對本書的讚譽

「我希望本書在我一開始從事機器學習生產時就已經存在了！本書為全面瞭解 ML 系統生產（尤其是 TFX）的最佳資源。Hannes 和 Catherine 直接與 TensorFlow 團隊合作獲取最準確的資訊，並將其含括在本書中，且用清晰與簡潔的說明和範例來進行介紹。」

—*Robert Crowe, Google TensorFlow 開發者與擁護者*

「資料科學從業者皆知——現實世界的機器學習不僅只涉及機器學習模型的訓練。本書揭開現代機器學習工作流程中隱藏之技術債的神秘面紗，將使您得以將實驗室和工廠資料科學模式投入至可重複的工作流程中。」

—*Josh Patterson, Patterson Consulting CEO*
Deep Learning: A Practitioner's Approach 與 Kubeflow Operations Guide 的共同作者

「如您想瞭解如何建構自動化、可擴展與可重現的 ML 管道，本書絕對值得一讀！無論您是資料科學家、機器學習工程師、軟體工程師還是 DevOps，您都能從中學到一些有用的知識。另外，其涵蓋 TFX 及其元件的最新功能。」

—*Margaret Maynard-Reid，機器學習工程師，*
Tiny Peppers, ML GDE（Google 開發者專家）, GDG 西雅圖主要組織者

《建構機器學習管道》可讀性極佳，其不僅是一本幫助資料科學家與 ML 工程師建構自動化與可重複 ML 管道的綜合指南，更是該主題唯一的權威書籍。本書概述成功建構 ML 管道所需定義的元件，並以實用的方式引導您完成範例程式碼。」

—*Adewale Akinfaderin，AWS 資料科學家*

「我真的很喜愛《建構機器學習管道》這本書。隨著 TFX 的不斷發展，且在 Google 內部使用 TFX 好多年，我必須說好希望那時就能擁有此書，而不是自行解決痛點。此書本該為我省去好幾個月的努力和困擾。感謝作者完成一本高品質的使用指南！」

—*Lucas Ackerknecht*，機器學習專家
Anti-Abuse Machine Learning，*Google*

「我們身邊存在著一些令人驚嘆的原型模型（prototype models）。本書將介紹可幫助原型投入生產的工具與技術。不僅如此，還可建構圍繞該原型之完整端對端管道，以便在未來自動化並順利交付出任何增強功能。對於希望將自身技能提升至新水平或與更大團隊合作，用以實現創新模型價值之 ML 初學者而言，這會是一本非常棒的書籍。」

—*Vikram Tiwari*，*Omni Labs, Inc.* 共同創辦人

「身為一位只使用 TensorFlow 為深度學習模型框架的本人而言，閱讀此書後對 TensorFlow 生態系統所提供的管道功能感到驚訝。本書是關於 TFX 用於分析與部署的所有工具之最佳指南，對於希望使用 TensorFlow 來完成第一個機器學習管道的人來說，它易於閱讀和使用。」

—*Jacqueline Nolis* 博士
Brightloom 首席資料科學家與 *Build a Career in Data Science* 的共同作者

「本書提供機器學習的深入探討。您將找到關於建構生產就緒的 ML 基礎架構，並提供深富說服力與實際使用的範例介紹。對於任何打算將 ML 應用於現實世界問題的工程師或資料科學家而言，我認為此書必讀。」

—*Leigh Johnson*
Slack 機器學習服務部門工程師

前言

每個人都在談論機器學習。它已經從一門學科搖身一變成為最令人興奮的技術之一。從理解汽車自動駕駛的影像訊號到個人化的藥物治療，機器學習在每個產業都變得非常重要。雖然模型架構和概念已經得到了很多關注，但機器學習還沒有像過去 20 年來軟體產業所經歷的標準化流程。本書將告訴您如何建構標準化的機器學習系統，此系統是自動化的，其結果為可重複性的。

什麼是機器學習管道？

近年來，機器學習領域的發展令人驚嘆。隨著圖形處理器（GPU）的普及，以及類似像 Transformers 的深度學習概念的興起，例如 BERT（*https://arxiv.org/abs/1810.04805*），或 Generative Adversarial Network（GANs），如深度卷積 GANs，人工智慧（AI）專案數目急遽上升與 AI 新創公司數量日益增加。有越來越多的組織將最新的機器學習概念應用到各種商業問題中。在這種對高性能機器學習解決方案的競相追逐中，我們發現了一些不太受眾人關注的部分——即資料科學家和機器學習工程師缺乏良好的概念和工具，無法加速、再利用、管理和部署模型。我們需要的是機器學習管道的標準化。

