

人工智慧 Artificial Intelligence

Dr. Steve Lai
Mathison Intelligence
2024/07 @ FinTech of TABF

Resume

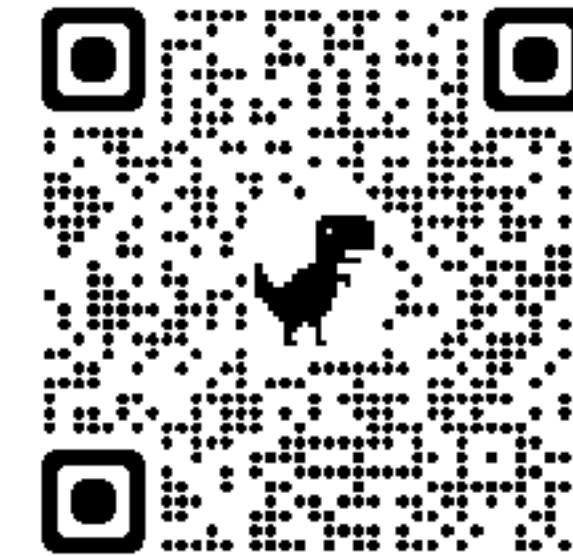
- Steve Lai (賴昭榮)
- 學經歷：
 - Founder and CEO at **Mathison Intelligence** (麥錫森智能)
 - ◆ plusForm — 線上課程品牌

- 台灣大學資訊工程博士
- 中央大學資訊工程碩士

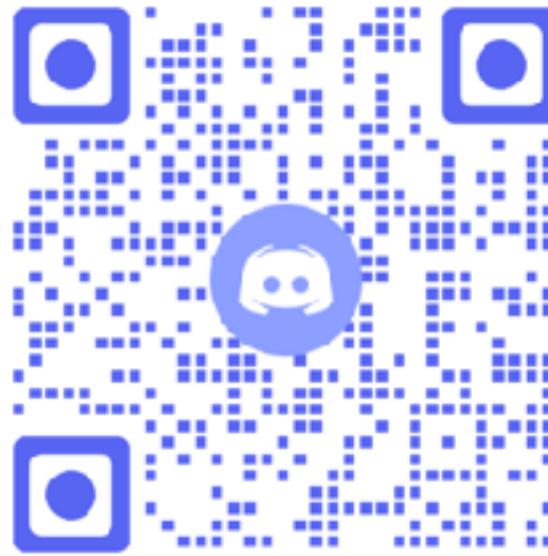
→加入 plusForm 討論群

→AI 科普相關頻道與討論群

→Youtube



Discord



LINE
openChat



機器的智慧

- 像人一樣思考
 - 強調機器在思維過程中模仿人類的方式，是否能夠模擬人類的認知過程，包括解決問題、推理和學習。
 - 圖靈測試
- 像人一樣行動
 - 關注機器在行為上的表現是否與人類相似，能否在現實世界中執行複雜的任務，如機器人操作、語音識別和自然語言處理等
- 理性地思考
 - 強調機器在推理和決策過程中的理性，意味著機器能夠基於邏輯和數據進行推理，並做出最佳決策。
- 理性地行動
 - 這個面向關注機器在行動過程中的理性表現，意味著機器能夠根據環境和目標做出最佳行動選擇，並且能夠適應變化的環境。

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"。
- 讓機器與人類進行對話，判斷機器是否能夠像人類一樣思考。如果機器的回答讓人類無法分辨其身份，那麼它就被認為具有智慧。

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

Alan Turing, 1950

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，並促進了許多重要的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。

"Artificial intelligence (AI) is the science and engineering of making intelligent machines, especially intelligent computer programs.

It is related to the similar task of using computers to understand human intelligence, but AI does not have to confine itself to methods that are biologically observable". "

John McCarthy, 1956

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「智慧」。其中 1957年，Perceptron提出。
- **第一次 AI 寒冬：1974 - 1980**。AI研究者們對其課題的難度未能作出正確判斷，加上過高的期望與有限的計算能力，對 AI 的資金縮減，研究進展放緩。
- **專家系統興起：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

1987年 - 至今

- **第二次 AI 寒冬：1987 - 1993。**專家系統的侷限性顯現，包含知識獲取困難、缺乏創造性與學習力、開發與維護成本的問題。
- **AI 復興與大數據：1993 - 2011。**機器學習、大數據等技術興起。利用統計學理論來處理和分析資料，從中提取有用的關連與知識。雲端運算與分散式平行運算的進步，加速資料的來源與大數據的處理分析。

人工智慧的歷史

深度學習，大資料和人工智慧：2011至今

- 進入21世紀，得益於大資料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。
- 語音辨識、影像辨識等領域的突破性進展。大型語言模型（Large Language Models, LLMs）與生成式 AI（Generative AI, GAI）的突破，朝向通用人工智慧（Artificial General Intelligence，AGI）發展。

人工智慧的里程碑

1. 1943年 - 沃倫・麥卡洛克和沃爾特・皮茨提出神經網絡 (neural network) 概念，為神經網絡和連結主義奠定基礎。
2. 1950年 - 艾倫・圖靈提出"圖靈測試"，用於判斷機器是否具有智慧。
3. 1956年 - 達特茅斯學院舉行人工智慧研討會，正式提出"人工智慧"這一術語，標誌著AI作為一門學科的誕生。
4. 1997年 - IBM的深藍(Deep Blue)戰勝西洋棋世界冠軍卡斯帕羅夫，成為里程碑事件。
5. 2011年 - IBM的Watson系統在智力問答節目Jeopardy!中戰勝人類冠軍，展現了問答和自然語言理解能力。

人工智慧的里程碑

6. 2012年 - 多倫多大學Geoffrey Hinton團隊的深度學習系統在ImageNet圖像識別挑戰賽中大幅超越之前的最佳成績。
7. 2016年 - Google DeepMind的AlphaGo擊敗頂尖圍棋選手李世石，在複雜的棋類遊戲中展現了AI的巨大潛力。
8. 2017年 - Google 發表"Attention Is All You Need" 與 Transformer 模型。
9. 2018年 - 谷歌發布BERT模型，大幅提升了自然語言處理系統的性能，為NLP領域帶來突破。
10. 2020年 - OpenAI的GPT-3語言模型發布，以其驚人的語言生成和理解能力震驚業界。
11. 2023年 - ChatGPT問世，流暢的對話大型語言模型的能力 受到全世界矚目。

