和AI做朋友

相知篇

從0開始學AI

中小學人工智慧教育教學示範例系列 高中階段

李建樹・編 | 陳虹伶、陳瑞翎・執行編輯 | 黃仁暐・涂益郎・合著

序言

早在AlphaGo打敗圍棋棋王以前，「AI」已經逐漸進入我們生活周遭，例如手機上的語音助理、自駕車，甚至是以演算法預測顧客的喜好以提供更適切的商品，AlphaGo不過是這場AI大戲中較引人矚目的一個角色。

當AI從研究室走入產業界、走入你我生活，成為不可忽視的存在，教育部認為不該只著重高階研發人才的培育，也應該向下扎根。讓中小學生有機會體驗 AI、知道 AI 的應用與對自己未來及生活的影響；並針對AI 原理及技術有興趣的學生，提供進階學習的資源及管道。

基於這樣的理念，我們邀請計畫團隊依十二年國民基本教育課程綱要、教師執教經驗與中小學學生相關數理觀念建立情形，編纂《和AI做朋友》這套教材。編輯團隊依據學生回饋修正教材，也將這些內容編輯成教案，做為其他教師後續實施參考的依據。

《和AI做朋友》是教育部為有興趣學習與教授AI的學生及教師鋪墊的第一步，是想一窺AI奧秘的師生們的參考教材。後續將陸續發展不同主題的課程內容，以及相對應的數位課程、實作課程與融入式課程等等，以提供師生們更多元的學習需求。

謹向過去一年辛苦投入編纂《和AI做朋友》的所有教師與助理團隊致敬及致謝，相信我們的努力終將能提升臺灣回應AI時代挑戰的競爭力。

教育部部長

謹誌

中華民國108年8月

編者的話

為推廣人工智慧教育並於中小學奠基，培育學生成為未來引領AI世代的科技人才，教育部籌組學者專家團隊，執行「人工智慧技術應用與人材培育計畫—中小學分項：中小學推廣教育計畫」(以下稱為本計畫)，發展中小學人工智慧教材教案示範例。

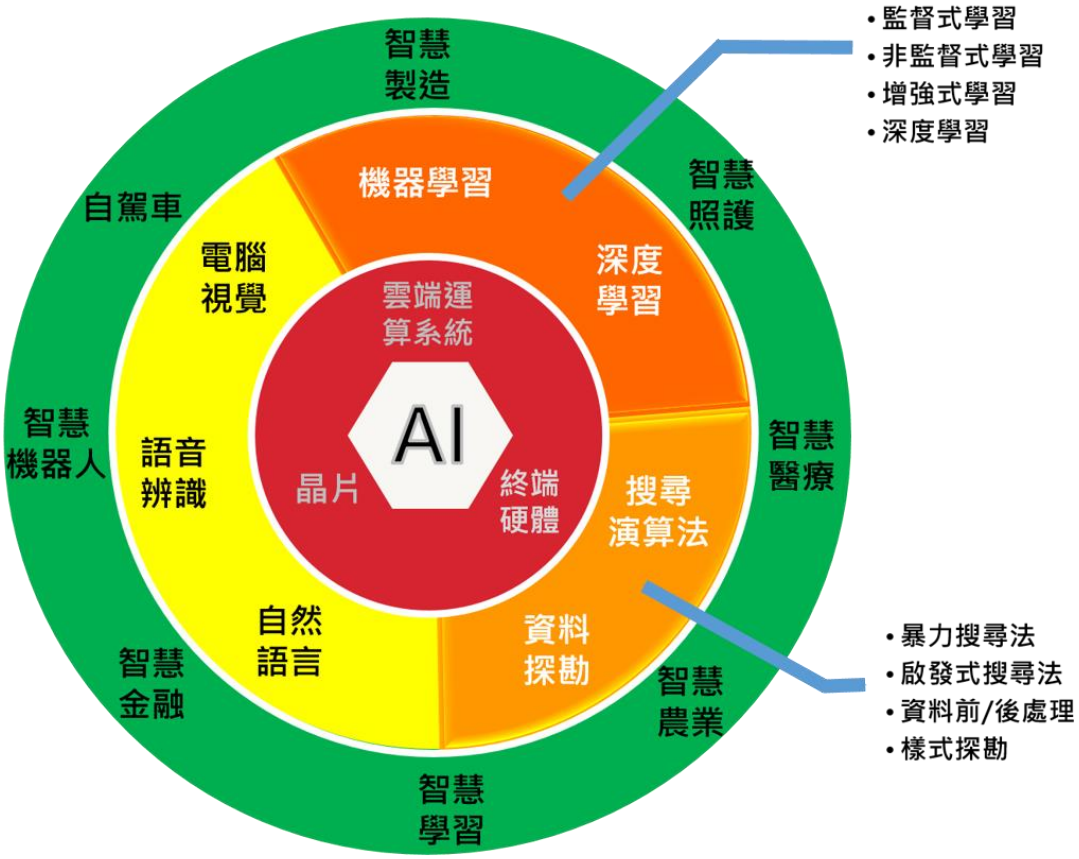
編輯團隊

為中小學設計適合的人工智慧教材，需要考量各年段學生的數理知識程度以及所要傳遞的人工智慧內涵，以適當規劃教材所要涵蓋內容的深度及廣度。計畫辦公室邀請相關領域的大學教授提供人工智慧的知識性內涵，並邀請教育現場第一線的教師，將上述人工智慧的內涵轉化為適合中小學學生及能配合十二年國民教育基本課程綱要的教材及教案。教材教案編輯團隊成員包括來自國立臺南大學、中華大學、國立成功大學、國立臺灣科技大學和南臺科技大學等大學教授，以及桃園市笨港國小、國立臺灣師範大學附屬高級中學國中部、國立臺南第二高級中學等現場教師共同合作，產出國小、國中、高中三階段的人工智慧教材教案。

為什麼是示範例？

人工智慧領域涵蓋的範疇甚廣，依據十二年國教課程綱要，資訊科技科目列為國、高中必修科目，其授課時數在國中階段每週1節、高中階段為2學分，高中可以選修方式列入課程，國中可採融入課程方式，國小則可於彈性課程實施或採融入方式進行教學，但以如此有限的時數要教授如此廣的內容確有其難度。團隊經過不斷思索研議後，決定先挑選具代表性的機器學習主題發展教材教案，若第一線教師有意願將此教材引入教學中，並在使用後覺得設計理念符合教學所需，願意投入後續發展，本計畫將逐步擴大教材發展團隊。這是「示範例」的第一個概念。

這波全球人工智慧熱潮大爆發起因於，Google的人工智慧圍棋軟體AlphaGo打敗世界棋王的新聞事件，進而讓AlphaGo背後的「深度學習」技術為世人所關注，並成為人工智慧的代名詞。事實上深度學習是機器學習的一個分支，機器學習歷經多年發展，體系相對完整，因此本計畫先以機器學習及深度學習兩主題發展教材教案，並預計在下一階段中，發展搜尋演算法及資料探勘等主題。這是「示範例」的第二個概念。



我們選擇的主題

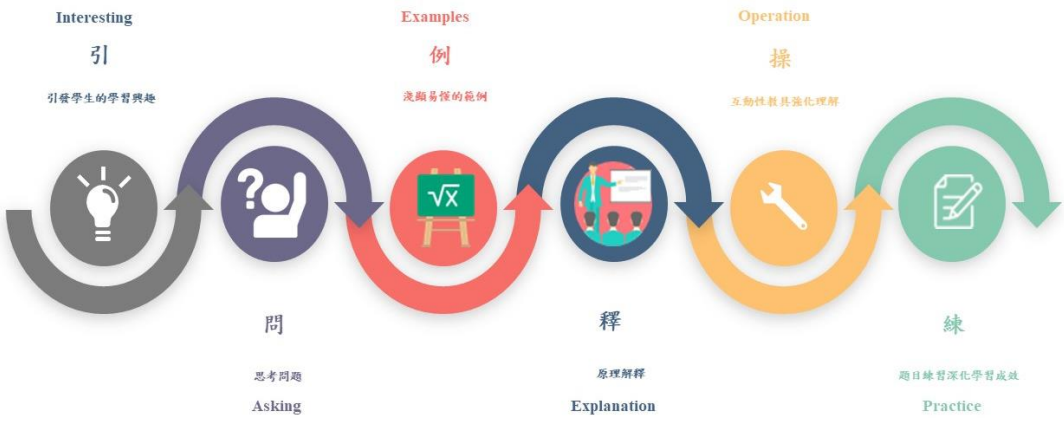
因人工智慧範疇廣袤且分支眾多，團隊依據運算資源及硬體設備、核心技術面、應用面等三個面向大致劃分出人工智慧現有範圍，以支持人工智慧運作最基本需求的運算資源及硬體設備為基底，向外延伸出機器學習、深度學習……等核

心技術面，而最外圍則是依據上述基礎所發展至生活中的各項應用。

本團隊考量高中學生已經具有足夠的數理基礎，所以著重介紹四種機器學習類型，包含：監督式學習、非監督式學習、增強式學習與深度學習。至於國中和國小教材，考量到學生的數理程度，著重於引起學生學習興趣並介紹人工智慧於生活中的應用，因此，除了簡介人工智慧及提供背景知識之外，主要課程內容是機器學習領域中最具代表性的監督式學習及非監督式學習介紹。

編輯理念說明

本計畫利用引、問、例、釋、操、練的順序設計教材，先導入多媒體來「引」發學生的學習興趣，再以提「問」的方式引導學生思考，並透過範「例」的講解，讓學生歸納重點後進行原理解「釋」，藉由互動性教具的「操」作強化理解，最後提供「練」習題目以深化學生的學習成效。



除了發展核心知識教材以外，我們也發展實作教材，提供高中學生開發人工智慧應用程式的機會，讓高中學生能透過實作對人工智慧有更深一層的體驗。實作教材使用Jupyter Notebook為執行環境，可以直接在Google Colab平台操作；學習完成實作教材的學生，若想藉由製作更大型人工智慧專案，以求更深入鑽研人工智慧技術，可透過指導老師申請使用教育部建置的AI虛擬雲端平台。另外，為提升國中及國小學生的學習興趣與成效，國中及國小階段教材有搭配實體教具，

讓學生能以動手操作的方式體驗機器學習的運作流程。此外，本教材也提供目前在中小學使用率較高的程式設計語言Scratch範例，讓學生進行操作演練。

給教師的話

對人工智慧感興趣的教師夥伴們，想將人工智慧融入教學中卻苦於始終找不到著力點嗎？本團隊除了開發高中、國中、國小各階段的教材之外，也依據教材開發出相應的教案，提供給對人工智慧教育感興趣的教師使用。所有階段的教材教案皆經過本團隊於現場實施教學後，檢討回饋再加以修正。依據團隊實際實施教學經驗，若是教師想要於現場進行教學，提供以下建議：高中階段教材所編撰的內容較深入，且教材的內容單元有搭配相應的上機實作教材，可採列入多元選修課程實施教學；國中階段教材可於彈性學習課中實施教學，並建議於國一或國二開設，若是上課時數不足，可以挑選數個活動任務，融入各領域課程之中，以符合跨領域（科技領域）精神，也可以開設一個學分的社團活動課程，挑選其中18小時進行一學期的AI人工智慧體驗課程進行教學；國小則可以採跨領域融入的方式進行教學，或是於彈性課程中實施，若是上課時數不足，則可以挑選適合的章節授課，以符合實際需求。

給學生及家長的話

對人工智慧有興趣的家長及同學，歡迎你們即將共同參與台灣教育史上嶄新的一頁。團隊所開發的教材包括：高中、國中、國小三個階段，考量現實情況，各階段學生大多是第一次接觸人工智慧，所以這三個階段的教材不具連續性，只有內容深淺的差異，可以獨立使用，且皆是由人工智慧簡介開始，由淺入深介紹人工智慧的核心知識及應用層面，讓各階段從未學習過相關知識的學生，能對人工智慧有基本認識。從未接觸過人工智慧領域的學生，若是有興趣想要學習此教材內容，有以下建議提供參考：若是高中學生，建議學習過高中一年級的數學課程，因高中階段教材需要用到向量、變異數等數學概念，若是有程式設計基礎者，

則在完成基本實作課程後，可至教育部AI雲端平台進行專案製作；若是國中學生，則建議國一以上，具備判別資料特徵的能力，以及最短距離的概念；若是國小學生，則以高年級較為合適，建議具備數數及簡單統計的概念，後面章節需要利用到表格判讀數據，及直尺作圖的能力。

希望這份教材可以成為同學認識人工智慧的啟蒙，並期待每位學習過此份教材的同學，能成為未來AI世代的人才。

結語

希望此份教材能對想將人工智慧概念帶入教學的老師有所幫助，並對將人工智慧教育往下扎根於中小學有所貢獻。由於教材設計牽涉的層面廣袤且相當繁複，難保不會有謬誤之處，若各位先進及對人工智慧教育具熱忱之教師及學子，發現有任何謬誤疏漏之處，祈請不吝指正，讓本教材能日趨完備。最後要感謝在此教材教案不同發展階段的審查委員們，由於他們所提供的寶貴專業意見，讓此教材教案得以更臻完備。

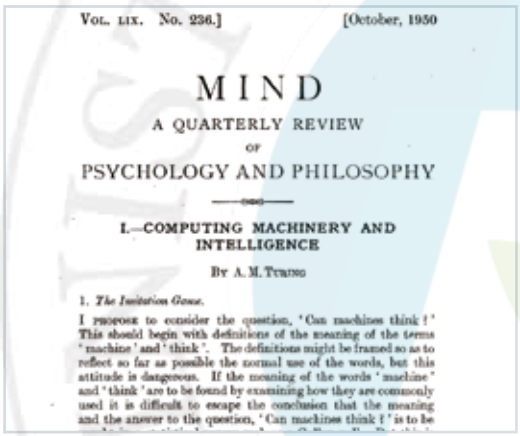
國立臺南大學資訊工程學系教授

中華民國108年8月

第一章

人工智慧簡介 Introduction

電腦科學與人工智慧之父圖靈在1950年提出的論文中，
提問「機器會思考嗎？Can Machines Think?」，
時至70年後的今日，人工智慧的發展真如圖靈大師的想像？
又或者那只是科學家窮極一生想要追尋的目標？
就讓我們一起踏上這趟探索AI的旅程吧！



Vol. LIX. No. 236.] [October, 1950

MIND

A QUARTERLY REVIEW

OF

PSYCHOLOGY AND PHILOSOPHY

I.—COMPUTING MACHINERY AND INTELLIGENCE

By A. M. Turing

1. *The Imitation Game.*

I propose to consider the question, 'Can machines think?' This should begin with definitions of the meaning of the terms 'machine' and 'think'. The definitions might be framed so as to reflect so far as possible the normal use of the words, but this attitude is dangerous. If the meaning of the words 'machine' and 'think' are to be found by examining how they are commonly used it is difficult to escape the conclusion that the meaning and the answer to the question, 'Can machines think?' is to be

圖1-1 圖靈發表之論文

1-1 思考如何避免思考

「人類一思考，上帝就發笑（Man thinks; God laughs.）」，關於思考，文學家米蘭・昆德拉（Milan Kundera）如是說。

而在數位時代有關於機器思考，一位數學家及邏輯學家的艾倫•圖靈（A. M. Turing）於1950年於MIND期刊發表了一篇在人工智慧領域深具里程碑意義的論文《Computing machinery and intelligence》。

這篇論文旨在探討「機器能否思考」，同時也為人工智慧的發展提供重要的思考方向，他的成就也獲世人的推崇，做為電腦科學與人工智慧之父，當之無愧！

圖1-2 人工智慧之父

1-2 人工智慧能聰明到什麼程度

在提出「機器能否思考」的同時，圖靈進一步定義了何謂「能思考」並提出了一套評估「能否思考」的方法：圖靈測試（Turing Test）。

該測試如下：假設有一個測試人員透過打字與密室裡的一臺電腦及一個人對話。如果測試人員分辨不出與他對話的是人或機器時，就表示與他對話的電腦通過了圖靈測試，代表了這臺電腦是具有思考的智慧能力。

在《Computing machinery and intelligence》一文中，圖靈預測：「我相信在50年內，人們能夠編出這樣一個程式，在經過五分鐘的詢問後，30%的詢問者都能覺得回答問題的是真人，而非一台機器」。

然而時至今日，仍然沒有一套電腦系統可以完全通過圖靈測試。這個事實顯露了圖靈的預估是過分樂觀的。

那麼時至今日，圖靈的評估方法是否仍適用於今日的人工智慧？有些研究者提出了質疑，例如MIT的電腦認知學教授Joshua Tenenbaum認為，這場測試完全沒什麼意義，只是針對特定標準去完成一個聊天程式；而紐約大學的Gary Marcus 教授則提出一現代版的「圖靈測試」：讓人工智慧系統觀看一段影片，再就影片中的內容對它進行詢問，如果它讓所有人都覺得是真人，便算通過。

1-3 人工智慧的繁花盛開－人工智慧廣為應用於生活上的各領域

「人工智慧（Artificial Intelligence,AI）」一詞，正式出現於1956年的達特茅斯夏季人工智慧研究計畫會議（Dartmouth Summer Research Project on Artificial Intelligence）中。近年來，人工智慧則漸被廣為應用在生活及各產業的產品上。例如瀏覽網站時（例如Facebook及Line網站）自動出現的推薦商品、你電腦裡安裝的防毒軟體…，其背後都有人工智慧的系統運作。人工智慧的效能甚至遠遠高於人類，例如：

1) 影像辨識：

Facebook 將人工智慧用來將使用者自動標記在照片中的「人臉辨識」、停車場或收費站的「車牌辨識」、「自動駕駛系統車」、無人商店的自助結帳，或是醫學影像診斷分析系統。

圖1-3 人臉辨識

圖1-4 自動駕駛系統車

2) 自然語言處理：

自然語言處理是讓電腦能理解人類語言，例如藉由分析詞意以及詞句間的關係，可以讓我們在網路搜尋（例如：使用Google搜尋）時找到與搜尋的字詞（關鍵詞）最相關的資料，又如機器翻譯、詐騙郵件偵測、Google的搜尋建議更正，抑或是那些用來分析社群媒體、娛樂產業、觀察網路網民活動的輿情分析系統等。

圖1-5 自然語言處理

3) 語音識別：

語音辨識技術為我們帶來最大的方便即是省去動手的麻煩，典型的應用就如我們常用手機語音助理（例如：Siri語音助理）所提供的語音撥號、語音輸入、語音問答等功能，又如家用電視機的語音操控介面，而Google於2018年5月在Google I/O大會首次向世人展示的Google Duplex則更進一步能透過語音助理撥電話與餐廳人員對話來完成訂位工作。

圖1-6 **語音辨識系統**

4) 醫療照護：

人工智慧在醫療照護方面的應用從基本的協助醫生進行醫療數據分析、疾病診斷，到病患生理數據即時監控、一般疾病諮詢，延伸至家庭或是老人照護陪伴等。

圖1-7 **照護型機器人**

想一想

人工智慧自動駕駛的技術已漸趨成熟，然而，若有一天無人車在路上不慎肇事傷人，你覺得責任應如何歸屬？是否有其他的人工智慧技術可能導致類似的法律問題？

人工智慧技術已悄悄進入我們的生活中，你覺得未來有哪些行業會需要用到人工智慧的技術？有哪些行業會因人工智慧的興起而逐漸消失？你覺得對於你未來想從事的行業類別，你需要具備哪些人工智慧的相關知識與技能？

1-4 起伏不斷的人工智慧發展歷程

不同於許多新興科技一經問市便成為市場寵兒，人工智慧的發展歷程頗為坎坷，期間的潮起潮落可大致歸納如下。

第一波浪潮（1956-1974年）：

在電腦被視為大型計算機的時期，電腦只要能展示出稍具智慧的能力（例如下棋），就會讓世人震驚不已，並衍生出無數美麗的幻想。自從AI一詞誕生於1956年那場會議後（該會議的發起人之一約翰・麥卡錫，John McCarthy，主要的研究方向正是電腦下棋），點燃了人們對人工智慧的熱情，同時迎來了第一波有關人工智慧的研究浪潮，感知器（Perceptron）之簡單型的「類神經網路」即為此時期的產物。

然而，期待愈高，失望也愈大。這時的人工智慧發展，在科學家過於樂觀的預估下，卻滿足不了人們的期待。「感知器」無法處理複雜的邏輯運算問題、也無法處理自然語言問題、更因其有限的計算能力應付不了快速增長的計算需求，面對這些當時難以突破的障礙，人工智慧的研究因而停滯不前，相關發展也急速降溫，在1970年代進入第一個寒冬。

第二波浪潮（西元1980-1987年）：

1980年起，由於統計思維的加入以及「專家系統」、「類神經網路」的新進展，人工智慧再次掀起另一波浪潮。在商業需求以及專業人士短缺的雙重激勵下，專家系統成為人工智慧商業應用的試金石。

專家系統是將專家的知識儲存在電腦裡，再結合規則，來回應人類的問題。換句話說，專家系統即是搜集現有的專業知識並加上推論規則，以產生如同專家一般的建議。舉例來說：生病時醫生會根據病人的症狀判斷出可能的疾病，進而採取對應的治療方式。

假如病人的症狀為「流鼻水、咳嗽、發燒」，就有比較大的機會是「感冒」並且給出感冒的建議治療方式。在蒐集許多醫療領域的專業知識之後，如果可以搭配一個良好的規則推論方法，就能從所蒐集的專業知識中推理，從而模仿醫師進行醫療診斷的過程。如此一來，只要輸入病人的徵狀，就可以從專家系統的推論中得知是什麼病因。一個存放專業知識的電腦系統，加上規則推論的方法，經過人工智慧不斷訓練，就可以成為一位專業的醫療人員，給予人們專業的建議。此醫療的專家系統最適用在醫療資源不足的偏鄉地區。

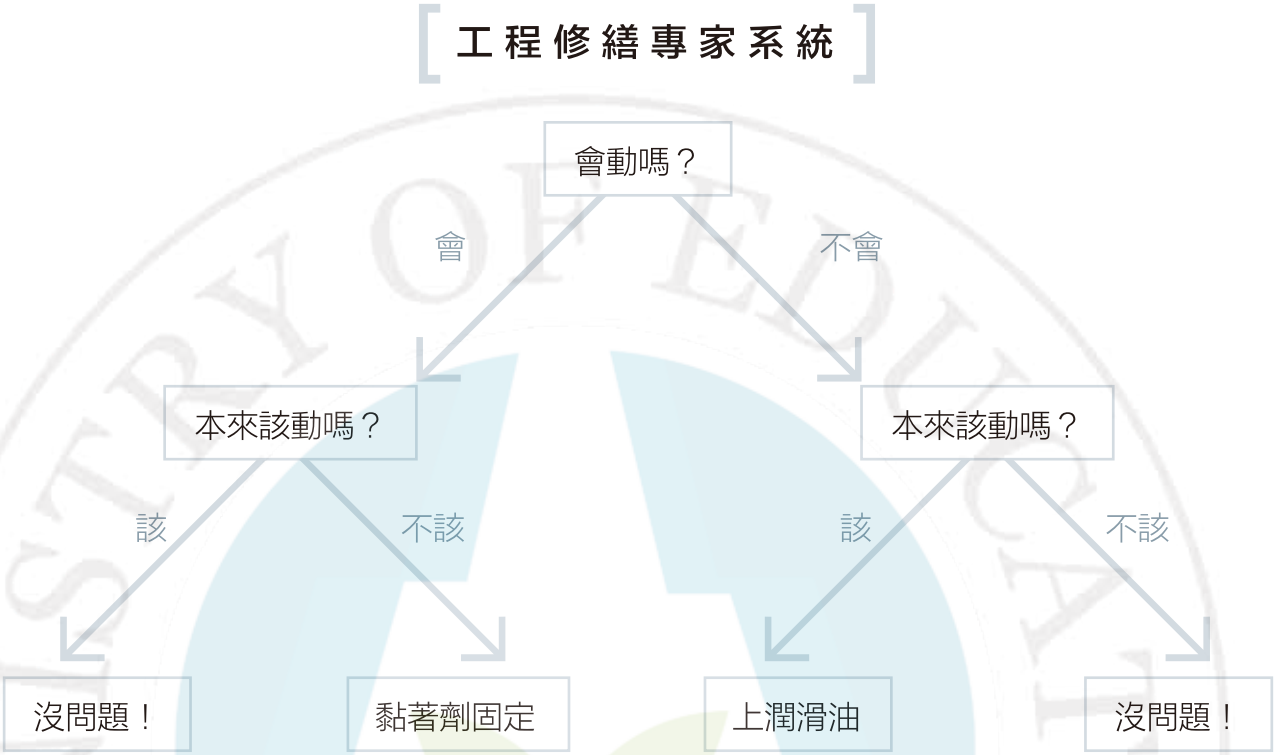


圖1-8 **專家系統**

於此同時，類神經網路也有新的進展，新式的類神經網路架構（如1986年Hinton等人提出的倒傳遞算法）陸續被用來學習與處理資料。在西元1980年代末期，因專家系統的開發及維護成本太高，而且其應用又僅侷限於非常狹小的專業領域，導致商業價值大減、研究經費被嚴重削減；另一方面，類神經網路的發展也在最佳化的過程遇上瓶頸，使得當時的「類神經網路」在學界幾乎成了人人避之唯恐不及的關鍵字。因此人工智慧的發展在1980年代末期再度進入了寒冬。

第三波浪潮（西元1993年-至今）：

1990年代後，人工智慧在發展方向上有了策略性變革，不同於以往的人工智慧致力於要讓電腦具備智慧。而是將著力點放在有目的性的「弱人工智慧」。弱人工智慧的任務是透過電腦把事情做的更好，提升人類處理事情的準確度，或是具有解決現有問題的能力。

在這一時期，網路的崛起，加上硬體設備的進步，促使電腦的計算能力越來越快，也使得資料的蒐集更加地容易。此一時期的人工智慧是以分析大量資料並配合機器學習為主要模式，在解決專門問題上已有長足的進步。例如：深藍電腦（Deep Blue）在1997年第二次挑戰並成功擊敗西洋棋世界冠軍、DARPA挑戰提供獎金給能夠橫越內華達沙漠的自動車、RoboCup的機器人足球比賽、資料探勘領域最重要的KDD Cup（臺大資工系團隊即曾多次榮獲此競賽冠軍），以及IBM的Waston益智問答等競賽。

2016年，由臺灣人為主要開發者的AlphaGo電腦圍棋以4比1打敗了獲得多次世界職業圍棋公開賽冠軍的韓國棋王，讓人們意識到人工智慧的發展可以藉由優異的網路系統、硬體設備提供強力的運算速度搭配現今最熱門的深度學習類神經網路獲得良好的成果。深度學習類神經網路為擁有許多層的類神經網路，藉以模擬人類大腦神經元的連繫方法，這種多層的類神經網路對於影像辨識相關的效果甚至超越人類，這也是深度學習這幾年來為什麼可以這麼熱門的原因。這波人工智慧浪潮現正持續不斷，而且未顯疲態。

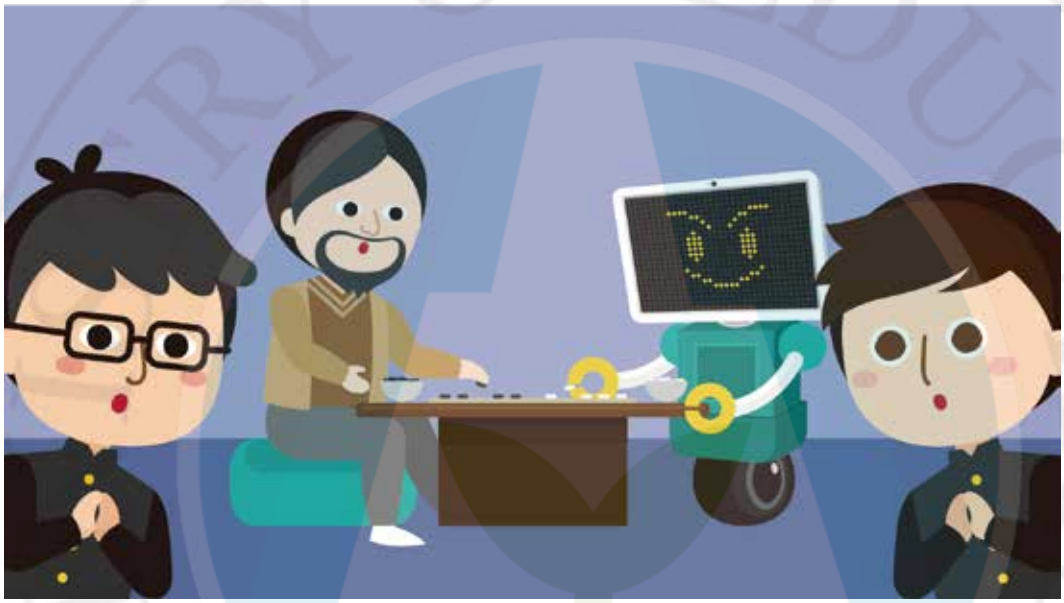


圖1-9 模擬對弈畫面

1-5 人工智慧的強弱之分

自進入第三波興盛以來，人工智慧的發展至今未見頹勢，實為史上少見歷經數次起落而依然屹立不搖的研究領域。人工智慧的研究趨勢也從以往的美麗幻想走向更務實的生活應用了。

一般而言，我們可將人工智慧分成三個不同的層次：**弱人工智慧、強人工智慧、強人工智慧假說**。

1) 弱人工智慧（Weak AI）：

也被稱為狹隘人工智慧（Artificial Narrow Intelligence，ANI），意即在某些限制下可以表現得很有智慧。從我們所熟知的Google AlphaGo、ImageNet視覺辨識競賽、IBM的Watson益智問題系統、自動駕駛，到辨識手寫數字等，均屬此類。它們或許能在單一領域媲美甚至超過人類的成就，但卻不能解決其他對人類而言相對容易的問題，例如：泡一杯香濃的咖啡，甚至擁有人類三歲小孩的智慧，表現相對應的行為舉止。

2) 強人工智慧（Strong AI）：

也稱之為通用人工智慧（Artificial General Intelligence，AGI）。可以跟人一樣有很廣泛的智慧，具備執行智慧行為的能力，但缺乏「心靈」或「自我意識」。其中一種強人工智慧的判斷指標為「咖啡測試」，指將一部機器帶到有咖啡機的家庭中，讓它在很自然的情境下，學會泡好一杯咖啡。這需要在陌生的環境中認識咖啡機、辨識咖啡和水、加入咖啡豆、找到適合的杯子放在咖啡機的正確位置上，和正確操作咖啡機。上述對人類十分簡單的動作卻需要整合機器人學、影像辨識等複雜的理論知識。現今強人工智慧仍是人工智慧的發展目標，尚未能實現。

圖1-10 **強人工智慧目前尚未實現。**

3) 強人工智慧假說（Strong AI Hypothesis）：

由約翰・羅傑斯・希爾勒（John Searle）提出，如同我們常在科幻電影或小說所看到的機器人，在這個層次中，電腦需要擁有跟人類一樣的「心靈」，需要認知自我並可以跟人類一樣思考。然而，在我們能徹底理解人類的智慧及自我意識產生的運作機制前，這很顯然仍是個不可能的任務。

到目前為止，強人工智慧尚未成熟，人工智慧仍無法表現出幼兒等級的通用智慧，像是AlphaGo下圍棋很出色，但如果你問它為什麼要下這一步的棋時，它並無力回答，甚至下棋時還需要人類協助擺放棋子。另外，像是IBM公司回答益智問題的Waston系統，它可以在益智問答上打敗所有人類，但是它也無力與你玩井字遊戲。而足球機器人可以把球踢的很厲害，但如果今天你告訴它「地震了，先不要踢了」，它還是不會理你繼續踢球，因為它並沒辦法做出非預設情境下的正確反應。

無論如何，在弱人工智慧的發展已漸趨成熟的情況下，現有的人工智慧技術都將是開發強人工智慧的基礎，而至於強人工智慧假說那種令人既期待又害怕受傷害的遠景，也許還要再等上很長一段時間，目前僅能在科幻電影中過過乾癮罷了！

