人工智慧
Artificial Intelligence

Dr. Steve Lai
Mathison Intelligence

什麼是人工智慧？

"I propose to consider the question 'can machines think'?"

Alan Turing, 1950

"Artificial intelligence is not, by definition, simulation of human intelligence"

John McCarthy, 1956

人工智慧的歷史

1950年 - 圖靈測試（Turing Test）

- 由英國數學家、邏輯學家艾倫·圖靈在1950年提出，在 "Computing Machinery and Intelligence"，在這篇論文中，提出了"機器能否思考"的問題，並設計了一種名為"模仿遊戲"（The Imitation Game）的遊戲，後來這個遊戲被稱為"圖靈測試"。
- 由人和機器進行對話，當無法由對話中分辨誰是機器，那麼該機器即被認為有人工智能。

人工智慧的歷史

1956年 - AI 的誕生

- 1956年，約翰·麥卡錫（John McCarthy）與其他幾位 AI 先驅者在達特茅斯學院舉辦了一個歷時兩個月的研討會。
- 約翰·麥卡錫首次提出了"人工智能"（Artificial Intelligence）這個術語，並定義了其研究目標：創造出可以執行人類智能任務的機器。
- 達特茅斯會議對人工智能的發展產生了深遠影響，開始了AI的第一個黃金時代，並促使了許多重要的AI研究領域的開創，包括機器學習、自然語言處理和電腦視覺等。

人工智慧的歷史

1956年 - 2011年

- **黃金年代：1956 - 1974**。計算機可以解決代數應用題，證明幾何定理，學習和使用英語。當時大多數人幾乎無法相信機器能夠如此「智能」。其中 1957年，Perceptron提出。
- **第一次AI低谷：1974 - 1980**。AI研究者們對其課題的難度未能作出正確判斷：此前的過於樂觀使人們期望過高，當承諾無法兌現時，對AI的資助就縮減或取消了。
- **繁榮：1980 - 1987**。一類名為"專家系統"的AI程序開始為全世界的公司所採納，而「知識處理」成為了主流AI研究的焦點。其中包含了**聯結主義**（神經網路）的重生。**反向傳播算法**，一種神經網絡訓練方法被推廣
- **AI：1993 - 2011**，AI拆分為各自為戰的幾個子領域。

人工智慧的歷史

深度學習，大數據和人工智慧：2011至今

- 進入21世紀，得益於大數據和計算機技術的快速發展，許多先進的機器學習技術成功應用於經濟社會中的許多問題。
- 2006, Geoffrey Hinton 發表了一篇展示如何訓練深度神經網路的論文，用以辨別手寫數字（MNIST），準確為當時最好的 98%以上。
  - 深度學習以神經網路為基礎，引領了人工智慧的新浪潮。

人工智慧的定義

By Google Cloud

- 人工智慧（AI）是打造電腦與機器的科學領域，這些電腦和機器可以進行**"推論、學習以及採取行動"**，而這類行動原本需要人類智慧判斷或涉及超出人為分析能力上限的資料規模。
- AI 是一個廣泛版圖，包含**許多不同的專業領域**，包括電腦科學、資料分析與統計資料、硬體與軟體工程、語言學、神經科學，甚至是哲學和心理學。
- 在業務營運方面，AI 是一組採用**機器學習技術和深度學習的技術**，可用於資料分析、預測和預報、物件分類、自然語言處理、推薦、智慧資料擷取等。

人工智慧的定義

By Amazon AWS

- 人工智慧 (AI) 是電腦科學的一個領域，致力於解決與人類智慧相關的常見認知問題，例如學習、解決問題和模式辨識。
- 人工智慧呈現出機器人或未來世界的景像，也就是說，AI 不再是科幻小說中虛構的機器人，而真正成為現代高階電腦科學中的現實。
- Pedro Domingos 教授是這個領域聲譽卓著的研究人員，提出機器學習的「五大流派」，包括：符號主義 (Symbolists)，源自於邏輯和哲學；聯結主義 (Connectionists)，神經科學的分支；進化主義 (Evolutionaries)，與進化生物學相關；貝葉斯派 (Bayesians)，採用統計學和概率方法；以及類比主義 (Analogizers)，源自於心理學。
- 最近由於**統計運算**效率的提高，所以貝葉斯派的學者成功地讓「機器學習」在很多領域更向前邁進。同樣地，網路運算方面的進展也讓聯結主義流派的學者進一步發展「深度學習」這個分支。**機器學習 (ML) 和深度學習 (DL)** 這兩個電腦科學領域都是從人工智慧這個學科衍生出來的。

人工智慧的定義

By Microsoft Azure

- 這是電腦系統用來模擬類似人類認知功能 (例如學習和解決問題) 的能力。
- 電腦系統使用數學與邏輯，來模擬人類用於從新資訊學習並做出決策的推理。
- 人工智慧電腦系統會根據現有資料中的模式進行預測或採取動作，然後可從其中的錯誤學習，來提高精確度。成熟的 AI 處理新資訊的速度非常快又精確，因此對於複雜的案例來說相當實用，例如自動駕駛汽車、影像辨識程式和虛擬助理。

人工智慧的定義

By ChatGPT

- AI 是 "Artificial Intelligence" 的縮寫，即 "人工智慧"。人工智慧是一個跨學科領域，主要目標是創建和應用智能機器或智能軟體模擬、增強或超越人類的智慧，以進行思考、解決問題、作出決策或完成一定工作之人工自動化系統
    - 機器學習：通過數據的輸入，機器可以自我學習和改進。這是最普遍的AI形式，並且被用於各種應用，例如推薦系統和自動駕駛汽車。
    - 自然語言處理：這使機器能夠理解和生成人類語言，被用於翻譯應用和語音助手（如 Siri 和 Alexa）。
    - 電腦視覺：AI 可以用來識別和處理圖像，被廣泛用於臉部識別和醫學影像分析。
    - 智能機器人：這些機器人可以學習如何在物理世界中操作，並能夠處理未知或未經過程式化的任務。
    - 專家系統：這些系統模擬人類專家的決策過程，並在特定領域（如醫學或法律）提供專業建議。

AI value creation by 2030

$13 trillion

AI 浪潮
現在進行式

- 第一波人工智慧浪潮（1950到1960年代）：符號邏輯，告訴電腦「人的思考邏輯」；人類無法釐清自己的思考過程，最終失敗。
- 第二波人工智慧浪潮（1980到1990年代）：專家系統，告訴電腦「人的所有知識」；人類無法解答所有問題、寫成規則，最終失敗。
- 第三波人工智慧浪潮（2010年至今）：機器學習，告訴電腦「人看見的事物」；正在發展中。
- 歷經兩次挫敗，在第三波人工智慧浪潮中，科學家發展出「機器學習」方法，終於讓AI科技有了突破。後來，人類又從機器學習的經驗中，找出「深度學習」技術，讓第三波AI浪潮開始大進展。
- 在第三波AI浪潮中，專家發現，圖形處理器（Graphics Processing Unit, GPU）比過去的中央處理器（Central Processing Unit, CPU）更適合執行深度學習，因此紛紛轉移目光。同時，GPU大廠Nvidia更成為深度學習的強力助攻，藉著提升硬體設備、發揮強大的運算能力，幫助AI深度學習能量大爆發。