機器學習管道是指採加速、再利用、管理與部署方式實作機器學習模型，並公式化此流程。十幾年前，隨著持續整合（CI）和持續部署（CD）的導入，軟體工程也經歷了同樣的變化。在過去，測試和部署一個 Web 應用程式是一段漫長的過程。如今，這些流程已被許多工具與概念大幅簡化。以前的 Web 應用程式部署需要 DevOps 工程師和軟體開發人員之間的合作。而目前應用程式則可在幾分鐘之內完成可靠的測試與部署。資料科學家和機器學習工程師可以從軟體工程中，學到許多關於工作流程的相關知識。引導

讀者從頭到尾瞭解整個機器學習管道，為機器學習專案的標準化做出貢獻則是本書最大目的。

依據個人經驗，多數將模型部署到生產的資料科學專案並不具備龐大的團隊。這使得內部很難從一開始就建立整個管道。這意味著機器學習專案將成為一次性的工作，而其中的模型表現會在經過一段時間後開始發生問題。資料科學家在底層數據發生變化時需花費大量時間來修復錯誤，而導致模型並未被廣泛運用。一個自動化、可重複的管道可以減少部署模型所需的心力。該管道應包括以下步驟：

- 有效地修改您的數據，並啟動新的模型訓練
- 驗證接收到的數據並檢查數據漂移（drift）情況
- 為模型訓練與驗證有效地預處理資料
- 有效地訓練機器學習模型
- 追蹤模型訓練
- 分析和驗證訓練和調整後的模型。
- 部署已驗證的模型
- 放大部署的模型
- 捕捉新的訓練數據，並透過反饋循環建立模型性能指標

上述列表遺漏一個重點：選擇模型架構，並假設您已經對此步驟擁有良好的工作知識。如您正開始學習機器或深度學習，以下參考資訊是熟悉機器學習很好的起點。

- 《*Fundamentals of Deep Learning: Designing Next-Generation Machine Intelligence Algorithms*》作者：Nikhil Buduma and Nicholas Locascio（O'Reilly）。繁體中文版《*Deep Learning 深度學習基礎｜設計下一代人工智慧演算法*》由碁峰資訊出版
- 《*Hands-On Machine Learning with Scikit-Learn, Keras*》，作者：Aurélien Géron（O'Reilly）。繁體中文版《*精通機器學習：使用 Scikit-Learn, Keras 與 TensorFlow 第二版*》由碁峰資訊出版

本書適合的讀者

本書鎖定的讀者是希望將將資料科學專案產品化，而不是訓練一次性之機器學習模型的資料科學家和機器學習工程師。這些讀者應該熟悉基本的機器學習概念，並至少熟稔一種機器學習框架（如 PyTorch、TensorFlow、Keras）。本書中的機器學習範例是基於 TensorFlow 與 Keras，但核心觀念可以運用於任何框架。

本書的次要讀者為希望加速資料科學專案開發的資料科學專案的經理、軟體開發人員與 DevOps 工程師。若想多瞭解自動化機器學習的生命週期，並使您的組織受益，本書將介紹一個工具集來回答這個問題。

為什麼選擇 TensorFlow 與 TensorFlow Extended？

本書所有管道的範例說明都將使用 TensorFlow 生態系統中的工具，尤其是 TensorFlow Extended（TFX）。選擇這個框架有許多重要原因：

- TensorFlow 是在撰寫本書時最廣泛用於機器學習的生態系統，包括多個有用的專案和支援套件。除了核心功能外，還包含如 TensorFlow Privacy 和 TensorFlow Probability。
- 在小型和大型的產品生產社群深受歡迎並廣泛被使用，且擁有一個由使用者組成的活躍社群。
- 從學術研究到產業運用的機器學習都有支援的案例。TFX 與 TensorFlow 平台核心緊密整合並支援生產的使用。
- TensorFlow 和 TFX 皆為開源工具，在使用方面沒有限制。

然而，本書描述的所有原則也與其他工具和框架相關。

各章概述

每一章將介紹建構機器學習管道的具體步驟，並透過案例專案來示範這些步驟如何進行。

第 1 章：導論　介紹機器學習管道的概況，討論何時該使用它們，描述構成管道的所有步驟，並介紹將在本書中使用的範例專案。

第 2 章：TensorFlow Extended 簡介　介紹 TFX 生態系統，解釋任務之間如何相互溝通，並描述 TFX 內部元件如何工作。另外，說明 ML MetadataStore 及其在 TFX 環境中的使用情況，以及 Apache Beam 如何在背後執行 TFX 元件。

第 3 章：數據擷取　討論如何以一致的方式將數據匯入至管道中，亦包括數據版本的概念。

第 4 章：數據驗證　說明 TensorFlow 數據驗證，如何有效驗證導入管道的數據。當新數據與之前的數據發生了實質性的變化，可能會影響您的模型性能時，此步驟能給您適當的警訊。