1-6 人工智慧與遊戲

在眾多人工智慧擅長的領域中，大概以其在遊戲上的進展最容易讓人們可以明顯感受到人工智慧的長足發展。由於一些傳統遊戲的規則很容易系統化，且環境較單純有限還可以重覆玩，可讓電腦不斷嘗試錯誤來學習並且越來越厲害。一般來說，遊戲類型可以分為完全訊息及部分訊息兩類：

完全訊息的遊戲：整個盤面沒有不確定性，對於盤面及對手的資訊能夠完全掌握，例如：圍棋、象棋及西洋棋…等。

部分訊息的遊戲：例如：德州撲克，因為不知道其他玩家的牌，必須去猜測環境；星海爭霸，可以看到周圍的訊息環境，但較遠的資訊無法得知。這類部分訊息的遊戲需要去猜測，搜尋環境會更大。

圖1-11 **德州撲克遊戲畫面**

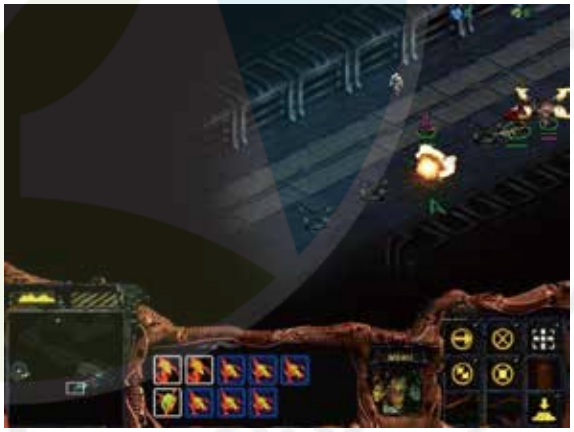


圖1-12 **星海爭霸遊戲畫面**

完全訊息的遊戲，近年來人工智慧已然攻克。對於不完全訊息的遊戲，人工智慧已有明顯的進步。2017年，微軟旗下新創公司Maluuba利用增強式學習（Reinforcement learning）的人工智慧技術，讓AI化身為150多個分身進行小精靈遊戲，透過評估每個分身行動的優劣，最後在考量整體利益下作出決策，拿到小精靈遊戲最高分999,900分。2018年，OpenAI公司也透過增強式學習技術，以一天訓練180年遊戲量的速度訓練AI程式，在熱門電玩遊戲Dota 2的5對5團體戰中，擊敗排名前1%的頂尖業餘玩家。

如上所述，可以想一想：如果有一天，你發現在某個網路遊戲中不斷慘敗，先別急著自暴自棄，也許網路的那一端並不是個有血有肉的真實玩家，是人或機器，有誰知道呢？

1-7人工智慧與機器學習

在看了前面的介紹之後，是不是對人工智慧如何運作產生興趣呢？是不是想要進一步了解其內涵呢？先不急，後面的章節將會逐步帶領大家領略人工智慧的內涵，但是在開始之前，先對人工智慧相關名詞之間的涵蓋關係有一較為整體性的概念是需要的。我想「深度學習」可能是2018年大家最耳熟的專有名詞之一，一般人也往往將深度學習跟人工智慧畫上等號，但事實上深度學習只是機器學習的一個分支，而機器學習又是人工智慧領域的一個分支，知道他們之間的關係後，日後就比較不會被這些名詞給弄糊塗了。

由於人工智慧領域所涵蓋的範疇甚廣，以高中教材編撰而言，要想完整涵蓋整個人工智慧範疇是困難的。鑑於最近的人工智慧熱潮大爆發，很大的原因之一是來自深度學習的令人驚艷表現，而其隸屬的機器學習歷經多年發展，體系相對完整，因此本教材會著重在機器學習領域中具代表性的四種機器學習類型介紹，包含：監督式學習、非監督式學習、增強式學習與深度學習。接下來的第二章會針對機器學習的眾多技術中，較常使用以及基本的知識先做介紹，第三章到第六章則是針對上述的四種機器學習類型做進一步介紹。

以下便對第二章到第六章的內容做摘要性的說明。

第二章　人工智慧的背景知識

這一章會先介紹一些人工智慧的基本知識及所需要的計算基礎等背景知識，例如：資料要如何收集及整理？特徵要如何選擇？特徵的距離要如何計算？有了這些背景知識之後，接下來就可以進一步探究一些人工智慧技術的運作原理了。

第三章　監督式學習

為方便理解，我們可以把監督式學習視為「有老師指導」的學習方法，因為老師擁有正確答案(或稱標記)，所以在訓練階段可以提供每個訓練樣本所對應的標記資訊，根據訓練樣本的對應標記資訊所設計的機器學習方法就叫做監督式學習。在這一章，將會學到監督式學習中的最短距離分類器、ＫＮＮ分類器以及決策樹的運作原理。

第四章　非監督式學習

在現實運用上，採用監督式學習便要取得每個訓練樣本所對應的標記資訊，這部份往往需要付出可觀的人力成本，以訓練機器辨識動物影像為例，要對每張訓練用的動物影像標記影像內容是哪種動物，便需要請專人做這件事。為避免此一限制，便有非監督式學習方法被提出，相對於監督式學習有老師指導，非監督式學習就是一種「沒有老師指導」的學習方法，也就是說，在訓練階段無需提供訓練樣本所對應的標記資訊，也由於欠缺此資訊，因此通常會就訓練樣本彼此之間的相似性進行聚類分群。在這一章，將會學到K-means以及階層式分群法。

第五章　增強式學習

換個角度來想，機器學習的方式是不是可以像心理學中的行為主義理論一樣，做對事情給予獎勵，做錯事情給予懲罰？增強式學習就是運用這一概念。即在環境所給予的獎勵或懲罰的刺激之下，對不同環境刺激有著不同的預期獎勵，並且逐漸調整趨向正確的方向學習。這一章是以典型的迷宮問題，來說明增強式學習的原理與運作，並介紹常用的Q-學習方法。

第六章　深度學習

人類的智慧在於大腦，而大腦的運作依賴著許許多多的神經元。是否電腦的計算處理也可以像人類大腦一樣的方式運作？後來仿造人類大腦神經元運作方式的電腦程式架構被開發出來，那就是類神經網路。多層類神經網路能藉由倒傳遞算法調整網路權重，達到分類最佳化。而近幾年令人矚目的深度學習中的卷積神經網路（CNN）則是將特徵擷取與分類整合，一起做最佳化，進而得出令人驚豔的效能。在這一章，將會學到卷積神經網路的原理，體會到它在影像辨識應用上的成效。

圖片來源：

圖 1-1：圖靈發表之論文。A. M. Turing, “Computing Machinery and Intelligence,” Mind, Vol. 49, 1950, pp. 433-460.

圖1-2：人工智慧之父。CyberHades (CC BY-NC 2.0)

圖1-3：人臉辨識。Katherine Scott (CC BY-NC 2.0)

圖1-4：自動駕駛系統車。Tony Hisgett from Birmingham, UK (CC BY NC 2.0)

圖1-5：自然語言處理。CC0

圖1-6：語音辨識系統。CC0

圖1-7：照護型機器人。University of Salford Press Office(CC BY 2.0)

圖1-9：模擬對弈畫面。截取自本計畫國中人工智慧簡介影片

圖1-11：德州撲克遊戲畫面。Albert Hsieh CC BY-NC 2.0

圖1-12：星海爭霸遊戲畫面。Gorekun CC BY-NC 2.0

第二章

背景知識 Background Knowledge

你曾經有上傳照片到臉書後，照片中人像被自動標註的經驗嗎？
對臉書的人像辨識系統來說，
人們上傳的照片，就如同機器學習所需的資料。
現在就讓我們一起來探索，
從資料取得到成為機器學習可用的素材，
究竟需要歷經哪些程序？

2-1 資料收集

機器學習需大量的資料，用以建立準確的模型，這些大量的資料可從以下管道取得：

- 自行收集
- 使用開放資料集
- 各家公司所提供的API
- 網頁爬蟲程式
- 直接向廠商購買

1) 自行收集：

即自行使用各種方法，從資料來源所收集到的第一手資料。其方法可透過問卷調查或藉由一些配備進行資料的蒐集，例如店家可將要瞭解消費者的相關問題設計成問卷，然後向消費者進行問卷調查。又例如要蒐集空汙資料，則可架設感測器來收集空氣的雜質濃度與成份等。

2) 使用開放資料集：

近年來，國內外已有由政府機關或是研究機構所提供的開放資料，供民眾或是研究者下載使用。這些公開資訊均有一定的資料格式，使用者只要遵循相關規定，即可在其開放資料平台自行下載使用。列舉一些公開資料網站如下：

- 政府資料開放平台：https://data.gov.tw/
- 臺南市政府資料開放平台：http://data.tainan.gov.tw/
- CHICAGO DATA PORTAL：https://data.cityofchicago.org/
- Kaggle：https://www.kaggle.com/datasets
- UCI Machine Learning Repository：https://archive.ics.uci.edu/ml/datasets.html

圖2-1 **政府資料開放平台提供民眾依資料集類別下載使用**

圖2-2 **台灣即時雨量資訊平台**

3) 各家公司所提供的API：

「API」為Application Programming Interface的簡稱。當我們要使用像Facebook、Twitter網路應用程式或服務時，一般的使用者，可以透過瀏覽器或是手機應用程式與之互動（發文、點讚等）；而對於熟知撰寫程式的人而言，則可以透過寫程式的方式與這些應用程式或服務互動。使用API的好處是可以使用程式碼向Facebook、Twitter等公司取得具結構化的資料，並可省去不少心力去整理資料。提供下面幾個知名網路服務的API文件網址：

- Twitter API：https://developer.twitter.com/en/docs
- Facebook API：https://developers.facebook.com/docs/graph-api/using-graph-api/
- Flickr API：https://www.flickr.com/services/developer/api/
- YouTube API：https://developers.google.com/youtube/v3/getting-started

4) 網頁爬蟲程式：

在講網頁爬蟲程式前，要先瞭解使用瀏覽器瀏覽網頁的工作原理。當我們在瀏覽器的網址列輸入網址或點選某個超連結網址時，代表我們正向該網址所指向的伺服器發出請求，當收到請求後伺服器就會回傳HTML原始碼的網頁內容，瀏覽器再將HTML的格式進行解析後呈現出內容。

然而，當我們想要的資料來源並未提供API讓程式直接連接，而是將資料放在網頁上時，我們也可以嘗試透過撰寫網頁爬蟲程式來取得資料。網頁爬蟲程式是透過程式，將含有我們所需資料網頁的HTML原始碼擷取下來，然後再透過分析該HTML原始碼取得我們想要的資料。

以經濟部能源局國際原油價格網頁「**http://gg.gg/dj046**」為例，可透過爬蟲程式去分析該段原始碼並取得所需的原油價格資料。擷取之網頁畫面如圖2-3。左圖為原始網頁呈現的內容，右圖為HTML原始碼。

圖2-3 爬蟲程式藉由程式解析取得原油價格資料

5）直接向廠商購買：

政府機構或許多私人企業皆有提供付費方式，來讓民眾取得所提供的資料，例如中央氣象局的歷年觀測資料（http://e-service.cwb.gov.tw/wdps/）、Twitter的完整關鍵字查詢資料等。

遙測氣象資料之項目及費額

項	目	計價單位	每單位資料金額（新臺幣）
一	氣象衛星黑白雲圖	每張	三百元
二	氣象衛星彩色雲圖	每張	四百元
三	氣象衛星影像存檔光碟片	每片	四百元
四	氣象衛星數位資料存檔光碟片	每片	四百元
五	氣象衛星單項儀器單一觀測路徑資料	每筆	一百元
六	氣象雷達資料光碟片	每片	四百元

圖2-4 **中央氣象局販售遙測氣象資料之項目及費額表**

2-2 資料整理與儲存

當我們從各種管道得到資料後，是否可以馬上就提供機器進行學習，並從中取得規律性呢？答案是不可以的，因為這些資料常是未經過處理而存在一些問題，例如：資料格式不一致、資料有缺漏、資料可能為無效或重複等問題。所以要提供機器學習的資料，必須要將上述的問題處理後方能使用，將資料格式化、資料缺漏、無效或重複的資料處理，此歷程稱為資料整理（Data Cleaning）。下表2-1是資料缺失的範例（每列代表一筆資料），對於資料有問題的部分，可依資料特性透過各種方法修正，以下介紹各種常用的方法。

表2-1 **資料缺失的範例**

正常的資料	2	3	4	2	1	1	4	3	2	4
部份欄位缺失	2	3	4	null	1	1	4	null	2	4
很多欄位缺失	2	3	4	null	null	null	null	null	2	4
所有欄位缺失	null	null	null	null	null	null	null	null	null	null
欄位尺度不同	2	4	yes	0.2	n	3	Hot	2	2	3
含有異常值	2	4	1	23	3	231	2	-78	4	3

- **插值法：**透過缺失值前後的資料分析，推測缺失欄位值的方法，適用於資料相依欄位具連續性的場合，如連續觀測的溫度、空氣品質資訊等。

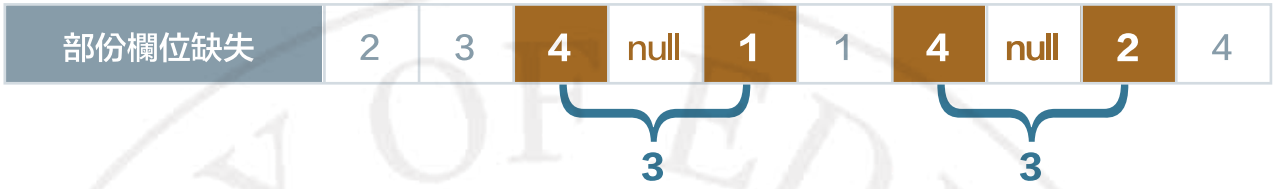


圖2-5 利用插值法推測缺失欄位的值

- **插入空值：**在無法推測欄位資料時，也可以藉由插入空值/零的方式處理，亦即宣告該筆資料不含有該項欄位具備之特徵。

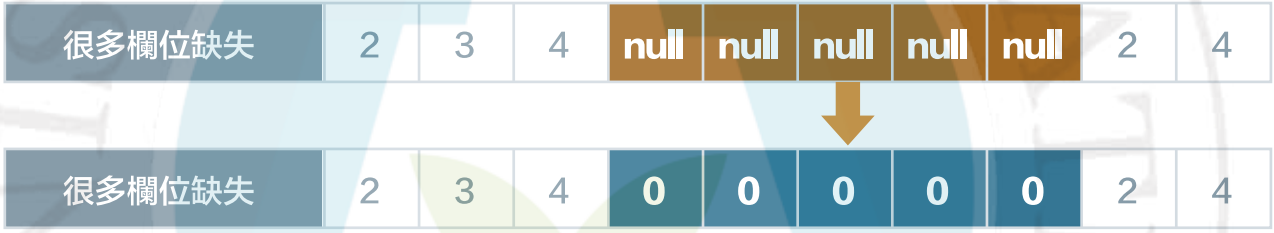


圖2-6 無法推測欄位時，插入空值也是修正資料的方法之一

- **插入平均值/中位數：**能計算資料各欄位的平均值或中位數。將之代入，即將該筆缺失的資料當作平均的情況看待。

- **將名目值量化：**當資料欄位的尺度為名目資料或次序資料時，可以透過量化的方式把它們轉化成易於分析的數值資料。

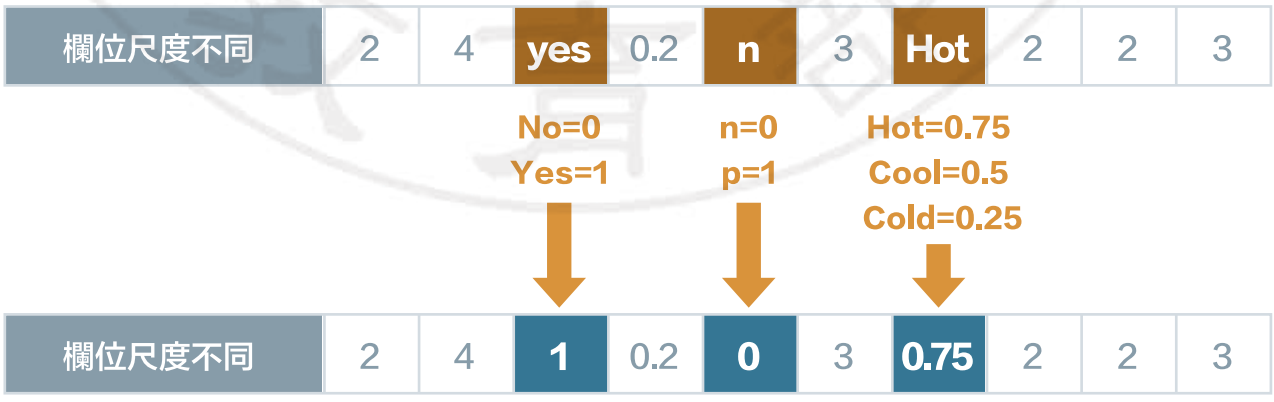


圖2-7 名目值資料透過量化轉為數值資料

- **移除異常資料：**在有異常值的情況下，如觀測儀器故障、問卷亂填等，大多會建議直接刪除該筆資料以保證資料的準確性。然而如何找到異常值呢？最簡單的做法是利用「標準差」。在一般情況下，觀測值常會呈現常態分布，如下圖2-8。由常態分布的性質可以知道99.8%的值都會分布在平均值正負 3 個標準差的範圍內，因此便可以利用這項基準，來辨別某項數值是否落入平均值正負3個標準差以外的範圍，而將之視為異常值。

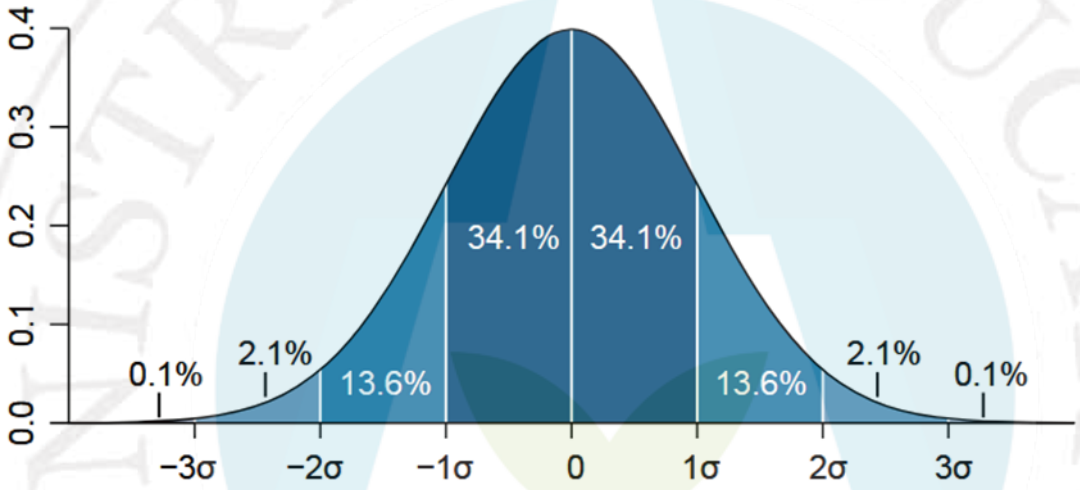


圖2-8 **常態分布圖**

資料經過整理完後，會儲存成檔案，並存放於「資料庫」，以供未來使用。對於資料的儲存，若是檔案內只含數值或文字資料，則可以儲存成文字檔。若是大量的資料，純文字檔的管理並不方便，很難快速查詢到所需的任一小段資料。此時，便可以將資料儲存至資料庫。存放於資料庫有很多好處，例如：方便管理大量資料、快速查詢特定資料、保障資料安全與完整等。如何將資料建立成資料庫，必須有資料庫的相關概念及建立技巧。此方面知識請參閱其他相關書籍或上網搜尋相關資訊。

2-3 特徵選擇

原始資料整理完畢後，接下來要檢視是否資料包含的所有特徵都是有用的。例如銀行在審核信用卡申請前，必定會針對申請者的個人資料進行判斷，判斷該申請者是否有還不出卡債的風險，個人資料可能包含性別、居住地、年齡、職業、年收入、家庭狀況等，每一項特徵對於判斷是否有風險可能有用、也可能無用。將所有特徵加入分析需要花費很長的時間，因此如何萃取資料，僅留下數個最有用的特徵便成為了一項課題。

如何檢驗特徵與特徵之間，以及特徵與類別之間的關聯性，其作法可透過統計中的皮爾森相關係數（Pearson Correlation Coefficient）來進行檢測。

- **皮爾森相關係數（Pearson Correlation Coefficient）：**可用於度量兩個變數x和y之間的相關（線性相依）性，其值介於-1與1之間。

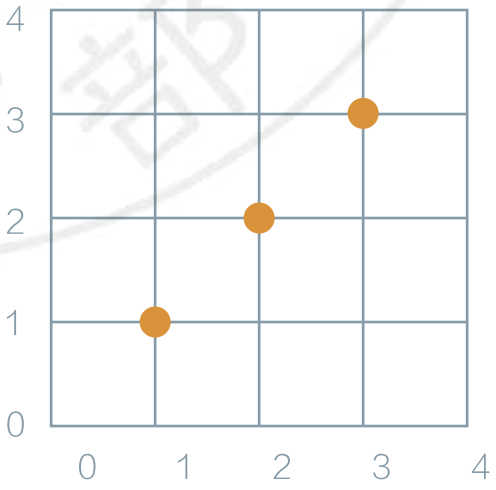
設兩特徵 x、y 各有n筆資料。形成$(x1, y1), (x2, y2)\cdots$定義皮爾森相關係數為：

$$r = \frac{\sum_{i=1}^{n}(x_i-\bar{x})\times(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2 \times \sum_{i=1}^{n}(y_i-\bar{y})^2}}$$

- **皮爾森相關係數計算範例：**假設兩特徵 x、y，現有三筆資料(1,1)、(2,2)與(3,3)，則此三筆資料帶入計算得到皮爾森相關係數如下：

$$r = \frac{(1-2)\times(1-2)+(2-2)\times(2-2)+(3-2)\times(3-2)}{\sqrt{[\,(1-2)^2+(2-2)^2+(3-2)^2\,]\times[(1-2)^2+(2-2)^2+(3-2)^2\,]}}$$

$$= \frac{1+0+1}{\sqrt{(1+1)\times(1+1)}} = \frac{2}{2} = 1$$



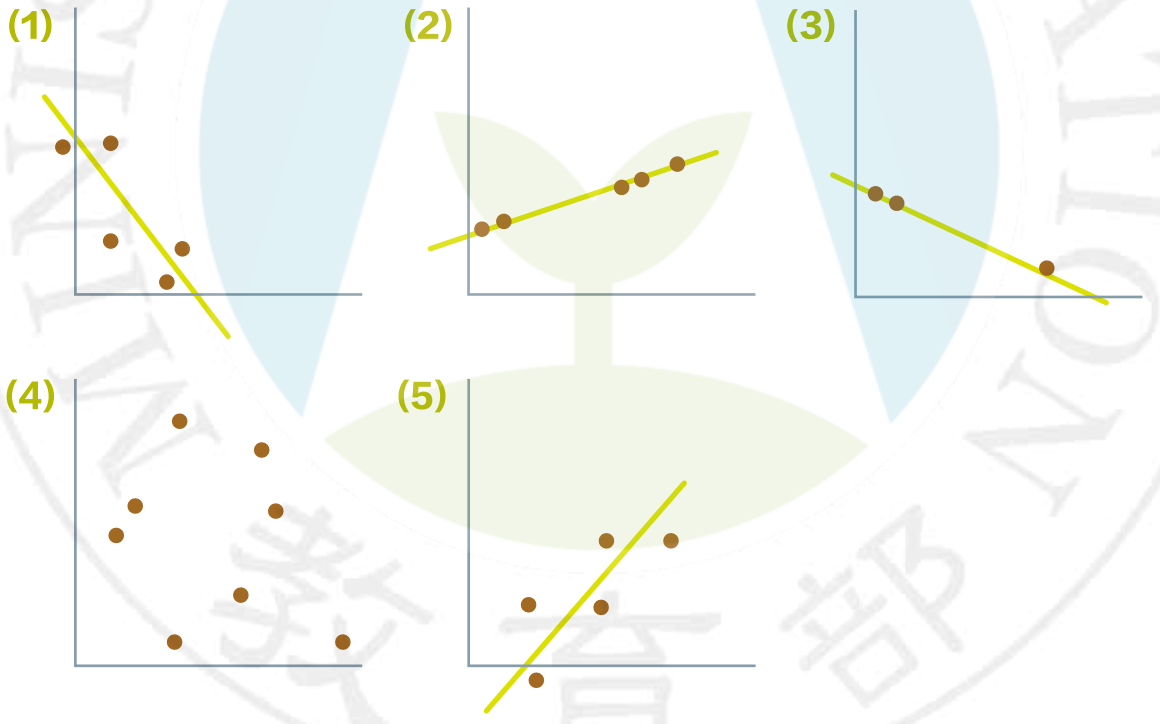
由上述範例可知，x 特徵與 y 特徵計算得到的皮爾森相關係數為1，意味著 x 和 y 所有的數據點都落在一條直線上，當 y 隨著 x 的增加而增加，即為「正相關」。反之，若計算得到之係數值為-1，意味著 x 和 y 所有的數據點都落在一條直線上，但 y 會隨著 x 的增加而減少，此為「負相關」。

課堂任務

請判斷下列各散布圖兩變數的相關係數範圍，完成配對。

A. $r = -1$　　B. $-1 < r < 0$　　C. $r = 0$

D. $0 < r < 1$　　E. $r = 1$



當相關係數絕對值愈高，兩特徵的散布圖呈現愈明顯的線性關係，因此我們可以藉由保留與類別之相關係數絕對值較高的特徵作為分類的依據。

2-4 特徵距離的計算

選定欲用來分析資料的特徵之後，接著要介紹特徵距離的計算，也就是資料間的相似度。在此將介紹兩種不同的特徵距離計算方式：

- 曼哈頓距離（Manhattan distance）
- 歐幾里得距離（Euclidian distance）

曼哈頓距離（Manhattan distance）：在如棋盤式街區的城市，要從這個街角開車到另一個街角，而開車的距離顯然不是兩點間的直線距離，而是順著馬路直行、轉彎、直行，這距離的總和就是「曼哈頓距離」。

在此是代表資料點間每個特徵距離的和，如下公式所示：

$$|a-b|_1=\sum_{i=1}^{n}|a_i-b_i|$$

例如：(1,2) 與 (3,4) 的曼哈頓距離是：$|1-3|+|2-4|=4$

歐幾里得距離（Euclidian distance）：是最容易理解的距離度測方法，在國中、高中計算的平面、空間上兩點的距離就是指歐幾里得距離，也是一般所理解的直線距離。

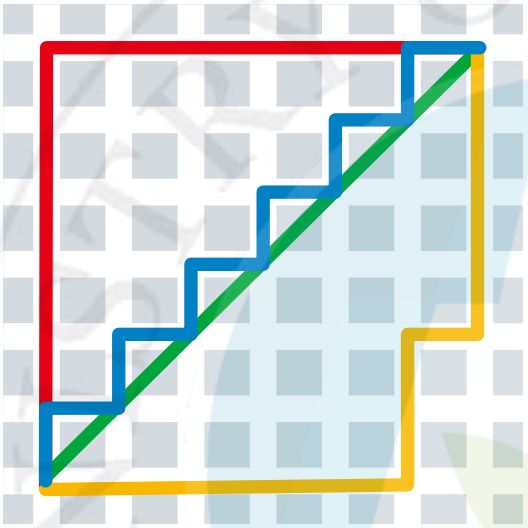
在此是代表資料點間每個特徵的直線距離，如下公式所示：

$$|a-b|_2=\sqrt{\sum_{i=1}^{n}|a_i-b_i|^2}$$

例如：(1,2) 與 (3,4) 的歐幾里得距離是 $\sqrt{|1-3|^2+|2-4|^2}=\sqrt{8}$

課堂任務

連連看：在下圖 紅、黃、藍、綠 共4條路線中，哪一條是曼哈頓距離？哪一條是歐幾里得距離？



曼哈頓距離
Manhattan distance

歐幾里得距離
Euclidian distance

▶ **以下列的例子介紹二維（具有2個特徵值的資料）的特徵距離計算：**

	甲同學	乙同學	丙同學
身高(公分)	172	166	170
體重(公斤)	60	54	72

由上表可知：　甲（身高,體重）=（$x1$,$y1$）=（172,60）

乙（身高,體重）=（$x2$,$y2$）=（166,54）

分別計算甲同學與乙同學的曼哈頓距離與歐幾里得距離為：

甲同學與乙同學的曼哈頓距離為：

$$|a-b|_1=\sum_{i=1}^{n}|a_i-b_i|=|172-166|+|60-54|=12$$

甲同學與乙同學的歐幾里得距離為：

$$|a-b|_2=\sqrt{\sum_{i=1}^{n}|a_i-b_i|^2}=\sqrt{|172-166|^2+|60-54|^2}$$

$$=\sqrt{36+36}=\sqrt{72}$$

課堂任務

延續上述範例，請分別計算出甲同學與丙同學間的曼哈頓距離與歐幾里得距離各為多少？

2-5 資料的標準化

假設我們有以下的3筆資料 ***a***、***b***、***c***，每筆資料分別包含 ***x***、***y***、***z***、***n***、***p*** 及 ***q*** 共六種特徵值：

	x	*y*	*z*	*n*	*p*	*q*
a =	2	3	4	2	1	15
b =	1	2	2	4	3	51
c =	1	4	3	2	2	35

若使用歐幾里得距離計算，可得兩兩資料間的特徵距離分別為：

$$d(a,b) = \sqrt{|2-1|^2+|3-2|^2+|4-2|^2+|2-4|^2+|1-3|^2+|15-51|^2}$$
$$\approx \sqrt{|15-51|^2}$$

$$d(b,c) = \sqrt{|1-1|^2+|2-4|^2+|2-3|^2+|4-2|^2+|3-2|^2+|51-35|^2}$$
$$\approx \sqrt{|51-35|^2}$$

$$d(a,c) = \sqrt{|2-1|^2+|3-4|^2+|4-3|^2+|2-2|^2+|1-2|^2+|15-35|^2}$$
$$\approx \sqrt{|15-35|^2}$$

由上述計算結果可以發現這些距離幾乎都被值最大的特徵所影響，為避免這種狀況，則可以對資料作「標準化（Normalization）」，將每個特徵的重要程度變成一致。

在此介紹以「Min-Max Normalization」做標準化：

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

就上述三筆資料而言，以特徵 x 為例，若要以 Min-Max 標準化，x_a、x_b、x_c 分別計算得：

$$x_a = \frac{2-1}{2-1} = 1$$

$$x_b = \frac{1-1}{2-1} = 0$$

$$x_c = \frac{1-1}{2-1} = 0$$

同理也可以對其餘的特徵都分別做標準化，標準化後的三筆資料為：

	x	y	z	n	p	q
a =	1	0.5	1	0	0	0
b =	0	0	0	1	1	1
c =	0	1	0.5	0	0.5	0.5556

此時以歐幾里得距離計算 a 與 b 的特徵距離就會是：

$$d(a,b) = \sqrt{\begin{array}{l} |1-0|^2 + |0.5-0|^2 + |1-0|^2 + \\ |0-1|^2 + |0-1|^2 + |0-1|^2 \end{array}}$$

比較資料標準化前後的特徵距離計算結果，可發現經過標準化後得到的特徵距離不會受值特別大的特徵影響太多。

2-6 資料集分割

經過一連串繁複的資料整理後，最後一步就是交由機器開始訓練模型。但，所有的資料都能夠直接拿去做訓練嗎？要如何驗證訓練出來模型的準確度呢？或許你會想說：那就把部分的訓練資料拿來當作測試就好啦！但是這樣會有一個問題，這些用來測試的資料早在訓練時都用過了，如果再用來進行驗證會不會有點「球員兼裁判」的感覺呢？