人工智慧發展簡史

第一波

1950-1960

符號邏輯

失敗

把人的思考邏輯放進電腦

由領域專家寫下決策邏輯。

人類還沒辦法清楚理解自己的思考過程，如何告訴電腦？

第二波

1980-1990

專家系統

失敗

把人的所有知識放進電腦

由領域專家寫下經驗規則。

太多難題人類無法解答、無法寫成規則、無法以程式碼表示。

專家系統

專家定義規則

第三波

2010-Present

機器學習

把人的所有看見放進電腦

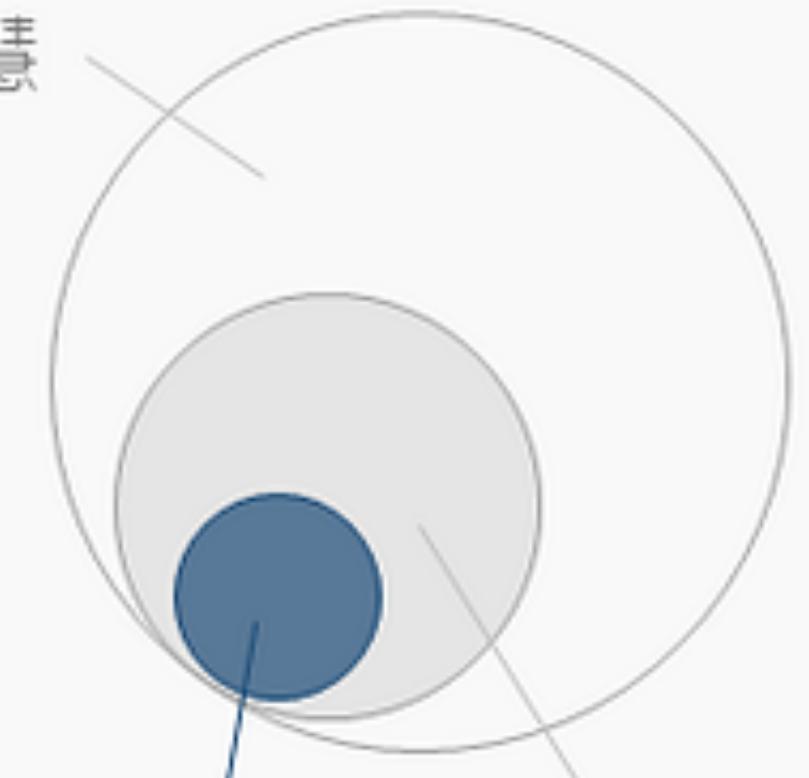
由領域專家提供歷史資料，讓電腦自己歸納規則。

傳統機器學習

(與深度學習區隔)

電腦定義規則
專家定義特徵

人工智慧



機器學習
(第三波人工智能的代表技術)

深度學習
(機器學習技術中成長最快、表現最佳)

深度學習

(多層類神經網路)

電腦定義規則 (更準)
電腦定義特徵

資料來源：《人工智慧在台灣》，劉奕酉整理

人工智慧 (AI)

- 人工智能 (Artificial Intelligence, 簡稱 AI) 是一個跨領域的科學，它的目標是讓機器能夠執行通常需要人類智慧的任務。這些任務包括學習、推理、問題解決、感知、語言理解和生成等。AI 包括但不限於以下幾個主要領域：
 - 機器學習 (Machine Learning, ML)：AI 的一個子領域，透過從資料中學習模式和規則。常見的機器學習技術包括監督學習、非監督學習和強化學習。
 - 深度學習 (Deep Learning, DL)：機器學習的一個子集，利用多層神經網路來模仿人腦的結構和功能。深度學習在影像辨識、聲音辨識和自然語言處理有很好的成果。
 - 自然語言處理 (Natural Language Processing, NLP)：自然語言處理是研究如何使電腦能夠理解、解釋和生成人類語言的技術。大型語言模型的成功即為其中的成果。
 - 電腦視覺 (Computer Vision)：使電腦能夠解讀和理解影像訊息，如圖片和影片的內容。

生成式AI（Generative AI）

- 生成式AI是一種人工智慧技術，能夠創建新的、類似於訓練資料的內容。這些模型使用大量的資料進行訓練，學習資料中的統計特徵，然後根據這些特徵生成新的資料。生成式AI廣泛應用於各種創意領域，如圖像生成、音樂創作、文本生成等。
- 文字生成
 - ChatGPT、Gemini、Llama、BERT
- 圖片生成
 - Midjourney、Dall-E

大型語言模型

- 大型語言模型（Large Language Models, LLMs）為生成式AI的一種主要的應用。利用大量的文本進行預訓練，進而理解語法語意，進而讀寫語言。可用於文本生成、翻譯、問答系統等。
- 可透過微調（Fine-tuning）、檢索增強生成（RAG）等方式改善回答品質
 - 微調：透過給予一定數量問題與答案的集合，改變LLMs的理解
 - 檢索增強生成（RAG）：針對提供資料產生答案

The Key AI Trends for Financial Services in 2024

- 生成式 AI
 - 風險評估、詐欺檢測報告等報告生成
 - 提供個性化財務計畫與投資建議
- 大型語言模型與自然語言處理
 - 智能客服與文件分析
- 機器學習
 - 風險管理與投資組合優化
- 聯邦學習 (Federated Learning) 與機密計算 (Confidential Computing)
 - 跨機構合作與資料隱私

The Key AI Trends for Financial Services in 2024

1. 生成式 AI 的應用增加：43% 使用生成式人工智慧，用於資料分析、投資洞察、市場營銷和個性化銀行服務等內外部應用。
2. 大型語言模型的使用：46% 使用大型語言模型，並結合檢索增強生成（RAG）獲得更好的結果。
3. AI 在金融服務業的普及：超過 75% 認為他們的 AI 能力處於行業領先或相同等級，並且 AI 正在推動風險管理、法規遵從和欺詐檢測等行業特定挑戰。
4. 資料分析和處理仍為主要應用：主要利用 AI 進行資料分析和處理，以及自然語言處理和大型語言模型。
5. 挑戰和障礙：最大的挑戰包括資料隱私與權利等問題（如隱私和數據主權問題）、招聘和留住人工智慧專家以及預算不足。
6. 安全和合規的新途徑：利用聯邦學習和保密計算技術來提高資料隱私和安全性，以滿足如 GDPR 和 CCPA 等合規要求。
7. 未來的投資和發展：計劃增加對 AI 使用、優化 AI 工作流程和生產周期，並增加基礎設施投資。

生成式 AI 於台灣銀行業現況

- 根據台灣金融研訓院 2023 年 10 月研究報告
 - 1. 約有五成銀行業者有專責人力在AI技術創新與應用相關職務，1% 以上相關人力銀行佔約21.5%
 - 2. 五成銀行業者至少與一家AI技術廠商合作
 - 3. 關於 ChatGPT 使用，約7成銀行制定明確規範與指導原則，約五成嚴禁公務使用
 - 4. 關於導入大型語言模型（LLM） ，約有一半觀望一半導入，已上線約佔11%