人工智慧的應用

應用場景四大面向

- 2017年，管理顧問公司麥肯錫（McKinsey & Company）將人工智慧的應用場景分成四大面向：
  - Project（計畫）：準確地預測與規劃，完成最佳生產計畫。
  - Produce（生產）：維持高品質、高效率的生產流程。
  - Promote（行銷）：精準目標銷售與市場分析。
  - Provide（供給）：提高客戶滿意度，帶動永續經營。

人工智慧的應用

From Data to AI

收集資料 Data Collection → 準備資料 Data Preparation → 選擇模型 Model Choosing → 模型訓練 Model Training → 模型評估 Model Evaluation → 參數優化 Parameter Tuning → 進行預測 Prediction

人工智慧的應用

台灣智慧醫療應用實例

- AI科技在醫療健康產業中，已開始協助臨床決策、疾病判斷，進一步跨入預防醫學、精準醫療等領域；除了減少醫護工作負擔、降低出錯率，也克服人類無法解決的醫療挑戰。
  - 精準醫學｜學者獲「AI界諾貝爾獎」：若有AI診斷，我的乳癌就能早兩年發現
  - 智慧醫院｜38年公家醫院，如何變成人人懂AI的智慧醫院？中榮讓員工做這件事
  - 心理健康｜疫情期間，AI如何撐起醫護心理健康，預防憂鬱自殺？
  - 科技防疫｜國旗登上WHO黑客松！AI秒讀新冠肺炎X光　成大擊敗全球奪勝

人工智慧的應用

台灣智慧交通應用實例

- 人工智慧能透過影像辨識技術，增強車輛辨識、號誌管理、交通安全管理等資訊整合。目前，台灣交通的人工智慧應用，已發展到自駕車、車流計算、路況安全預警、路網優化等領域。
  - 大眾運輸安全｜欠缺人力巡檢軌道？AI能幫忙
  - 自駕車｜破解台北都市傳說！深夜一輛公車開進信義區，卻沒人握著方向盤？
  - 智慧交通｜連假人擠人？引進這種交通AI就不必擔心塞車、買不到票

人工智慧的應用

其他智慧產業應用實例

- 生活中，常見智慧音箱及手機AI助理運用的語音辨識功能；Netflix、YouTube為你推薦的影音演算法；AI客服辨識客戶想法，提出個人化回覆……AI人工智慧應用早已無所不在，持續為你改善生活品質。
  - 能源科技｜在家工作讓用電量創新高！疫情、缺水又限電，台灣電力有解方？
  - 智慧金融｜股市菜雞救星！免費AI工具月更投資組合，客戶資產變13倍
  - 精準行銷｜薄酒萊如何新鮮準時送到你手中？酒莊用AI預測供需　退貨減80%
  - 媒體經營｜媒體業如何用AI人工智慧，實現流量變現？天下雜誌群的經驗談
  - 生活娛樂｜誰買社群假帳號衝人氣？AI揪灌水網紅　準確率破九成
  - 生活娛樂｜不只有Tinder！交友軟體用「AI戀愛教練」助70%男女脫單

台灣的人工智慧

台灣如何發展人工智慧？已有哪些發展？

- 台灣人工智慧年會 <2014>
- 台灣人工智慧學校 <2017>
- 人工智慧實驗室 <2017>
- 大學大量設立人工智慧學系 <2021>

Resume

- Steve Lai (賴昭榮)
- 學經歷：
  - Founder and CEO at **Mathison Intelligence**（麥錫森智能）
    - ✦ plusForm — 線上課程品牌
  - 台灣大學資訊工程博士
  - 中央大學資訊工程碩士
  - 加入 plusForm 討論群

Discord 語音討論群

Line openChat 討論群

AI 與機器學習

"Machine learning is the field of study that gives computers the ability to learn without being explicitly programmed."

Arthur Samuel, 1959

機器學習的分類

source: https://databasecamp.de/en/ml/reinforcement-learnings

監督式學習（Supervised Learning）

- 模型從標記過的訓練數據（即每筆數據都有明確的答案或標籤）中學習。透過這種方式，模型進行訓練，以便能夠基於輸入特徵預測出對應的輸出。
  - 標籤數據：在監督式學習中，我們需要標籤或標記的數據作為訓練集。例如，如果我們正在建立一個垃圾郵件檢測器，那麼我們將需要一組郵件，其中每封郵件都被標記為"垃圾郵件"或"非垃圾郵件"。
  - 特徵：特徵是用於進行預測的輸入變量。在上述垃圾郵件檢測器的例子中，特徵可能包括郵件的文本、發送者的地址、郵件的長度等。
  - 模型訓練：在訓練過程中，機器學習算法會嘗試找出輸入特徵和輸出標籤之間的關係或模式。這個過程通常涉及到損失函數（一種衡量模型預測結果和實際值差異的方式）和優化器（一種用於調整模型參數以最小化損失函數的方法）。
  - 預測：一旦模型訓練完成，它就可以接收新的、未標記的輸入並預測出對應的輸出。例如，我們的垃圾郵件檢測器可以接收一封新郵件並預測其是否為垃圾郵件。
- 監督式學習的兩個主要類型是分類（預測的輸出是離散的，如「垃圾郵件」或「非垃圾郵件」）和回歸（預測的輸出是連續的，如一個人的收入或房價）。

Input (X)	Output (Y)	Application
email	spam? (0/1)	spam filtering
audio	text transcripts	speech recognition
English	Spanish	machine translation
ad, user info	click? (0/1)	online advertising
image, radar info	position of other cars	self-driving car
image of phone	defect? (0/1)	visual inspection

Classification

Predicting Class Labels

- Binary classification

Two or more inputs

Regression

Predicting Continuous Outcomes

- Linear regression

Regression: Housing price prediction

非監督式學習（Unsupervised Learning）

- 使用未標記的數據來訓練模型。這種學習方式的目標是讓模型從輸入數據中學習到有用的結構或模式，而不是透過對應的輸出標籤來學習。
  - 數據聚類（Cluster）：將數據根據某種相似度度量分組的過程。舉例來說，商業中的市場細分就可以透過將顧客根據他們的消費習慣分組來進行。
  - 降維：目的是減少數據的維度（即特徵的數量），同時盡量保留數據的重要信息。降維常用於數據的視覺化或者是在進行監督式學習之前的數據預處理。
  - 異常檢測：在異常檢測中，模型被訓練來識別出那些與數據集中的其他樣本顯著不同的樣本。例如，信用卡公司可能使用異常檢測來識別可能的詐騙交易。
- 由於非監督式學習不依賴於標記數據，所以它在處理大量無標籤數據時，尤其有用。

Supervised learning
Learn from data labeled
with the "right answers"