因此，通常會將原始資料集分成「訓練集」與「測試集」，訓練集的資料通常會比測試集大上許多，是用來訓練模型的，待訓練完後，再以訓練時沒看過的測試集資料，測試訓練出來模型的準確度，如下圖2-9所示：

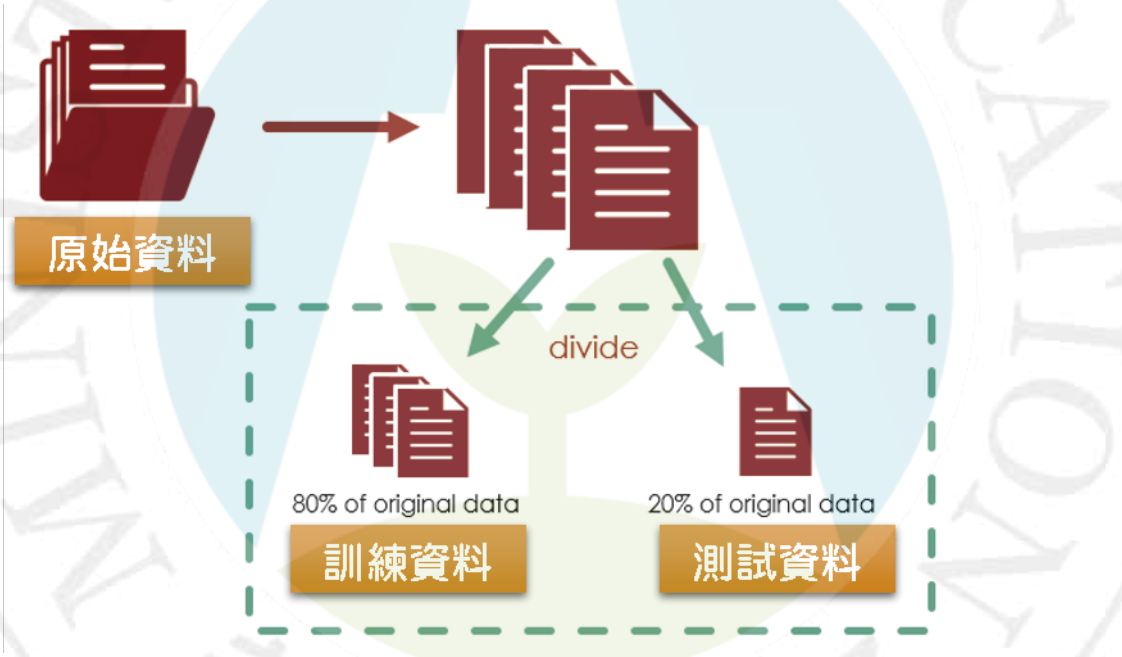


圖2-9 將原始資料集分為訓練資料集與測試資料集

有時為了強化訓練模型的準確度，也能在訓練過程中不斷去驗證模型，此時便可以再從訓練集中取少數資料形成「驗證集」，在訓練完模型之後自行使用驗證集資料來自我驗證，確定模型的有效性之後再拿測試集測試。如下圖2-10所示：

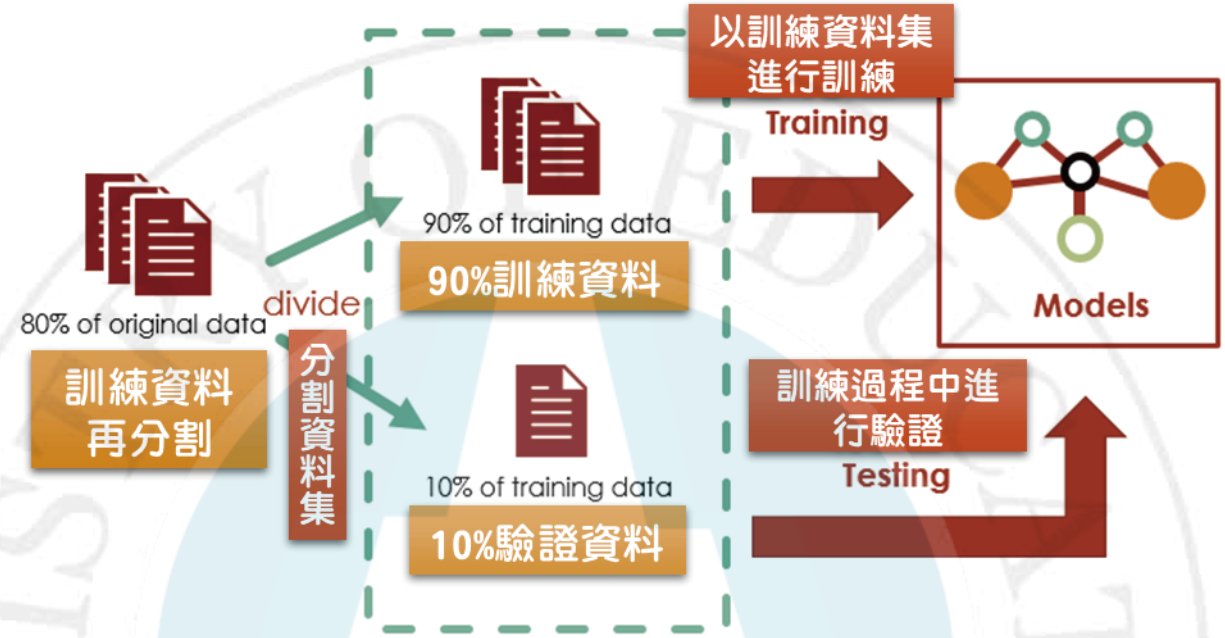


圖2-10 **將訓練資料集再分為大部分訓練資料集與少部分驗證資料集**

另外也能將訓練集切成數等份，並將每等份輪流作為驗證集，分別對模型做交叉驗證，如此可避免只使用一份驗證資料時，因為運氣的好壞造成不公平的驗證結果。最常見的交叉驗證手法即是K折交叉驗證（K-fold cross validation）。將訓練集平均分成K個集合，然後將某一個集合當做驗證資料，剩下的K-1個集合做為訓練資料，如此重複進行直到每一個集合都被當做驗證資料為止。最後再將此k個驗證結果取其平均值，便是此次交叉驗證所得之結果。透過這樣的方式，每一輪使用的訓練集與驗證集都不同，能更精準地驗證預測模型的有效性。

假設將所有的訓練資料切成四等份，就可以分別進行四次的模型驗證：

	Data 1	Data 2	Data 3	Data 4
第一輪驗證：	**驗證集**	訓練集	訓練集	訓練集
第二輪驗證：	訓練集	**驗證集**	訓練集	訓練集
第三輪驗證：	訓練集	訓練集	**驗證集**	訓練集
第四輪驗證：	訓練集	訓練集	訓練集	**驗證集**

如上表，第一輪使用 ***Data 2***、***3***、***4*** 來訓練模型，並使用 ***Data 1*** 做驗證；接著第二輪改為使用 ***Data 1***、***3***、***4*** 來訓練模型，並使用 ***Data 2*** 來驗證；以此方式將每份資料都驗證過一次，這樣的模式稱為 **4折交叉驗證**（4-fold cross validation）。

圖片來源：

圖 2-1：擷取自政府資料開放平台，
https://data.gov.tw/

圖 2-2：擷取自用數據看台灣網站中的台灣即時雨量資訊，
https://www.taiwanstat.com/realtime/rain/

圖 2-3：擷取自經濟部能源局國際原油價格查詢網頁，
http://gg.gg/dj046

圖 2-4：擷取自中央氣象局販售遙測氣象資料之項目及費額表，
http://e-service.cwb.gov.tw/wdps/cwb_fee.htm

圖 2-8：常態分布圖，圖片來源M.W.Toews，
https://commons.wikimedia.org/wiki/File:Standard_deviation_diagram.svg

第三章

監督式學習 Supervised Learning

「『狗』是人類最忠實的朋友」，你對狗的認識有多少？
又是從什麼時候開始認識「狗」的呢？
是爸媽、或是師長告訴你的，還是…？
在認識動物時，你是如何區辨狗與貓的差異？
而在人工智慧的領域中，
機器學習，是如何模仿人類對資料進行有效的分類呢？
這個單元，就讓我們一起進入「**監督式學習**」吧！

3-1 監督式學習簡介

這個單元，所要談的「監督式學習」，許多人可能會聯想到「全民公敵」這部電影的劇情，那就是我們是否會處在一個被電腦網路監視器隨時監控的環境？現在就讓我們透過「監督式學習」來一窺究竟。

小時候，當我們看到一隻鳥，不知道那是甚麼，也不知道怎麼描述，直到父母跟我們說：「那是一隻鳥」，而且是一而再、再而三跟我們說：「那是一隻鳥」；因此以後只要看到鳥或者鳥的圖片如圖3-1，我們就知道那「**是**」一隻鳥，而看到圖3-2就知道那「**不是**」鳥。

由此可知，我們從小就是在父母監督下學習到各種知識。

圖3-1 鳥

圖3-2 狗

圖3-3 **具有標準答案的監督式學習示意圖**

人工智慧的監督式學習就跟小孩子學習一樣，需要由外界給予大量的資料並提供資料對應的結果，讓電腦知道資料的相關性，進而從其相關性再探究是否具有因果關係。換句話說，電腦在學習的過程當中，接收到的每一筆資料都有對應的「標準答案」，當新的資料需要被判斷時，就會藉由既有的資料分布特性，進行資料的判別。

大家一定很好奇，如果要讓電腦判別圖片中的動物是否為一隻鳥？那麼其依據是什麼？那就是「特徵值」。

圖3-4 **監督式學習透過學習大量的資料來增加判別未知資料時的準確性**

特徵值

世界萬物都是獨特且具有差異性，但是同一種類事物，必然也存在著相同或相似之特性。當我們要訓練機器具有人工智慧，而提供了大量資料給機器時，該機器首要任務就是要先找出資料中的相同或相似特性。這個特性，即是資料的特徵值。

例如：想要電腦運用人工智慧識別出「貓」，那麼可以在訓練的過程中提供大量的圖片，然後設定特徵值，讓電腦比對學習。假設我們設定的特徵值是：四條腳、尖耳朵、耳朵在頭頂兩側、鼻子在嘴巴上緣、長鬍鬚…等，那麼電腦在進行監督式學習時，就會將這些圖片分別建立其特徵值，並加以分類形成資料庫。之後提供給電腦（機器）辨識的照片，就可以經由特徵值資料庫來比對該照片合乎了哪些特徵值，而評斷出該照片中的動物是不是貓。

圖3-5 學習的過程中藉由比對特徵值來找出相對應的類別

特徵值描述得愈清楚完整，就能有精準的判斷結果。所得結果不僅是貓的動物判別，甚至還可以更細緻地判別出不同品種的貓，例如：波斯貓、俄羅斯藍貓、蘇格蘭摺耳貓…等。若要達成貓的品種識別，則在訓練電腦的過程中，就必須提供完整的對應答案，使能區隔出貓的品種差異。

課堂任務

任務一：辨識「豬」的特徵值。

任務二：藉由任務一的練習結果，持續探討：

1、會不會有其他動物也有「豬」的特徵值，因而導致辨識錯誤？

2、為了增加判別的準確性，是否還有其他特徵值可以加入？

分類 Classification

電腦藉由特徵值的蒐集，將資料依照所設定的規則加以整理，然後產生一套區分該物件的原則，這一套原則可建構出分類器（Classifier），要被判別的物件就是藉由分類器來區辨是屬於哪一個分類。

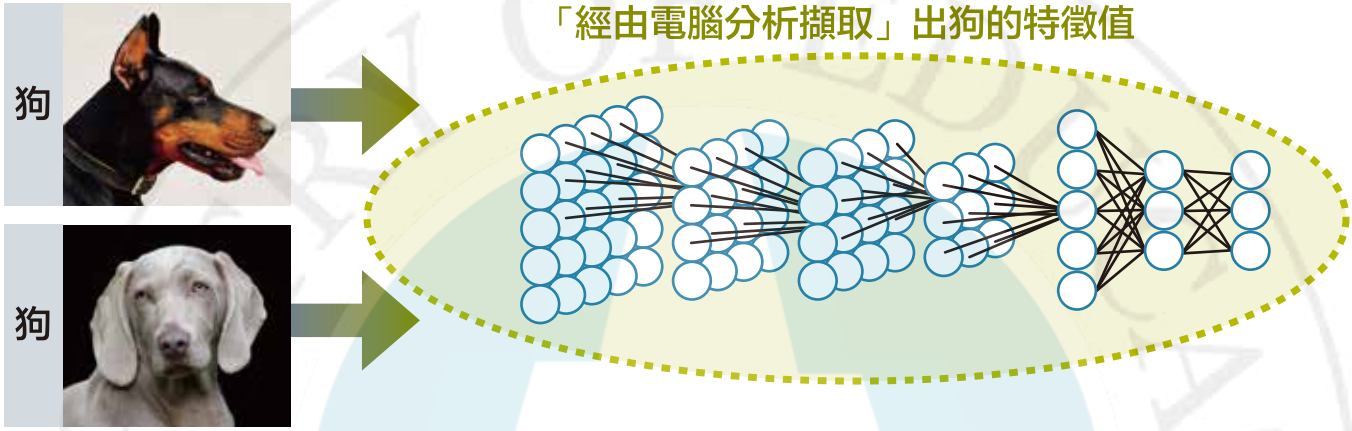


圖3-6 透過電腦分析每筆資料的特徵值將有助於建構分類器

接下來，讓我們來認識三種分類器的運作方式，使能對「分類」有進一步的瞭解！

3-2 最短距離分類器

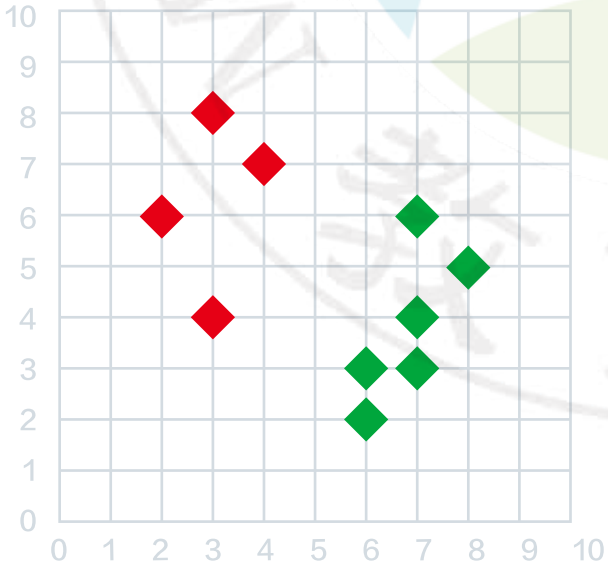


圖3-7 紅色及綠色菱形分別代表2種類別的資料

在介紹最短距離分類的運作方式前，必須先將接受訓練的資料數值化，讓每一個訓練資料皆能展現於空間座標中，這邊所說的空間座標是N維（其中N大於等於1）的空間座標，至於N為多少，則須視訓練資料的複雜程度而定。

圖3-7中，紅色及綠色的菱形分別代表2種不同類別的資料，而座標中X軸與Y軸則代表該筆資料的2種特徵值。以資料數值化的角度來看，若每個菱形以X軸及Y軸的數值表示(X,Y)，則紅色類別的資料分別是(2,6)、(3,4)、(3,8)與(4,7)；綠色類別的資料分別是(6,2)、(6,3)、(7,3)、(7,4)、(7,6)與(8,5)。

接下來針對已知的紅色及綠色類別，找出每個類別的中心點。依據紅色、綠色2個類別中的每個點X、Y值進行平均，就可以得到每個類別的中心點。

表3-1 依據 **紅色**、**綠色** 2個類別的X、Y值，取得平均數來得到該類別的中心點

類別	X	Y	類別	X	Y
紅色	2	6	綠色	6	2
紅色	3	4	綠色	6	3
紅色	3	8	綠色	7	3
紅色	4	7	綠色	7	4
平均值	**3**	**6.25**	綠色	7	6
			綠色	8	5
			平均值	**6.83**	**3.83**

依據表3-1之計算結果，將紅色及綠色2種類別的中心點繪製於座標軸上，可得圖3-8之結果。

若有一筆新的資料位於座標軸上(4,2)，則該筆資料與紅色、綠色2種類別之中心點距離何者最短，則屬該類別，在此假設以「歐幾里得距離」為兩點距離之計算方式，計算結果如表3-2。

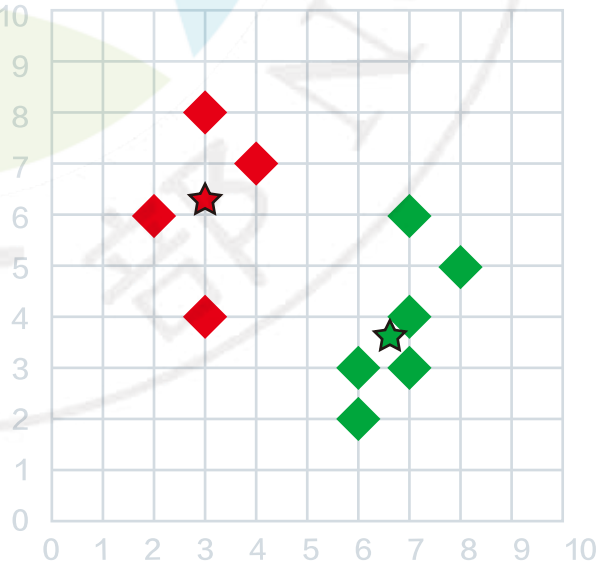


圖3-8 填充**紅色**與**綠色** ☆分別代表該類別中心點所在位置

歐幾里得距離：

在N維的座標中，點 $a=(a_1,...,a_N)$ 和 $b=(b_1,...,b_N)$ 之間的歐幾里得距離為：

$$d(a,b)=\sqrt{(a_1-b_1)^2+(a_2-b_2)^2+...+(a_N-b_N)^2}$$

表3-2 新資料(4,2)分別與2種類別中心點之歐幾里得距離

新資料座標	類別中心點座標	2點距離
(4,2)	紅色(3,6.25)	$d=\sqrt{(4-3)^2+(2-6.25)^2}=4.37$
(4,2)	綠色(6.83,3.83)	$d=\sqrt{(4-6.83)^2+(2-3.83)^2}=3.37$

依據表3-2的計算結果顯示，(4,2) 與綠色類別之距離較短，因此歸屬於綠色類別的資料。

課堂任務

任務一：依據最短距離分類器的運作，請問若有一筆新的資料在座標軸上(5,5)，請問該筆資料應該屬於哪一個類別？

任務二：想想看，什麼樣的狀況下，最短距離分類器不能獲得理想的分類結果？是什麼因素對其造成影響？

3-3 KNN分類器

KNN分類器（K-Nearest Neighbor Classifier）仍然會將資料以數值化的方式進行表示，且已知的每筆資料都有對應的所屬類別。這款分類器在碰到一筆新的資料時，就會找出與該筆資料最接近的K個資料，在此假設以歐幾里得距離來判別兩點距離。此分類器的運作方式就像「投票表決」般，當K個資料中的某個類別的數量較多時，則新的資料就會歸屬於該類別。

當K值設定為1，也就是說要找到距離綠色點最近的1個點，由圖3-9發現，有找到1個藍色的點距離綠色點最近，所以就會判斷這個綠色點是屬於B分類。

當K值設定為5，也就是說要找到距離綠色點最近的5個點，由圖3-9發現，可找到2個藍色跟3個橘色的點與綠色點最近，所以判定這筆新的資料是屬於A分類。

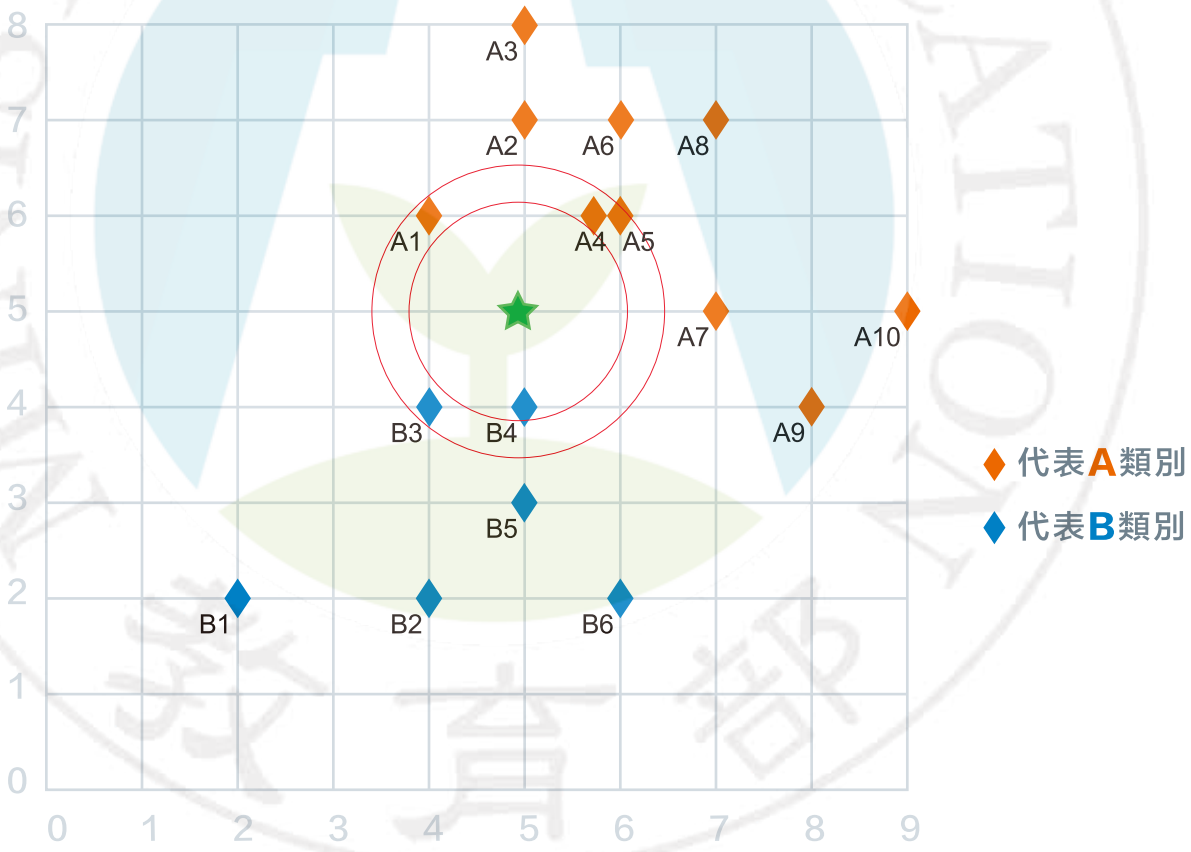


圖3-9 **橘色及藍色**菱形分別代表A、B類別的資料

從上面的例子中，可知不同的K值會導致不同的分類結果，那麼K值到底要設為多少才是最好的呢？其實並沒有一定最好的K值，而是要取決於資料以及應用的情境。但很肯定的是，如果以圖3-9為例，共有A、B兩種類別，則建議K值要儘量挑選「奇數」，以避免在判定類別時，出現數量相同的狀況。

課堂任務

任務一：依據圖3-9，列出A1至A10及B1至B6中所有的點跟綠色點(5,5)之歐幾里得距離。

表3-3 由學生填空，完成下表歐幾里得距離之計算

圖3-9 中綠色點(5,5)與其他各點之歐幾里得距離			
A 類別	距離	B 類別	距離
A1(4,6)		B1(2,2)	
A2(5,7)		B2(4,2)	
A3(5,8)		B3(4,4)	
A4(5.8,6)		B4(5,4)	
A5(6,6)		B5(5,3)	
A6(6,7)		B6(6,2)	
A7(7,5)			
A8(7,7)			
A9(8,4)			
A10(9,5)			

任務二：延續圖3-9，若K值為「9」，則綠色點將分屬於哪一個類別？

3-4 決策樹 Decision Tree

決策樹是一種條件式的分類器，可視為專門處理分類問題的樹狀結構。依據現有的資料建構出一棵決策樹，通常採用「由上而下」的方式，將整群資料從某個特徵開始，根據該特徵值分為數個子群，各個子群再根據某個特徵，將子群再分為更小的子群，直到子群內的資料都是同一個類別上為止。

為了能更加瞭解決策樹的運作方式，現在就讓我們化身為3C賣場的員工，針對消費者是否購買筆記型電腦的消費記錄，建構出一棵決策樹。消費記錄如表3-4。

表3-4 消費者是否購買筆記型電腦的消費記錄

年紀	收入	是否為學生	購買筆電與否
<=30	高	否	否
31…40	高	否	是
>40	中	否	是
>40	低	是	否
31…40	低	是	是
<=30	中	否	否
<=30	低	是	是
<=30	中	是	是
31…40	中	否	是
31…40	高	是	是
>40	中	是	否

由表3-4得知，除了可以知道消費者是否購買筆記型電腦外，亦可得知每筆消費記錄中的消費者年齡、收入狀況與是否仍保有學生身份等三種特徵值。那麼我們接著就必須思考，依據是否購買筆記型電腦的結果，到底要用哪一個特徵值來啟動建構決策樹的開端呢？而決策樹中不同階層又要以哪些特徵值來作為中間節點，才能建構出分類結果？

特徵值選擇

為了得知哪一種特徵值可以建構出較好的決策樹，比較簡單的做法是將每個特徵值都當作分類的條件，一一去建構決策樹，以列舉的方式將所有的特徵值排列組合出很多的決策樹，再進一步分析哪一種可以有最好的分類結果，但是這種做法是沒有效率的。因此，較佳的做法是先對資料的特徵值進行分析。同時能提供電腦做選擇時的優先順序條件，使能減少判斷條件的次數。那麼就能更有效率地將訓練資料清楚地分類出幾個類別，也讓決策樹能精簡易讀。

評斷特徵好壞的依據——「熵」

透過圖3-10來說明「熵」的意義：

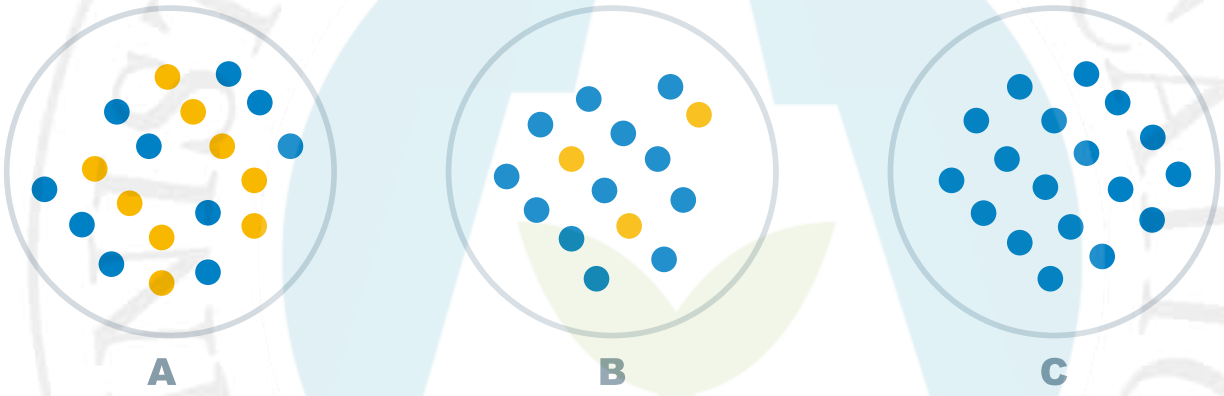


圖3-10 A、B及C三個資料集

從圖3-10可以看出，資料集C不需多做說明即可看出都屬於同一種類型，而資料集B則需要部分的說明來解釋資料集中3個不同類型資料的意義。但資料集A，則需要最多的說明來解釋在同一資料集中不同類型資料間的差異。由此得一結論：淨度越高的資料不需要過多資訊即可表示，而越雜亂的資料則需要較多的資訊才能夠完整呈現資料的意義。

在資訊理論（Information theory）中度量資訊而衍生出「熵」（Entropy）的概念，當所有資料的類型都是一致時，得到的熵為「0」，但如果分屬二類資料類型的數量各自對半時，熵為「1」。

「熵」的 計算方式如下：

$$Info\,(D)_2 = -\sum_{i=1}^{m} p_i \log_2 (p_i)$$

其中D代表某一個特徵值，而這個特徵值會有1到m種類別，p就是某個類別在這個特徵值中出現的機率，另外在取對數時一般會以2為底，源自於資訊的編碼大多是以0/1二進位的方式編碼。以表3-4為例，我們來計算一下「購買筆電與否」這個特徵值的熵。

表3-5 消費者是否購買筆記型電腦的消費記錄

購買筆電與否	出現次數	出現機率	熵
是	7	7/11	Info（原始資料）=I（7,4） $=-(\frac{7}{11})\log_2(\frac{7}{11})-(\frac{4}{11})\log_2(\frac{4}{11})$ $=0.425+0.531=0.956$
否	4	4/11	

※註：Info（購買筆電與否）可簡化表示方式為I（7,4），其中的參數7、4分別代表7筆有買筆電及4筆沒有買筆電的資料。

但如果要計算在「收入」這個特徵值下「購買筆電與否」的熵，計算時需要考量不同的收入類別出現的機率，則計算方式如下：

表3-6 計算以收入為特徵值下購買筆電與否的熵

收入	購買筆電	未購買筆電	熵
高	2	1	$Info_{收入}$（原始資料） $=\frac{3}{11}I(2,1)+\frac{5}{11}I(3,2)+\frac{3}{11}I(2,1)$ $=\frac{3}{11}(-(\frac{2}{3})\log_2(\frac{2}{3})-(\frac{1}{3})\log_2(\frac{1}{3}))$ $+\frac{5}{11}(-(\frac{3}{5})\log_2(\frac{3}{5})-(\frac{2}{5})\log_2(\frac{2}{5}))$ $+\frac{3}{11}(-(\frac{2}{3})\log_2(\frac{2}{3})-(\frac{1}{3})\log_2(\frac{1}{3}))$ $=0.250+0.441+0.250=0.941$
中	3	2	
低	2	1	

同樣的，我們也可以計算在「年紀」這個特徵值下「購買筆電與否」的熵：

表3-7 計算以年紀為特徵值下購買筆電與否的熵

年紀	購買筆電	未購買筆電	熵
<=30	2	2	$\text{Info}_{\text{年紀}}$（原始資料） $=\frac{4}{11}\text{I}(2,2)+\frac{4}{11}\text{I}(4,0)+\frac{3}{11}\text{I}(2,1)$ $=\frac{4}{11}(-(\frac{2}{4})\log_2(\frac{2}{4})-(\frac{2}{4})\log_2(\frac{2}{4}))$ $+\frac{4}{11}(-(\frac{4}{4})\log_2(\frac{4}{4})-(\frac{0}{4})\log_2(\frac{0}{4}))$ $+\frac{3}{11}(-(\frac{1}{3})\log_2(\frac{1}{3})-(\frac{2}{3})\log_2(\frac{2}{3}))$ $=0.364+0+0.250=0.614$
31...40	4	0	
>40	1	2	

最後，使用「是否為學生」做為特徵值，計算「是否購買筆電」的熵：

表3-8 計算以是否為學生為特徵值下購買筆電與否的熵

學生	購買筆電	未購買筆電	熵
是	4	1	$\text{Info}_{\text{是否為學生}}$（原始資料） $=\frac{5}{11}\text{I}(4,1)+\frac{6}{11}\text{I}(3,3)$ $=\frac{5}{11}(-(\frac{4}{5})\log_2(\frac{4}{5})-(\frac{1}{5})\log_2(\frac{1}{5}))$ $+\frac{6}{11}(-(\frac{3}{6})\log_2(\frac{3}{6})-(\frac{3}{6})\log_2(\frac{3}{6}))$ $=0.408+0.545=0.953$
否	3	3	

經過上述的計算，表3-5若單看「是否購買筆電」的熵為0.956；在「收入」這個特徵值獲得「是否購買筆電」的熵為0.941；在「年紀」這個特徵值獲得「是否購買筆電」的熵為0.614；而在「是否為學生」這個特徵值得到「是否購買筆電」的熵為0.953。