第 5 章：資料預處理　主要介紹使用 TensorFlow Transform，將原始數據轉換為適合訓練機器學習模型特徵的資料預處理（特徵工程）。

第 6 章：模型訓練　討論如何在機器學習管道中訓練模型，並說明模型調校的概念。

第 7 章：模型分析和驗證　介紹瞭解生產中模型的有用指標，包括那些讓您發現模型預測發生偏誤的指標，並提供解釋模型預測的方法。第 122 頁的「TFX 中的分析和驗證」，說明當新模型可改善模型表現時的版本控管。管道中的模型可以自動更新到新版本。

第 8 章：TensorFlow Serving 的模型部署　重點介紹如何有效地部署機器學習模型。從簡單的 Flask 實作開始，我們將強調這種自訂模型應用的局限性，並介紹 TensorFlow Serving 以及如何配置您的服務實例。還將討論批次處理的功能，並在請求模型預測時說明如何進行客戶端設定。

第 9 章：TensorFlow Serving 的高級模型部署　討論如何優化模型部署及如何進行監控。此章節涵蓋優化 TensorFlow 模型以提高性能的策略，並使用 Kubernetes 進行基本的部署設定。

第 10 章：進階 TensorFlow Extended　介紹機器學習管道客製化元件的概念，則可將不受限於 TFX 的標準元件的功能。無論是想加入額外的數據擷取步驟，或是將導出的模型轉換為 TensorFlow Lite（TFLite），本章節將說明創建此元件的必要步驟。

第 11 章：管道第一部分：Apache Beam 與 Apache Airflow　接續前幾章的內容，本章將討論如何將設定的元件變成管道，及如何為您選擇的編排平台進行設定，並說明在 Apache Beam 與 Apache Airflow 運作的端對端管道。

第 12 章：管道第二部分：Kubeflow 管道　延續上一章的內容，並透過 Kubeflow 管道和 Google 的 AI 平台說明端到端管道。

第 13 章：反饋循環　討論如何將模型管道，轉換為可透過最終產品使用者的反饋進行改進的循環。本章將討論該捕捉何種型態的數據改進為未來模型版本，及如何將數據反饋至管道中。

第 14 章：機器學習的數據隱私　介紹重要性快速成長的機器學習隱私保護領域，並討論三種重要的方法：差別隱私、聯合學習和加密機器學習。

第 15 章：管道的未來與下一步　提供機器學習管道在未來對技術發展產生的影響，以及該如何思考未來機器學習工程的變化。

附錄 A：機器學習的基礎架構介紹　對 Docker 和 Kubernetes 進行簡要介紹。

附錄 B：在 Google Cloud 上設置 Kubernetes 集群　提供在 Google Cloud 上設置 Kubernetes 的補充資料。

附錄 C：操作 Kuberflow 管道的技巧　介紹操作 Kubeflow 管道設置的實用技巧，並包括 TFX 命令列的簡介。

本書編排方式

本書使用下列的編排方式：

斜體字（*Italic*）

代表新術語、URL、email 地址、檔名，與副檔名。中文以楷體表示。

定寬字（`Constant width`）

在長程式中使用，或是在文章中代表變數、函式名稱、資料庫、資料型態、環境變數、陳述式、關鍵字等程式元素。

定寬粗體字（**`Constant width bold`**）

代表應由使用者親自輸入的命令或其他文字。

定寬斜體字（*`Constant width italic`*）

應換成使用者提供的值，或由上下文決定的值的文字。

這個圖案代表提示或建議。

這個圖案代表註解。

第一章

導論

本章將介紹何謂機器學習管道：概述建立管道的所有步驟，並解釋將一個機器學習模型從實驗階段轉移至穩健的生產系統需要進行哪些步驟。本章將展示專案範例，並在其他章節使用該範例說明本書所提出的原則。

為何要選擇機器學習管道？

機器學習管道最主要的優勢在於模型生命週期的自動化。當加入新的訓練數據集時，應觸發包括數據驗證、資料預處理、模型訓練、分析和部署等工作流程。我們發現為數眾多的資料科學團隊手動完成上述步驟；這不僅成本高昂，更是許多錯誤的來源。接著將說明採用機器學習管道的許多優點：

更專注於新模型開發，而不只是維護現有模型

自動化的機器學習管道將使資料科學家從維護現有模型中解放出來。我們觀察到有太多的資料科學家將時間花在維護過去所開發的模型上。他們手動執行腳本（script）來預處理訓練數據集、手動調整他們的模型，並編寫一次性的部署腳本。自動化管道將允許資料科學家從事工作中最有趣的部分—開發新模型，進而提高人員在競爭激烈的就業市場的工作滿意度與留職率。

預防程式碼的錯誤

自動化管道可避免程式碼的出錯，誠如將在後續章節所介紹的，新模型將與一組被版本控管的數據資料綁定；而資料預處理則與開發模型綁定。這意指：如果收集到新的數據，將產生新模型；如「資料預處理」被更新，訓練數據將變得無效，亦產生新模型。在手動機器學習工作流程中，一個常見的錯誤（bug）來自於當在模型訓