AI 與機器學習

Artificial
Intelligence

Machine
Learning

Deep
Learning

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

Arthur Samuel, 1959

人工智慧的應用

- 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

- 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

- 人工智能能透過影像辨識技術，增強車輛辨識、號誌管理、交通安全等資訊整合。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

- 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

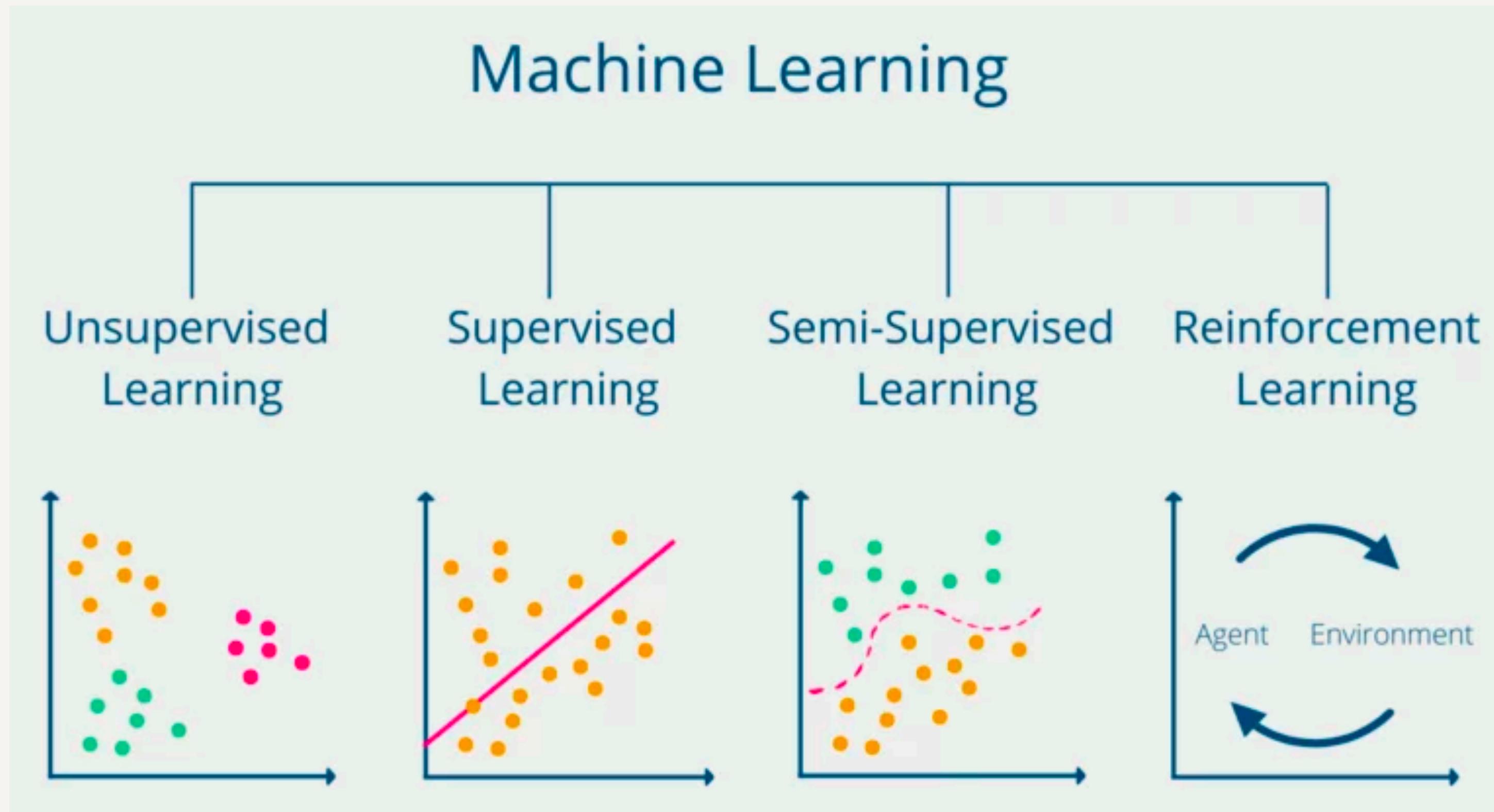
台灣的人工智慧

- 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>

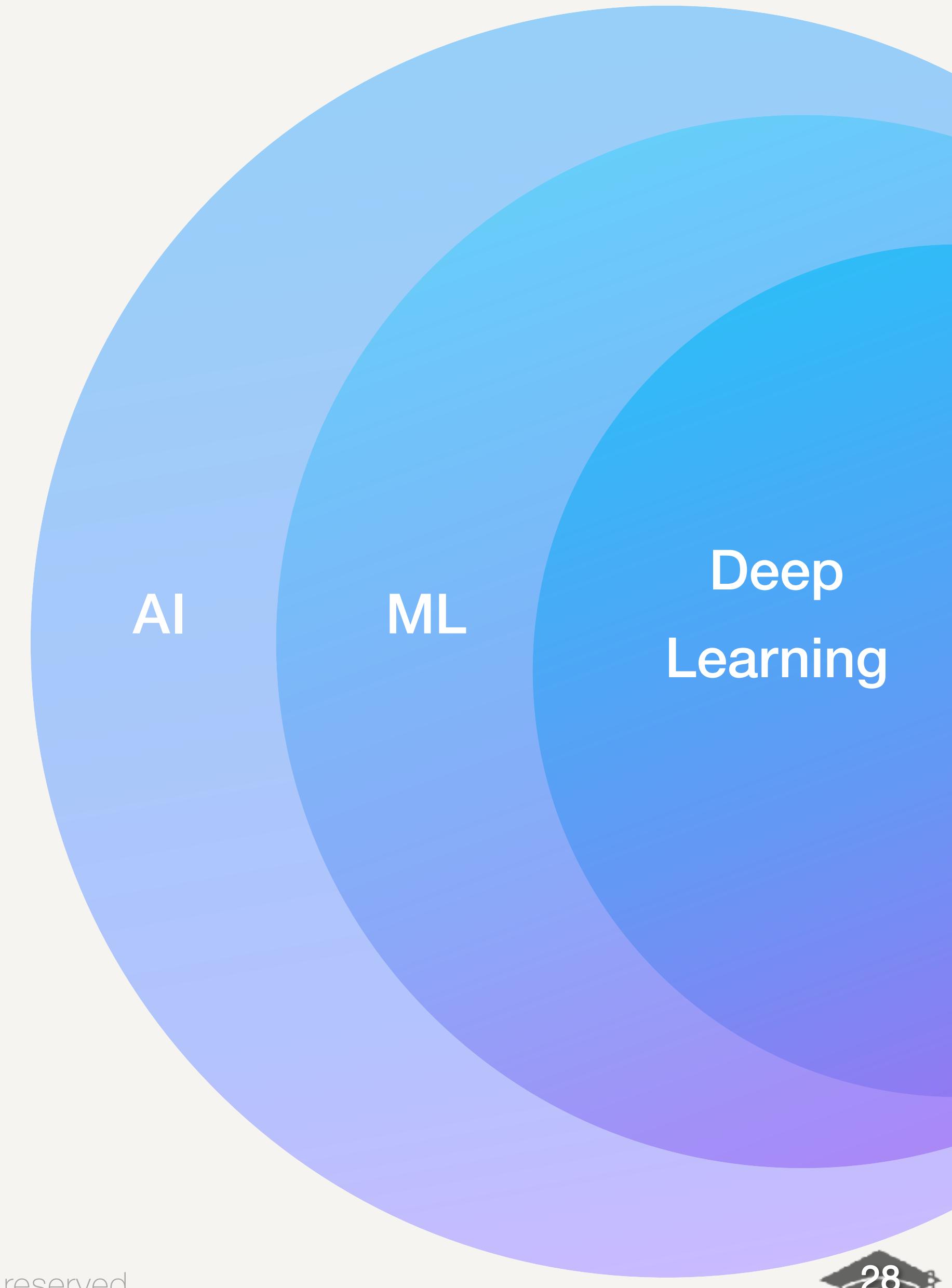
From Data to AI



AI 與機器學習



source: <https://databsecamp.de/en/ml/reinforcement-learnings>



機器學習的分類

- 機器學習技術包括監督學習、非監督學習、半監督式學習和強化學習。
- 監督式學習 (Supervised Learning)