Unsupervised learning

Clustering: Grouping customers

Clustering: Grouping customers

Anomaly Detection

- Detecting unusual credit card transactions to prevent fraud,
- Catching manufacturing defects
- Automatically removing outliers from a dataset

Classical Supervised & Unsupervised Learning

Supervised learning
Input data: x
Model
Training dataset
Model update
Predict output (ŷ)
Compare
Expected output (y)
Error

Unsupervised learning
Input data: x
Model
Generated example

半監督學習（Semi-supervised Learning）

- 結合了監督式學習和非監督式學習的特點。在半監督學習中，模型被訓練在少部分被標籤的數據用以建構基礎模型，和大量的未被標籤的數據上用以產生新的數據。
  - 利用未標籤的數據：雖然未標籤的數據不能直接用於預測目標變量，但它可以提供有關數據分布或結構的有用信息，這可能有助於提高模型的預測性能。
  - 自我訓練：這是一種常見的半監督學習技術，其中模型首先在標籤數據上進行訓練，然後使用該模型來產生未標籤數據的預測，這些預測接著被作為新的標籤再次用於訓練。
  - 標籤傳播：在標籤傳播中，模型使用圖形表示來模擬數據間的關聯，並利用這種關聯性將標籤從已知的項目傳播到未標籤的項目。
- 半監督學習尤其適合在有大量未標籤數據和少量標籤數據的情況下使用。
- 由於標籤數據的收集過程通常既費時又昂貴，所以在許多實際情況中，半監督學習可以提供一種有效的解決方案。

Semi-Supervised Learning

- Two classes (triangles and squares)
- The unlabeled examples (circles) help classify a new instance (the cross)

強化式學習（Reinforcement Learning）

- 主要涉及的是一個智能體（agent）在與環境（environment）的互動中學習如何行動（action），以便在長期中最大化某種形式的累計獎勵（reward）。
  - 智能體（Agent）：這是在環境中執行操作的實體或程序。
  - 環境（Environment）：智能體與其互動的世界或上下文。
  - 行動（Actions）：智能體可以在給定的狀態下採取的可能操作。
  - 狀態（States）：這是智能體可以處於的環境條件或位置。
  - 獎勵（Reward）：這是智能體在採取行動後從環境中接收的反饋。
  - 政策（Policy）：這是智能體用來從當前狀態選擇行動的策略。

source: https://www.synopsys.com/ai/what-is-reinforcement-learning.html

- 在強化學習中，智能體的目標是學習一個最佳政策，該政策將在給定狀態下選擇能最大化長期獎勵的行動。
- 這個學習過程通常涉及到探索（嘗試新的行動）和利用（採取已知最佳行動）之間的平衡，以及處理即時獎勵與長期獎勵之間的權衡。
- 強化學習的範例包括棋盤遊戲如圍棋、遊戲、機器人導航，以及在不確定環境中進行資源管理等許多其他類型的問題。

Reinforcement Learning

- The learning system, called an agent.
- Observe the environment, select and perform actions, and get rewards in return.
- Learn by itself what is the best strategy, called a policy, to get the most reward over time.
- A policy defines what action the agent should choose when it is in a given situation.

深度學習（Deep Learning）

- 機器學習的一種子領域，特別專注於利用稱為類神經網路（Neural Network, NN）的模型方法來模擬人腦神經元的工作方式，從而在數據中學習並找到模式。
- "深度"一詞，來自於這種神經網路可以由多個"層"組成，每一層都對應於更高層次的特徵或抽象概念。
- 深度學習模型通過這樣的多層神經網絡來進行學習和預測。
  - 自動特徵學習：深度學習模型可以自動從原始數據中學習和識別有用的特徵或模式，無需人工預定義。
  - 多層次抽象：深度學習模型可以從數據中學習出多層次的抽象，這些抽象往往更好地捕捉了數據的結構和規律。
  - 大數據與計算能力：深度學習模型通常需要大量的數據和計算能力來進行訓練，但在現代計算硬件和數據收集能力的發展下，這已經不再是問題。
  - 寬泛應用：深度學習模型已經被成功應用在很多領域，包括影像辨識、語音識別、自然語言處理、遊戲等。

where $z = \boldsymbol{w}^T\boldsymbol{x} + b$

X1₁
X1₂
X1₃
X1[...]
X1[...]
X1[...]

生成式 AI

- 生成 AI 是一種可以產生各種類型內容的 AI，包括文本、圖像、音頻和合成數據。可以根據從現有內容中學習到的內容創建新的內容。
- 深度學習模型可以分為兩種類型：生成和判別。
  - 判別模型對數據點進行分類或預測標籤，通常在標記數據點的數據集上進行訓練。判別模型在不同類型的數據實例之間進行區分。
  - 生成模型根據現有數據的學習概率分佈生成新的數據實例。
  - 生成 AI 模型使用統計模型來預測，並根據他們訓練數據的基礎結構生成新的內容。他們可以根據接收到的輸入生成文本、圖像、音頻和決策的新組合。生成語言模型通過訓練數據學習模式和語言，可以預測一個句子的下一部分。例如 GPT 和 Bard。

Gen AI Supervised, Semi-Supervised & Unsupervised Learning

Not GenAI when y is a:

- Number
- Discrete
- Class
- Probability

Is GenAI when y is:

- Natural language
- Image
- Audio

Deep Learning Model Types

Discriminative

- Used to classify or predict
- Typically trained on a dataset of labeled data
- Learns the relationship between the features of the data points and the labels

Generative

- Generates new data that is similar to data it was trained on
- Understands distribution of data and how likely a given example is
- Predict next word in a sequence

Data
Labels
Predictive ML Model
Learns relationship between data and label
Output
Label
Input
Unstructured Content
GenAI Model
Learns patterns in unstructured content
Output
New content

Not GenAI when y is a:

- Number
- Discrete
- Class
- Probability

Is GenAI when y is:

- Natural language
- Image
- Audio

機器學習系統建立流程圖

人工智慧的演算法

監督式學習

- 分類
  - 感知器（perceptron）
  - Logistic regression
  - 支援向量機（SVM）
  - 決策樹（Decision Tree）
  - 隨機森林（Random Forest）
  - K鄰近演算法（KNN）

分類演算法

感知器(perceptron)

Perceptron

- 感知器學習規則是一種用於訓練感知器的監督學習方法，其核心思想是不斷地根據預測錯誤來更新模型的權重。
- 以下是具體的步驟：
    1. 初始化權重：我們首先需要初始化感知器的權重，這可以是隨機數字或者為零。
    2. 對於每個訓練樣本，進行以下操作：
        - 計算輸出：計算感知器對訓練樣本的預測輸出。感知器使用一個步驟函數（unit step function）作為其激活函數（activation function），對於二分類問題，這個步驟函數可以定義為：當加權輸入（net input）大於或等於某個閾值時，輸出 1，否則輸出 0。
        - 更新權重：根據感知器的預測結果與實際目標值的差異來更新權重。更新規則可以定義為：新權重 = 舊權重 + 學習率 * (目標值 - 預測值) * 特徵值。其中，學習率是一個超參數（H=hyperparameter），用於控制模型學習的速度。
    3. 重複第二步，直到模型的預測錯誤達到一個滿意的程度，或者達到預設的迭代次數。
- 這種學習規則的一個重要特性是，如果訓練數據是線性可分的（也就是說，存在一條直線或一個超平面可以完美地將兩個類別分開），那麼感知器學習規則保證能夠找到一組解決方案。然而，對於不可分的數據，感知器學習規則可能會陷入無窮的權重更新循環中。