資訊獲利

計算完熵後便可利用其計算結果來探討「資訊獲利」（Information Gain），所謂的資訊獲利就是用來衡量特徵值於分類資料的能力。延續表3-4的範例，各項特徵值的資訊獲利計算方式，就是將「是否購買筆電」的熵扣掉在「某個特徵值下是否購買筆電」的熵。根據不同特徵值得到的資訊獲利越高，表示該特徵值內資料的凌亂程度越小，用來分類資料效果越佳；反之，若資訊獲利越低，表示該特徵值內資料的凌亂程度越大，用來分類資料效果較差。

依據表3-5至表3-8的範例得到各項特徵值的資訊獲利如下：

- **特徵值「收入」的資訊獲利 = 0.956 − 0.941 = 0.015**
- **特徵值「年紀」的資訊獲利 = 0.956 − 0.614 = 0.342**
- **特徵值「是否為學生」的資訊獲利 = 0.956 − 0.953 = 0.003**

由上述結果可知，若欲將表3-9之資料以決策樹方式呈現，則優先於根節點選擇「年紀」這個特徵值為判斷條件，可以帶來較佳的分類結果。

建構決策樹

在挑選完根節點的特徵值作為分類的判斷條件後，接著要進入第二層的分支節點，只要繼續反覆透過計算資訊獲利的方式去決定下一個特徵值，直到每一筆資料都有對應到決策樹上的每個條件，即可完成決策樹之建構。

為了繼續建構決策樹，依據根節點特徵值的選擇，將原始的表3-4切割為表3-9至表3-11。

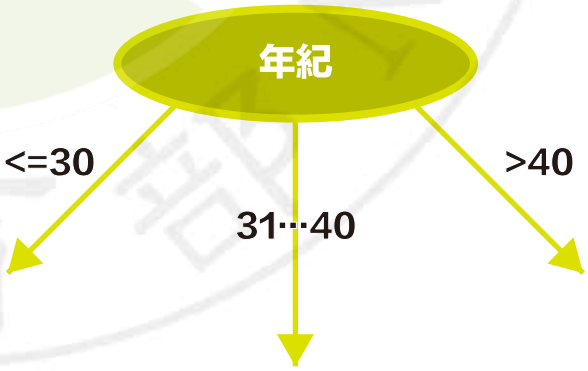


圖3-11 以「年紀」特徵值做為決策樹的根節點

表3-9 將表3-4中 **年紀<=30歲** 的資料擷取出來

年紀	收入	是否為學生	購買筆電與否
<=30	高	否	否
<=30	中	否	否
<=30	低	是	是
<=30	中	是	是

表3-10 將表3-4中 **年紀介於31歲到40歲** 的資料擷取出來

年紀	收入	是否為學生	購買筆電與否
31…40	高	否	是
31…40	低	是	是
31…40	中	否	是
31…40	高	是	是

表3-11 將表3-4中 **年紀>40歲** 的資料擷取出來

年紀	收入	是否為學生	購買筆電與否
>40	中	否	是
>40	低	是	否
>40	中	是	否

以「年紀<=30」的這個分支節點來看，如何在「收入」及「是否為學生」這兩種特徵值間做挑選，並作為下一層分類的判斷依據。選擇的依據，仍然要透過資訊獲利的多寡決定，其計算方式如下：

在特徵值「年紀<＝30」中「購買筆電與否」的熵為：

表3-12 計算 **年紀<=30歲** 的資料中購買筆電與否的熵

購買筆電與否	出現次數	出現機率	熵
是	2	2/4	$\text{Info}(\text{原始資料}) = \text{I}(2,2)$ $= -(\frac{2}{4})\log_2(\frac{2}{4}) - (\frac{2}{4})\log_2(\frac{2}{4})$ $= 0.5 + 0.5 = 1$
否	2	2/4	

再細看「年紀<＝30」的4筆資料中，特徵值「收入」的熵為：

表3-13 計算 **年紀<=30歲** 的資料中以收入為特徵值購買筆電與否的熵

收入	購買筆電	未購買筆電	熵
高	0	1	$\text{Info}_{\text{收入}}(\text{原始資料})$ $= \frac{1}{4}\text{I}(0,1) + \frac{2}{4}\text{I}(1,1) + \frac{1}{4}\text{I}(1,0)$ $= \frac{1}{4}(-(\frac{0}{1})\log_2(\frac{0}{1}) - (\frac{1}{1})\log_2(\frac{1}{1}))$ $+ \frac{2}{4}(-(\frac{1}{2})\log_2(\frac{1}{2}) - (\frac{1}{2})\log_2(\frac{1}{2}))$ $+ \frac{1}{4}(-(\frac{1}{1})\log_2(\frac{1}{1}) - (\frac{0}{1})\log_2(\frac{0}{1}))$ $= 0 + 0.5 + 0 = 0.5$
中	1	1	
低	1	0	

接著看「年紀<=30」的4筆資料中，特徵值「是否為學生」的熵為：

表3-14 **計算 年紀<=30歲 的資料中以是否為學生之特徵值購買筆電與否的熵**

學生	購買筆電	未購買筆電	熵
是	2	0	$\text{Info}_{\text{是否為學生}}(\text{原始資料}) = \frac{2}{4}\text{I}(2,0) + \frac{2}{4}\text{I}(0,2)$ $= \frac{2}{4}\left(-\left(\frac{2}{2}\right)\log_2\left(\frac{2}{2}\right)-\left(\frac{0}{2}\right)\log_2\left(\frac{0}{2}\right)\right)$ $+ \frac{2}{4}\left(-\left(\frac{0}{2}\right)\log_2\left(\frac{0}{2}\right)-\left(\frac{2}{2}\right)\log_2\left(\frac{2}{2}\right)\right)$ $= 0 + 0 = 0$
否	0	2	

經由計算，在特徵值「年紀<=30」的4筆資料中，再以不同特徵值計算，可得資訊獲利如下：

- 特徵值「收入」的資訊獲利 = 1 − 0.5 = 0.5
- 特徵值「是否為學生」的資訊獲利 = 1 − 0 = 1

由計算得到之資訊獲利可知，決策樹在「年紀<=30」的下一支點將以「是否為學生」為判斷條件，繼續建構決策樹如圖3-12。

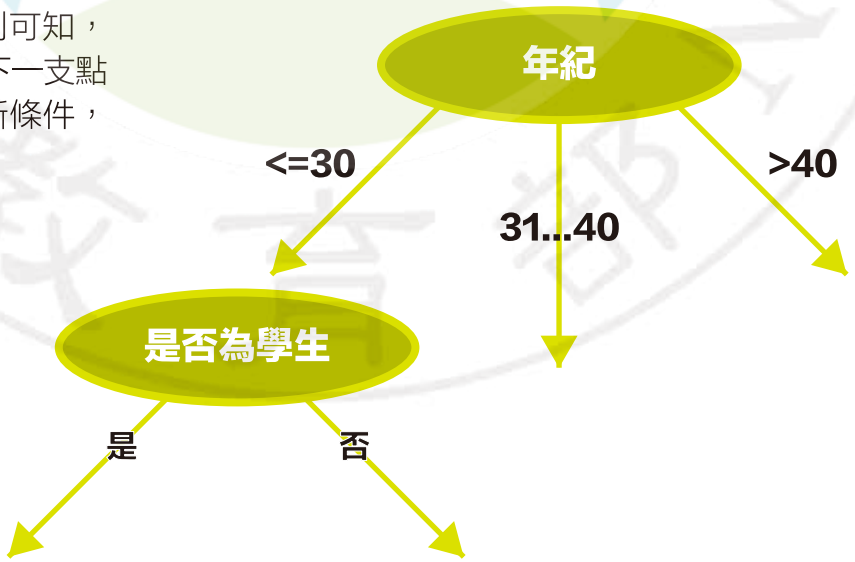


圖3-12 「年紀<=30」的根節點下，以特徵值「是否為學生」建構決策樹

經過第二層特徵值的選擇後，原先以「年紀<=30」的資料表分割出來的4筆資料，現在又可以被「是否為學生」的特徵值分割為表3-15與表3-16。

表3-15 此表為 **年紀<=30且不是學生** 的資料表

年紀	收入	是否為學生	購買筆電與否
<=30	高	否	否
<=30	中	否	否

表3-16 此表為 **年紀<=30且是學生** 的資料表

年紀	收入	是否為學生	購買筆電與否
<=30	低	是	是
<=30	中	是	是

由於在特徵值「是否為學生」加入後，根據分類結果可以觀察到在年紀小於等於30歲的資料中，只要是學生都會購買筆電；若不是學生，則都不會購買筆電，最後建構出的決策樹如圖3-13。

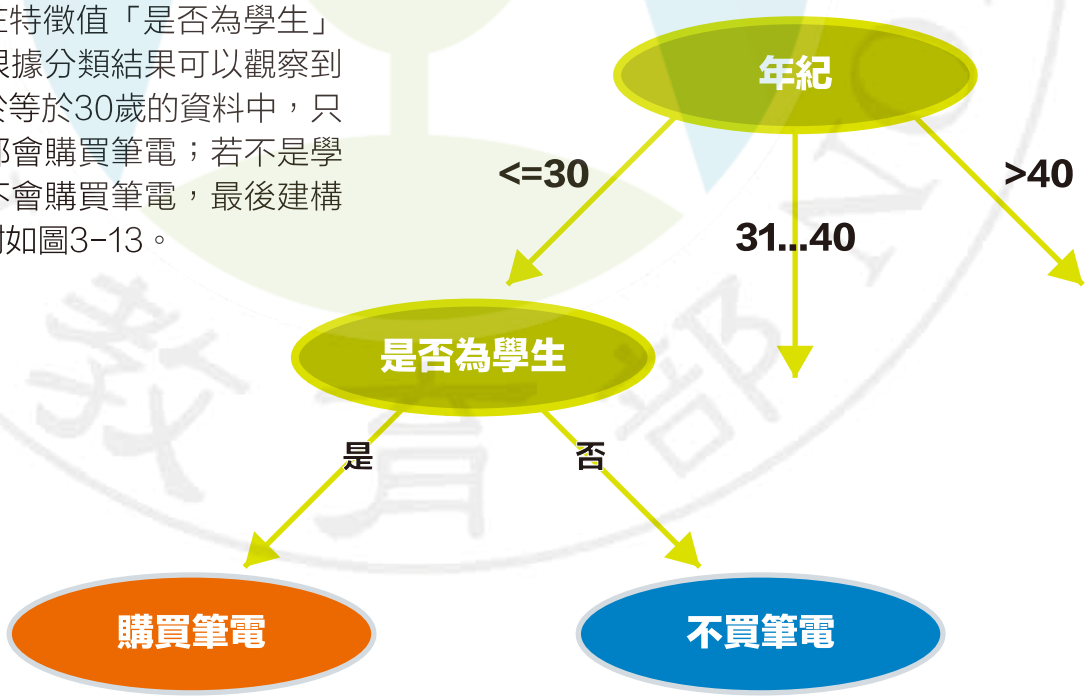


圖3-13 針對「年紀<=30」分支節點所建構之決策樹

接著，將表3-4中「年紀31…40」的資料擷取出來如下表3-10，可以明顯觀察到在年紀介於31至40歲的資料皆會購買筆電，由於都屬於同一類別，不需再挑選其他特徵值，即可建構出決策樹如下圖3-14。

表3-10 將表3-4中 **年紀介於31歲到40歲** 的資料擷取出來

年紀	收入	是否為學生	購買筆電與否
31…40	高	否	是
31…40	低	是	是
31…40	中	否	是
31…40	高	是	是

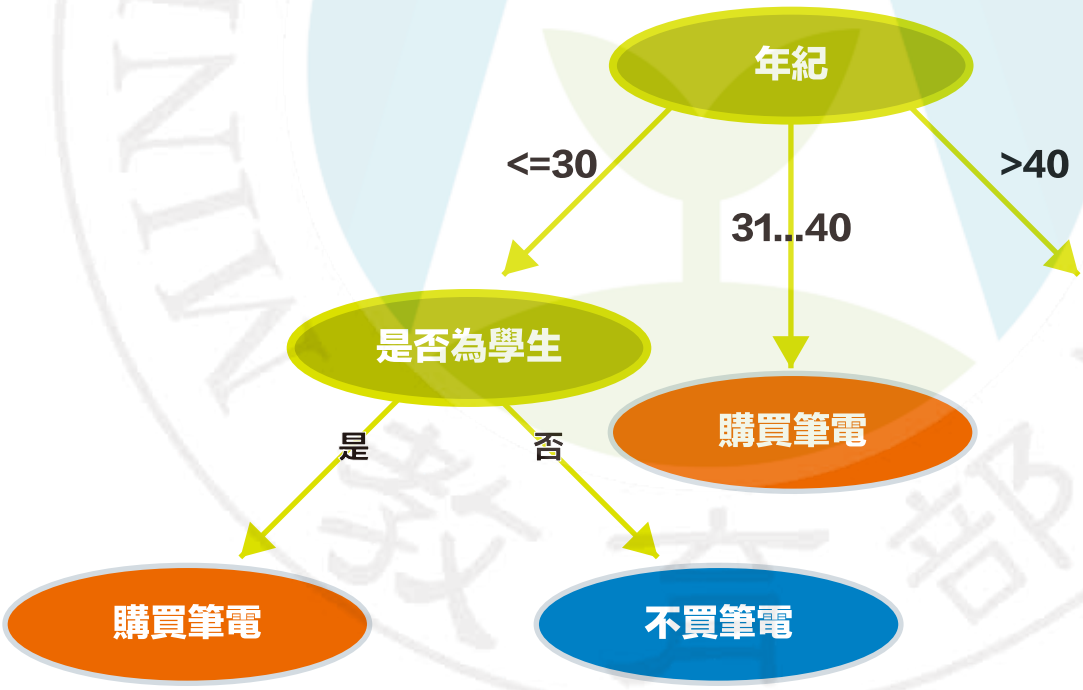


圖3-14 針對「年紀31…40」分支節點下建構之決策樹

課堂任務

任務一：延續先前範例，請計算出下表3-11「年紀>40」的資料表中，分別以「收入」及「是否為學生」作為分類條件，其資訊獲利為多少？

表3-11 將表3-4中 **年紀>40歲** 的資料擷取出來

年紀	收入	是否為學生	購買筆電與否
>40	中	否	是
>40	低	是	否
>40	中	是	否

在「年紀>40」的特徵值中，「購買筆電與否」的熵為：

表3-17 計算 **年紀>40歲** 的資料中購買筆電與否的熵

購買筆電與否	出現次數	出現機率	熵
是	1	1/3	Info（原始資料）= I（1,2）
否	2	2/3	$= -\left(\frac{\square}{\square}\right)\log_2\left(\frac{\square}{\square}\right) - \left(\frac{\square}{\square}\right)\log_2\left(\frac{\square}{\square}\right)$
			=

再細看「年紀>40」的3筆資料中，特徵值「收入」的熵為：

表3-18 計算 **年紀>40歲** 的資料中以收入為特徵值購買筆電與否的熵

收入	購買筆電	未購買筆電	熵
中	1	1	$Info_{收入}$（原始資料）=
低	0	1	

接著看「年紀>40」的3筆資料中，特徵值「是否為學生」的熵為：

表3-19 計算 **年紀>40歲** 的資料中以是否為學生之特徵值購買筆電與否的熵

是否為學生	購買筆電	未購買筆電	熵
是	0	2	$Info_{是否為學生}$（原始資料）=
否	1	1	

課堂任務

經由計算，在「年紀>40」特徵值的3筆資料中，再以不同特徵值計算可得資訊獲利如下：

- 特徵值「收入」的資訊獲利 = _____
- 特徵值「是否為學生」的資訊獲利 = _____

任務二：根據任務一的結果，建構出「年紀>40」分支節點下的決策樹。

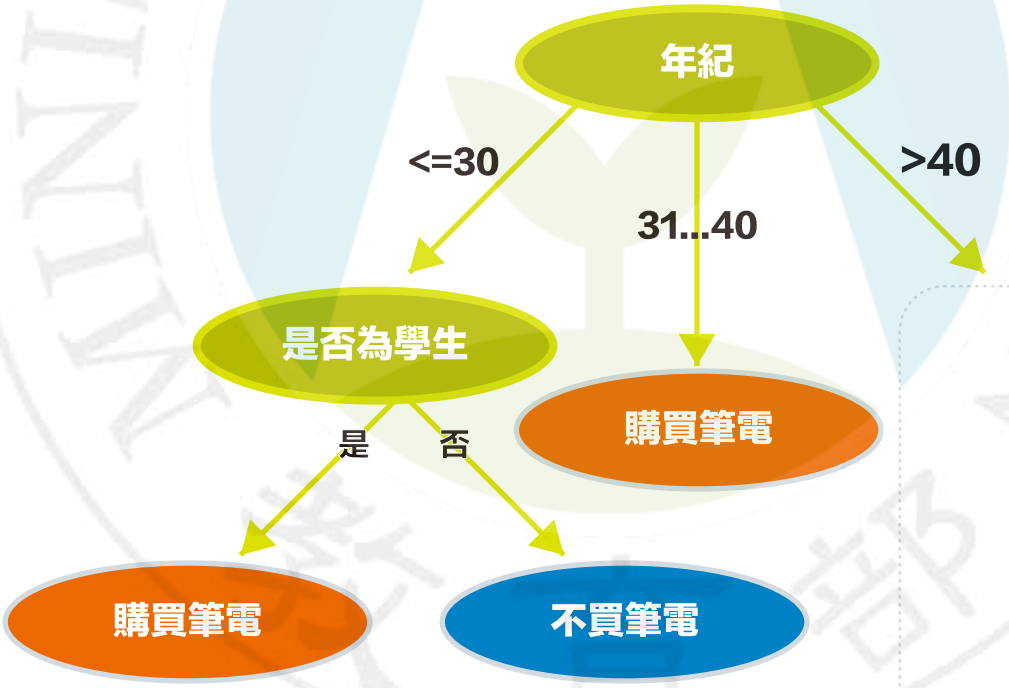


圖3-15 針對「年紀>40」分支節點下建構之決策樹

Note

圖片來源：

圖3-1：鳥，來源網址 http://gg.gg/dlmzu，擷取自Pixabay，免費下載可支援商業用途請無須署名。

圖3-2：狗，來源網址 https://pxhere.com/en/photo/947471，CC0授權使用。

圖3-3：台南二中109級美術班學生繪製並授權使用。

圖3-4：台南二中109級美術班學生繪製並授權使用。

圖3-5：貓，來源網址 http://gg.gg/dln17，免費下載且無須署名。

第四章

非監督式學習 Unsupervised Learning

我們常聽到一句話「人類是群居的動物」，
每個人都應是獨立的個體，卻又因為什麼緣故而聚集在一起呢？
這個有趣的分群問題，
也是「非監督式學習」中需要思考的重要概念，
現在就跟著我們一起進入非監督式學習吧！

4-1 非監督式學習簡介

這個章節中我們要談到「非監督式學習」，藉由前一章節的介紹，大家應該都可以清楚知道所謂的監督式學習，是指電腦在學習的過程中每一筆資料都有對應的結果，也就是一種有「標準答案」的學習。那麼，單就字義上來說「非監督式學習」不就是一種「沒有標準答案」的學習嗎？接著看下去就知道囉！

如果要用再更淺顯易懂的方式來區分監督式學習與非監督式學習的概念，監督式學習就好像我們如果要訓練電腦能夠分辨蘋果跟梨子，在訓練的過程中就必須給予大量蘋果跟梨子的圖片，而且還要清楚的告訴電腦每一張圖片所屬的水果種類是什麼，經過這樣的訓練後，當我們再給定一張新的圖片時，電腦就能夠透過先前的訓練做出判別。而以相同的例子來說，非監督式學習則是給予電腦大量的圖片時，並沒有給定每一張圖片是何種水果，反而是希望電腦能夠在大量的圖片中自動去解析並且歸納出規則，接著自動分辨出每張圖片所屬的類別。

這樣說起來，有沒有覺得「非監督式學習」很神奇？居然可以透過計算資料間相互的關係就對資料進行分析或標記。這也是非監督式學習的最大特色，因為沒有事先給定的類別標記，只能透過資料間彼此相互比對，計算出關連性，所以我們很常藉由非監督式學習來處理「分群（Clustering）」問題。利用現有資料的特徵分成若干個不同的群體，每個群體中資料的特徵會是相近的。為了讓相近的資料可以聚集在一起，通常還是會將資料的特徵值數值化，再透過計算資料間的相近程度(稱之為「距離」)來進行分群。而計算「距離」的方法有許多種，像前一章節提到的「歐幾里得距離」就是一種常見的距離計算方式。以下我們將介紹兩種常見的分群演算法，分別是「K-means演算法」與「階層式分群法（Hierarchical Clustering）」。

4-2 K-means演算法

俗話說「物以類聚」，分群的概念簡單來說就是將相近的資料彼此分在同一群體，接下來我們將以圖4-1來介紹K-means演算法的運作原理。在下圖4-1中，每一個點都視為一筆資料，而座標平面上將以X軸與Y軸來代表該筆資料的2種特徵值。

因此可以得到圖上共有16個點，座標值(X,Y)分別是(1,3)、(1,4)、(2,2)、(2,5)、(2,6)、(3,2)、(3,3)、(3,8)、(4,4)、(4,6)、(5,0)、(5,5)、(6,2)、(6,6)、(7,2)及(7,4)。

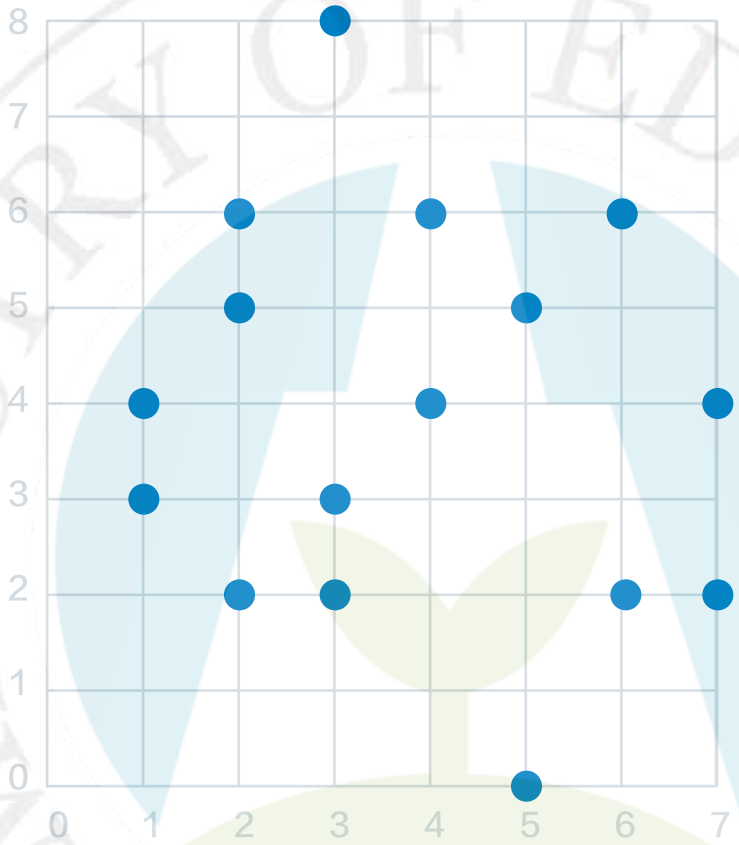


圖4-1 座標平面上共有16個點，X軸與Y軸代表2種特徵值

K-means演算法的執行步驟如下：

步驟1： K-means演算法的第一步，需要先選定「K」的值，代表接下來要將資料分為K群。在此假設K為2。

步驟2： 上一步驟已經決定出K值，接著就任意在座標平面上指定K個點作為初始的分群中心點，在此隨機擇定以(2,2)及(6,6)此2點作為分群中心。

步驟3： 既已擇定K個(K=2)分群中心，座標上的各點可以分別計算與各分群中心間的「距離」，來決定歸屬於哪一個群體。在此距離計算方式採取「歐幾里得距離」，前一章節已有介紹，在此將直接進行計算各點與各分群中心的距離，如下表4-1。

表4-1 計算座標平面上每筆資料與分群中心之距離

資料座標 (X,Y)	與(2,2)距離	與(6,6)距離	歸屬之分群中心
(1,3)	$\sqrt{(1-2)^2+(3-2)^2}=1.44$	$\sqrt{(1-6)^2+(3-6)^2}=5.83$	(2,2)
(1,4)	$\sqrt{((1-2)^2+(4-2)^2}=2.24$	$\sqrt{(1-6)^2+(4-6)^2}=5.39$	(2,2)
(2,5)	$\sqrt{(2-2)^2+(5-2)^2}=3$	$\sqrt{(2-6)^2+(5-6)^2}=4.12$	(2,2)
(2,6)	$\sqrt{(2-2)^2+(6-2)^2}=4$	$\sqrt{(2-6)^2+(6-6)^2}=4$	(2,2)或(6,6)皆可
(3,2)	$\sqrt{(3-2)^2+(2-2)^2}=1$	$\sqrt{(3-6)^2+(2-6)^2}=5$	(2,2)
(3,3)	$\sqrt{(3-2)^2+(3-2)^2}=1.44$	$\sqrt{(3-6)^2+(3-6)^2}=4.24$	(2,2)
(3,8)	$\sqrt{(3-2)^2+(8-2)^2}=6.08$	$\sqrt{(3-6)^2+(8-6)^2}=3.61$	(6,6)
(4,4)	$\sqrt{(4-2)^2+(4-2)^2}=2.83$	$\sqrt{(4-6)^2+(4-6)^2}=2.83$	(2,2)或(6,6)皆可
(4,6)	$\sqrt{(4-2)^2+(6-2)^2}=4.47$	$\sqrt{(4-6)^2+(6-6)^2}=2$	(6,6)
(5,0)	$\sqrt{(5-2)^2+(0-2)^2}=3.61$	$\sqrt{(5-6)^2+(0-6)^2}=6.08$	(2,2)
(5,5)	$\sqrt{(5-2)^2+(5-2)^2}=4.24$	$\sqrt{(5-6)^2+(5-6)^2}=1.44$	(6,6)
(6,2)	$\sqrt{(6-2)^2+(2-2)^2}=4$	$\sqrt{(6-6)^2+(2-6)^2}=4$	(2,2)或(6,6)皆可
(7,2)	$\sqrt{(7-2)^2+(2-2)^2}=5$	$\sqrt{(7-6)^2+(2-6)^2}=4.12$	(6,6)
(7,4)	$\sqrt{(7-2)^2+(4-2)^2}=5.39$	$\sqrt{(7-6)^2+(4-6)^2}=2.24$	(6,6)

註：在表4-1中可觀察到，(2,6)、(4,4)及(6,2)等3點與分群中心(2,2)及(6,6)計算後之距離相同，面臨此狀況時，可隨機將該筆資料分入任一分群中心所屬之群體中，在此假設(2,6)及(4,4)分於分群中心(2,2)的群體，而(6,2)分於分群中心(6,6)的群體。

步驟4：經過上一步驟計算每筆資料與分群中心之距離後，即可針對距離之遠近來判別該筆資料所屬的分群結果，如下圖4-2，其中為辨識方便，以分群中心(2,2)之分群資料以「橘色」標記，以分群中心(6,6)之分群資料則以「綠色」標記。

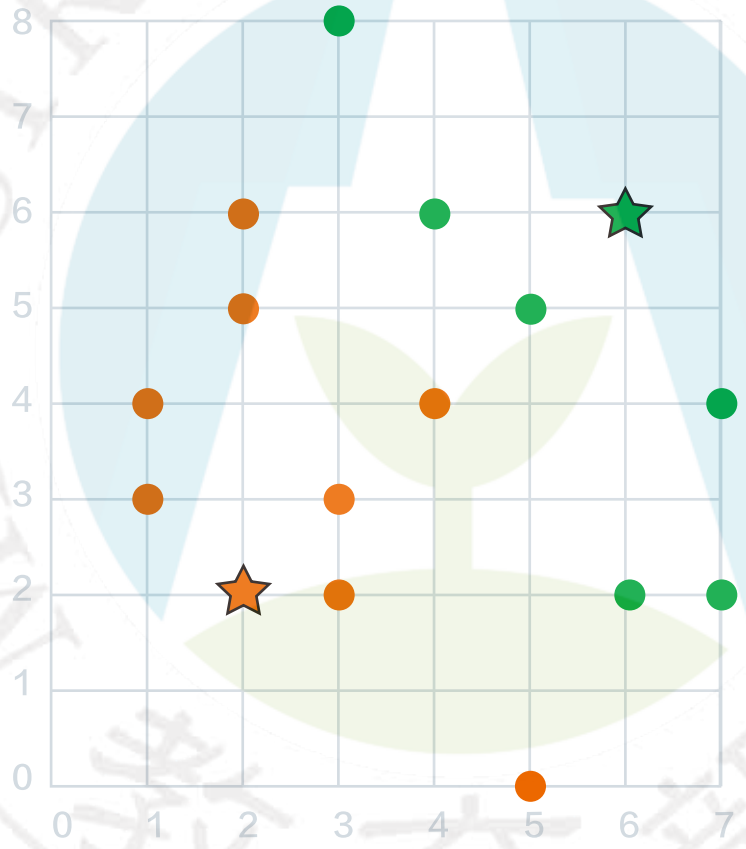


圖4-2 K-means演算法以(2,2)及(6,6)為分群中心得到之分群結果

步驟5：依據步驟4之分群結果重新計算分群中心，在此常以各點座標之「算數平均值」作為新的分群中心之座標值，因此可得新的分群中心為：(2.56,3.22)與(5.43,4.71)，計算方式如下表4-2。

表4-2 以算數平均數計算新分群中心之X、Y值

	(1,3)、(1,4)、(2,2)、(2,5)、(2,6)、(3,2)、(3,3)、(4,4)、(5,0)	(3,8)、(4,6)、(5,5)、(6,2)、(6,6)、(7,2)、(7,4)
新分群中心之X值	$\frac{1+1+2+2+2+3+3+4+5}{9}=2.56$	$\frac{3+4+5+6+6+7+7}{7}=5.43$
新分群中心之Y值	$\frac{3+4+2+5+6+2+3+4+0}{9}=3.22$	$\frac{8+6+5+2+6+2+4}{7}=4.71$

步驟6：經過上述步驟，可以根據分群結果得到新的分群中心，接著重複步驟3到5，座標軸上的各筆資料又會以新的分群中心重新計算距離，再以計算後的結果來進行分群，如此反覆的程序直到分群的結果不再變動，且各分群的中心也都不改變時，即代表完成分群過程。

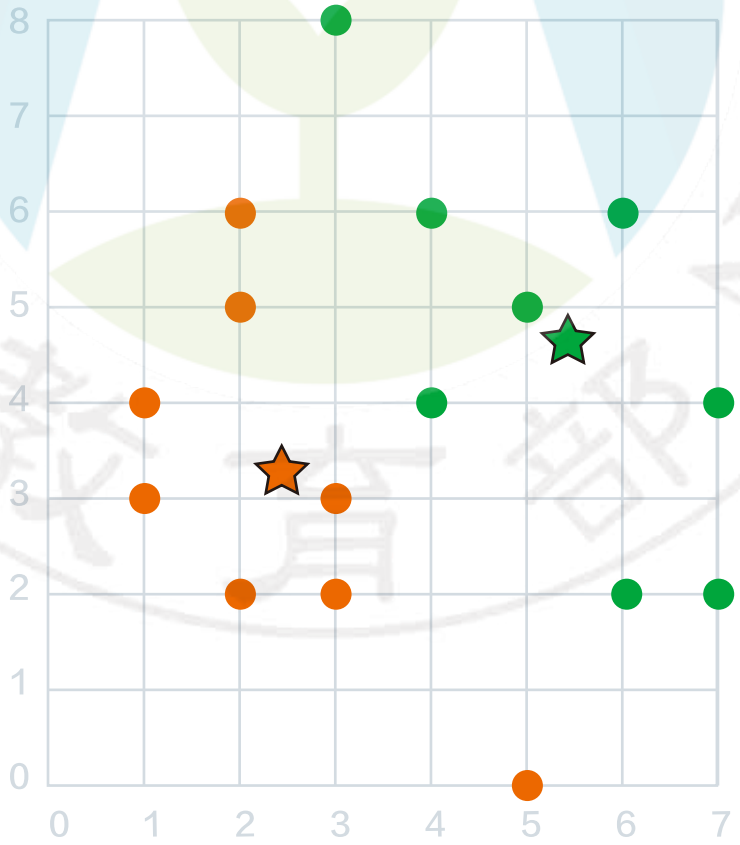


圖4-3 以(2.56,3.22)與(5.43,4.71)為新分群中心後得到的分群結果

根據圖4-3的結果可以發現，經過第一次分群後得到的新分群中心(2.56,3.22)與(5.43,4.71)，倘若再以此進行分群，資料中的(4,4)會由原本的群體中被分到另一群，也可藉此看出在K-means演算法運作的過程中，依據每次分群結果得到的新分群中心，若再據以進行分群後，各資料所歸屬的群體可能會發生轉變。