 - 使用帶有正確答案標籤的資料來訓練模型，使其能夠預測新資料的結果。
- 非監督式學習 (Unsupervised Learning)
 - 利用沒有標籤的資料，讓模型自動尋找資料中的模式和結構，如叢集和降維。
- 半監督式學習 (Semi-Supervised Learning)
 - 結合少量帶標籤的資料和大量無標籤的資料來訓練模型，以提高學習效率和準確性。
- 強化學習 (Reinforcement Learning)
 - 通過讓智能體在環境中採取行動並根據獲得的獎勵或懲罰來學習最佳策略。

監督式學習 (Supervised Learning)

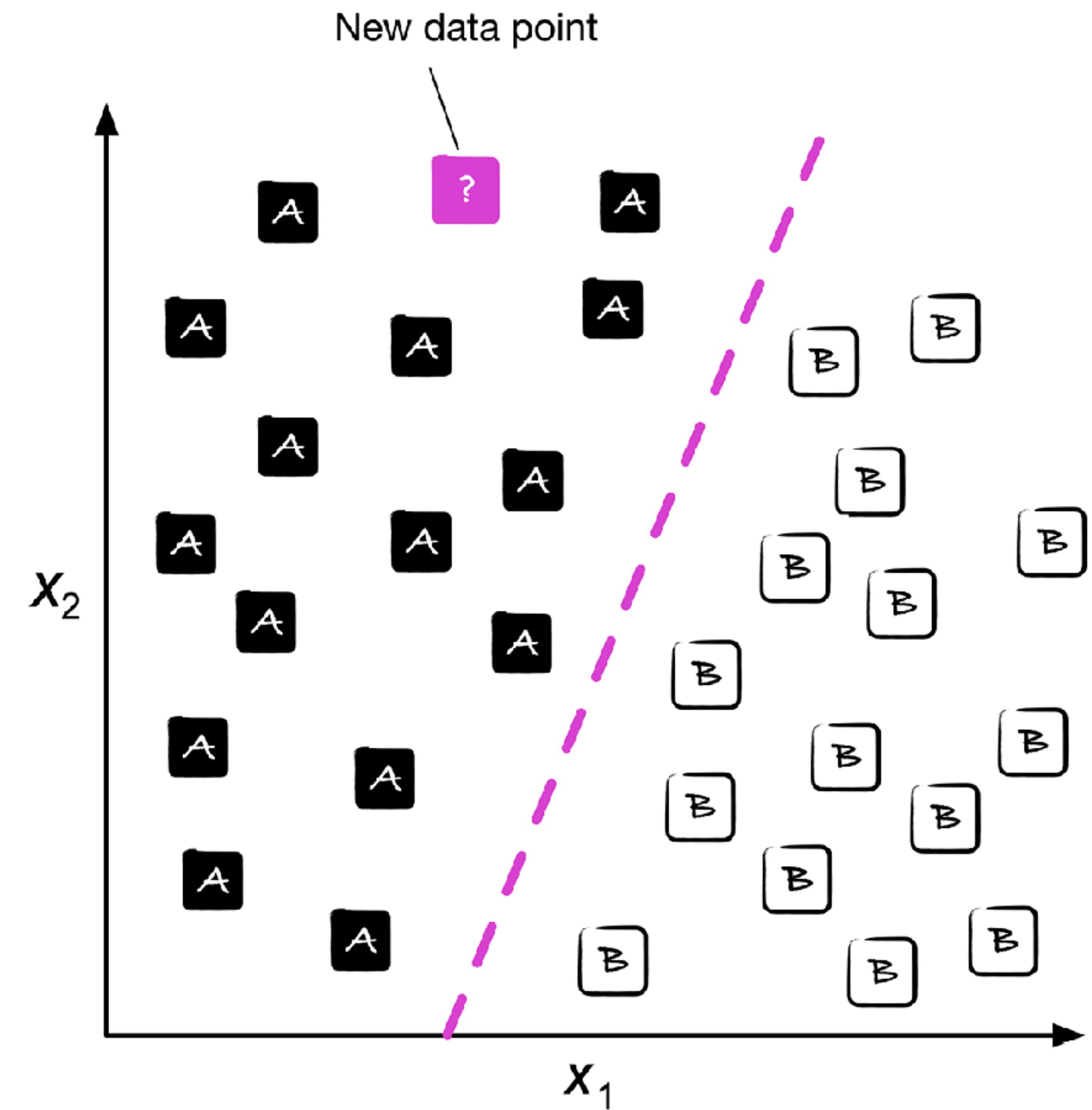
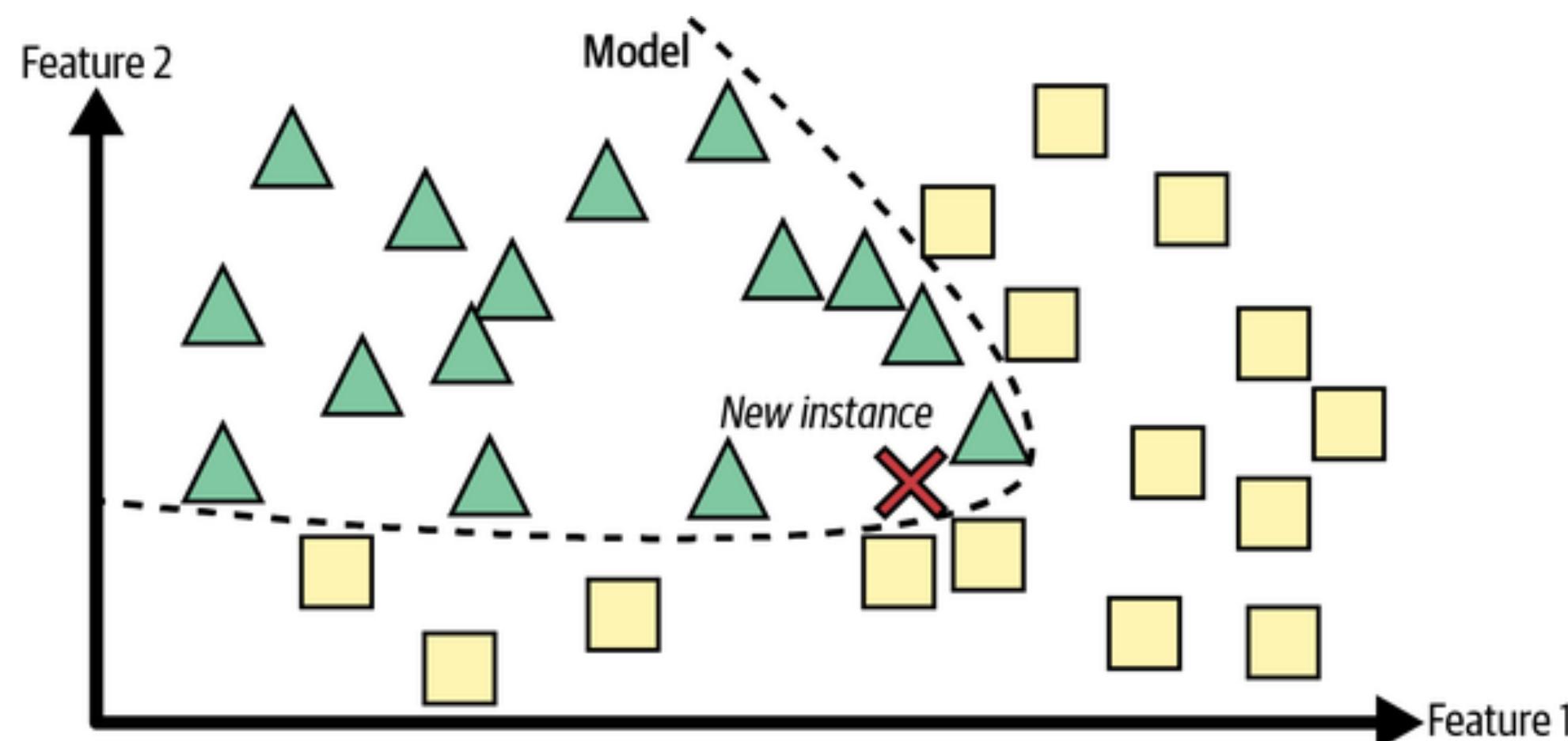
- 模型從標記過的訓練資料（即每筆資料都有明確的答案或標籤）中學習。透過這種方式，模型進行訓練，以便能夠基於輸入特徵預測出對應的輸出。
 - 標籤資料：建立一個垃圾郵件分類器，需要一組郵件，每封郵件都有"垃圾郵件"或"非垃圾郵件"的標記。
 - 特徵（屬性）：垃圾郵件分類器中，特徵可能包括郵件的內容、發送者的mail、信件的長度、標題等。
 - 模型訓練：在訓練過程中，演算法會嘗試找出輸入特徵和輸出標籤之間的關係。
 - 預測：一旦模型訓練完成，它就可以接收新的、未標記的輸入並預測出對應的輸出。垃圾郵件分類器可以接收一封新郵件並預測其是否為垃圾郵件。
- 監督式學習的兩個主要類型是分類（預測的輸出是離散的，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