練之後改變「資料預處理」步驟。在這種情況下，我們會部署一個來自不同的處理指令的模型，而這個模型則與過去訓練的模型不同。這些錯誤是很難除錯的，因為模型的設定仍然可以運作，但結果可能不正確。有了自動化的工作流程，這些錯誤即可避免。

有用的文件追蹤

實驗追蹤與模型發佈管理會產生模型發生變化的文件追蹤紀錄。實驗將會記錄模型超參數（hyperparameter）的變動、採用的數據集及由此產生的模型測量指標（如損失（loss）或準確度（accuracy））的變化。模型發佈管理將追蹤最後部署哪個模型。當資料科學團隊重新創建模型或追蹤模型的性能時，這樣的文件追蹤將更有價值。

標準化

具標準化的機器學習管道可以改善資料科學團隊的工作體驗。由於標準化流程的設置，資料科學家可迅速到職與轉換團隊、找到共同的開發環境、提高開發效率，並降低新專案設置的時間。另外，投入設置機器學習管道的時間亦可改善人員的留存率。

管道的商業案例

自動化機器學習管道的實施將為資料科學團隊帶來三個關鍵影響：

- 更多的新模型開發時間
- 更簡單的模型更新流程
- 減少重置模型的時間

上述部分將大幅降低資料科學專案的成本。進一步而言，自動化機器學習管道亦將：

- 檢查數據集或訓練模型中的潛在偏誤。偏誤的發現可避免與模型相關之人員對其造成傷害。例如，由機器學習驅動的亞馬遜（Amazon）履歷篩選器（*https://oreil.ly/39rEg*）被發現對女性求職者產生偏見。
- 如因數據保護法（如歐洲通用數據保護條例（Europe's General Data Protection Regulation, GDPR））發生問題時，使用文件追蹤（透過實驗追蹤和模型發佈管理）將提供助益。
- 為資料科學家騰出更多的開發時間，並提升工作滿意度。

何時該考慮機器學習管道？

機器學習管道具有多種優勢，但並非每個資料科學專案都需要管道。有時，資料科學家只是想嘗試新的模型、研究新的模型架構，或重現一個最新的應用等。在這些情況下，管道就派不上用場。然而，當模型正在使用時（例如，正在應用程式中執行的模型），就需要持續更新和微調。在這種情況下，又回到前面所討論的持續更新模型和減輕資料科學家負擔的場景之中。

隨著機器學習專案的蓬勃發展，管道的角色也變得更加重要。如需要大量的數據集或資源需求時，本書所討論的方法就可以輕鬆進行基礎架構擴展。當「重複性」是重要的考量時，則可透過自動化的機器學習管道和審查追蹤來提供。

概述機器學習管道的步驟

機器學習管道從擷取新的訓練數據開始，到接收新訓練模型的某種反饋而結束。此反饋可以是某種性能指標或是產品使用者的反饋。管道包括許多步驟，包括資料預處理、模型訓練、模型分析及模型部署。您可以想像如手動進行這些步驟會有多麻煩，且容易出錯。本書將介紹一些工具與解決方案來自動化機器學習管道。

從圖 1-1 中可以看出，管道其實是一種往復循環的過程，其不斷地收集數據並更新機器學習模型。更多的數據流入意味著模型持續地改良。而由於數據的不斷的更新，自動化即是其中的關鍵。在現實狀況的應用，必須經常重新訓練模型。如果您不這樣做，在多數情況下因訓練數據與模型進行預測的新數據不同，模型準確率將會降低。如重新訓練為人工作業，即需手動驗證新訓練數據與分析更新後的模型，資料科學家或機器學習工程師將沒有時間為全新的業務問題開發新模型。