Perceptron

- $z = w_1 x_1 + w_2 x_2 + \ldots + w_m x_m$
- $w = \begin{bmatrix} w_1 \\ \vdots \\ w_m \end{bmatrix}, x = \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix}$
- Decision function $\sigma(z)$
- $\sigma(z) = \begin{cases} 1 & if\ z \geq \theta \\ 0 & otherwise \end{cases}$

Perceptron

- $w_j := w_j + \Delta w_j,\ b_j := b_j + \Delta b_j$

- $\Delta w_j = \eta(y^{(i)} - \hat{y}^{(i)})x_j^{(i)}$
- $\Delta b_j = \eta(y^{(i)} - \hat{y}^{(i)})$

$$y^{(i)} = 0,\quad \hat{y}^{(i)} = 0,\quad \Delta w_j = \eta(0-0)x_j^{(i)} = 0,\quad \Delta b = \eta(0-0) = 0$$
$$y^{(i)} = 1,\quad \hat{y}^{(i)} = 1,\quad \Delta w_j = \eta(1-1)x_i^{(i)} = 0,\quad \Delta b = \eta(1-1) = 0$$
$$y^{(i)} = 1,\quad \hat{y}^{(i)} = 0,\quad \Delta w_i = \eta(1-0)x_i^{(i)} = \eta x_i^{(i)},\quad \Delta b = \eta(1-0) = \eta$$
$$y^{(i)} = 0,\quad \hat{y}^{(i)} = 1,\quad \Delta w_j = \eta(0-1)x_j^{(i)} = -\eta x_j^{(i)},\quad \Delta b = \eta(0-1) = -\eta$$

Adaptive Linear Neurons

Adaline VS Perceptron

Adaptive Linear Neurons

Adaline VS Perceptron

- 權重更新的方式的不同：
  - Perceptron: 權重的更新是基於實際輸出和預期輸出的差異。如果預測是正確的，則權重保持不變。如果預測是錯誤的，權重就會更新。
  - Adaline 模型中，權重的更新是基於損失函數（loss function，或稱成本函數 cost function）的梯度下降。這種方法不僅考慮是否錯誤，而且還考慮錯誤的程度。Adaline 使用的是梯度下降（gradient descent）來最小化連續的代價函數。
- 在 Adaline 中，目標最小化錯誤的損失函數 L，可將其定義為 mean squared error(MSE)這個函數的值越小，代表模型的預測結果與實際結果越接近，模型的表現越好。在訓練過程中，我們的目標是優化（通常是最小化）成本函數。
- $$L(\boldsymbol{w}, b) = \frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - \sigma\left(z^{(i)}\right)\right)^2$$

Minimizing Loss Functions with Gradient Descent

Minimizing Loss Functions with Batch Gradient Descent

$$\nabla L(\boldsymbol{w}, b) \begin{cases} \boldsymbol{w} := \boldsymbol{w} + \Delta \boldsymbol{w} \\ b := b + \Delta b \end{cases}$$

$$\Delta \boldsymbol{w} = -\eta \nabla_w L(\boldsymbol{w}, b), \quad \Delta b = -\eta \nabla_b L(\boldsymbol{w}, b)$$

$$\Delta w_j = -\eta \frac{\partial L}{\partial w_j} \text{ and } \Delta b = -\eta \frac{\partial L}{\partial b}$$

$$\frac{\partial L}{\partial w_j} = -\frac{2}{n} \sum_i \left(y^{(i)} - \sigma\left(z^{(i)}\right) \right) x_j^{(i)}$$

$$\frac{\partial L}{\partial b} = -\frac{2}{n} \sum_i \left(y^{(i)} - \sigma\left(z^{(i)}\right) \right)$$

when learning rate is too large

if you want to know the derivation of MSE

$$\begin{aligned}
\frac{\partial L}{\partial w_j} &= \frac{\partial}{\partial w_j} \frac{1}{n} \sum_i \left(y^{(i)} - \sigma\left(z^{(i)}\right) \right)^2 = \frac{1}{n} \frac{\partial}{\partial w_j} \sum_i \left(y^{(i)} - \sigma\left(z^{(i)}\right) \right)^2 \\
&= \frac{2}{n} \sum_i \left(y^{(i)} - \sigma\left(z^{(i)}\right) \right) \frac{\partial}{\partial w_j} \left(y^{(i)} - \sigma\left(z^{(i)}\right) \right) \\
&= \frac{2}{n} \sum_i \left(y^{(i)} - \sigma\left(z^{(i)}\right) \right) \frac{\partial}{\partial w_j} \left(y^{(i)} - \sum_j \left(w_j x_j^{(i)} + b \right) \right) \\
&= \frac{2}{n} \sum_i \left(y^{(i)} - \sigma\left(z^{(i)}\right) \right) \left(-x_j^{(i)} \right) = -\frac{2}{n} \sum_i \left(y^{(i)} - \sigma\left(z^{(i)}\right) \right) x_j^{(i)}
\end{aligned}$$

Improving Gradient Descent through Feature Scaling

Standardization

- 特徵縮放（Feature Scaling）是調整數據的尺度以確保所有特徵都在相同的範疇。這對於許多機器學習算法都是必要的，因為它們在計算過程中使用距離或相似度度量。如果特徵的尺度不一致，那麼某些特徵可能會比其他特徵對最終結果產生更大的影響。
- 標準化（Standardization）是一種特定的特徵縮放方法，將特徵調整為均值為0，標準差為1的標準常態分佈。這種方法不僅將特徵的尺度統一，還確保特徵遵循正態分佈，這對於許多機器學習算法，尤其是基於假設輸入數據為高斯分佈的算法（如線性回歸，邏輯回歸，和支持向量機等）非常有利。
- 對於梯度下降（Gradient Descent）來說，進行特徵縮放和標準化有以下幾個主要好處：
  - 更快的收斂：當我們使用梯度下降進行優化時，如果所有特徵都在相同的尺度範圍內，算法通常會更快地收斂到最小值。否則，如果特徵的尺度差異很大，梯度下降可能需要更多次的迭代才能找到最小值，因為它需要在一個特徵的範圍內進行大量的微小調整。
  - 避免數值穩定性問題：標準化可以防止數值穩定性問題，這些問題可能在特徵的尺度非常不一致時出現。當某些特徵的尺度相對於其他特徵非常大時，它可能會主導成本函數並導致算法無法正確地學習其他特徵。
  - 改善模型性能：當輸入特徵被標準化，模型的性能通常會得到改善，特別是對於那些對輸入特徵範圍敏感的模型，如支持向量機（SVM）或者 k-近鄰（k-NN）等。