課堂任務

任務一：依據圖4-3以(2.56,3.22)與(5.43,4.71)為新分群中心後得到的分群結果，請分別找出標記「橘色」及「綠色」兩個分群之新的分群中心(X,Y)為何？

	標記橘色資料之分群中心	標記綠色資料之分群中心
新分群中心之X值		
新分群中心之Y值		

藉由上述課堂任務得到的新分群中心，按照K-means演算法的執行步驟，再各自與每筆資料計算距離進行分群，可發現即便有了新的分群中心，但所有資料分群結果已經不再改變，如下圖4-4。當分群中心不再變動，各資料所屬的分群結果也不再變化，則代表分群已經完成。

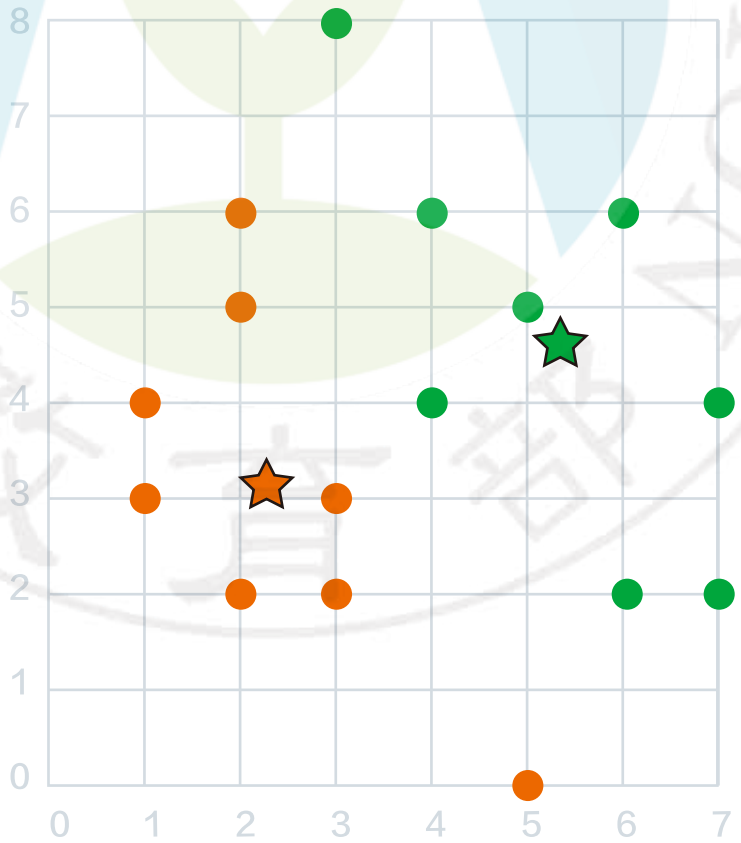


圖4-4 經過課堂任務一計算新分群中心後可得到的分群結果

課堂任務

任務二：以圖4-1為依據，倘若將K值訂為「3」，且隨機指定平面座標中的(1,3)、(3,8)及(7,2)為分群中心（下圖4-5中已用星號標註），請在下圖中將執行K-means演算法「第一次」分群後屬於同一群體的資料圈起來。

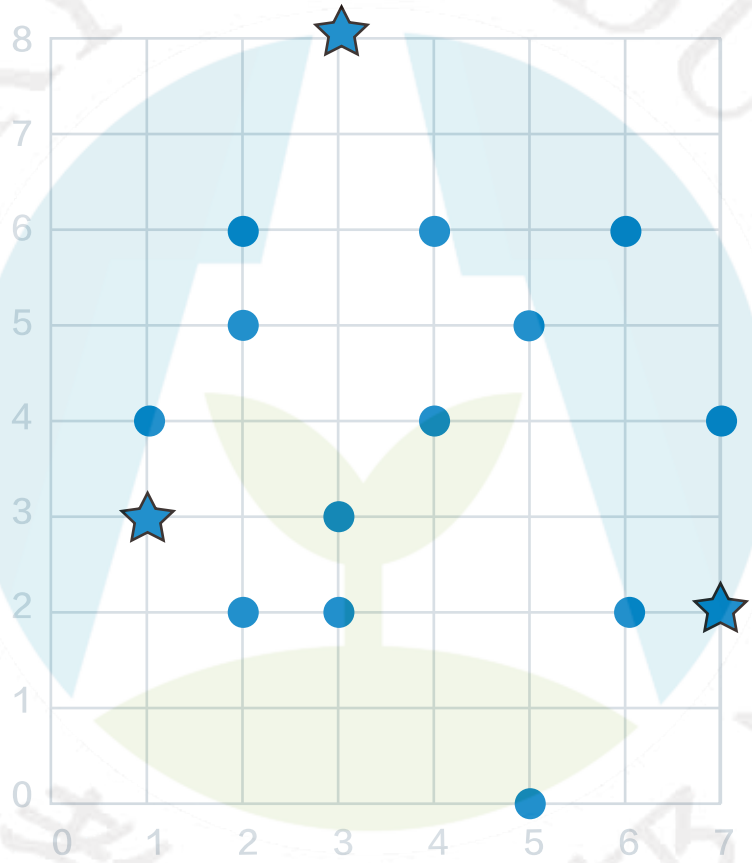


圖4-5 利用K-means演算法並將K值訂為3

利用K-means演算法來進行分群時，有時候可能無法展現良好的分群效果，尤其碰到資料分布狀況比較特別的時候如下圖4-6。當資料分布呈現環狀（或甜甜圈狀）時，會造成分群中心不斷地在環上移動，而出現不會停止（收斂）的狀況。

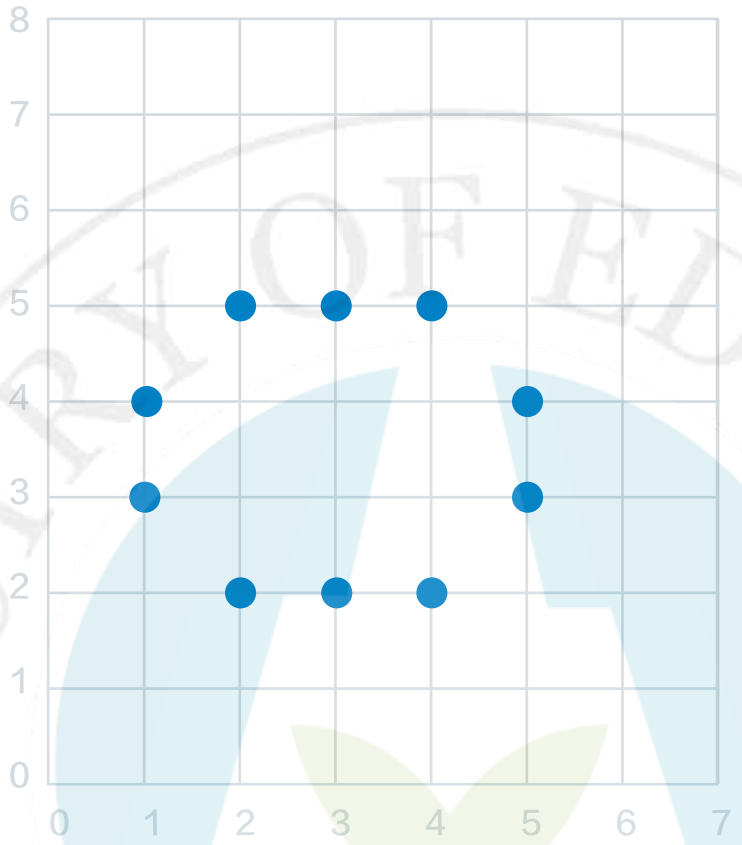


圖4-6 環狀分布的資料易造成分群中心不斷在環上移動

4-3 階層式分群法

看完前一節介紹的K-means演算法後，瞭解到K-means演算法通常需要事先給定分群的數量，也就是K值，但有時候我們難免會面臨難以決定分群數量的狀況。假設我們將資料分群時，想要分成2群，又想要分為3群，如此三心二意下又覺得分為4群比較好。如果這時候我們用K-means演算法的話，分成3群、4群、5群，就必須做三次不同的操作，而且這三次分群的結果，彼此之間的分群不一定有其關聯性。這時透過階層式分群法（Hierarchical Clustering）就可以彈性地依據群內的狀況決定群數，而在此所說的「階層」代表的是分群數量的階層。單就資料來看，我們可以將所有的資料包含在一起變成同一個分群，也可以個別將每一筆資料視為一個群體，端看分群者對群內資料點密集程度的定義而定。

其中，階層式分群法又可分為兩種：聚合法和分裂法。

- **聚合法（Agglomerative）：**「異中求同」，將所有資料先視為個別不同的群，再從這些群之中，找到最相似的2群，將其結合成一個新的群。合併為一新群後，再與其他群比較，一樣找最相似的2群，再結合成一個新群，如此不斷地重複聚合步驟，直到聚合後的群數達到目標群數。如果以階層的方式來看，是一種「由下而上」的執行策略。

- **分裂法（Divisive）：**「同中求異」，將所有資料先視為同一群，再依據群內的相異，分裂成2群。接著，再從2群中，找群內相異度最高的那群，再分裂一次，變成3群…，重複操作直到分出來的群數達到目標群數。如果以階層的方式來看，是一種「由上而下」的執行策略。

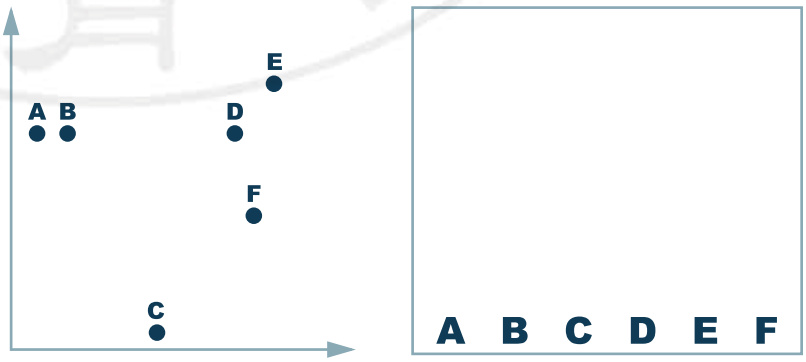
在上述2種方法中，不管分裂或聚合，其中所稱的「相似」或「相異」，指的就是資料在特徵值上的差異，如果以數值化的方式轉換到座標平面上，指的就是資料點之間的距離。由於聚合法與分裂法作法相似，前者「由下而上」而後者「由上而下」，因此在此我們先以聚合法為例進行說明。

如果以人類社會來比擬，最早期的社會每個人都是獨立個體，為了生存人們要互助合作，選擇上總是找與自己最近的人先開始，形成各個聚落後，不同聚落間有可能會結盟合作，但會產生結盟的條件就是彼此在距離還是利益等因素要夠接近。當聚落結盟不斷推展下去，最後就會形成各個不同的聯邦甚至國家。

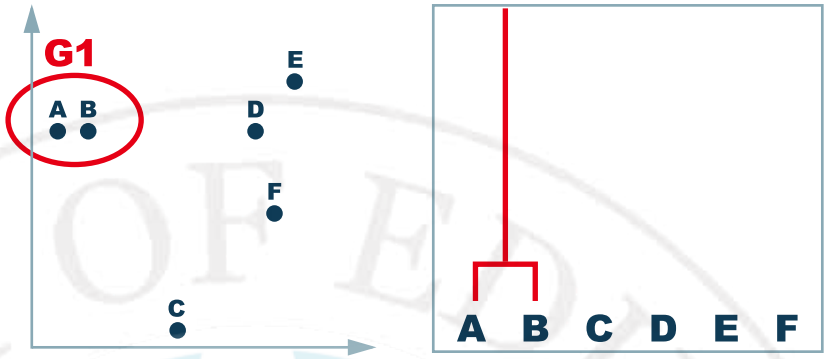
聚合法階層式分群法的操作：

1. **將各個資料點先視為個別的「群」。**
2. **比較各個群之間的距離，找出距離最短的２個群。**
3. **將其合併變成一個新群。**
4. **不斷重複直到群的數量符合我們所要求的數目。**

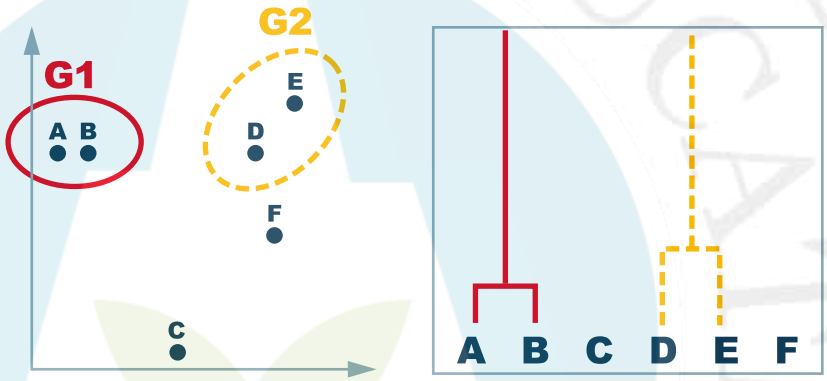
假設現在有6筆資料，分別標記A、B、C、D、E及F，且每筆資料都是一個群。首先找距離最近的2個群，在此例為A、B。



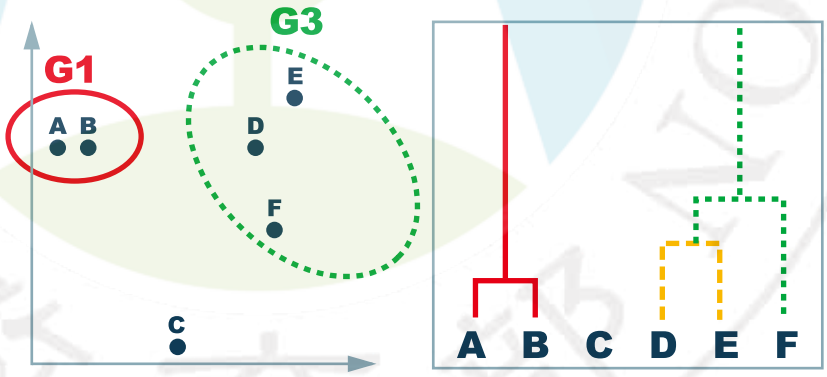
將A與B結合為新的一群G1，這樣，就將這些點分成5群了，其中有4群還是單獨的點。接著，再繼續找距離最近的2個群，依此範例應為D與E。



同樣地，將D與E結合為新的一群G2。現在，我們已經分成4群了。接下來，會需要思考一個問題：群和群之間的距離要怎麼定義呢？倘若我們先以直覺去操作，之後再討論分群間的距離計算方式。因此在直覺上，最近的應該是F和G2。



接著將F與G2合而為新的群G3。這時，這些資料已經被分為3群了。



按照上述範例，如果我們還想要再繼續聚合變成兩群，那勢必要面對群和群之間距離的定義，因為使用不同的定義，就可能造成不同的結果。

由於數值化的關係，我們通常將資料視為座標平面上的點，當群還只是單個點的時侯，群與群的距離，可以看成是點和點之間的距離，運用前面章節講述過的歐幾里得距離就能夠明確計算出來。但是當一個分群內有多個點的時候，其距離的定義方式就更加複雜。

通常要定義不同群體間的距離，還是會利用原有的資料點與點之間距離定義，在此我們將介紹三種常用的群間距離計算方式，分別是單一連結single-linkage、完整連結complete-linkage、平均連結average-linkage。舉例說明如下：若A、B為2點，則d(A,B)表示為A點到B點之間的距離。若G1、G2為2個群，則d(G1,G2)表示為G1與G2的群間距離。而接下來，將分別說明三種不同的群間距離計算方式，延續上述範例，已知座標平面上的6個點，先依據距離遠近，將A、B聚合為G1；D、E聚合為G2；而將G2加入F後形成G3。

▶ **單一連結（Single-linkage）：**

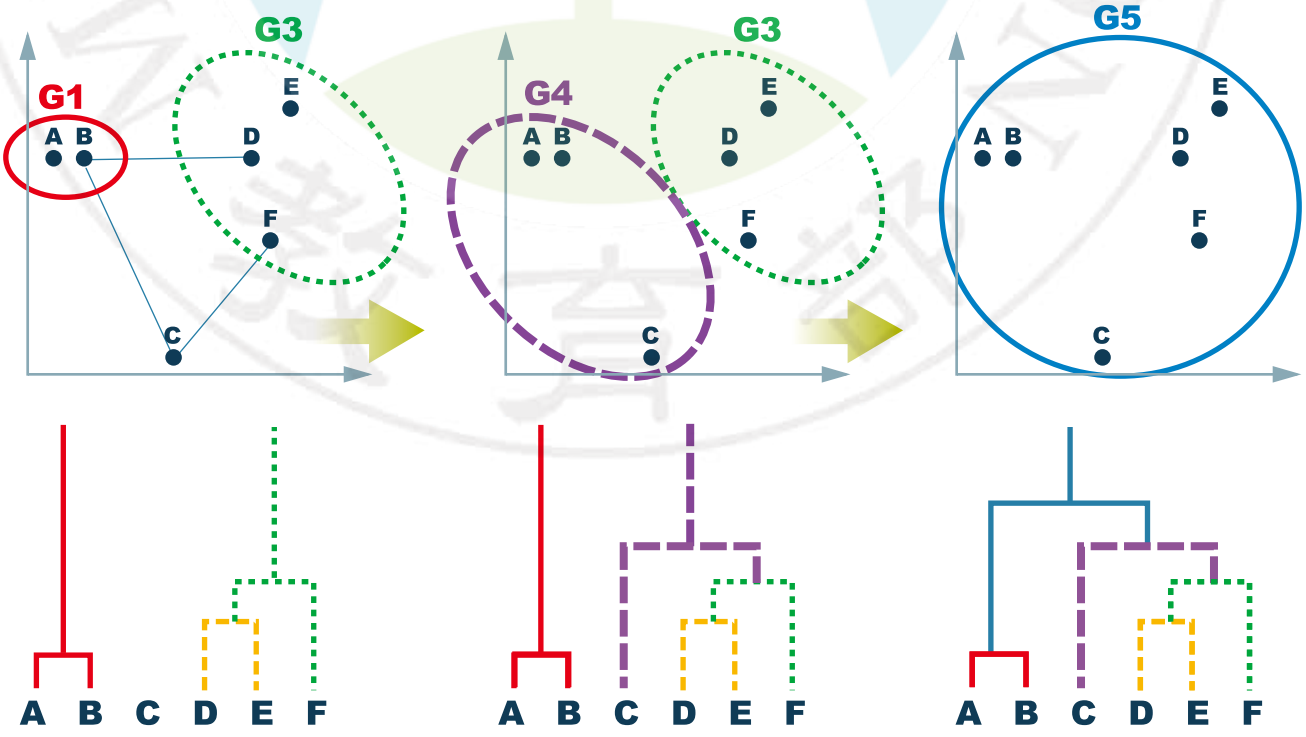
在分屬不同的兩群中，選擇最接近的兩點之距離，即代表兩群間的距離。因此在群與群間進行聚合時，依據此值最小者做為選取下一步結合之對象。

$$d(G1,G2)=\min_{A\subset G1,B\in G2} d(A,B)$$

G1、G3與C之間如何聚合？

- G1與C之間的距離d(G1,C) = d(A,C)
- G3與C之間的距離d(G3,C) = d(E,C)
- G1與G3之間的距離d(G1,G3) = d(A,E)

計算完各群間的距離後，可知d(G1,C)為最短距離，因此G1將與C聚合，成為新群G4。倘若要再聚合，由於剩下G1與G4，可聚合成為G5。



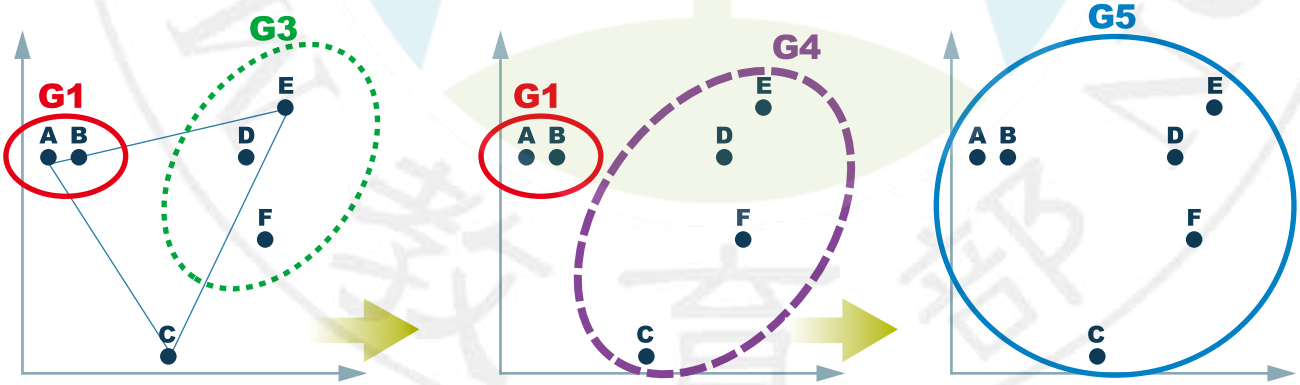
▶ **完整連結（Complete-linkage）：**
在分屬不同的兩群中，選擇最遠的兩點之距離，即代表兩群間的距離。但在群與群間進行聚合時，仍然依據此值選擇最小者做為下一步結合之對象。

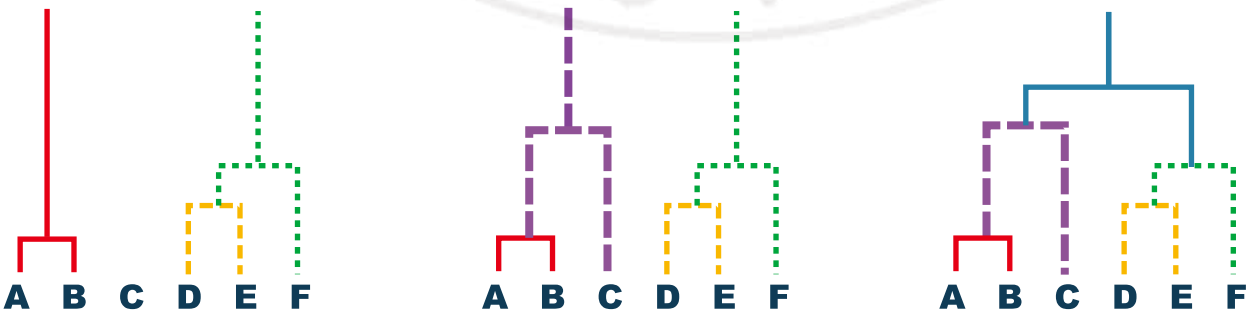
$$d(G1,G2) = \max_{A \in G1, B \in G2} d(A,B)$$

G1、G3與C之間如何聚合？

- ▶ G1與C之間的距離d(G1,C) = d(B,C)
- ▶ G3與C之間的距離d(G3,C) = d(F,C)
- ▶ G1與G3之間的距離d(G1,G3) = d(B,D)

計算完各群間的距離後，可知d(G3,C)為最短距離，因此G3將與C聚合，成為新群G4。倘若要再聚合，由於剩下G1與G4，可聚合成為G5。





- **平均連結（Average-linkage）：**
  在分屬不同的兩群中，各點之距離的平均即代表兩群間的距離。而在群與群間進行聚合時，仍然依據此值選擇最小者做為下一步結合之對象。

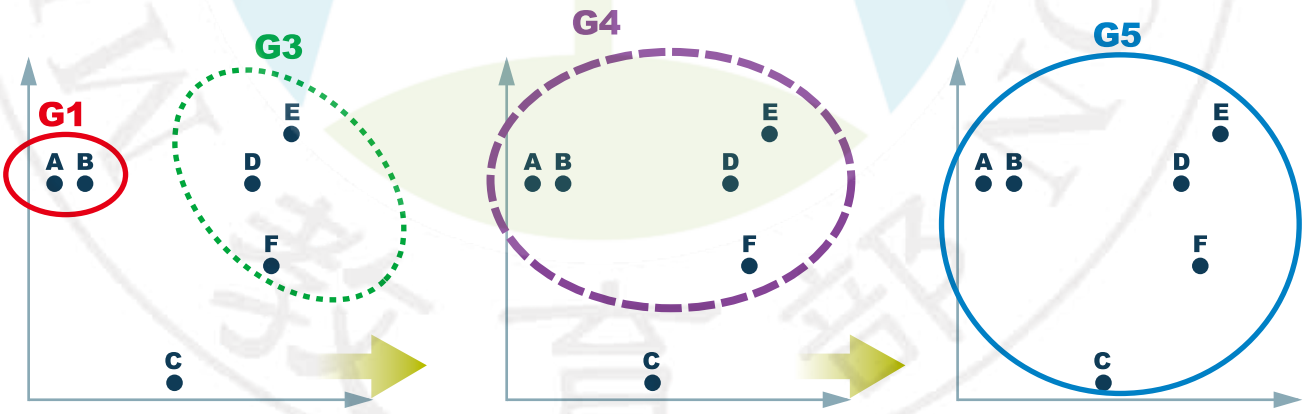
$$d(G1,G2)=\frac{\sum_{A\in G1,B\in G2} d(A,B)}{|G1|\times|G2|}$$

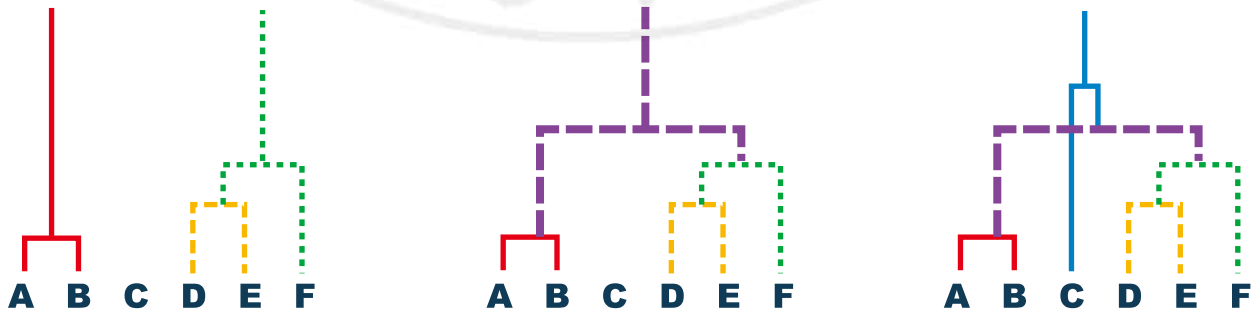
G1、G3與C之間如何聚合？

- G1與C之間的距離 $d(G1,C)=\frac{d(A,C)+d(B,C)}{2\times1}$
- G3與C之間的距離 $d(G3,C)=\frac{d(D,C)+d(E,C)+d(F,C)}{3\times1}$
- G1與G3之間的距離

$$d(G1,G3)=\frac{d(A,D)+d(A,E)+d(A,F)+d(B,D)+d(B,E)+d(B,F)}{2\times3}$$

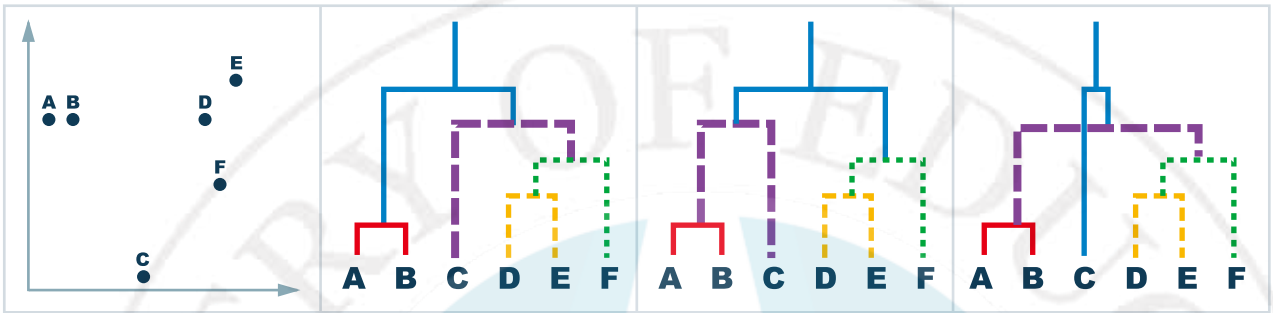
計算完各群間的距離後，可知d(G1,G3)為最短距離，因此G1將與G3聚合，成為新群G4。倘若要再聚合，由於剩下G4與C，可聚合成為G5。





由上述範例，可得到不同的群間距離計算方式，產生之階層式分群結果如下表4-3。

表4-3 **依據不同群間距離計算方式得到的階層式分群結果**



不管使用什麼樣的群間距離計算方式，聚合式階層式分群法皆可得到上表4-3中的樹狀圖結果，接著就可以依照使用者的群數需求或相似度要求，來決定要在哪一層時停止聚合資料。如下圖4-7，若以完整連結的群間距離計算方式為例，圖上的虛線代表不同的群數，端看使用者需求來決定。