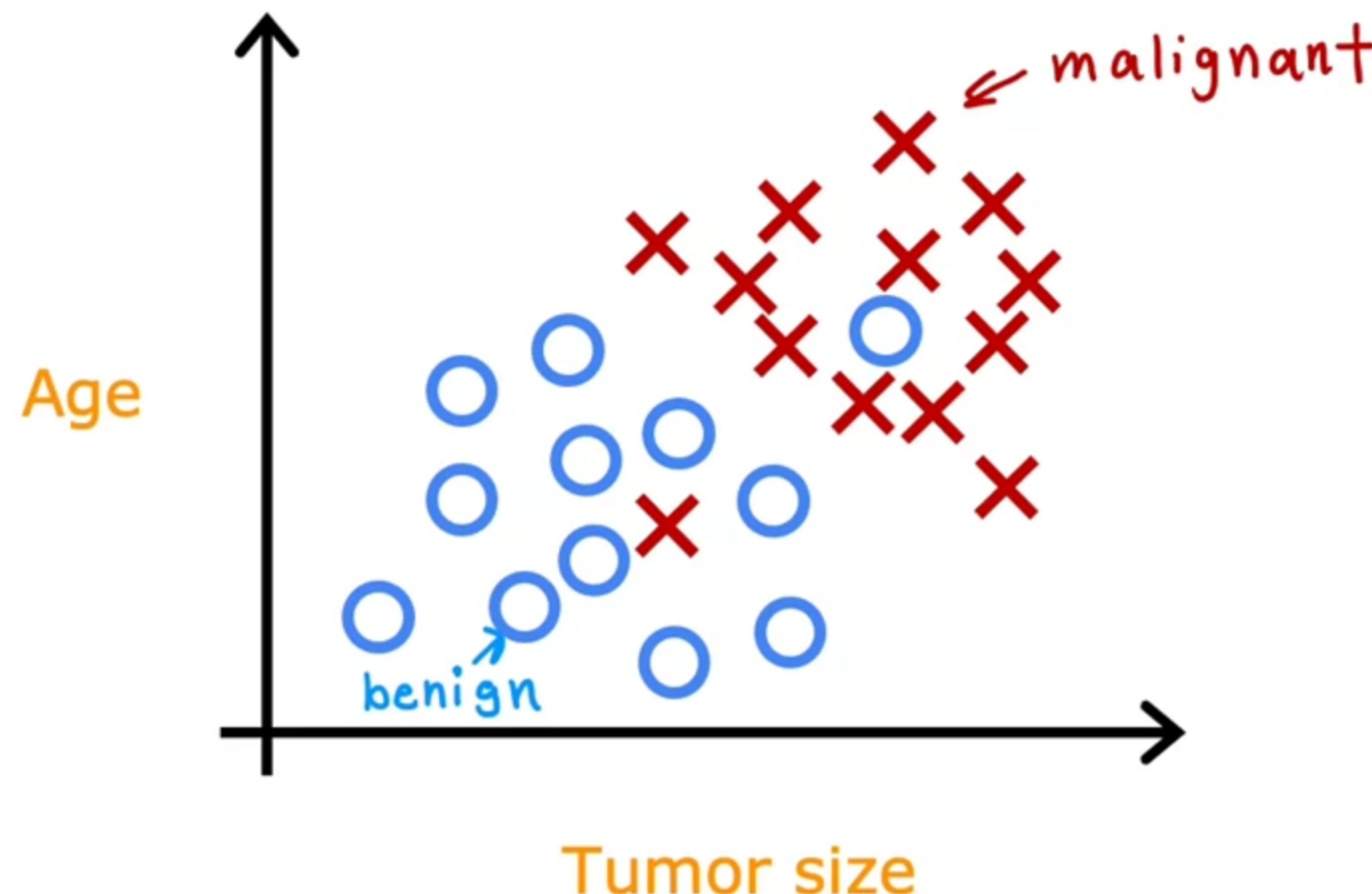
Classification

Predicting Class Labels

- Binary classification