如何選擇不同的演算法

- 選擇特徵並收集帶有標籤的訓練樣本
- 選擇性能評估指標
- 選擇學習算法並訓練模型
- 評估模型的性能
- 調整算法的設定並調整模型

評估模型

K-fold

- 評估機器學習模型的的一種經典且常用方法是留出法（holdout method）。
  - 使用留出法，我們將初始數據集分割成獨立的訓練和測試數據集，前者用於模型訓練，後者用於評估模型的性能。
- 調整和比較不同參數設定，以進一步提高在未見數據上做預測的性能，這個過程被稱為模型選擇（model selection）。
- 在模型選擇過程中反覆使用同一個測試數據集，該數據集將變成我們的訓練數據，因此模型將更有可能過度擬合。
- 一種更好的使用留出法進行模型選擇的方法是將數據分為三部分：訓練數據集（training set）、驗證數據集（validation set）和測試數據集（test set）。訓練數據集用於訓練不同的模型，並在驗證數據集上的性能用於模型選擇。
- 在訓練和模型選擇步驟中擁有模型未見過的測試數據集的優點是，我們可以獲得對其對新數據能力的較無偏見的評估。
- 留出法的一個缺點是，性能估計可能對我們如何將訓練數據集劃分為訓練和驗證子集非常敏感；該估計將會因數據的不同示例而變化。

評估模型

K-fold cross-validation

- 在 K-fold cross-validation中，我們將訓練數據集無放回地隨機分割成k fold。這裡，k – 1 fold，為 training fold，用於模型訓練，另外的 test fold，用於性能評估。這個過程重複 k 次，因此我們得到 k 個模型和評估。
- 然後，我們根據不同、獨立的 test fold 計算模型的平均性能，以獲得一個與留出法相比對訓練數據的子分區較不敏感的性能估計。通常，使用 K-fold cross-validation 進行模型調整，找尋較佳的 hyperparameter。
- 一旦我們找到了滿意的超參數值，我們就可以在完整的訓練數據集上重新訓練模型，並使用獨立的測試數據集獲得最終的性能估計。
    - 在 K-fold cross-validation 後將模型擬合到整個訓練數據集的原理是，首先，我們通常對一個單一的最終模型（而不是k個獨立模型）感興趣，其次，提供更多的訓練實例給學習算法通常會產生更準確和穩健的模型。
    - 由於 K-fold cross-validation 是一種無放回的重採樣技術，這種方法的優點是在每次迭代中，每個資料將被使用一次，且training fold 和 test fold 是不相交的。此外，所有的 test fold 也是不相交的。
- 如訓練集相對較小，增加 fold 可能是有益的。
    - 如果我們增加 k 的值，每次迭代將使用更多的訓練數據，這會導致對評估偏差減小。較大的 k 值也會增加交叉驗證算法的運行時間。
- 如果我們正在處理大型數據集，我們可以選擇較小的 k 值，例如k = 5，並且仍然能獲得對模型平均性能的準確估計，同時減少在不同 fold 上評估模型的計算成本。

Logistic Regression

Classification Algorithm

- 邏輯回歸（logistic regression）使用一個邏輯函數（通常為 Sigmoid 函數）將線性回歸的輸出轉換為一個概率值。這個概率值可以被用來預測類別。
- $\sigma(z) = \dfrac{1}{1+e^{-z}}, z = w^T x + b$
- $\hat{y} = \begin{cases} 1 & if\ \sigma(z) \geq 0.5 \\ 0 & otherwise \end{cases}$

Logistic Regression in Detail

Logistic Regression in Detail
Training

- loss function:

$$L(\boldsymbol{w}, b|\boldsymbol{x}) = \sum_i \frac{1}{2}\left(\sigma\left(z^{(i)}\right) - y^{(i)}\right)^2$$

- Log loss function

$$L(\boldsymbol{w}, b) = \frac{1}{n}\sum_{i=1}^{n}\left[-y^{(i)}\log\left(\sigma\left(z^{(i)}\right)\right) - \left(1 - y^{(i)}\right)\log\left(1 - \sigma\left(z^{(i)}\right)\right)\right]$$

Tackling Overfitting

Regularization

- 過度擬合（overfitting）是機器學習中常見的問題，當模型在訓練數據上表現良好，但在未見過的數據（測試數據）上泛化能力差時，我們就說模型過度擬合。過度擬合往往是由於模型有太多參數，導致模型過於複雜。
- 模型也可能出現欠擬合（underfitting）的問題，即模型不夠複雜，無法很好地捕捉訓練數據中的模式，因此在未見數據上的表現也不佳。
- 找到好的偏差-變異性權衡的一種方法是通過正規化（Regularization）來調整模型的複雜性。正規化是一種非常有用的方法，用於處理共線性（特徵之間的高度相關），濾除數據中的噪聲，並最終防止過度擬合。
- 正規化的概念是引入額外的資訊來懲罰極端的參數（權重）值。最常見的正規化形式是所謂的L2正規化（有時也稱為L2收縮或權重衰減）。在這裡，λ是所謂的正規化參數，它被加入到模型的損失函數中，以懲罰過大的權重。

$$L(\boldsymbol{w}, b) = \frac{1}{n}\sum_{i=1}^{n}\left[-y^{(i)}\log\left(\sigma\left(z^{(i)}\right)\right) - \left(1 - y^{(i)}\right)\log\left(1 - \sigma\left(z^{(i)}\right)\right)\right] + \frac{\lambda}{2n}\|\boldsymbol{w}\|^2$$

Support Vector Machines
支持向量機（SVM）

- 最大邊際（margin）分類器：SVM是一種最大邊際分類器。這意味著SVM旨在找到一個超平面，該超平面能最大化正確分類樣本的邊際，即最接近超平面的樣本點到超平面的距離最大。這個特性使得SVM有著良好的泛化能力，即在未知數據上的預測能力。
- 支持向量是訓練樣本中距離超平面最近的樣本點，這些點在定義決策邊界上起著關鍵作用。只有這些支持向量才會影響最終的決策邊界，其他的點（即使被移除）也不會影響決策邊界的位置。
- 核技巧（kernelized）：在某些情況下，數據不是線性可分的。為了解決這個問題，SVM使用了一種稱為"核技巧"的技術，透過將數據映射到一個更高維度的空間中使其變得可分。這樣做可以使SVM能夠處理非線性分類問題。
- 軟間隔和正則化：在現實世界中，數據往往是帶有噪聲的，可能存在一些難以正確分類的異常值。SVM引入了軟間隔的概念，允許一些點被錯誤分類。此外，通過正則化參數C，可以控制間隔的寬度和錯誤分類點的數量之間的權衡。