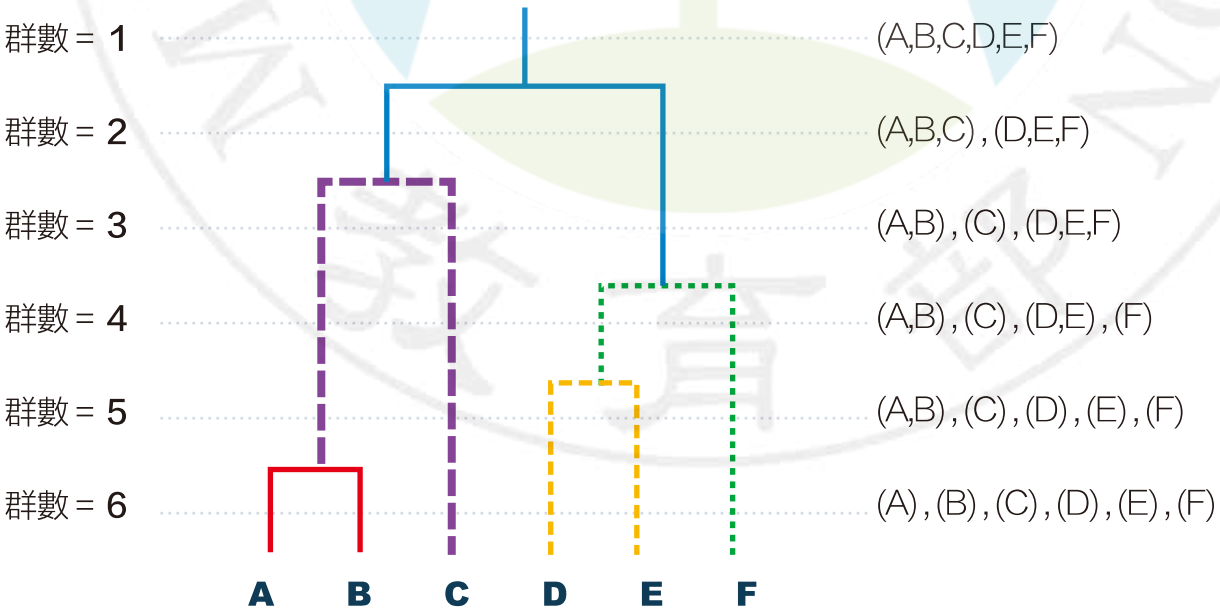
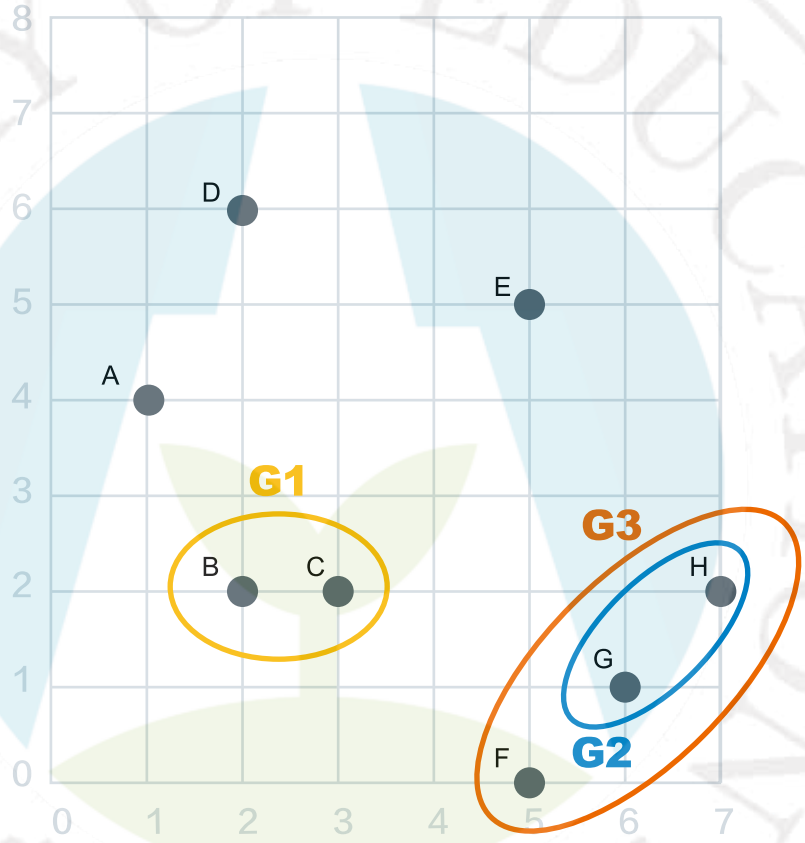


圖4-7 **透過聚合式階層式分群法可得到不同群數的分群結果**

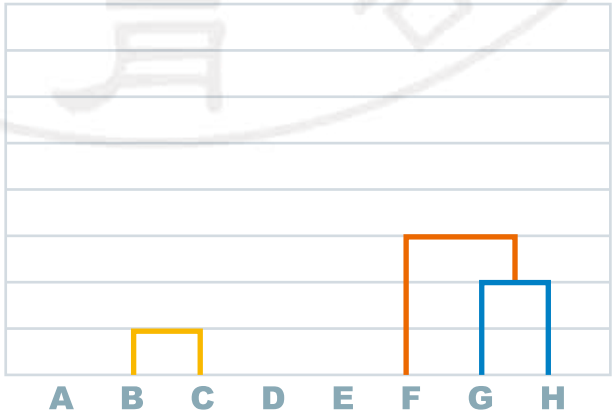
課堂任務

任務三：在此給定資料並以數值化座標平面表示，其中包含A、B、C、D、E、F、G及H共8個點。假設B與C點合併為G1；G與H點合併為G2，而G2加入F點後形成G3。

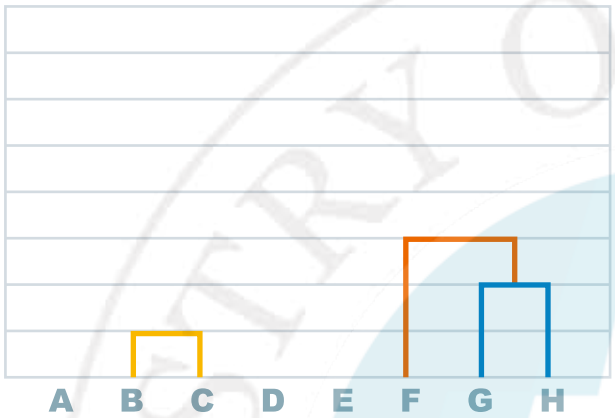


請分別繪製出以「單一連結」、「完整連結」及「平均連結」等3種群間距離計算方式得到的階層式分群結果。

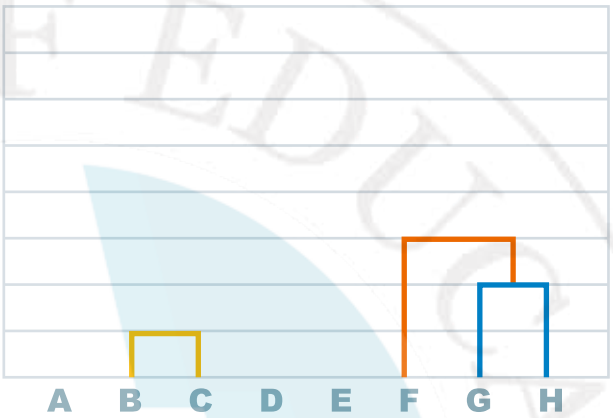
▶ 單一連結



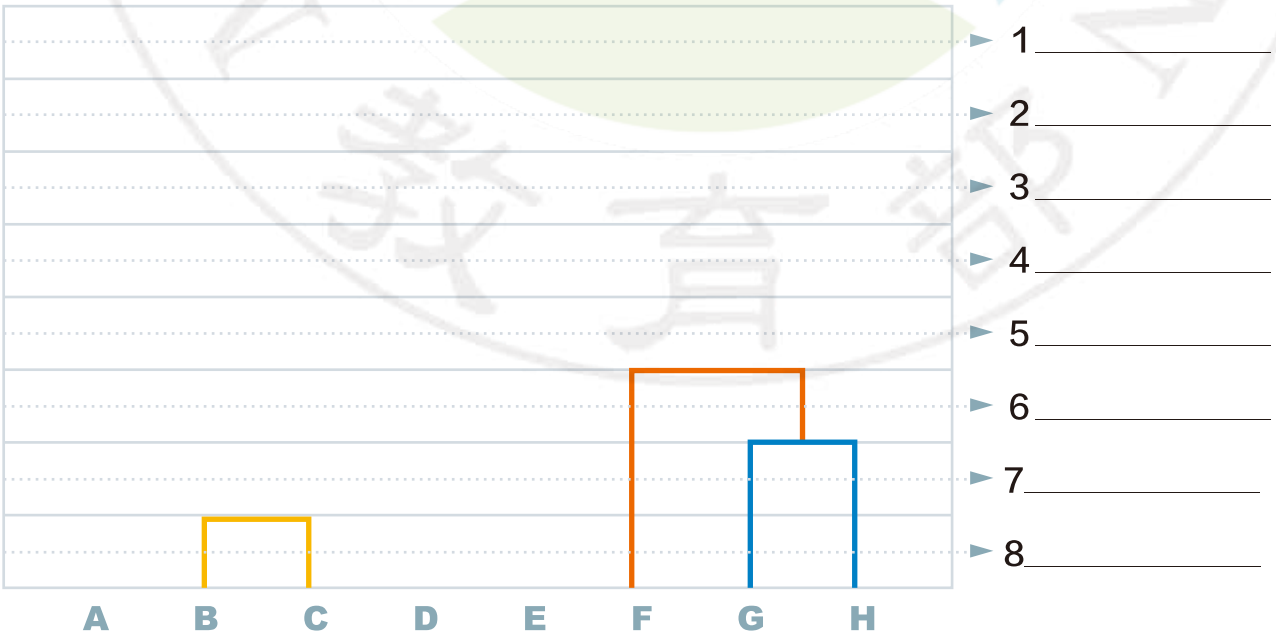
▶ 完整連結



▶ 平均連結



任務四：請以「單一連結」完成之聚合式階層式分群結果，寫出各種不同分群數量時，各群所包含的資料內容。



第五章

增強式學習 Reinforcement Learning

還記得在小時候，當我們做錯事情時會被父母處罰，
而當表現好時就會被父母獎勵的往事嗎？
「賞功罰過」便是大多數人從小接受到的教導方式，
從嘗試中得到經驗進而學習的歷程，
這與增強式學習又有甚麼關聯呢？
現在就讓我們一起來瞭解增強式學習吧！

5-1 增強式學習簡介

在機器學習的子領域中，除了「監督式學習」及「非監督式學習」外，另有一個重要的領域就是「增強式學習」（Reinforcement learning），在訓練的過程中以獎勵來強化機器的行為，強調與環境的互動，以取得最大的預期利益為行動方式。增強式學習的構想來自心理學中的「行為主義」理論，當一個有機體受到來自環境給予的獎勵或懲罰等刺激時，形成對不同環境刺激有著不同的預期利益，並產生能獲得最大利益的行為。此一利用過往的經驗，獎勵回饋並強化所作的行為來進行學習，在不少領域上都十分受用，因此運用十分廣泛，例如：博弈論、資訊理論、統計學以及遺傳算法…等。

下圖是增強式學習的簡單概念圖，我們假設機器是人類的代理人（Agent），來協助人類做決策，而在某個問題的情境（Environment）下，機器需要對當前的狀態（State）做出行動（Action），以到達能獲取較高獎勵（Reward）的狀態。代理人在環境中會獲得現在的狀態及可能的獎勵，根據狀態與獎勵來決定代理人的行為。代理人行為過後，將造成環境的狀態改變、獎勵變更之後亦將回饋給代理人，以決定下一步如何行動，如此循環不已。在增強式學習中，目標是找到一個適當的行為模型，能將代理人所累積獲得的報酬最大化。

圖5-1 增強式學習模型

5-2 由代理人例子來了解如何進行最佳行動

我們用一個簡單的例子來解說在增強式學習的過程中，機器也就是上述所指稱的「代理人」如何在環境中挑選最佳行動，以獲取最大的利益。假設今天在一個迷宮中，裡頭有著寶藏（treasure）和怪物（monster），而迷宮裡的路錯綜複雜，我們在入口處並不知道要怎麼走，而此時代理人的任務則是要最快地取得寶藏並避開怪物。如下圖5-2所示，代理人每次只能往上、下、左、右任一方向前進一格，灰色底的位置則是不能走的障礙物。

圖5-2 增強式學習中代理人所處環境之介紹

由於一開始代理人沒有任何的知識來教導它如何才能最快地到達寶藏，所以在迷宮內的某個點，它只能用嘗試各種可能性的方法選擇一個行動。代理人在不同的狀態下如何走下一步？又可以採取什麼策略呢？

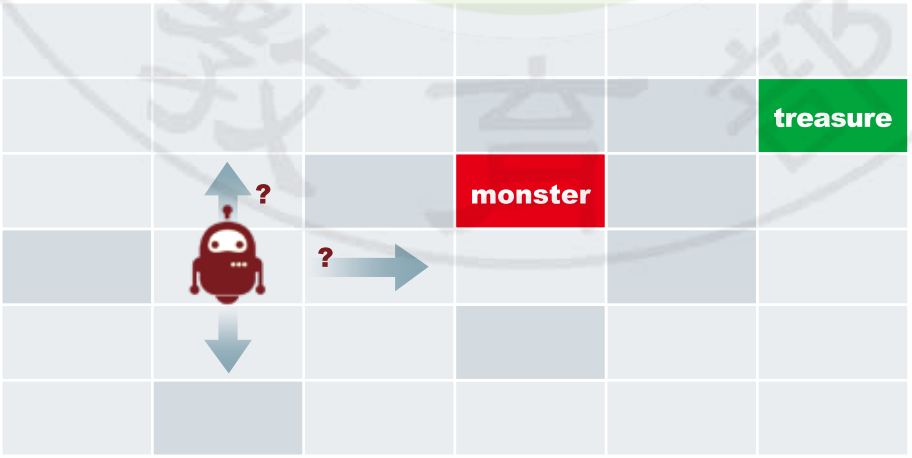


圖5-3 不同狀態下代理人如何挑選下一步

如圖5-4，這時如果往上走，可能就很快能找到寶藏；反之，如圖5-5，如果選擇往右，則會遇到怪物。

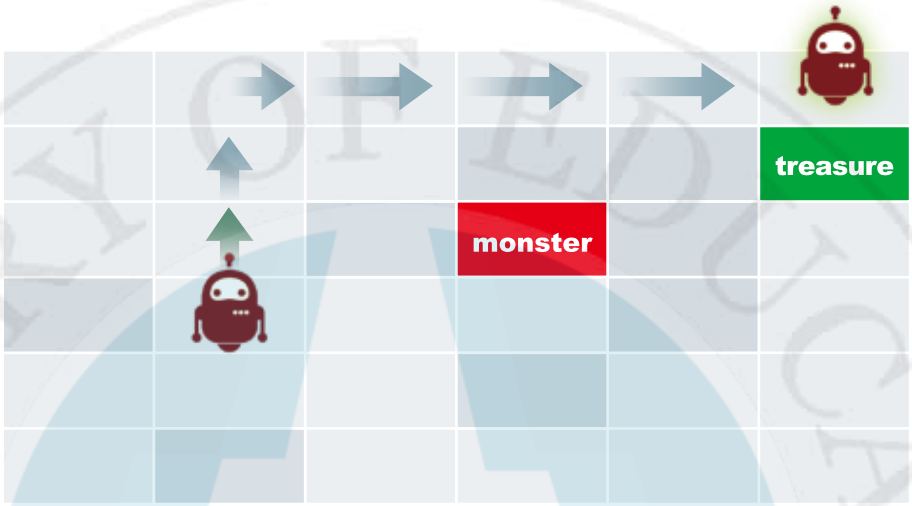


圖5-4 代理人採取往上走策略可獲得寶藏

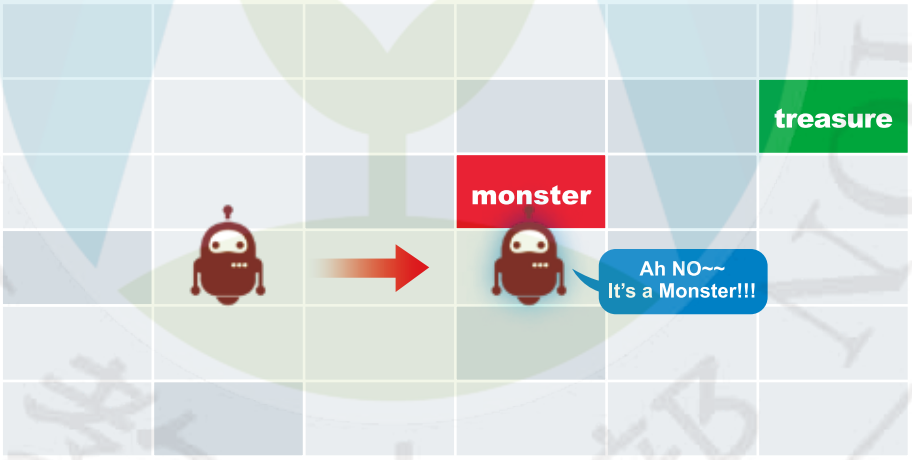


圖5-5 代理人採取往右走策略會遇到怪物

因此，代理人在多次嘗試之後，希望在該點時能夠選擇往上走而非往右走。我們可以使用類似獎勵的方式，在每一個點（狀態）上都設定一個跟最終獎勵相關的「值」如下圖。讓代理人可以儘量往值較高的地方移動，也就是儘量接近可以獲得獎勵的寶藏處。簡單來說，可以將該值視為最後會成功得到寶藏的「可能性」。

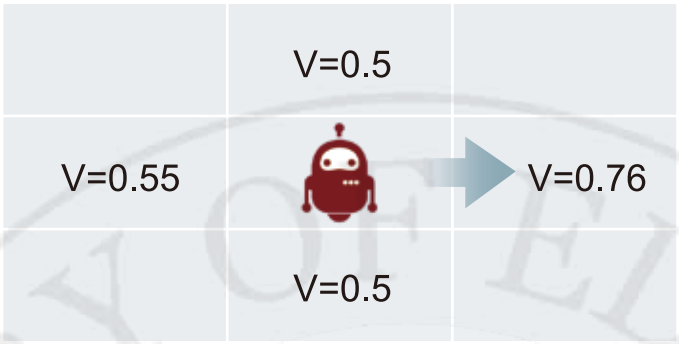


圖5-6 **代理人依據值的高低作為下一動作決策之依據**

但是每個狀態上的值要怎麼設計呢？在這個環境中，存在兩種行動結果，分別是「得到寶藏」與「遇到怪獸」。因此，我們可以分別針對在環境上的行動結果訂定獎勵。如下圖所示，針對環境定義的獎勵，將獲得寶藏的獎勵定成1，受到怪物攻擊的獎勵（也就是懲罰）定成-1。

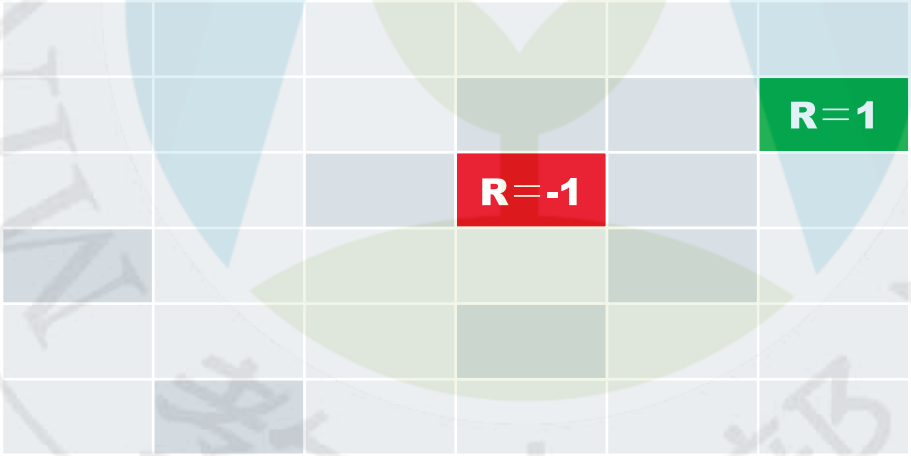


圖5-7 **針對環境定義的獎勵**

既然最終的獎勵已經設定完畢，那麼如何設定環境中每個點的值，便成為一個很重要的問題。我們預期將環境中每個點的值設定好後，代理人可以依此來找出一條路徑以獲取寶物，但到底要如何設定呢？對於在某個狀態的代理人而言，一個好的行動，依賴以下兩個因素：

1. **做了這個行動之後能夠立即取得的獎勵；**
2. **做了這個行動之後，轉移到下一個狀態時，未來有可能接著獲得獎勵（也就是下個狀態的值會更高）。**

在這樣的前提下，希望可以透過較少步驟獲得更高的獎勵，找出環境中的最佳的路徑。

圖5-8 **不同的狀態之下所設定的獎勵值**

上圖為兩種代理人在不同狀態上值的設定，我們通常會希望能挑選最短並且獲得最大獎勵的路線，但像上圖下方的設置方法就違背了這種原則，雖然一樣可以獲取寶藏，但就路徑長短的比較上可能並不是好的設計方式。讓代理人能循著前面提到的兩個因素行動時，基本的想法就是讓狀態的值在離寶藏愈近時愈大，愈遠時愈小。實務上假設我們已經獲得地圖的全貌與寶藏資訊，然而在一開始每個狀態下的值是未知的，必須透過探索後所取得的獎勵來回推估計每個狀態上的值。依照這個想法，可借助「貝爾曼方程（Bellman Equation）」的概念來設計狀態下的值。

貝爾曼方程：

亦稱為「動態規劃方程（Dynamic Programming Equation）」，運用這種數學化方法能夠達到路徑搜尋最佳化。貝爾曼方程的公式如下：

$$V(S) = \max_{a}(R(S,a) + \gamma V(S'))$$

其中，V(S)代表當前狀態的值；R(S,a)表示在當前狀態S中，選擇行動a的獎勵；V(S')則是在選擇了行動a時，轉移到下個狀態S'的值；而γ代表折現率（discount rate），通常介於0到1之間，基本概念是要控制下個狀態的值對當前狀態值的影響力，γ值愈大，下個狀態的值對當前狀態的值影響愈大。

接下來以貝爾曼方程來說明前面尋找寶藏的例子。貝爾曼方程通常使用在描述此類決策問題的決策過程，根據這個方程式（在此假設γ=0.9），寶藏所在位置的前一格之值為1，因為 $V(S) = \max_{a}(R(S,a) + \gamma V(S')) = 1+0.9\times 0 = 1$ 表示由該狀態前往下一個狀態時就能獲得立即的獎勵（寶藏），且因為已經獲取寶藏，並沒有下一個狀態S'。而依序能算出各個狀態下的值，如下圖所示：

圖5-9 **將寶藏前一格的值訂為1**

計算路徑前一步的值，該計算過程為：$V(S) = \max_{a}(R(S,a) + \gamma V(S')) = 0+0.9\times1 = 0.9$ 因為，該狀態選擇行動a後並不能獲取寶藏，加上γ為0.9乘以下一個狀態S'的值1，所以經過貝爾曼方程計算後得到的值為0.9，如下圖所示。

圖5-10 由寶藏回推前兩格計算得到的值為0.9

若再繼續回推上一步的值應為0.81，計算過程為：$V(S) = \max_{a}(R(S,a) + \gamma V(S'))$ $= 0+0.9\times0.9 = 0.81$ 因為，該狀態無法在下一個行動後獲取寶藏，加上γ值乘以下一個狀態S'的值為0.9，因此得到該狀態的值為0.81，如下圖所示。

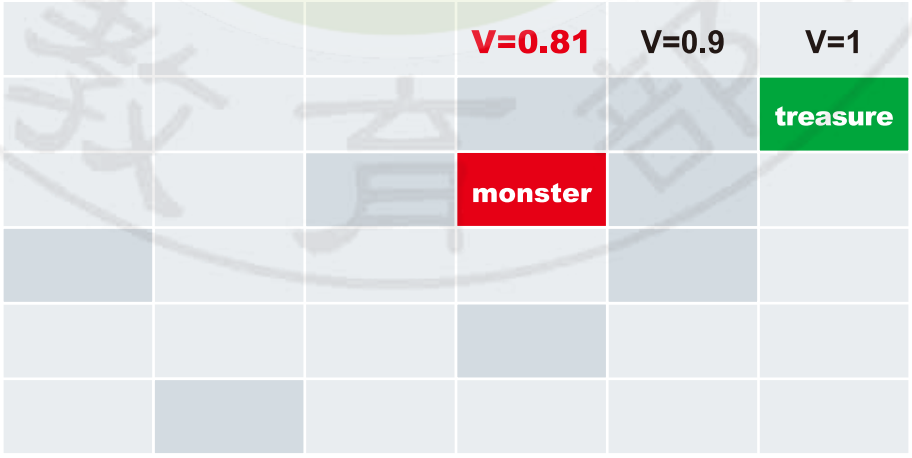


圖5-11 由寶藏回推前三格計算得到的值為0.81

課堂任務

經過上述範例，請試著依照貝爾曼方程的公式，依序算出路徑上各個狀態的值，如下圖紅框處，並將套用貝爾曼方程的計算過程寫下來。

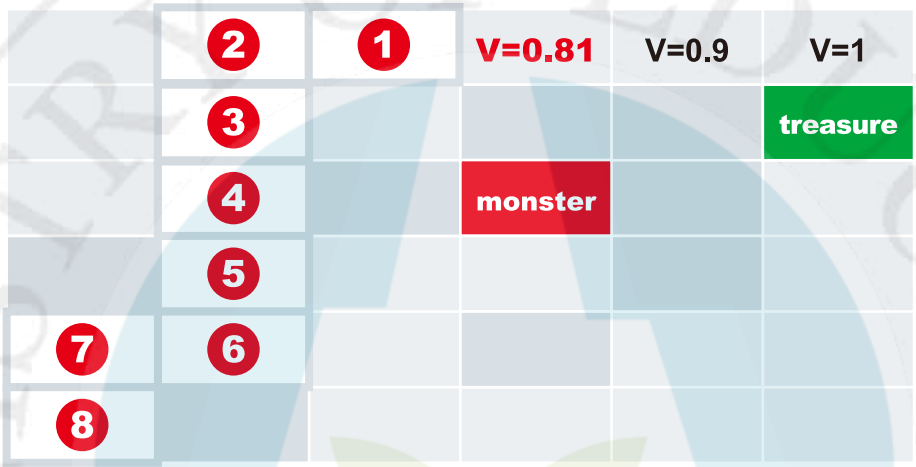


圖5-12 配合課堂任務填寫各狀態的值

1. $V(S) = \max_{a}(R(S,a) + \gamma V(S')) =$ _____
2. $V(S) = \max_{a}(R(S,a) + \gamma V(S')) =$ _____
3. $V(S) = \max_{a}(R(S,a) + \gamma V(S')) =$ _____
4. $V(S) = \max_{a}(R(S,a) + \gamma V(S')) =$ _____
5. $V(S) = \max_{a}(R(S,a) + \gamma V(S')) =$ _____
6. $V(S) = \max_{a}(R(S,a) + \gamma V(S')) =$ _____
7. $V(S) = \max_{a}(R(S,a) + \gamma V(S')) =$ _____
8. $V(S) = \max_{a}(R(S,a) + \gamma V(S')) =$ _____

經過課堂任務的練習後，依序算出路徑上各個狀態的值，如下圖所示。

	V=0.66	V=0.73	V=0.81	V=0.9	V=1
	V=0.59				treasure
	V=0.53		monster		
	V=0.48				
V=0.39	V=0.43				
V=0.35					

圖5-13 路徑上各狀態的值

此外還可以延伸計算路徑旁邊的其他狀態值，如下圖的值為0.66。在該狀態中，可以選擇往左或往上移動，而前述提到之概念，當值越高代表離寶藏越近且獲取寶藏的機會較大，故在該狀態中會選擇往上移動，其值計算過程為：$V(S) = \max_{a}(R(S,a) + \gamma V(S'))$ = 0+0.9×0.73 = 0.66

	V=0.66	V=0.73	V=0.81	V=0.9	V=1
	V=0.59	V=0.66			treasure
	V=0.53		monster		
	V=0.48				
V=0.39	V=0.43				
V=0.35					

圖5-14 計算其他路徑上狀態的值

可利用貝爾曼方程之公式 $V(S) = \max_{a}(R(S,a) + \gamma V(S'))$ 把地圖上可能路徑算出其值結果如下圖，並畫出在不同的狀態下的最佳路徑圖。

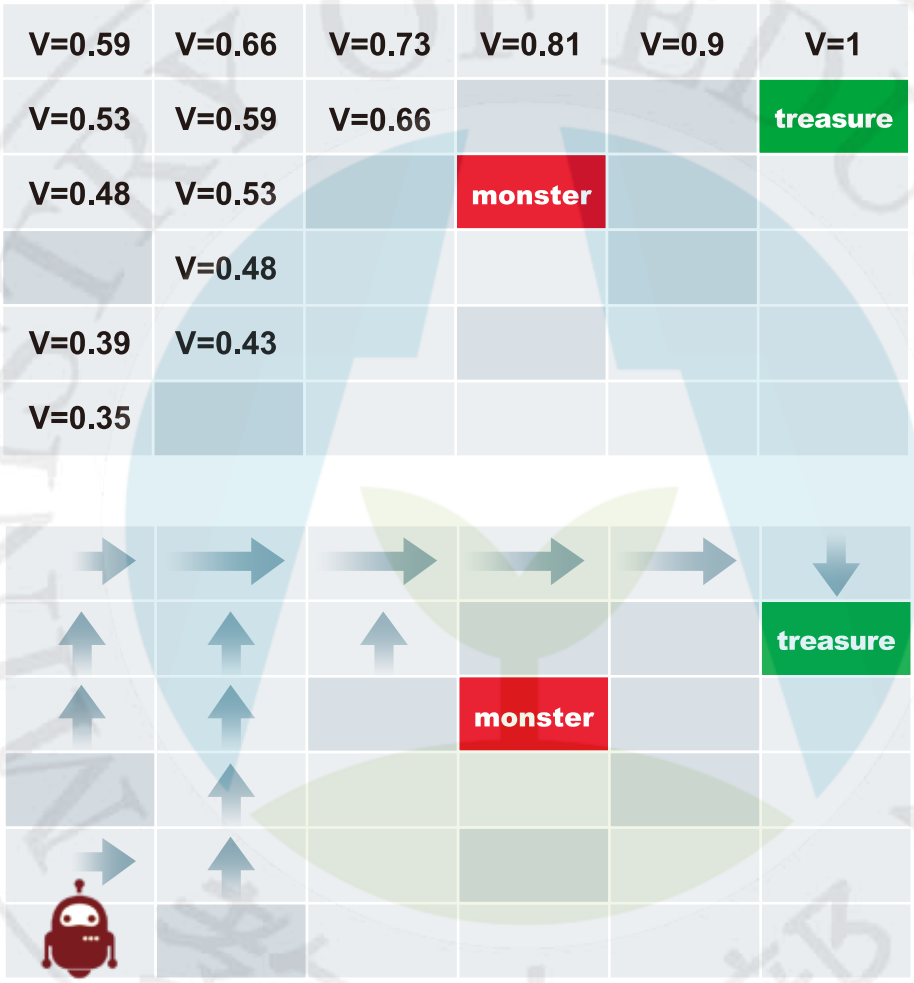


圖5-15（上）地圖上各狀態的值（下）各狀態採取之行動

但根據圖5-15的結果，會產生另一個問題，假設永遠只選擇往值高的路線走，則圖5-16上的兩個狀態，只能依循這個原則選擇往左移動。

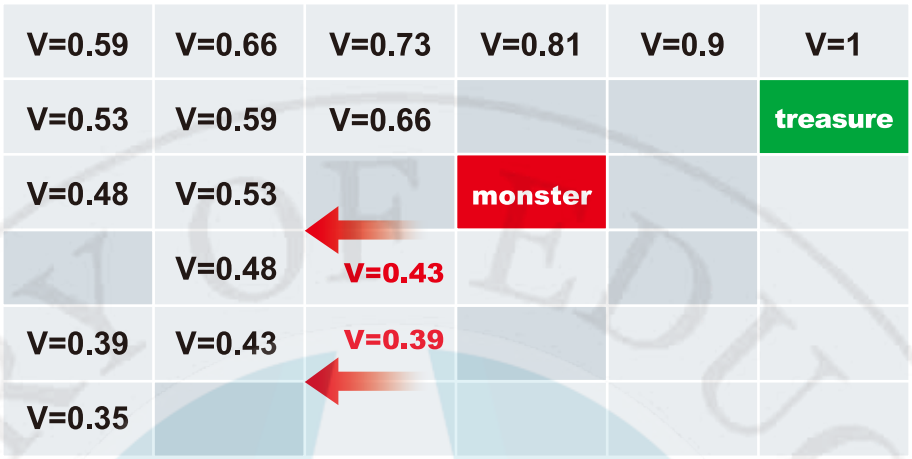


圖5-16 只選擇當下狀態最佳行動的路徑

這麼一來，就永遠也找不到下圖這條與目前最佳路線一樣快的路線了（獲取寶藏之過程同樣移動11格）。

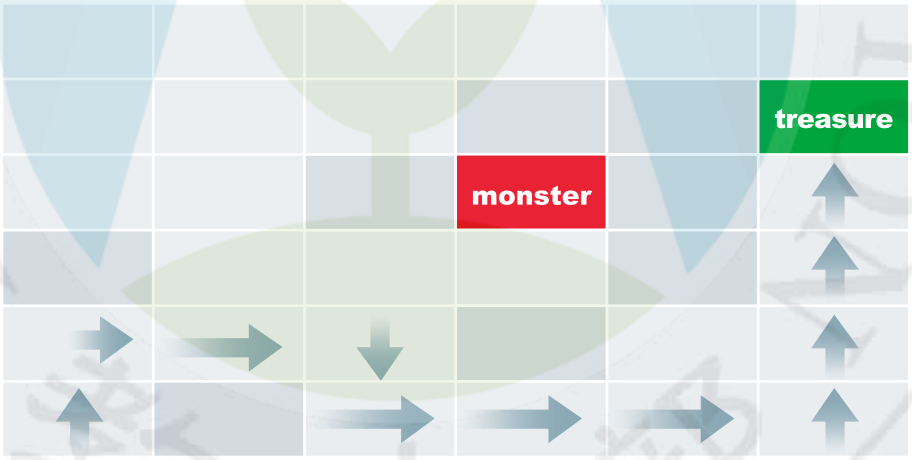


圖5-17 與現有最佳路徑移動相同距離獲得寶藏的另一條路徑

然而，我們利用貝爾曼方程找出的路徑，可能只是一條不錯的路徑，而未必能找到真正的最佳路徑。原因在於此公式是找出「當下狀態」最佳的行動，有時得到的只是局部最佳路徑，而非整體最佳路徑。為解決此問題可用「馬可夫決策過程（Markov Decision Process，縮寫為 MDP）」來解決。