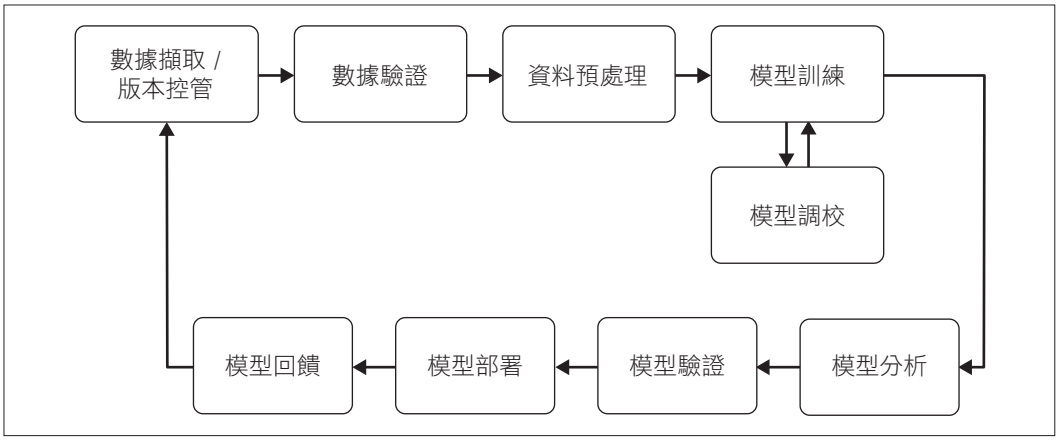

圖 1-1　模型生命週期

一個機器學習管道通常包括以下步驟：

數據擷取與數據版本控管

正如第 3 章所描述，數據擷取是每個機器學習管道的開始。在管道流程中，我們將數據轉換為下一個元件可以接受的資料格式。數據擷取並不執行任何特徵工程（這在數據驗證步驟之後發生）。這也是對導入數據進行版本控管的好時機——可將資料快照（data snapshot）與管道末端訓練完成之模型做連結。

數據驗證

在訓練新模型版本前，則需對新資料進行數據驗證。數據驗證（第 4 章）主要是檢查新數據的統計量是否符合預期（例如，全距、類別變數個數和類別變數的分佈）。當檢查到任何異常情況，「數據驗證」將會向資料科學家發出警訊。例如，當您訓練一個二元分類模型時，此訓練數據可能包含 50% 的 X 類別樣本與 50% 的 Y 類別樣本。如果類別間的分割發生變化，數據驗證工具將會提出警訊。當採非平衡的訓練集來訓練模型時，如樣本類別 X 或 Y 過多 / 不足，且資料科學家並無調整模型的損失函數，則模型預測可能會偏向優勢類別。

常用的數據驗證工具亦可比較不同的數據集。如您具有顯性標籤的數據集，並將該數據集拆分為訓練集和驗證集，則必須確保兩個數據集之間的標籤分割大致相同。數據驗證工具將允許您比較數據集與異常的部分。

如數據驗證發現任何異常，則可在此處停止管道並向資料科學家發出警告。如檢查出數據發生變化，則資料科學家或機器學習工程師可以更改各個類別的抽樣方式（例如，從每個類別中挑選相同數量的範例）、更改模型的損失函數，啟動新的模型建構管道，並重新啟動模型的生命週期。

資料預處理

您很有可能無法直接使用剛收集的數據進行機器學習模型的訓練。在大部分情況下，都必須對數據進行預處理以便執行模型訓練。標籤（label）通常需要轉換為 one-hot vector 或 multi-hot vector[1]，而此同樣適用於其他模型輸入。如果從文字數據訓練模型，則必須將文字字串轉換為索引（index），或將文字標記（text tokens）轉換為詞向量（word vector）。由於資料預處理只需要在模型訓練之前進行，而非在每個訓練期都需要，因此在訓練模型之前進行預處理才是最合理。

資料預處理工具的範圍可以從簡單的 Python 腳本到複雜的圖形工具。雖然大多數資料科學家關注於他們首選工具的處理能力，但對於預處理步驟的修改能夠與處理後的數據產生關聯也很重要，反之亦然。這意味著如果有人修改了一個預處理步驟（例如，在 one-hot vector 再增加一個標籤），之前的訓練數據應將變得無效，並強制更新整個管道。我們將在第 5 章描述這個步驟。

模型訓練與調校

模型訓練（第 6 章）是機器學習管道的核心。此步驟訓練模型，使其接受輸入並以極小化誤差的方式預測輸出。對於較大的模型，尤其是大的訓練集，此步驟可能很快變得難以管理。因電腦記憶體通常是進行計算時的有限資源，故有效分配資源對於模型訓練將變得至關重要。

模型調校在近期備受關注，因為它可以顯著地提高模型表現並提供競爭優勢。根據您的機器學習專案，您可以在開始機器學習管道前選擇調校模型，或者將模型調校作為管道的一部分並進行調整。因機器學習管道的基礎架構具有可擴展性（scalable），故可採平行（parallel）或按順序建立大量模型。其可為最後的生產模型挑選出最適模型超參數（hyperparameter）。

1 在以多個類別作為輸出的監督式分類問題中，通常需要將分類轉換為向量，例如 (0,1,0)，此為 one-hot vector，或者從分類列表轉換為向量，如 (1,1,0)，而此為 multi-hot vector。

模型分析

一般而言，我們會使用準確度（accuracy）或損失（loss）來決定最適模型參數集。但當確定模型的最終版本時，對模型的性能表現進行更深入的分析是非常有用的（在第 7 章中描述）。上述包括其他性能指標，如精確率（precision）、召回率（recall）和 AUC（area under the curve，曲線下面積），或計算出比訓練時使用之驗證集更大的數據集的性能指標。