Decision Tree
決策樹

- 樹結構：決策樹模型的結構就像一棵反轉的樹，有根節點、內部節點和葉節點。每一個內部節點代表一個特徵或屬性，每一個分支代表一個決策規則，而每一個葉節點代表一個結果。
- 分支標準：決策樹的建立基於特徵的分裂。該分裂是通過某種度量（如信息增益（entropy）、增益比率（classification error）或吉尼不純度（gini impurity））來決定的，該度量可以衡量分裂後的數據純度。目的是在每次分裂後，我們希望類別之間的差異更加明顯。
- 可解釋性：決策樹的一大優點是它的可解釋性。由於每一個決策都對應於一個明確的規則，所以決策樹模型非常直觀，人們可以很容易地理解模型的決策過程。
- 剪枝：決策樹有可能過於複雜，導致對訓練數據過度擬合。為了避免這種情況，我們可以使用一種稱為"剪枝"（pruning）的技術，來減少決策樹的複雜性並增強其泛化能力。

Random Forests
隨機森林

- 隨機森林屬於整體學習（Ensemble Learning）的範疇，其基本思想是結合多個弱學習器（在此情況下是決策樹）的預測，以創建一個強大且穩定的模型。
- 隨機森林由多棵決策樹組成。每棵樹都是在數據的不同子集上訓練的。通常，這些子集是通過有放回的抽樣（bootstrap sampling）從訓練數據中選擇的。
- 特徵隨機性：在每個分裂點，隨機森林不是考慮所有特徵來做分裂，而是從所有特徵中隨機選擇一個子集，並基於這個子集找到最佳分裂。這增加了模型的多樣性，通常會得到更好的性能。
- 投票機制：對於分類問題，每棵決策樹做出自己的預測，然後隨機森林通過多數投票的方式做出最終預測。對於回歸問題，則是計算所有樹的預測的平均值。
- 過擬合的降低：由於隨機森林結合了多個決策樹的預測，它通常不會像單一決策樹那樣受到過擬合的影響，特別是當數據集很小或決策樹很深時。
- 特徵重要性：隨機森林可以用來估計特徵的重要性。這是通過觀察當特徵被用於樹的分裂時，對預測性能的影響來計算的。
- 可擴展性：隨機森林可以有效地處理大數據集和具有大量特徵的

K-Nearest Neighbors (KNN)

K-最近鄰演算法

- 基於距離的學習：KNN演算法的核心概念是以距離為基礎來進行學習。對於一個給定的新資料點，找出訓練集中與其最接近（即最相似）的「K」個樣本點。
- K的選擇：K是KNN方法中的一個超參數，代表從訓練集中選取的鄰近樣本點的數量。K的值必須在開始使用演算法之前設定，並會影響到最終的結果。選擇適合的K值通常需要透過交叉驗證來實現。
- 投票機制：在分類問題中，KNN通過投票機制來進行預測。即選取最接近的K個鄰居，然後看這些鄰居中哪一個類別的數量最多，該類別就被預測為新資料點的類別。
- 平均方法：在回歸問題中，KNN則是將最接近的K個鄰居的目標值進行平均，得到的平均值就是新資料點的預測值。
- 距離度量：KNN演算法中使用到的距離可以有多種度量方法，例如歐氏距離、曼哈頓距離等，選擇哪一種距離度量方式需要根據數據特性來決定。
  - Manhattan distance

$$d\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}\right) = \sqrt[p]{\sum_k \left|x_k^{(i)} - x_k^{(j)}\right|^p}$$

- 特性縮放：由於KNN演算法是基於距離來進行學習的，因此如果數據集中的特徵維度的尺度（scale）差異大，可能會影響到結果。在使用KNN前，通常需要先進行特性縮放（例如正規化或標準化）。
- 懶惰學習：KNN屬於懶惰學習演算法，也就是說它在訓練階段不進行明顯的學習。而是在需要進行預測時，才根據訓練數據來決定新數據點的類別。
- 計算密集型：由於KNN演算法需要計算新數據點與所有訓練數據點的距離，因此當訓練數據量非常大時，其預測階段的計算量會非常大。

Natural Language Processing，NLP
自然語言處理

- 自然語言處理（Natural Language Processing，NLP）是人工智慧和語言學領域的一個交叉科學，專門研究如何讓機器能夠理解、處理和產生人類語言。以下是主要的部分：
  - 語言模型（language modeling）：語言模型試圖對語言（字詞的序列）的概率分布進行建模。例如，給定前面的一些字詞，預測下一個字詞是什麼。這種模型是語音識別、機器翻譯、拼寫檢查等應用的基礎。
  - 詞性標註（part-of-speech, POS tagging）：詞性標註是給語句中的每個字詞分配語法角色（如名詞、動詞、形容詞等）的過程。這是許多NLP任務（包括句法解析、實體識別等）的關鍵步驟。
  - 句法解析（syntactic parsing）：句法解析是對句子結構的分析，透過此過程能夠找出句子中的主語、動詞、直接與間接受詞等等。
  - 命名實體識別（named entity recognition, NER）：命名實體識別是從文本中識別和分類具有特定意義的實體，如人名、地名、組織名、日期等。
  - 情感分析（sentiment analysis）：情感分析也稱為情緒分析，目的是從文本中識別和提取主觀信息，例如作者對某一主題的情緒、觀點或情感。
  - 自然語言生成（natural language generation）：自然語言生成是一種從數據中創建自然語言文本的方法。這可能涉及文本摘要、聊天機器人的回答生成、文章的自動生成等。
  - 機器翻譯（machine translation）：機器翻譯是使用自然語言處理技術將一種語言的文本自動翻譯成另一種語言。
- 自然語言處理的挑戰在於人類語言的複雜性和歧義性，包括語意、語境、文化和俚語等各種因素。現代的自然語言處理技術多依賴機器學習（特別是深度學習）來理解語言的這些細微差異。

Language Models

- 語言模型（Language Models）是用於預測語言序列概率的模型，通常用於應用如語音識別、機器翻譯等等。以下是一些主要的語言模型：
    1. N-gram Models: 這是最早期且最簡單的語言模型之一。N-gram模型將語句分割成N個單詞的序列（或稱為"gram"），並根據先前的N-1個單詞來預測下一個單詞。然而，這種模型有一個明顯的缺點，那就是它不能捕捉到長範圍的依賴關係。
    2. Hidden Markov Models (HMMs): 這種模型將語言視為一種馬可夫過程，其中每個單詞的產生都依賴於先前的狀態。HMMs在語音識別和詞性標註等任務上有廣泛的應用。
    3. Neural Network Language Models: 隨著深度學習的興起，神經網路語言模型已經變得非常流行。這些模型使用神經網路來捕捉語言中的複雜模式和長範圍的依賴關係。
        A. Recurrent Neural Networks (RNNs): RNNs具有處理時間序列數據的能力，使它們成為處理語言模型的理想選擇，因為語言可以被視為一種時間序列。
        B. Long Short-Term Memory (LSTM): LSTM是一種特殊的RNN，設計來解決RNN在學習長範圍依賴關係時的問題。
        C. Gated Recurrent Unit (GRU): GRU是另一種特殊的RNN，它與LSTM有相似的性能，但結構較簡單。
        D. Transformers: Transformer模型在處理長範圍依賴關係方面特別強大，目前已經成為語言模型的主流。這些模型（如GPT-3、BERT）使用了自注意力（self-attention）機制來獲取語句中的全局信息。