馬可夫決策過程：

在動態規劃與增強式學習的研究領域中，馬可夫決策過程在找尋最佳化問題上是一個有用的工具。馬可夫決策過程的基本概念是賦予決策者在決策過程時，除了按照既有的狀況選擇最佳決策外，還有一些隨機性。在演算法領域上，我們先前所學習的做法，稱之為「貪婪法（Greedy method）」，此法意指在每一個決策點上永遠只選擇當下最好的選擇；而透過馬可夫決策過程的引入，代理人會隨機選擇一個動作進行，在一定程度的機率下會按照原先貪婪法的決策過程，選擇當下的最佳策略。但也存在某些可能，選擇當前不是最佳行動的狀態。以下圖為例：在值的設定上有10%機率為0.65、10%機率為0.7，而有80%機率為0.8，所以有三種的選擇。在原本的規劃中，接下來的行動會以值最高的為行動選擇之依據，但導入馬可夫決策過程後，可以想像成有10隻籤，V=0.65的有1隻、0.8的有8隻、0.7的有1隻，還是有很大的機會會選到V=0.8，但也有機會出現不同的選擇。

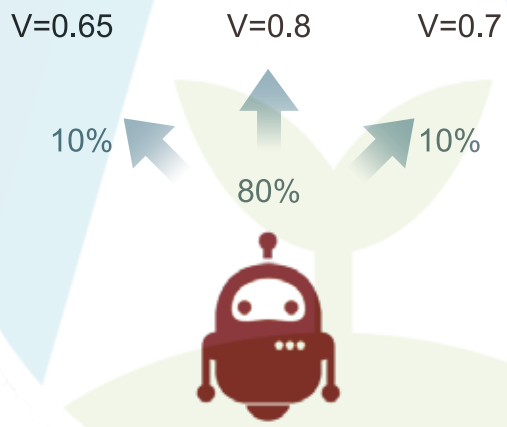


圖5-18 **馬可夫決策過程之機率概念**

這個做法的用意是為了探索出更多的可能路線，在數學上我們則可以視為避開「局部最佳解（Local optimum）」的方法。通常在最佳化問題（Optimization problem，例如這個例子中的找出最佳路線）中會有所謂的全域最佳解（Global optimum）與其它次佳（Suboptimal）的可行方法，而透過貪婪法往往只能找到次佳的解，因此，此刻就需要一些技巧，如馬可夫決策過程，來尋找其它可能的解法。

5-3 Q-學習

在了解增強式學習的概念後，接著介紹一個增強式學習中常用的方法，稱為Q-學習（Q-Learning）。原先增強式學習中，只關注每個狀態的值，Q-學習則評估每個狀態下、每個動作對應的Q值（以Q代表是取當下做決策的「品質」Quality的字首）來決定代理人下一步的行為。上一節提到狀態下價值函數的評估是根據探索而回推的，在Q-學習中，我們要把價值函數的概念推廣到不只是每個狀態下的值，而是每個狀態下、對於不同行動的值。

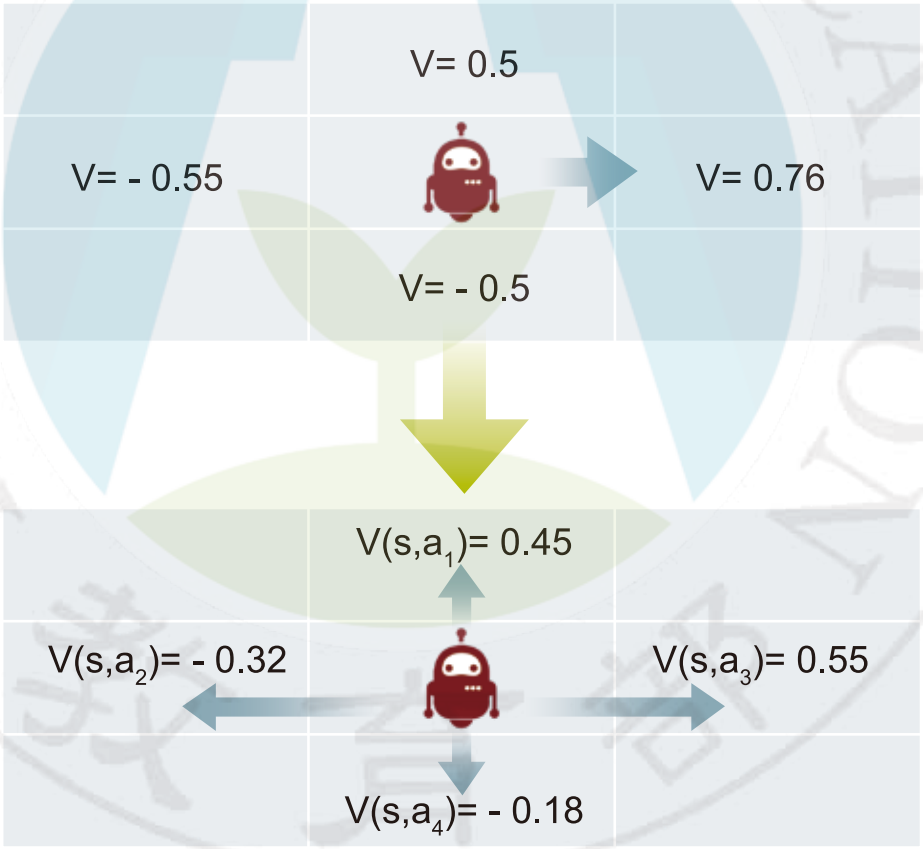


圖5-19 **Q-學習關注該狀態每一個行動的值**

因此，Q-學習需記錄每個狀態下所對應每個行動的值，簡稱「**Q值（Q-value）**」所形成的Q值表（Q-table），做為代理人在行動決策上的依據，如下表。

表5-1 **使用Q表來記錄每個狀態下個行動的Q值**

	a_1	a_2	a_3	a_4	...	...	a_m
s_1	0.32	-0.25	0.56	-0.05	...	...	0.25
s_2	-0.54	0.22	0.35	-0.08	...	...	-0.36
s_3	0.65	-0.95	-0.98	0.95	...	...	0.22
s_4	-0.56	0.45	0.32	0.88	...	...	-0.02
⋮	⋮	⋮	⋮	⋮	...	...	-0.03
⋮	⋮	⋮	⋮	⋮	...	...	-0.14
s_n	0.35	0.22	0.03	0.95	...	...	0.25

在每個狀態S下，執行行動a的 **Q值** 定義如下：

$$Q(S,\mathrm{a}) = R(S,\mathrm{a}) + \gamma \max_{\mathrm{a}'}(Q(S',\mathrm{a}'))$$

基本上跟前一節介紹的相似，其中R(S,a)代表在此狀態S中，採取行動a會獲取寶藏（或遇到怪物）的獎勵，加上下個行動能獲得的最大Q值組合而成。在一開始尚未訓練時，由於原先完全沒有任何關於該環境的相關資訊，每個狀態下每個動作的Q值可能都只是隨機給定的，故Q值的更新是透過一次又一次的迭代，並依照下一步的行動結果來更新針對這個行動的Q值。例如在某個狀態S下，倘若選擇某個行動a之後發現會遇到怪物，這時該狀態的獎勵值可能會是-1，這時就能依照這個情況更新上一步的Q值，將之降低。反之，若執行某個行動a之後獲得寶藏，這時該狀態的獎勵值可能會是1，這時就能依照這個情況更新上一步的Q值，將之提高。

更新Q表中的Q值：

接下來將以範例來說明更新Q值的作法。以表5-2的Q表為例，若目前在S1這個狀態中，假設在此狀態做出任何行動後尚無法獲得寶藏，此時由Q表中S1選擇a3這個行動，因其獲得寶藏的可能性較高。若在S1這個狀態下執行a3會進入S2，則根據「回推」的原則，S2中Q值最高的a3就是更新S1中Q值之關鍵。依據上述定義，其Q值更新計算過程如下（假設$\gamma = 0.9$）：

$$Q(S1,a3) = R(S1,a3)+\gamma \max_{a'}(Q(S2,a')) = 0+0.9\times0.35 \cong 0.32$$

表5-2 搭配課程範例之Q表

	a_1	a_2	a_3	a_4
s_1	0.35	-0.25	0.56	-0.05
s_2	-0.54	0.22	0.35	-0.08

雖然已經知道更新Q值的方式，但我們通常不會直接以上述計算得到的結果來直接更新Q值，反而會透過「時值差異（Temporal Difference，簡稱TD）」的方式更新。時值差異TD的定義如下：

$$TD(S,a) = R(S,a)+\gamma \max(Q(S',a'))-Q(S,a)$$

由時值差異TD的定義，可發現就是將上述計算更新後Q值的結果減掉原先的Q值，藉此找出兩者之間的落差。而得到時值差異TD後，就可以用來更新Q表中的Q值，如下所示：

$$Q_t(S,a) = Q_{t-1}(S,a)+ \alpha TD(S,a)$$

因此在某個狀態S下執行行動a時，其Q值的變化在既有的Q值中，又參照一定比例的時值差異TD，來推估出新的Q值。其中，α是一個介於0到1之間的值，當α越大，代表從時值差異TD學習的幅度越大，受下一步Q值的影響也越大。延續表5-2的範例，得到的Q值更新計算過程如下（假設$\gamma = 0.9$、$\alpha = 0.9$）：

$$TD(S1,a3) = 0+0.9\times0.35-0.56 = -0.245$$

$$Q_t(S1,a3) = 0.56+0.9\times(-0.245) = 0.3395 \cong 0.34$$

藉由上述運算過程，即可將S1中的a_3之Q值由0.56更新為0.34，接著再繼續於不同的狀態下，依循Q表採取其他行動，直到找到寶藏（或遭遇怪物）為止。

接下來，將透過一個較完整的範例（見表5-3），示範如何透過一步步迭代的方式更新Q表中的Q值。

表5-3 **Q表中紀錄4個狀態及每個狀態上各行動的Q值**

	a_1	a_2	a_3	a_4
s_1	0.35	-0.25	0.56	-0.05
s_2	-0.54	0.22	0.35	-0.08
s_3	0.85	-0.95	-0.98	0.95
s_4	-0.56	0.45	0.32	0.88

在開始更新Q表前，必須先瞭解每個狀態在進行不同的行動後，所具有的狀態改變規則，如下圖5-20。每個狀態執行甚麼樣的行動，進而進入哪一個狀態，都是由環境自訂的，下圖只是記錄下規則，以便在更新Q表的過程中，作為狀態轉換之依據，例如：S1在進行 a1的行動後，將進入S4。

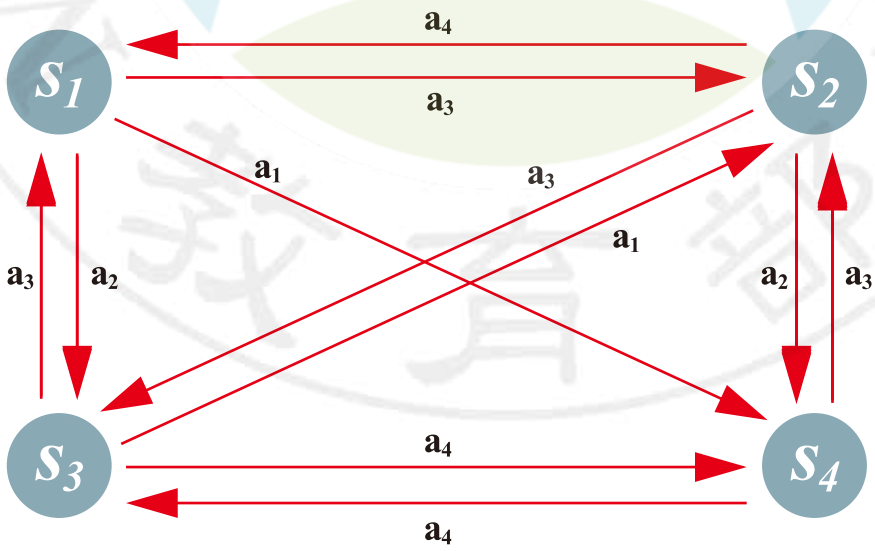


圖5-20 **環境中各狀態在執行不同行動後的狀態轉變圖**

若此時由S1開始進行Q值的更新，則計算過程如下（假設 $\gamma = 0.9$、$\alpha = 0.9$）：

S_1

原始Q表：

	a_1	a_2	a_3	a_4
s_1	0.35	-0.25	0.56	-0.05
s_2	-0.54	0.22	0.35	-0.08
s_3	0.85	-0.95	-0.98	0.95
s_4	-0.56	0.45	0.32	0.88

行動選擇：a_3

狀態轉換：依據環境中的規則，S1 執行 a3後進入 S2。

$TD(S1, a3) = 0+0.9\times0.35-0.56 = -0.245$

$Q_t(S1, a3) = 0.56+0.9\times(-0.245) = 0.3395 \cong \mathbf{0.34}$

更新後Q表：

	a_1	a_2	a_3	a_4
s_1	0.35	-0.25	**0.34**	-0.05
s_2	-0.54	0.22	0.35	-0.08
s_3	0.85	-0.95	-0.98	0.95
s_4	-0.56	0.45	0.32	0.88

S_2

原始Q表：

	a_1	a_2	a_3	a_4
s_1	0.35	0.25	**0.34**	-0.05
s_2	-0.54	0.22	0.35	-0.08
s_3	0.85	-0.95	-0.98	0.95
s_4	-0.56	0.45	0.32	0.88

行動選擇：a_3

狀態轉換：依據環境中的規則，S2執行a3後進入S3。

$TD(S2,a3)$ = 0+0.9×0.95-0.35 = 0.505

$Q_t(S2,a3)$ = 0.35+0.9×(0.505) = 0.8045 ≅ **0.80**

更新後Q表：

	a_1	a_2	a_3	a_4
s_1	0.35	-0.25	**0.34**	-0.05
s_2	-0.54	0.22	**0.80**	-0.08
s_3	0.85	-0.95	-0.98	0.95
s_4	-0.56	0.45	0.32	0.88

S_3

原始Q表：

	a_1	a_2	a_3	a_4
s_1	0.35	0.25	**0.34**	-0.05
s_2	-0.54	0.22	**0.80**	-0.08
s_3	0.85	-0.95	-0.98	0.95
s_4	-0.56	0.45	0.32	0.88

行動選擇：a_4

狀態轉換：依據環境中的規則，S3執行a4後進入S4。

$TD(S3,a4)$ = 0+0.9×0.88-0.95 = -0.158

$Q_t(S3,a4)$ = 0.95+0.9×(-0.158) = 0.8078 ≅ **0.81**

更新後Q表：

	a_1	a_2	a_3	a_4
s_1	0.32	-0.25	**0.34**	-0.05
s_2	-0.54	0.22	**0.80**	-0.08
s_3	0.65	-0.95	-0.98	**0.81**
s_4	0.56	0.46	0.32	0.88

課堂任務

延續上述的計算，當S3執行a4後進入S4，接下來要如何繼續更新Q值呢？請完成下方的填空。

S_4

原始Q表：

	a_1	a_2	a_3	a_4
s_1				
s_2				
s_3				
s_4				

行動選擇：_____

狀態轉換：依據環境中的規則，S4執行_____ 後進入_____。

$TD(S4,$_____$) =$ _____

$Q_t(S4,$_____$) =$ _____

更新後Q表：

	a_1	a_2	a_3	a_4
s_1				
s_2				
s_3				
s_4				

透過以上方法，我們可以在一次又一次的迭代之中，藉由獎勵值的影響而改變Q值，進而影響到決策的方向。以上面的走迷宮當例子，在一開始由於沒有任何的資訊，所以可能會亂走，亂走後可能會遇到怪物，此時的獎勵值是-1，而根據Q-學習的方法，會將任何可能走到怪物的決策之Q值調降，這時的機器就會變的比較聰明些，會避開傾向怪物的決策。過了一段時間之後可能就能走到寶藏，而寶藏的獎勵值是1，此時傾向走向寶物的決策之Q值就會大幅提升，因此機器就會更聰明地開始直接走向寶物。

在Q-學習中有兩個參數可以調整，分別是 α 與 γ ，且這兩個值的範圍皆在0到1之間。從公式可以看出， α 愈大，時值差異TD對原先的Q值的影響就愈大，也就是原先Q值的改變幅度愈大，可以視為從時值差異TD學習的幅度愈大。而 γ 在前一節有提過，是要控制下個狀態的值對當前狀態值的影響力；也就是Q-學習中，當前狀態的最佳決策對上一次決策的影響，這可以視為「未來的決策」影響目前決策的幅度。以迷宮的例子而言，就像是在上一次迭代中，發現走這條路就可以獲得寶藏，而在下一次迭代時，要不要依照上一次迭代的經驗繼續選擇這個決策以獲得獎勵的考量。

總而言之，增強式學習就是在一次又一次的試誤中，找到最好的方法以完成決策。以Q-學習為例，為了增加學習的效率，可以讓機器人從多個狀態中出發，多工同步更新Q值。而完成學習後的Q-表就可以提供各狀態作為採取行動之決策，以最快的方式達成目標，但需注意的是環境一旦改變，Q表就需要重新訓練。

Note

第六章

深度學習 Deep Learning

當AlphaGo屢敗世界棋王李世乭與柯潔後，
在世人一片驚呼與讚嘆聲背後，「深度學習」也開始受到注目！
許多人將「人工智慧」與「深度學習」劃上等號，
或許答案不然，但也是因這層緊密的關連，更引起人們的興趣，
現在就讓我們一起探究深度學習的奧妙吧！

6-1 深度學習簡介

說起深度學習（Deep learning），可能多數人會有種一頭霧水的感覺，但若是提及2016年人工智慧圍棋程式AlphaGo與世界圍棋冠軍李世乭那場世紀之戰，相信大家都仍記憶猶新。然而不同於二十年前的另一場世紀對決：1996年IBM的深藍超級電腦（Deep Blue）擊敗西洋棋世界冠軍卡斯帕洛夫（Garry Kasparov），此次電腦不再憑藉著當時每秒可評估三億棋步的高速運算能力，AlphaGo藉以致勝的關鍵技術之一便是「深度學習」。

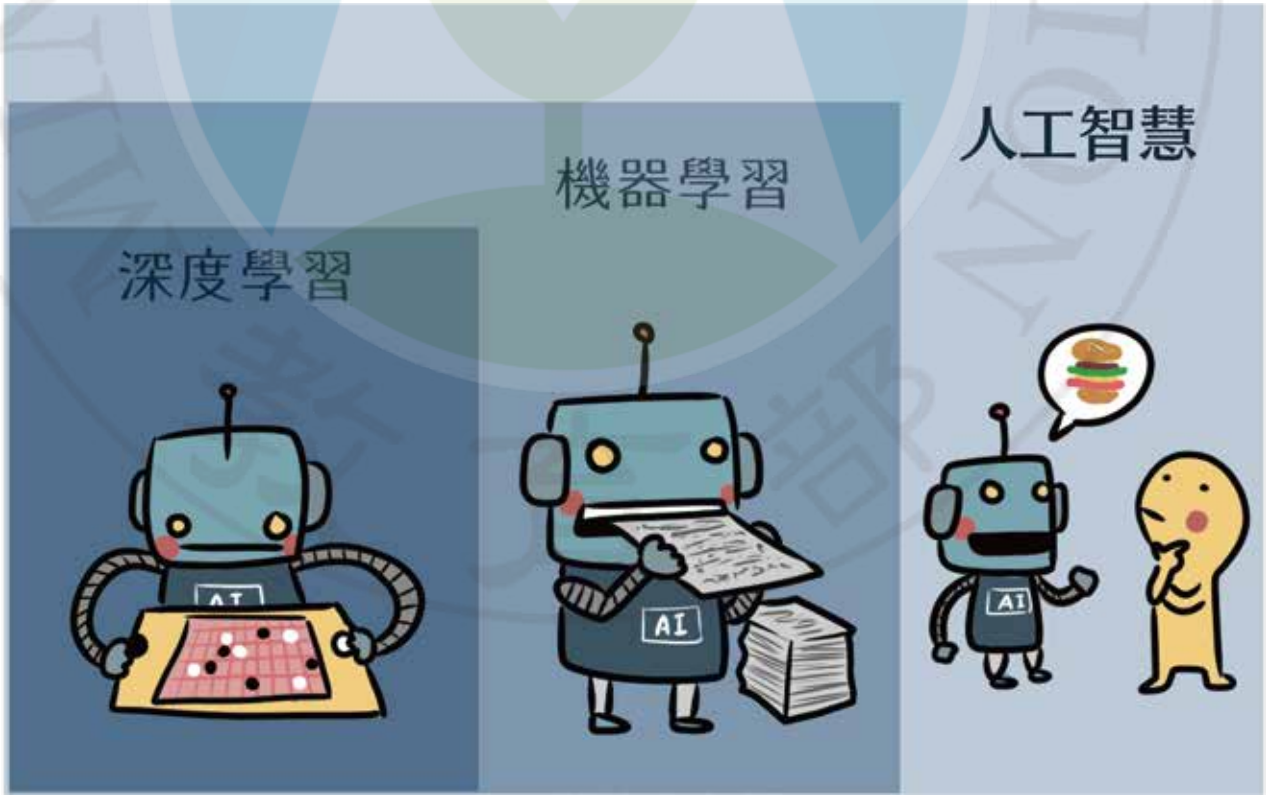


圖6-1 人工智慧、機器學習、深度學習的範圍

深度學習為機器學習的一個分支，二者均為人工智慧領域的重要理論。雖然人工智慧的概念直到近年才廣為人知，實際上，早在1943年便有以數學模型模擬生物神經網路的概念，此種模擬出的成品即稱「類神經網路」。到了1958年，心理學家Rosenblatt在類神經網路的結構中加入了訓練修正參數的機制，賦予了類神經網路學習能力，至此，其基本架構已然成形。如圖6-2所示，類神經網路的神經元從前端收集到各種訊號（模擬生物神經的樹突），然後將各個訊號根據權重加權後加總轉換成新訊號傳送出去。

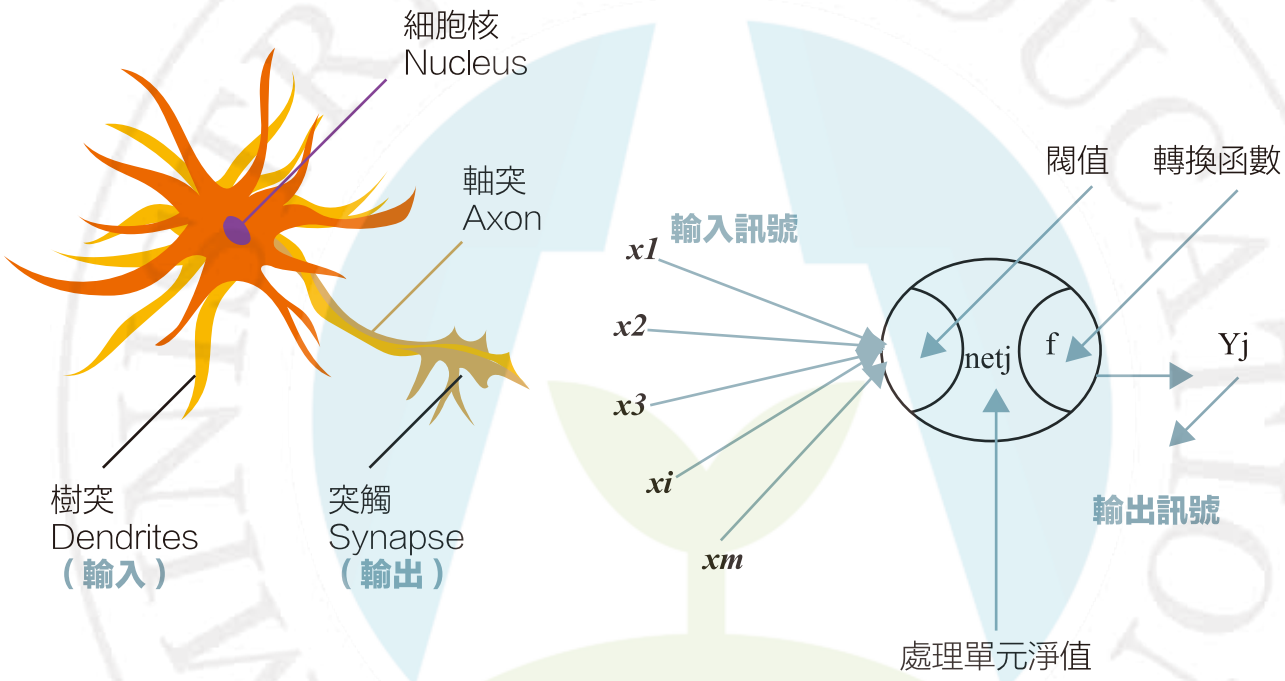


圖6-2 生物神經網路與類神經網路

典型的類神經網路架構如圖6-3，可分為輸入層（Input layer，表示輸入變數）、隱藏層（Hidden layer）、輸出層（Output layer，表示要預測的變數）；隱藏層的每個神經元分別代表著一個抽象特徵，神經元之間都有連結，各自擁有「權重」，用以處理訊號的加權。相信大家看到這裡，對於神經元的「權重」應該還是一知半解吧？簡單的來說，一開始先隨機產生每個神經元的特徵與該特徵的權重，然後輸入訓練資料後，先依照網路中初期的特徵與權重，算出輸出層的向量，再根據「經過計算的向量」與「我們期待的向量」間的差異，調整每個神經元的權重，最終使「經過計算的向量」與「我們期待的向量」能趨於一致。

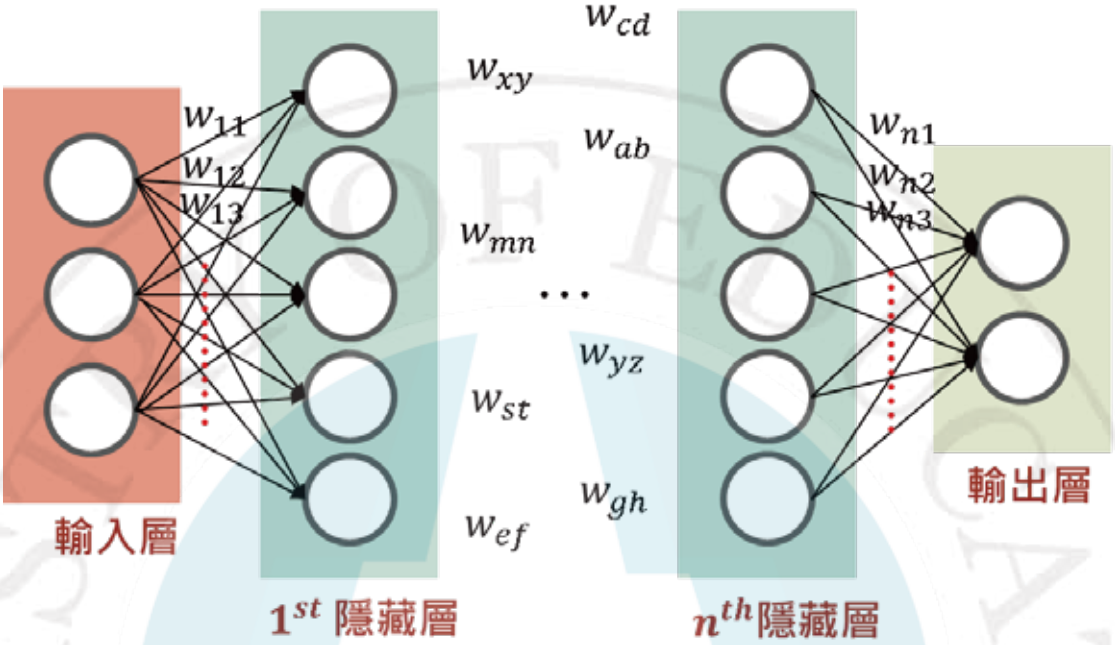


圖6-3 類神經網路架構

深度學習的基本原理即是透過賦予資料特徵不同權重來對資料進行分類。在第三章的「監督式學習」裡，我們已學習到不論是分類器或是決策樹，均仰賴人力訂定出特徵值（如先前以使用者購買筆電與否的消費記錄建構決策樹時，談到的收入、學生與否），而且對於分類的準確性而言，這些特徵值的參考價值各不相同。與傳統分類器不同，深度學習類神經網路中的特徵值與權重均可由數學演算法自行推估，並透過大量的訓練資料取得最理想的成果。

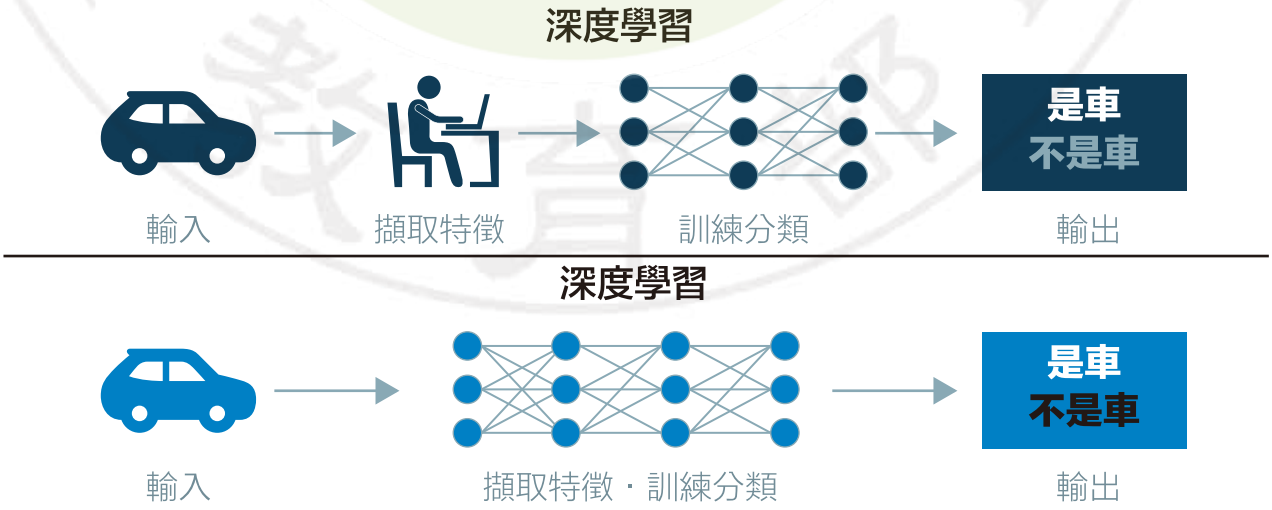


圖6-4 深度學習類神經網路與傳統分類器的架構比較

深度學習運用了分層次抽象的想法，這個想法認為更高層次的概念可以從低層次的概念學習到，所以會在輸入與輸出層中增加隱藏層的層數達到這一目的。當層數愈多，網路看起來就愈深，故而被稱做深度學習。然而隨著神經元與隱藏層層數的增加，所需要的計算也更為龐大，雖然深度學習概念很早就被提出，然而在八〇年代的硬體運算速度不敷應付的狀況下，其應用仍極其有限。