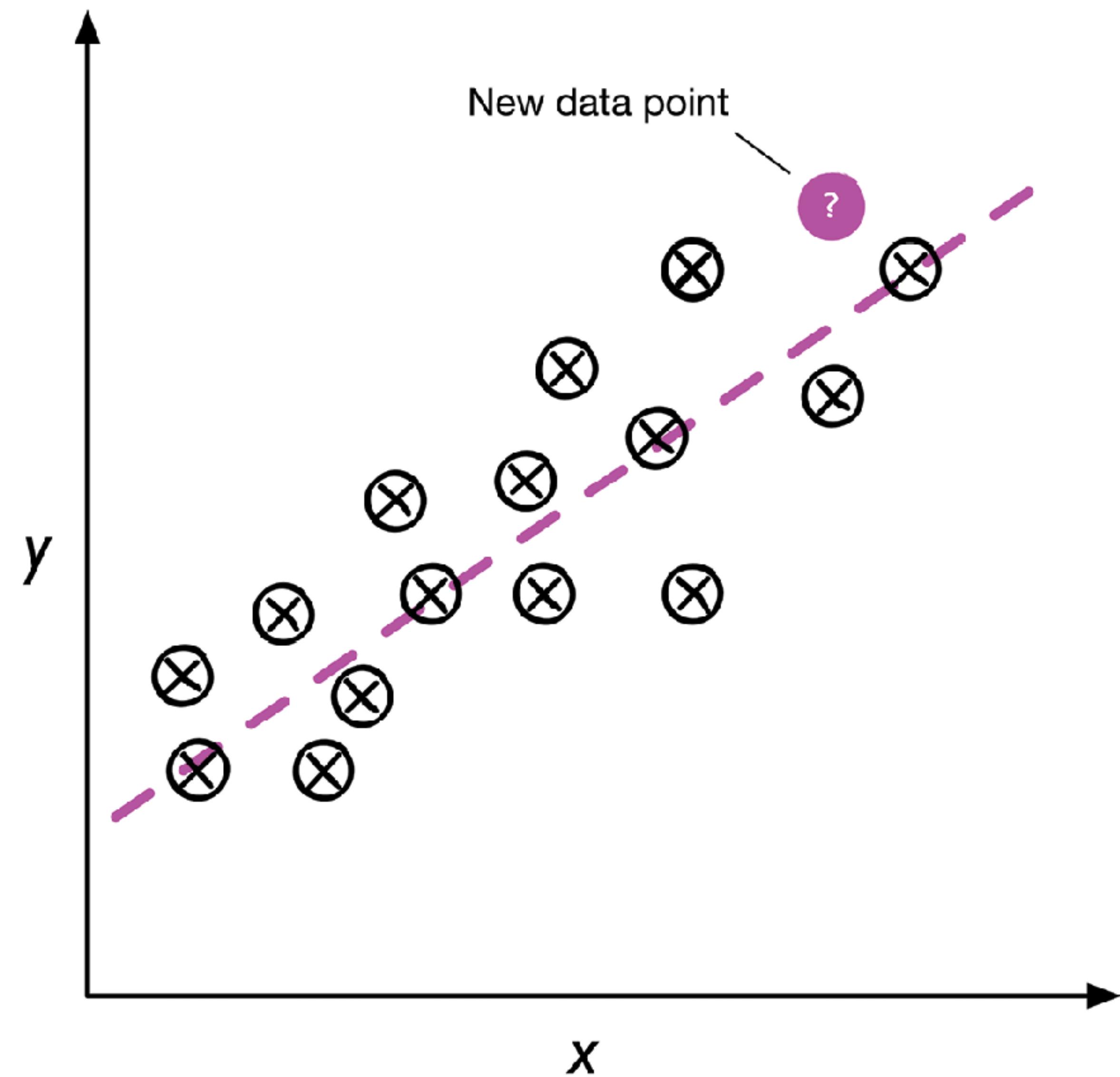
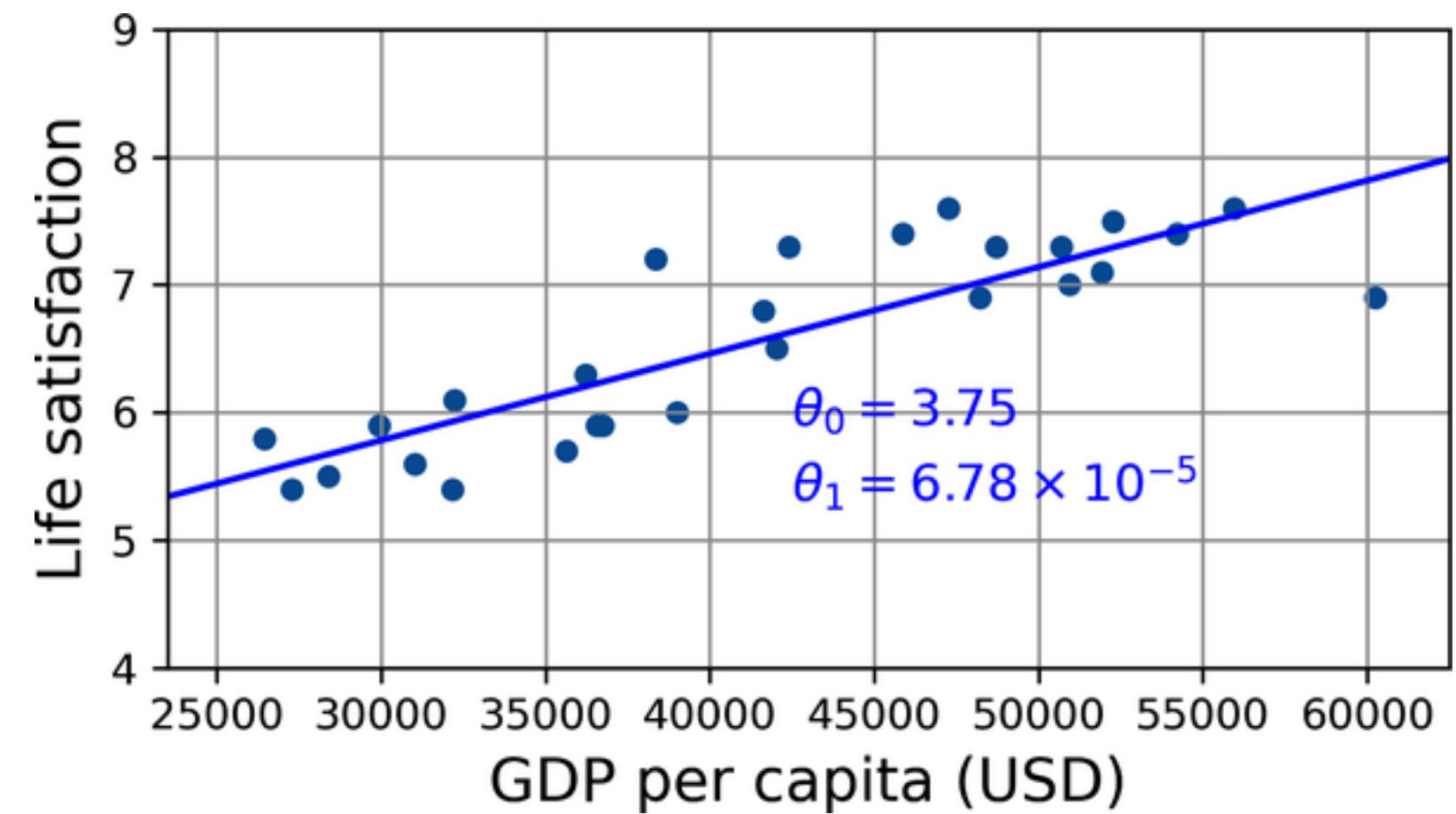
Two or more inputs