文字表達

- 在自然語言處理（NLP）中，我們經常需要將文字轉換成數字形式，這樣機器才能進行處理。以下是一些常用的文字編碼或表達方式：
  - One-hot Encoding: 在這種編碼方式中，每個單詞被表示為一個長度等於詞彙表大小的向量，向量中的一個位置為1（該位置與該單詞相對應），其餘位置都為0。雖然這種編碼方式很直觀，但當詞彙表非常大時，會產生非常稀疏的向量，而且無法捕捉單詞之間的相似性。
  - Bag of Words: 這是一種非常簡單的表示，通常用於文檔分類任務。詞袋模型忽略了文本中的語詞順序和語法，僅將每個文本表示為詞頻的集合。這種方法的缺點是它忽略了文本的結構。
  - TF-IDF (Term Frequency-Inverse Document Frequency)**: TF-IDF是一種統計方法，用於反映一個詞對文本集或一個語料庫中一份文件的重要性。TF-IDF的值會隨著詞語出現的頻率增加而增加，並隨著詞語在語料庫中出現的文件數量增加而減小。
  - Word Embeddings: 單詞嵌入將詞彙映射到密集的向量，這些向量可以捕捉單詞之間的語義相似性和關聯性。最常見的單詞嵌入方法是Word2Vec和GloVe。這些模型可以獲得單詞的分佈式表示，這些表示能捕捉到許多語義屬性，如單詞間的同義詞、反義詞等。
  - Contextual Word Embeddings: 在Word2Vec和GloVe等傳統單詞嵌入方法中，每個單詞只有一個嵌入，這意味著每個單詞的所有含義都被壓縮到一個單一的向量中。而上下文單詞嵌入（例如BERT和ELMo）解決了這個問題，它們根據單詞出現的上下文生成單詞嵌入，因此一個單詞可以有多個嵌入，每個嵌入對應一種不同的含義。

大型語言模型

Large Language Models，LLMs

- 大型語言模型是利用深度學習在自然語言處理（NLP）和自然語言生成（NLG）任務中的基礎模型。這些模型通過在大量的數據上進行預訓練，來學習語言的複雜性和聯繫。預訓練後，這些模型可以透過微調、上下文學習、少量示例學習等技術，來適應下游（特定）任務。
- 大型語言模型可以應用於許多不同的用途和行業，包括醫療保健、零售、技術等。其中包括的用途有：文本摘要、文本生成、情感分析、內容創建、聊天機器人、虛擬助理和對話AI、命名實體識別、語音識別和合成、圖像註釋、文本轉語音合成、拼寫更正、機器翻譯、推薦系統、詐騙檢測、代碼生成等。
- 大型語言模型也有一些挑戰和限制，包括可靠性和偏見、上下文窗口、系統成本和環境影響。例如，這些模型的能力僅限於其訓練數據中的文本，這意味著它們對世界的了解有限。訓練數據中可能包含錯誤的信息、種族、性別和性別偏見以及有毒語言。此外，每個大型語言模型只有一定數量的記憶體，因此它只能接受一定數量的代幣作為輸入。例如，2023/06之前，ChatGPT 3.5 的限制是4096 個 tokens（約3000個英文單詞），GPT-4 為 8192 tokens。
- 還有一些開源的大型語言模型，如BLOOM、NeMO LLM、XLM-RoBERTa、XLNet、Cohere、GLM-130B等，可以在本地或私有雲中部署，這有助於快速的業務採用和強大的網絡安全

BERT
Bidirectional Encoder Representations from Transformers

- BERT（Bidirectional Encoder Representations from Transformers）是 Google 於2018年提出的一種新型深度學習模型，專為自然語言處理（NLP）設計。BERT 模型的主要創新在於其雙向性，可以從兩個方向（即左右兩側）來理解文本語境。
- BERT的設計基於一種稱為 Transformer 的架構，該架構在處理序列數據時具有優越的性能，並且可以並行處理整個序列，這與傳統的循環神經網路（RNN）需要按照特定順序處理數據的方式不同。
- BERT的特點包括：
    1. 雙向上下文理解：傳統的語言模型通常只能從一個方向（例如從左到右或從右到左）來理解文本。然而，BERT模型可以從兩個方向來理解文本，從而得到更準確的語境理解。
    2. 大規模非監督式預訓練：BERT模型在大規模的文本數據集上進行非監督式預訓練，這包括兩種任務：被遮蔽的語言模型（Masked Language Model）和下一句預測（Next Sentence Prediction）。在預訓練階段，模型學習到豐富的語言表示，這些表示可以在各種下游NLP任務中進行遷移。
    3. 微調（fine-tuning）適用於多種NLP任務：一旦預訓練完成，BERT模型可以在特定的NLP任務上進行微調，例如文本分類、命名實體識別或問答系統等。在進行微調時，模型的所有參數都會被更新，但是通過預訓練獲得的語言表示為下游任務提供了一個強大的初始點，從而大大提高了下游任務的性能。
- BERT 在多種NLP任務中都取得了當時最好的效果，包括問答、文本理解和情感分析等，它的出現對自然語言處理領域產生了深遠的影響。

GPT
Generative Pre-trained Transformer

- GPT（Generative Pretrained Transformer）模型是OpenAI所提出的一種語言模型，其目的是生成與輸入文本語義相關且連貫的文本。GPT 的核心理念是：先在大量未標記的文本數據上進行預訓練，然後針對特定的下游任務進行微調。
- 以下為 GPT 模型的幾個關鍵特點：
  - 生成式模型：GPT 是一種生成式模型，這意味著它可以生成新的文本序列。例如，給定一段初始的文本，GPT 可以產生與該文本相關的新文本，且這段新文本在語義、語法和風格上都應該與初始文本保持一致。
  - 基於 Transformer 的架構：GPT 的核心網路結構基於 Transformer，該架構使用自注意力機制處理輸入的文本，能夠將每個單詞與文本中的所有其他單詞的關係考慮進來，對於捕獲文本的長範圍依賴性特別有效。
  - 遷移學習：GPT 模型首先在大量的未標記文本數據上進行預訓練，這階段的訓練目標是預測文本中的下一個單詞，然後在特定的下游任務（例如情感分析、文本分類或者命名實體識別等）上進行微調。

自然語言處理與金融應用

- 智能客服：NLP可以使聊天機器人和虛擬助手理解用戶的問題並提供相關的回答。實體識別可以幫助從用戶的問題中提取關鍵資訊，如產品名稱和時間等；情感分析可以理解用戶的情緒，進而提供更個性化的服務；自然語言生成則可以產生人類可以理解的回答。
- 反洗錢（AML）模型：金融機構可以利用NLP來分析客戶的交易記錄和通訊記錄，以檢測任何可疑的行為。例如，話語分析可以尋找暗示不誠實或欺詐行為的語言模式，主題模型則可以尋找在文本數據中出現的主題，這可能表示不正常的行為。這樣，金融機構可以更有效地檢測和防止洗錢活動。
- 防詐欺模型：NLP也可以用於防詐欺模型。例如，保險公司可能會使用NLP來分析索賠申請，以檢測可能的詐欺行為。這可能包括情感分析（詐欺者可能會使用不同的情感語言）、實體識別（找出在索賠申請中出現的特定模式，如同一個地址在多個索賠中出現）以及文本分類（基於先前已知欺詐案例，將新的索賠分類為正常或可疑）。這樣，公司可以更早地發現可能的詐欺行為。