模型需進行深入分析的另一個原因：需檢查模型預測是否公允。除非對數據集進行切片，並計算每個切片的性能，否則無法瞭解模型對不同使用者群體的表現。我們還可以深究模型對於用於訓練的特徵的相依性，並探索當改變訓練集的特徵時，模型預測會發生怎樣的變化。

與模型調校和最終選擇性能最好的模型類似，此步驟需要資料科學家的審查。然而，我們將示範如何將整個分析進行自動化，其中，只有最後的審查由人力完成。自動化將使模型的分析保持一致性，並可與其他模型分析進行比較。

模型版本控管

模型版本控管和驗證的目的是追蹤已被選定的模型、超參數集合與數據集，並做為下一次部署的版本參考。

當 API 中進行不兼容的更改或加入主要功能時，軟體工程中的語意化版本管理（semantic versioning）會要求您增加主要版本號；否則，增加次要版本號。模型發佈管理還有一個自由度：數據集。在某些情況下，透過在訓練過程中提供更多和 / 或更好的數據，則可在不更改單一模型參數或模型架構下，大大改善模型性能。而性能的提高是否需對主要版本進行升級？

儘管這個問題對每個資料科學團隊而言可能都不一樣，但將所有輸入註記在一個模型版本（超參數，數據集，模型架構）上，並在此發佈流程進行追蹤是相當重要的。

模型部署

模型在完成訓練、調校和分析之後，隨即進入重要的黃金階段。可惜的是太多模型都是一次性的部署，這使得更新模型變得非常困難。

現代的模型伺服器允許不用編寫 Web 應用程式即可完成模型部署。一般而言，它們提供了多個 API 介面（interface），如 representational state transfer（REST）或 remote procedure call（RPC）協定等，並允許同時托管同一模型的多個版本。同時托管多個版本允許在模型上進行 A/B 測試，並提供具參考價值的反饋。

模型伺服器還允許在不需重新部署應用程式的情況下更新模型版本，這將減少應用程式的停機時間，並減少應用程式開發和機器學習團隊之間的往復溝通。我們將在第 8 章和第 9 章中討論模型部署。

反饋循環

機器學習管道的最後一步經常會被忽略，但這對資料科學專案的成功與否至關重要——結束整個管道循環。我們還可以衡量新部署的模型之有效性與性能。在這個階段，我們可以捕捉關於模型性能的寶貴訊息。在某些情況下，還可以加入新的訓練數據來增加數據集並更新模型。這部分可能涉及人員參與或是自動完成。我們將在第 13 章中討論反饋循環。

除了兩個人工檢查的步驟（模型分析和反饋）之外，我們還可以自動化整條管道。資料科學家應該能更專注於新模型的開發，而非對現有模型的更新與維護。

資料隱私

在撰寫本書時，資料隱私的考量並不在標準的機器學習管道當中。預計隨著消費者對數據使用的關注越來越大，將引入新法律來限制個人數據的使用，現在的情況將在未來發生改變，並使得隱私保護法被整合至建構機器學習管道的工具之中。

我們將在第 14 章討論目前在機器學習模型中增加隱私保護的幾種選擇：

- 差異化隱私（differential privacy）：當中的數學確保模型預測不會洩露使用者資訊
- 聯合學習（federated learning）：原始數據不會離開使用者的裝置
- 加密機器學習（encrypted machine learning）：整個訓練過程在加密環境中進行，或對原始數據訓練的模型進行加密

管道編排（Pipeline Orchestration）

上一節提及的所有元件都需要被執行，也就是強調的：需被編排（Orchestrated），以便元件按正確順序執行。在執行元件之前，必須先計算每個元件的輸入。這些步驟的編排是由如 Apache Beam、Apache Airflow（在第 11 章中討論）或類似用於 Kubernetes 基礎架構的 Kubeflow 管道（在第 12 章中討論）之類的工具來執行。

在數據管道工具編排機器學習管道步驟的同時，管道的工件（artifact）儲存（如 TensorFlow ML MetadataStore）捕獲各個步驟的輸出。第 2 章將概述 TFX 的 MetadataStore，並瞭解 TFX 及其管道元件背後的原理。

為什麼要進行管道編排？

2015 年，來自 Google 的機器學習工程師團隊提出以下結論：機器學習專案經常失敗的原因在於，大多數專案都帶有自訂程式碼，用以彌補機器學習管道步驟之間的差異[2]。然而，這種自訂程式碼並不容易從一個專案移轉至另一個專案。研究人員在論文「機器學習系統中隱藏的技術債務」中總結了他們的發現[3]。作者在這篇論文中認為：管道步驟之間的**膠水程式碼**（*glue code*）相當脆弱，且自訂的腳本無法擴展到某一特定專案之外。隨著時間的推移，類似像 Apache Beam、Apache Airflow 或 Kubeflow 管道等這樣的工具已經被開發出來，而這些工具可用來管理機器學習管道任務。上述工具可對標準化編排和對任務之間的膠水程式碼進行抽象化。