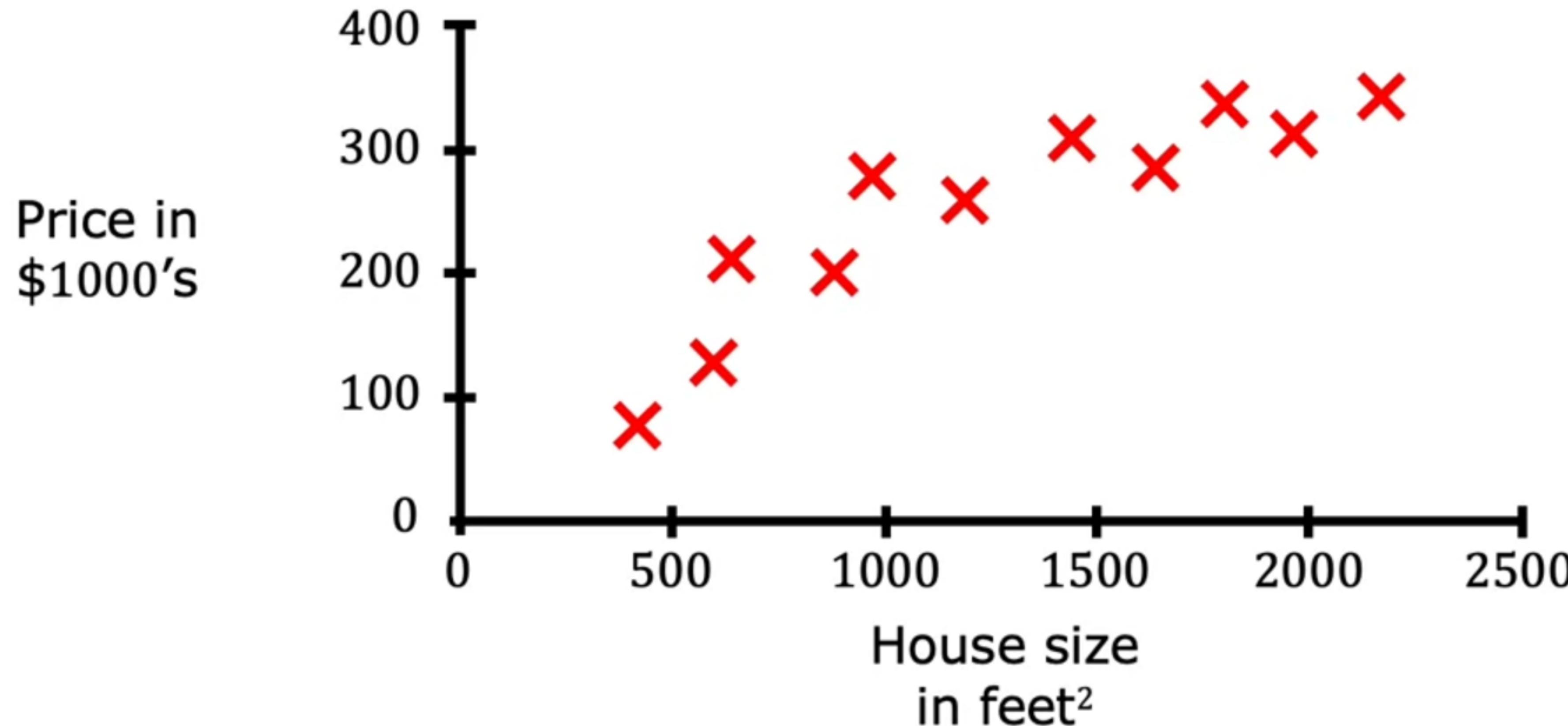
Regression

Predicting Continuous Outcomes

- Linear regression



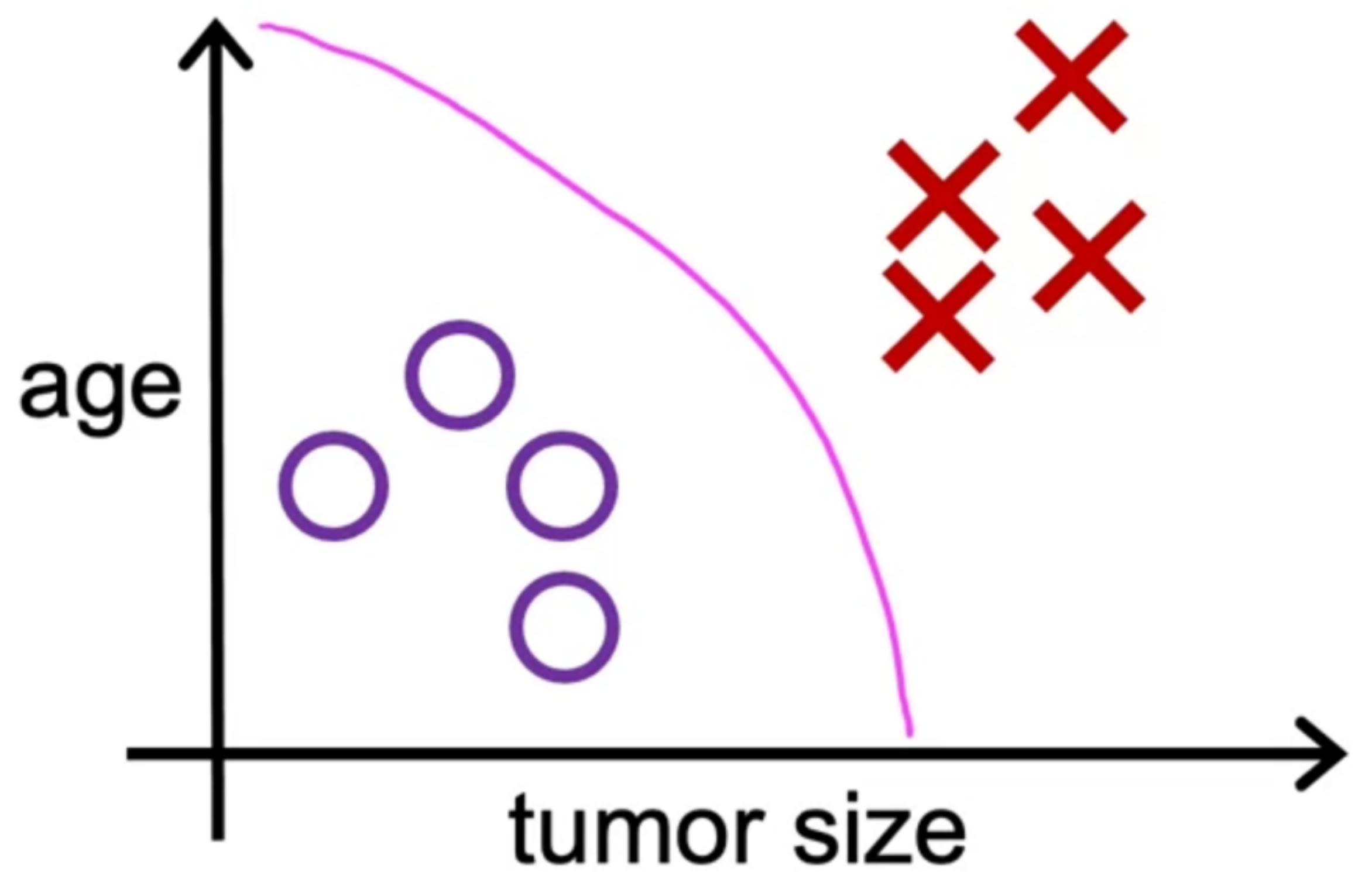