中文斷詞與文字雲

- https://github.com/lzrong0203/youtube_data/blob/master/ptt_crawler.ipynb
- 2020 年 2 月，"PTT大家都在說什麼防疫話題？"

OpenAI API

- https://platform.openai.com/overview
  - 註冊登入
- https://platform.openai.com/account/api-keys
  - 建立 API Keys
- https://platform.openai.com/docs/api-reference
  - 參考文件
- https://platform.openai.com/playground
  - 練習場與內容

Models

- GPT-3 (text-XXXXX-003)
  - Ada是OpenAI提供的最小的語言模型，訓練數據集為40GB，擁有6億個參數。它主要用於文本分類、標記和生成等任務。由於Ada的規模較小，因此它的預測速度較快，但是它在執行更複雜的任務時可能會失去精度。
  - Babbage是OpenAI提供的第二個語言模型，訓練數據集為570GB，擁有1.2億個參數。與Ada相比，Babbage更適用於執行更複雜的自然語言處理任務，例如文本摘要、機器翻譯和問答系統等。由於它的規模較小，因此在執行大規模任務時可能會受到限制。
  - Curie是OpenAI提供的第三個語言模型，訓練數據集為800GB，擁有1.6億個參數。與Babbage相比，Curie更適用於執行更複雜的自然語言處理任務，例如生成自然語言文本、情感分析和文本摘要等。由於Curie的規模更大，因此它在執行大規模任務時通常比Babbage表現更好。
  - Davinci是OpenAI提供的最大和最先進的語言模型，訓練數據集為近1000GB，擁有1.75億個參數。它在自然語言處理任務中表現出色，例如生成自然語言文本、情感分析和問答系統等。由於它的規模較大，因此在執行大規模任務時它通常比其他模型表現更好，但也需要更長的時間來完成推理。

Requests (GPT-3)

```python
prompt = "哈囉你好，我是Steve"
response = openai.Completion.create(
  model="text-davinci-003",
  prompt=f"{prompt}",
  max_tokens=256,  # default: 16
)
print(response)
```


```
{
  "choices": [
    {
      "finish_reason": "stop",
      "index": 0,
      "logprobs": null,
      "text": " \n\n\u5f88\u9ad8\u8208\u898b\u5230\u4f60\uff01\u6709\u4ec0\u9ebc\u53ef\u4ee5\u5e6b\u52a9\u4f60\u7684
\u55ce\uff1f"
    }
  ],
  "created": 1685909689,
  "id": "cmpl-7NoPhkelklPRd1R6lzOldeyRFsw1p",
  "model": "text-davinci-003",
  "object": "text_completion",
  "usage": {
    "completion_tokens": 42,
    "prompt_tokens": 16,
    "total_tokens": 58
  }
}
```


```python
response["choices"][0]["text"]
```

' \n\n很高興見到你！有什麼可以幫助你的嗎？'

Requests (GPT-3.5)

```python
prompt = "哈囉你好，我是Steve"
messages = [{"role": "user", "content": prompt}]
response = openai.ChatCompletion.create(
            model="gpt-3.5-turbo",
            messages=messages,
            temperature=0
        )
print(response)
```


```
{
  "choices": [
    {
      "finish_reason": "stop",
      "index": 0,
      "message": {
        "content": "\u4f60\u597d\uff0cSteve\u3002\u6211\u662fAI\u52a9\u624b\uff0c\u6709\u4ec0\u9ebc\u6211\u53ef\u4ee5
\u5e6b\u52a9\u4f60\u7684\u55ce\uff1f",
        "role": "assistant"
      }
    }
  ],
  "created": 1685909900,
  "id": "chatcmpl-7NoT6CvuWgdL04wev6KRHcqUpvGJF",
  "model": "gpt-3.5-turbo-0301",
  "object": "chat.completion",
  "usage": {
    "completion_tokens": 30,
    "prompt_tokens": 17,
    "total_tokens": 47
  }
}
```


```python
response.choices[0].message["content"]
```

'你好，Steve。我是AI助手，有什麼我可以幫助你的嗎？'

Prompts Engineering

- 兩個prompt的原則
  - 寫出清楚明確的指示
  - 給模型思考的時間
- 寫出清楚明確的指示
  - 使用界定符號清楚指示輸入的不同部分
  - 要求結構性的輸出
    - JSON
  - 要求模型檢查條件是否滿足
  - 提供小提示
- 給模型思考的時間
  - 指定完成任務所需的步驟
    - 指定回答格式
  - 在模型匆忙下結論之前，指示模型製定出自己的解決方案，

情緒分析

- 原始回答會一個句子回覆
- 指定回答的方式（Positive or Negative）
- 列出有的情緒
- 辨認文章是否出現某情緒
- 從文章萃取資訊、整理成JSON格式
- 一次多個任務
  - 所有問題一次問是可以的，只要條列清楚問題，並加上清楚的指示。

實作參考內容

課程 DEMO

- https://github.com/lzrong0203/AI-202306

參考資料
Websites

1. 許鈺屏，Apr 2023, "人工智慧是什麼？AI應用案例、技術、未來發展都有的必修知識包來了", 未來城市@天下，<https://futurecity.cw.com.tw/article/2228>
2. 彭文暉，Dec 2022, 人工智慧理財之監理問題研析，立法院法制局，<https://www.ly.gov.tw/Pages/ashx/File.ashx?FilePath=~/File/Attach/226127/File_19775280.pdf>
3. Google Cloud Tech, May 2023, Introduction to Generative AI, <https://youtu.be/G2fqAlgmoPo>
4. Ask ChatGPT, <https://chat.openai.com/>
5. DeepLearning.AI & Stanford University, Machine Learning Specialization, <https://www.coursera.org/specializations/machine-learning-introduction>
6. 人工智慧, viewed June 2023, wiki, <https://zh.wikipedia.org/zh-tw/人工智慧>
7. 人工智慧史, viewed June 2023, wiki, <https://zh.wikipedia.org/zh-tw/人工智慧史>
8. StatQuest , Logistic Regression <https://www.youtube.com/playlist?list=PLblh5JKOoLUKxzEP5HA2d-Li7IJkHfXSe>

參考資料

Books

- Géron, A. (2022). Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow. " O'Reilly Media, Inc.".
- Raschka, S., Liu, Y. H., Mirjalili, V., & Dzhulgakov, D. (2022). Machine Learning with PyTorch and Scikit-Learn: Develop machine learning and deep learning models with Python. Packt Publishing Ltd.
- Stephan Raaijmakers, (2022). Deep Learning for Natural Language Processing. Manning Publications