。

10.6 在大型稀疏圖形中，平均而言，節點的度數是很低的，請問使用 SimRank 量測的相似矩陣依然是稀疏的嗎？如果是，意思為何？如果不是，又是為何原因？請解釋你的答案。

10.7 請比較 SCAN 演算法 (10.3.3 節) 與 DBSCAN (9.4.1) 演算法，它們的相似處與相異處為何？

10.8 考慮分割式分群與下述的群集上的限制：在每一個群集內的物件數量必須在 $n/k(1-\delta)$ 與 $n/k(1+\delta)$ 之間，其中 n 是全部物件的數量，k 是期望群集的數目，

而參數 δ 落在 $[0,1)$ 之內，你能否擴充 k-means 演算法來處理此限制？請探討此限制是硬式或軟式的情況。