雖然學習新工具（如 Beam 或 Airflow）或新框架（如 Kubeflow），並設置額外的機器學習基礎架構（如 Kubernetes）似乎很麻煩，但投入的時間很快就會得到回報。如不採標準化的機器學習管道，資料科學團隊將面臨個別的專案設定、任意的日誌檔案位置、特定的除錯步驟等。繁瑣的情況將會是無止盡的。

有向無環圖（Directed Acyclic Graphs）

管道工具如 Apache Beam、Apache Airflow 和 Kubeflow 管道，透過任務相依關係的圖形表達方式來管理任務的流動。

2 Google 於 2007 年啟動了一個名為 Sibyl 的內部專案，用以管理內部機器學習生產管道。然而，於 2015 年 D. Sculley et al. 提出時，該主題得到更廣泛的關注。並發表他們對機器學習管道的瞭解：「Hidden Technical Debt in Machine Learning Systems」（*https://oreil.ly/qVlYb*）。

3 D. Sculley et al., “Hidden Technical Debt in Machine Learning Systems,” Google, Inc. (2015).

如圖 1-2 中的範例圖所示，管道的步驟是有方向的。這意味著管道以任務 A 為起點，以任務 E 為終點，並保證執行路徑是由任務的相依關係明確定義。有向圖（directed graph）避免任務在沒有完全計算出所有相依關係的情況下就開始執行的情況。由於在訓練模型之前，必須對訓練數據進行預處理，故將其作為有向圖進行預處理，將避免訓練任務在預處理步驟完成之前被執行。

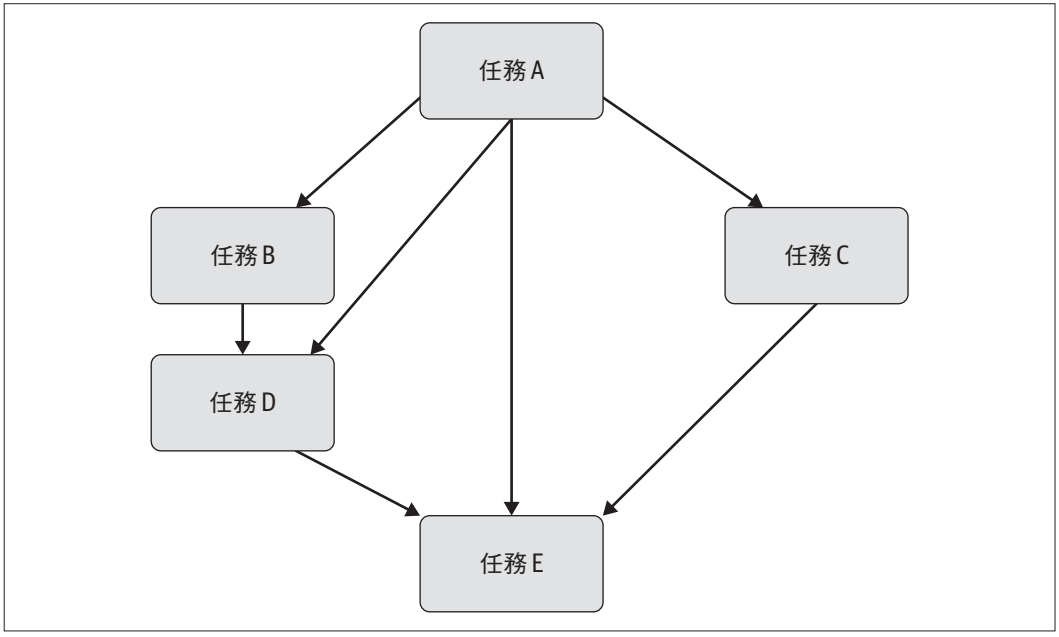

圖 1-2　有向無環圖範例

管道圖也必須是無環（acyclic）的，這意味著圖形並不會連接到先前已完成的任務。其意指管道可以無窮盡地進行，故不會結束工作流程。

依據上述兩個條件（有向（*directed*）和無環（*acyclic*）），管道圖被稱為有向無環圖（*directed acyclic graphs, DAG*）。您會發現 DAG 是大多數工作流程工具背後的核心概念。我們將在第 11 章和第 12 章中更詳細地討論這些圖形將如何被執行。

本書範例專案

為了配合本書的學習，我們採用開源資料庫並建立一個範例專案。該數據集為美國消費者對金融商品投訴的資料。它包含了結構化數據（分類 / 數字數據）和非結構化數據（文字）的混合。數據來自消費者金融保護局（Consumer Finance Protection Bureau）（*https://oreil.ly/0RVBG*）。