2012年，傑佛瑞・辛頓（Geoffrey Hinton，人稱深度學習之父）的兩位學生於每年舉辦的「ImageNet圖像識別大賽」中以「16.42%」錯誤率的驚人成績取得冠軍，其背後技術便是利用GPU來快速處理深度學習會用到的大量矩陣運算。2013年，Google斥資收購Hinton與其兩位學生所成立的公司，也宣告了深度學習的春天正式來臨。

如今，深度學習的應用領域除了最為常見的影像識別，同時也適用於語音辨識、自然語言處理、音訊辨識與文本分析，另外還可應用於自動導航、判讀醫療影像，甚至在訓練後成為電玩高手，在在顯示深度學習已在資訊技術發展上佔有舉足輕重的地位。

6-2 如何將待解問題數學化

既然深度學習是透過「數學運算」來進行，那麼，我們要如何將一個所面臨的問題以數學形式表現出來呢？例如，一個圖像識別的問題要如何轉換為數學運算進行識別分類呢？讓我們先來看一個簡化的圖形識別問題：讓電腦來讀入「/」及「\」兩個圖像並進行識別。

首先，我們要先理解，電腦眼中的圖像皆是由像素（Pixel）組成，因而「/」及「\」兩個圖像在電腦眼中其實是如以下的矩陣所示，其中「+1」表示有像素存在，反之則以「-1」表示。

\

+1	-1
-1	+1

矩陣 A

/

-1	+1
+1	-1

矩陣 B

現在，我們面臨第二個問題：如何藉由數學運算來將上述兩個矩陣區分為「/」及「\」兩個圖像？倘若將矩陣裡的數值進行相加或相乘似乎都無達到要求，因為兩個矩陣元素相加或相乘的結果都相同；另一種思考方向則為加入另一個新的矩陣K。

在此我們先定義兩個矩陣如何進行乘積，假設給定兩個相同維度的矩陣M、N，如下：

$$M=\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}、N=\begin{bmatrix}0 & 1\\ 5 & 4\end{bmatrix}$$

透過「阿達馬乘積（Hadamard product）」或稱做分素乘積（Entrywise product）進行矩陣的運算，且將兩矩陣的乘積運算標記為「M。N」，其計算方式就是將兩矩陣中對應位置的元素相乘，因此可得到以下矩陣乘積結果：

$$\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}\circ\begin{bmatrix}0 & 1\\ 5 & 4\end{bmatrix}=\begin{bmatrix}0 & 2\\ 15 & 16\end{bmatrix}$$

瞭解矩陣如何進行乘積運算後，我們可以初步擬定一條分類規則：分別將矩陣A與矩陣B與一矩陣K進行乘積，並將運算後得到的矩陣所有元素相加，若相加所得值大於0，則此圖像為「\」；反之則判定為「/」。

矩陣乘積	矩陣乘積後之元素和	判定圖像
矩陣A與矩陣K之乘積 $\begin{bmatrix}+1 & -1\\ -1 & +1\end{bmatrix}\circ\begin{bmatrix}+1 & -1\\ -1 & +1\end{bmatrix}=\begin{bmatrix}+1 & +1\\ +1 & +1\end{bmatrix}$	$(+1)+(+1)+(+1)+(+1)=4$	\
矩陣B與矩陣K之乘積 $\begin{bmatrix}-1 & +1\\ +1 & -1\end{bmatrix}\circ\begin{bmatrix}+1 & -1\\ -1 & +1\end{bmatrix}=\begin{bmatrix}-1 & -1\\ -1 & -1\end{bmatrix}$	$(-1)+(-1)+(-1)+(-1)=-4$	/

註：假設矩陣K為 $\begin{bmatrix}+1 & -1\\ -1 & +1\end{bmatrix}$

然而，若是電腦所讀入的「/」及「\」兩符號書寫過於潦草，那麼我們的判斷是否仍然正確呢？假設所讀入的「\」圖像在右上角的彎曲幅度過大，以致於佔去了三個像素（如下圖矩陣A）；而「/」圖像又因寫的太短，僅佔了左下角一個像素（如下圖矩陣B）。

矩陣 A

矩陣 B

經由如下的計算結果，我們發現仍可將二者進行正確分類。此時，我們可以宣稱這個方法是「強韌的（robust)」。

矩陣乘積	矩陣乘積後之元素和	判定圖像
矩陣A與矩陣K之乘積 $\begin{bmatrix} +1 & +1 \\ -1 & +1 \end{bmatrix} \circ \begin{bmatrix} +1 & -1 \\ -1 & +1 \end{bmatrix} = \begin{bmatrix} +1 & -1 \\ +1 & +1 \end{bmatrix}$	$(+1)+(-1)+(+1)+(+1)$ $=2$	\
矩陣B與矩陣K之乘積 $\begin{bmatrix} -1 & -1 \\ +1 & -1 \end{bmatrix} \circ \begin{bmatrix} +1 & -1 \\ -1 & +1 \end{bmatrix} = \begin{bmatrix} -1 & +1 \\ -1 & -1 \end{bmatrix}$	$(-1)+(+1)+(-1)+(-1)$ $=-2$	/

上述範例中，矩陣K對矩陣A、B所進行的運算即為「卷積（Convolution）」，矩陣K在深度學習中稱之為「卷積核（Convolution　kernel）」，其作用即在於萃取出資料特徵。藉此，我們達成了利用數學運算來擷取圖像特徵，也可以理解到為何深度學習能夠過濾資料雜訊而完成圖像識別。

課堂任務

思考一下，如果你想要區分「及」兩個符號，你要如何制訂你的卷積核？你能驗證你的卷積核夠強韌嗎？

+1	+1
+1	-1

矩陣 A

-1	+1
+1	+1

矩陣 B

6-3 卷積神經網路

卷積神經網路（Convolutional Neural Network, CNN）為目前深度學習技術中極具代表性的類神經網路結構之一，其優勢在於無需人力介入訂定特徵值，可直接處理原始數據。上述提及之ImageNet圖像識別競賽中，許多成功的模型均是基於卷積神經網路，而AlphaGo也是利用CNN來判斷落子位置與評估當前勝率。

卷積神經網路的基本架構

卷積神經網路（以下簡稱CNN）的基本架構大致分為卷積層（Convolution layer）、池化層（Pooling layer）與全連接層（Fully connected layer），其典型結構如下圖所示。

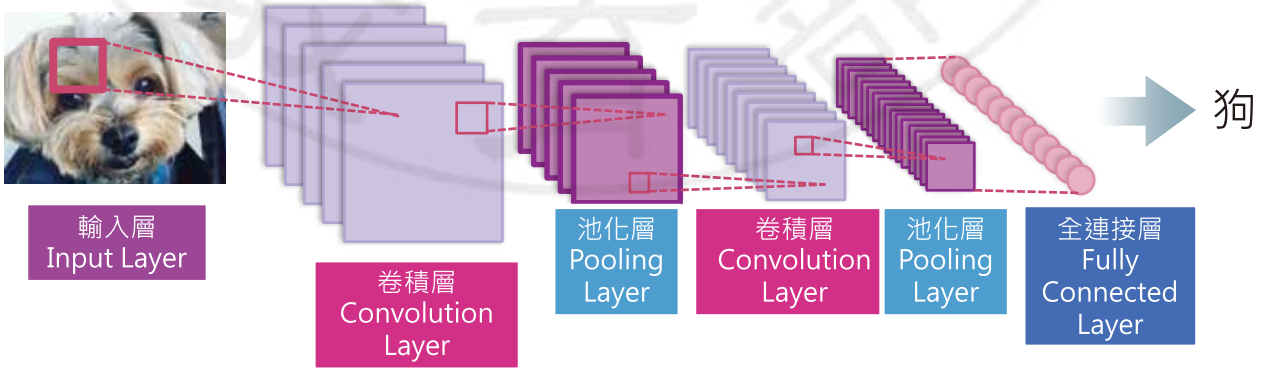


圖6-5 卷積神經網路的基本結構

在實務應用上，常會使用CNN來進行圖片的辨識，因此在輸入層上會使用圖片作為輸入。而一張圖片會由許多像素組成，以下是一個以「矩陣」形式所呈現的概念圖：

3	1	2	5	8	4
4	5	2	6	1	3
1	3	2	1	4	6
2	5	8	5	4	2
2	4	2	1	3	5
4	5	1	1	3	2

圖6-6 **輸入層的圖片以矩陣形式呈現**（僅為概念而非真實對應）

卷積層（Convolution layer）

在卷積層的運作中，最重要的就是透過卷積核，在原始影像上進行卷積運算，藉此萃取原始影像上的特徵。卷積核可以與一般神經網路相同，隨機產生矩陣，倘若現有一個卷積核如右：

-2	3	1
2	2	-3
1	-1	-1

則依據前述提到的「阿達馬乘積」，在原始影像中取出與卷積核相同維度的矩陣進行乘積，可得到：

3	1	2	5	8	4
4	5	2	6	1	3
1	3	2	1	4	6
2	5	8	5	4	2
2	4	2	1	3	5
4	5	1	1	3	2

$\circ$

-2	3	1
2	2	-3
1	-1	-1

$=$

-6	3	2
8	10	-6
1	-3	-2

接著，將卷積運算後得到的矩陣中所有元素值進行加總：(−6)+3+2+8+10+(−6)+1+(−3)+(−2)＝7，此為原始影像經過第一次卷積運算後得到的結果。

在上述的範例中，我們從原始影像的最左上角開始進行卷積運算，接著可繼續向右平移一格，再進行一次卷積運算如下：

3	1	2	5	8	4
4	5	2	6	1	3
1	3	2	1	4	6
2	5	8	5	4	2
2	4	2	1	3	5
4	5	1	1	3	2

。

-2	3	1
2	2	-3
1	-1	-1

＝

-2	6	5
10	4	-18
3	-2	-1

延續先前步驟，一樣將卷積運算後得到的矩陣中所有元素值進行加總：(−2)+6+5+10+4+(−18)+3+(−2)+(−1)＝5。

如果繼續往右平移一格直到原始影像的右邊界，則可以得到以下的四個矩陣乘積及其元素值加總結果。

卷積運算結果	-6	3	2	-2	6	5	-4	15	8	-10	24	4
	8	10	-6	10	4	-18	4	12	-3	12	2	-9
	1	-3	-2	3	-2	-1	2	-1	-4	1	-4	-6
元素值加總	**7**			**5**			**29**			**14**		

當然除了水平方向平移外，也可以在垂直方向繼續平移，例如向下方平移一格，可以得到：

3	1	2	5	8	4
4	5	2	6	1	3
1	3	2	1	4	6
2	5	8	5	4	2
2	4	2	1	3	5
4	5	1	1	3	2

。

-2	3	1
2	2	-3
1	-1	-1

＝

-8	15	2
2	6	-6
2	-5	-8

接著，將卷積運算後得到的矩陣中所有元素值進行加總：(−8)+15+2+2+6+(−6)+2+(−5)+(−8)＝0，此為原始影像經過第一次卷積運算後得到的結果。倘若繼續往右平移一格，直到原始影像的右邊界，亦可得到如下四個矩陣及其元素值加總結果。

卷積運算結果	-8	15	2	-10	6	6	-4	18	1	-12	3	3
	2	6	-6	6	4	-3	4	2	-12	2	8	-18
	2	-5	-8	5	-8	-5	8	-5	-4	5	-4	-2
元素值加總	**0**			**1**			**8**			**-15**		

如此一來，原先維度6×6的原始影像矩陣，經過維度3×3的卷積核進行運算後，就可以得到16個矩陣乘積及其所有元素值加總結果。

卷積運算結果	-6	3	2	-2	6	5	-4	15	8	-10	24	4
	8	10	-6	10	4	-18	4	12	-3	12	2	-9
	1	-3	-2	3	-2	-1	2	-1	-4	1	-4	-6
元素值加總	**7**			**5**			**29**			**14**		

卷積運算結果	-8	15	2	-10	6	6	-4	18	1	-12	3	3
	2	6	-6	6	4	-3	4	2	-12	2	8	-18
	2	-5	-8	5	-8	-5	8	-5	-4	5	-4	-2
元素值加總	**0**			**1**			**8**			**-15**		

卷積運算結果	-2	9	2	-6	6	1	-4	3	4	-2	12	6
	4	10	-24	10	16	-15	16	10	-12	10	8	-6
	2	-4	-2	4	-2	-1	2	-1	-3	1	-3	-5
元素值加總	**-5**			**13**			**15**			**21**		

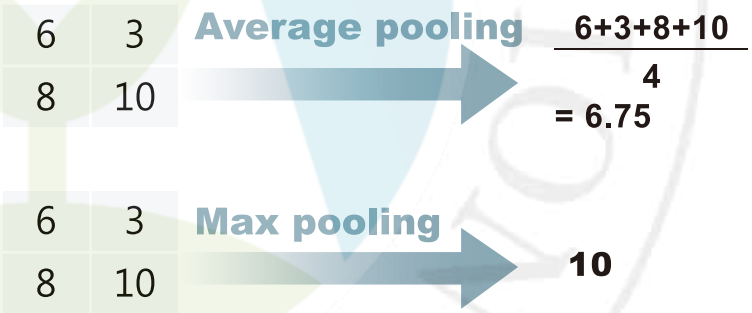
卷積運算結果	-4	15	8	-10	24	5	-16	15	4	-10	12	2
	4	8	-6	8	4	-3	4	2	-9	2	6	-15
	4	-5	-1	5	-1	-1	1	-1	-3	1	-3	-2
元素值加總	**23**			**31**			**-3**			**-7**		

因此，藉由上述步驟將矩陣中所有元素值加總後，可以得到一個4 × 4的矩陣，此矩陣也是原始影像透過卷積核進行卷積運算後得到的結果，如右：

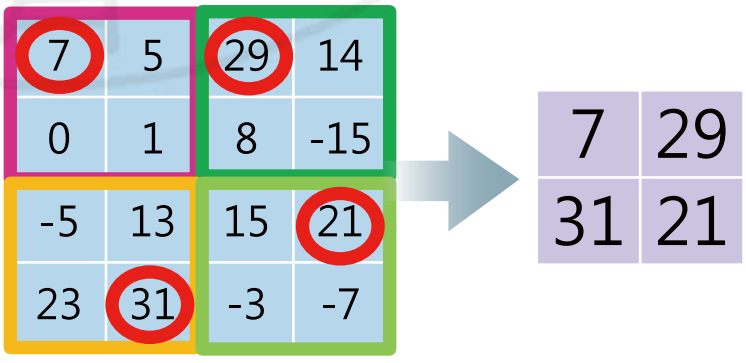
7	5	29	14
0	1	8	-15
-5	13	15	21
23	31	-3	-7

池化層 Pooling layer

經過卷積層後得到的矩陣運算結果就會進入池化層，而池化層的主要目的是保留主要特徵與降維。池化有許多種做法，在此挑其中兩種講解，分別是「平均值池化（Average pooling）」與「最大值池化（Max pooling）」。平均值池化以計算目標矩陣中所有元素的平均值作為新元素；而最大值池化則以目標矩陣中的元素最大值作為新元素。



現今大多使用最大值池化的做法來降低維度，因為這樣能保留最顯著的特徵，計算上也較為簡便。延續卷積層所使用的範例，經過最大值池化後的結果如下（在此以維度2 × 2作為池化層中子矩陣大小之設定）：



原始的影像經過卷積運算後，最後得到的結果就是以該卷積核在原始影像中獲得的特徵圖（Feature map）。

3	1	2	5	8	4
4	5	2	6	1	3
1	3	2	1	4	6
2	5	8	5	4	2
2	4	2	1	3	5
4	5	1	1	3	2

。

-2	3	1
2	2	-3
1	-1	-1

=

7	29
31	21

由最後得到的特徵圖所呈現的矩陣內容可發現，經過池化層不但保留了卷積運算後各子矩陣中的主要特徵，另外也從原始影像為維度6×6的矩陣，變成維度2×2的特徵圖矩陣。呼應前述所說，保留主要特徵及降低維度之效。

課堂任務

任務一：

若右方矩陣A代表原圖影像，矩陣K代表卷積核，請寫出矩陣A與矩陣K進行卷積運算後得到的結果為何？（請依序寫出尚未進入池化層前的卷積運算結果）

2	1	4	6	0
8	5	4	2	10
2	1	3	5	9
1	1	3	2	0
7	6	0	4	11

矩陣 A

−2	3
4	−1

矩陣 K

任務二：

延續任務一的卷積運算結果，若在此以最大值池化為作法，則經過池化層後得到的特徵圖為何？（在此以維度2×2作為池化層中子矩陣大小之設定）

關於卷積核的兩三事

經過上述的介紹，對於CNN的運作機制是不是更加清楚了！針對愈大量、愈複雜的圖片，我們也能透過增加的卷積核的數量（在此稱為「深度（Depth）」）來萃取出不同的特徵，讓訓練出來的模型更加準確。

圖6-7 **藉由不同的卷積核萃取出原始影像中的各種特徵**

一般來說，卷積核的選擇通常可以隨機產生，進而與原始影像的矩陣進行卷積運算，但有時候也可以透過一些特殊的矩陣來提取「邊緣特徵」，最有名的例子就是「Prewitt kernel」：

$$G_X = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} 、 G_Y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

其中 G_x 用於提取水平邊緣、G_y 用於提取垂直邊緣，這樣的卷積核設計使得對原始影像做卷積運算產生的結果，如同邊緣探測的效果，如下圖6-8，左邊是原圖，右邊則是邊緣探測後的結果。

圖6-8 **左邊為原圖；右邊為經過Prewitt kernel卷積運算後的圖像**

另外，卷積核的大小也能夠自行定義，例如下方以兩種不同維度的卷積核（分別是3×3與2×2）在原始影像的矩陣中進行卷積運算，假設在卷積運算中每次都位移一格，則維度3×3的卷積核在此可以得到36個卷積運算的矩陣結果，而維度2×2的卷積核則有49個。倘若再經過池化層後，最後得到的特徵圖，其矩陣維度分別為6×6與7×7。藉此可發現，若選擇的卷積核維度較大，卷積運算的數量較少，速度較快，但在特徵萃取的精準度上也不如卷積核維度較小的效果好。

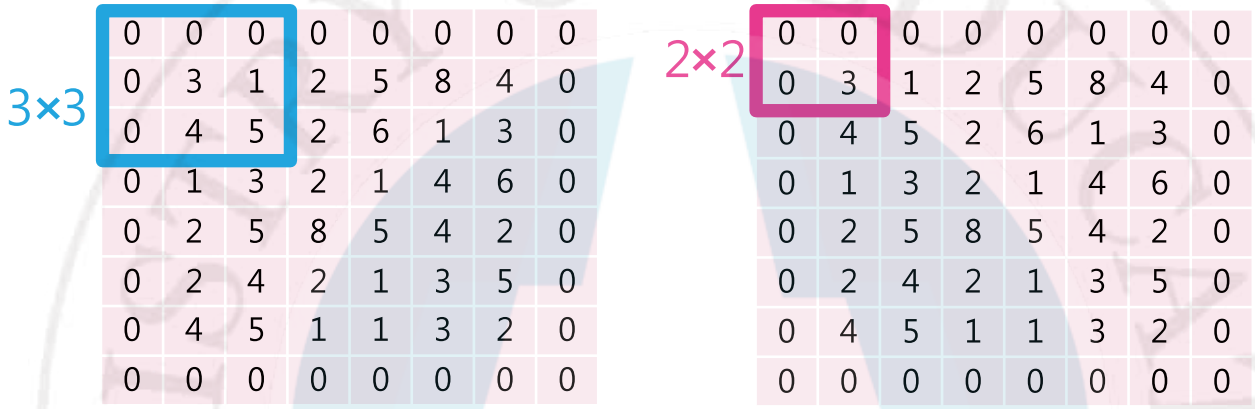


圖6-9 左圖以維度3×3；右圖以維度2×2的卷積核進行卷積運算

除了可以控制卷積核的維度外，在進行卷積運算的過程中，還可以設定每次卷積核移動的步伐（stride）。由前面講述卷積運算原理的例子可以看出，每次卷積核在原始影像的矩陣上做完卷積運算後，都會往右或往下「平移一格」，這樣的平移就是以步伐「stride = 1」的概念進行運作。然而在卷積運算的過程中，也可以調整卷積核移動的步伐大小，以下圖為例，就是將卷積核移動的步伐設定為2。因為卷積核移動的步伐變大，如此一來可以減少卷積運算的數量使整體運算速度加快，但同樣的在特徵萃取之精準度上也不如卷積核移動步伐較小的效果好。

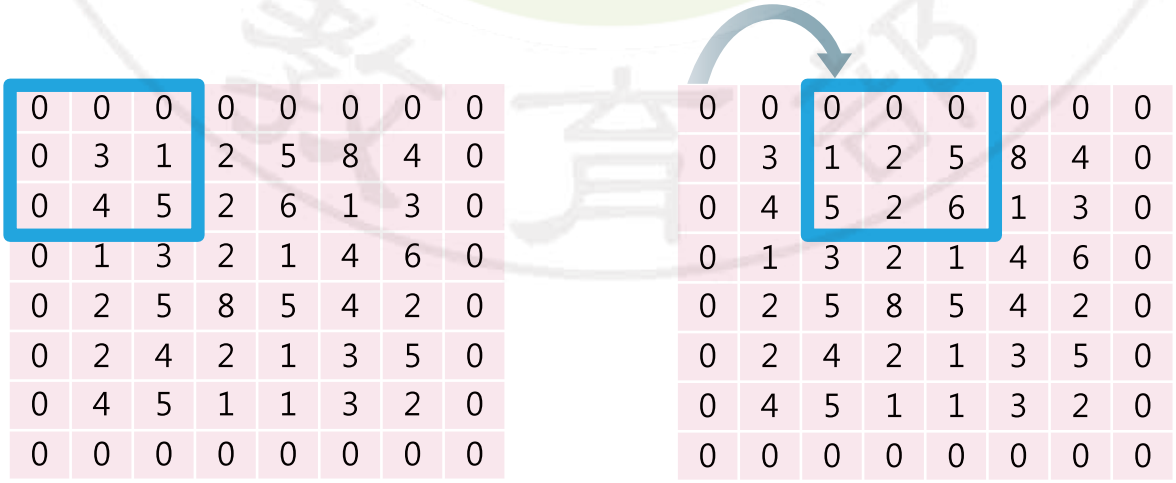


圖6-10 設定卷積核在原始影像的矩陣上移動步伐為2

經過卷積層與池化層運算後得到的特徵圖，會發現在矩陣的維度上變小了，倘若後續要繼續進行再一次的卷積、池化，則發現原本萃取出來的特徵圖邊緣區域，接下來再進行卷積運算時，無法作為卷積核的中心（如下圖）。這麼一來，特徵圖邊緣在整個卷積運算上的影響力就變小了，為了解決這個問題，可以採取邊緣補零（Zero-padding）的方法讓特徵圖與原圖的大小一致。

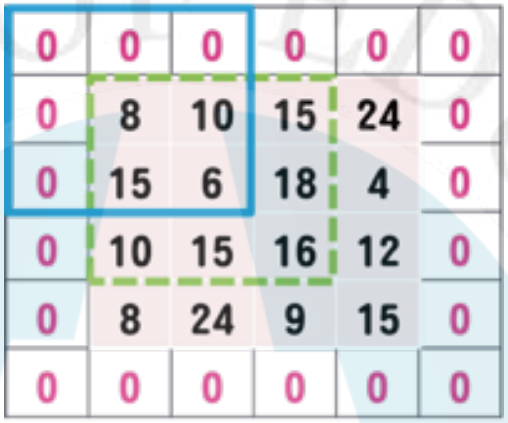


圖6-11 **藉由邊緣補零強化特徵圖邊緣的影響力**

激活函數

實務上，我們在做卷積運算後通常會將得到的特徵圖以一個函數做整理，讓各神經元的參數能更快地達到收斂（愈快訓練完成），同時提高分類的準確率。在一般類神經網路的概念，我們稱這個函數為「激活函數（Activation function）」。

激活函數有許多種選擇，ReLU（Rectified Linear Unit, 線性整流函數）即為使用率最高的激活函數之一。ReLU的數學定義為f(x)=max(0,x)，簡單來說就是讓輸入值為0以下的值歸零，而0以上值不變。

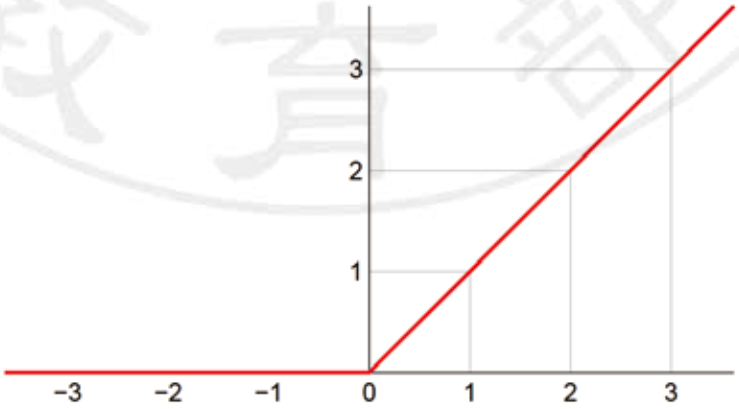


圖6-12 **ReLU激活函數**

下圖ReLU激活函數的使用範例，在萃取特徵的過程中，獲得的值越大代表特徵越明顯，ReLU函數將負數變為0也是凸顯特徵的做法之一。

圖6-13 **ReLU激活函數使用範例**

全連接層（Fully connected layer）

進行了數層的卷積與池化後，接下來要把結果放入全連接層中運算。全連接層基本上與本章開頭提到的類神經網路相同，但在CNN中，要將以上結果放到普通的類神經網路前，還需要進行攤平（Flatten），讓許多二維的特徵圖可以變成單一個一維的矩陣。假設在經過一連串的卷積與池化後，我們得到3個維度為2x2的特徵圖，則可進行以下操作：

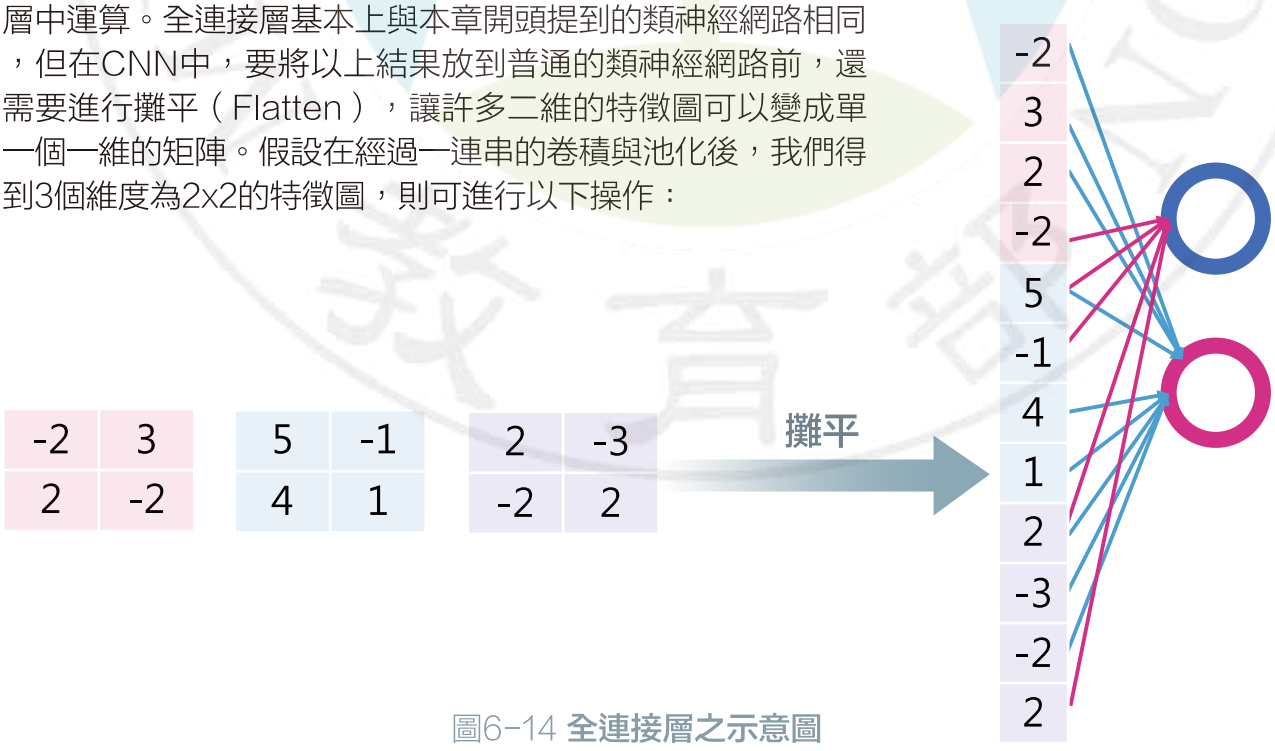


圖6-14 **全連接層之示意圖**

最後將攤平後的一維矩陣用來當成神經網路的輸入做訓練，就是整個CNN的運作過程。至於需要多少卷積層、池化層，與在全連接層中應該使用多少層、每層多少神經元，也都是需要考慮的問題，但具體來說目前仍是依照經驗去設定，沒有嚴謹的理論參考，只能依照應用的不同，依經驗去做調整。

本章節在介紹時多以圖像資料為例，但透過上述對於矩陣運算的過程，我們可以發現：只要原始數據的相鄰元素間存在某些關係，便能使用卷積式類神經網路進行學習，如聲音、文字等等；反之，若原始數據的相鄰元素之間並不存在任何關連性，則不適用於卷積式類神經網路。如一個包括姓名、電話、收入、顧客購買記錄的資料庫，這些欄位之間並沒有任何的關聯性，因此利用卷積式類神經網路並無法很有效的學習。

上述範例中，為了方便解說，所有的卷積核均為事先設定。然而，透過不斷隨機產生卷積核、測試其辨識特徵成效、再進行學習調整卷積核，CNN便神奇的幫我們省去了這一困難的工作。總之，CNN的本質即是在透過數學運算來找出資料存在的特徵，進而對資料進行識別分類，相較於第三章提到的監督式學習，同樣用來處理分類問題，確有著截然不同的作法與效果，這或許就是人工智慧的奧妙之處，等待我們繼續細細品味與挖掘。

Note

感謝 Thanksgiving

以下人員參與教材教案討論，提供專業建議與共同編撰內容：

成功大學電機系：李修齊同學、劉發元同學

南台科技大學資工系：李育強教授

台中科技大學資工系：張家瑋教授

台南二中：鄭忠煌校長、蔡安靖老師
謝明佳老師、謝敏雄老師
陳國雄老師、黃楨智老師
莊吳慧瑩老師、郭育里老師
宋政蒲老師、顏黃柔老師

台南一中：顏永進老師

玉井工商：洪志菁老師

聖功女中：毛全良老師

德光高中：王秀鶯老師

書　　名：和 AI 做朋友-相知篇
副 標 題：從 0 開始學 AI
著　　者：黃仁暐、涂益郎
主　　編：李建樹
副 主 編：李育強
執行編輯：陳虹伶、陳瑞翎
封面設計：君傳媒工作室
插圖設計：君傳媒工作室、設計超市多媒體 DESIGNGOGO、夢享製造所
發 行 人：潘文忠
發　　行：教育部
出版機關：教育部
地　　址：臺北市中正區中山南路 5 號
電　　話：02-7736-6666
網　　址：https://www.edu.tw/
出版年月：中華民國 108 年 8 月
版　　次：初版
其他類型版本說明：本書另有電子版本，請至教育部教育雲網站（https://cloud.edu.tw/）申請。
定價：新臺幣 150 元

GPN：1010801347　　ISBN 978-986-05-9905-3

本計畫所規畫之人工智慧(AI)導入高級中學之教材，目的為使未接觸過人工智慧相關知識之高中學子，了解人工智慧的基本概念和原理，並搭配應用與實作課程，使學生在學習過後具備有運用人工智慧的相關技術幫助解決日常生活中問題的能力。各校可規劃於校本課程之校定必修規劃實施，或於多元選修中開設提供有興趣的學生選修。

AITC
教育部人工智慧技術及應用人才培育計畫
Artificial Intelligence Talent Cultivation Program

教育部AI技術及應用人才培育計畫
中小學分項計畫｜中小學推廣教育