圖 1-3 展示來自此數據集的樣本。

	product	issue	consumer_complaint_narrative	company	state	company_response	timely_response	consumer_disputed
0	Mortgage	Loan servicing, payments, escrow account	My mortgage servicing provider (XXXX) transf...	SunTrust Banks, Inc.	TX	Closed with non-monetary relief	Yes	No
1	Debt collection	Cont'd attempts collect debt not owed	I HAVE NEVER RECEIVED ANY FORM OF NOTIFICATION...	ERC	CA	Closed with non-monetary relief	Yes	No
2	Debt collection	Disclosure verification of debt	i contacted walmart and the manager there said...	Synchrony Financial	MA	Closed with non-monetary relief	Yes	No
3	Credit reporting	Credit reporting company's investigation	I have filed multiple complaints XXXX on this ...	TransUnion Intermediate Holdings, Inc.	NY	Closed with explanation	Yes	Yes
4	Bank account or service	Account opening, closing, or management	Sofi has ignored my request to stop sending me...	Social Finance, Inc.	TX	Closed with explanation	Yes	No

圖 1-3　資料樣本

此機器學習問題為在給定投訴的數據下，預測該投訴是否會被消費者提出異議。因該數據集只有 30%的投訴有爭議，故此數據集為非平衡資料。

專案內容架構

我們將範例專案以 GitHub 儲存庫（*https://oreil.ly/bmlp-git*）方式提供，您可使用以下程式指令進行下載：

```
$ git clone https://github.com/Building-ML-Pipelines/\
            building-machine-learning-pipelines.git
```

Python 套件版本

為建立本書的範例專案，我們採用 Python 3.7-3.8 版本、TensorFlow 2.2.0 與 TFX 0.22.0。我們會盡可能地更新最新版至 GitHub 儲存庫，但無法保證此專案在其他套件版本下執行無誤。

此範例專案包含以下內容：

- 章節（*chapters*）資料夾，包含第 3、4、7 和 14 章單機範例使用的 notebook。
- 包含常見元件程式碼的元件（*components*）資料夾，如模型定義。
- 完整的互動管道（interactive pipeline）。
- 機器學習實驗範例，此為管道的起點。
- 採用 Apache Beam、Apache Airflow 和 Kubeflow 進行管道編排的完整管道範例。
- 附下載數據腳本程式的工具（*utility*）資料夾。

接下來的章節將引導您完成必要的步驟，並將機器學習實驗（在範例中是一個帶有 Keras 模型架構的 Jupyter Notebook）轉變為完整端對端的機器學習管道。

機器學習模型

深度學習專案的核心是由範例專案之 `components/module.py` 腳本中的函數 `get_model` 所產生的模型。該模型利用以下特徵預測消費者是否對投訴有爭議。

- 金融產品
- 子產品
- 公司對投訴的答覆
- 消費者投訴的問題
- 美國州名
- 郵政編碼（zip code）
- 申訴的內容（陳述文）

為建構機器學習管道，假設模型架構設計已完成，且不會修改模型。我們將在第 6 章詳細討論模型架構。但對於本書來說，模型架構是一個非常次要的問題。本書的重點為當模型確定之後，您可以利用它做哪些事。

範例專案的目標

本書將示範必要的框架、元件和基礎架構要素，以持續訓練機器學習模型範例。我們將在圖 1-4 所示的架構圖中使用堆棧：

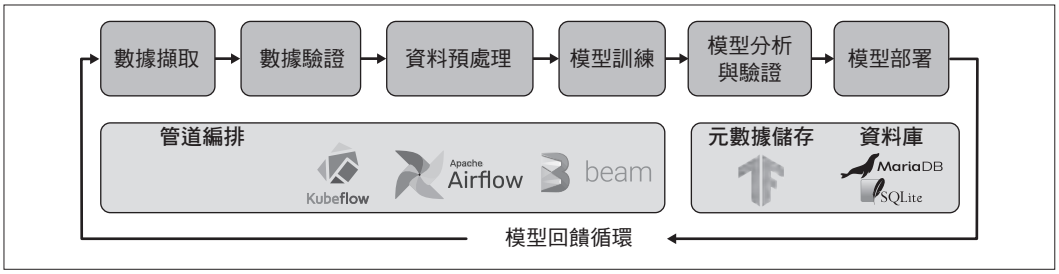

圖 1-4　範例專案的機器學習管道架構

本書試圖實現一個通用的機器學習問題，其可輕易使用特定問題做為代替。機器學習管道的結構和基本設置保持不變，但可輕鬆移轉至實際案例。每個元件都需要一些的客製化（例如，從哪裡擷取數據）。但正如之前的討論，客製化的需求將受到限制。

總結

本章介紹機器學習管道的概念、解釋各個步驟內容，並展示將這個過程自動化的好處。此外，概述每一章的大綱與範例專案，並為後續的章節建立基礎。下一章我們將開始建構管道！