Regression: Housing price prediction



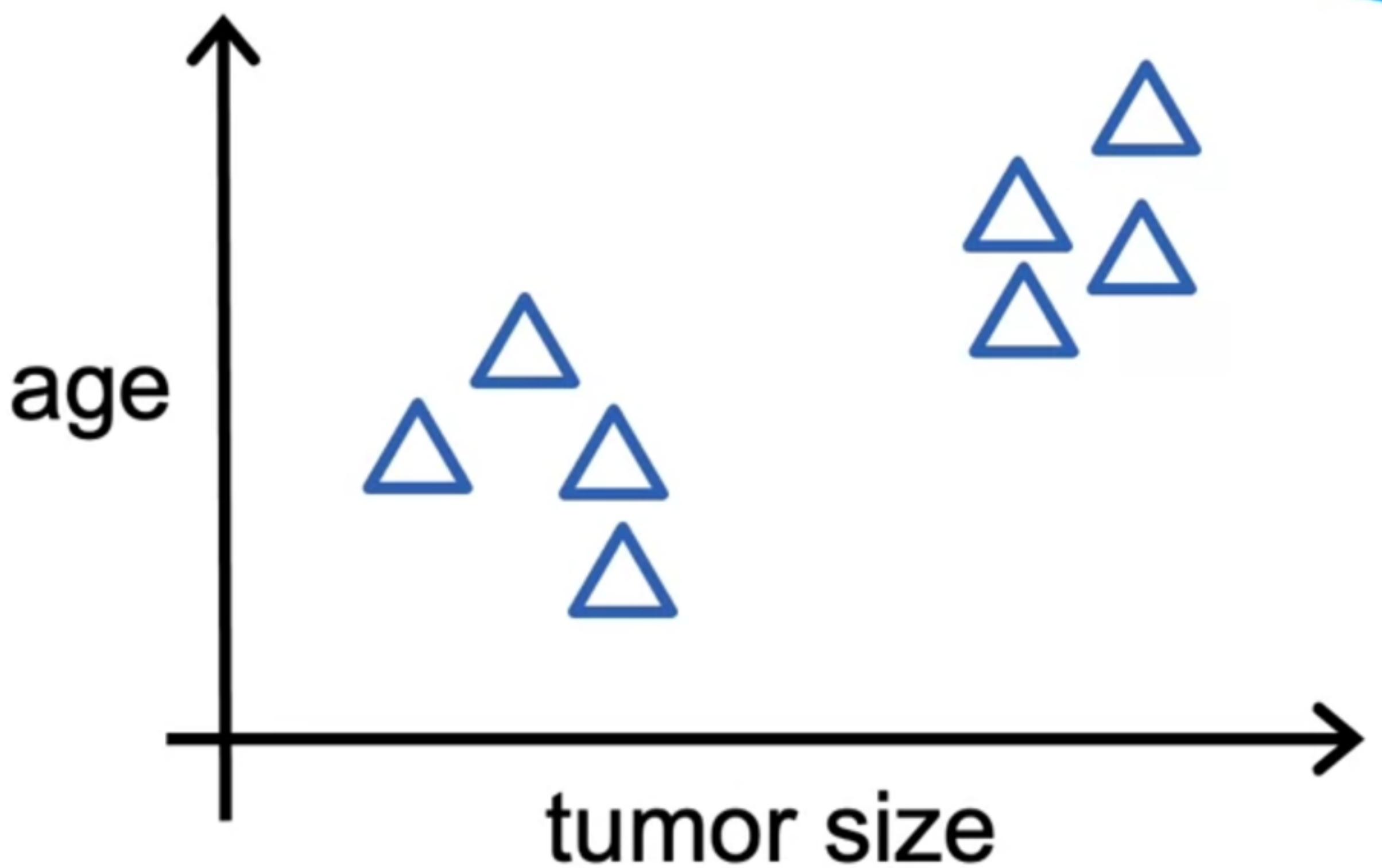
非監督式學習 (Unsupervised Learning)

- 使用未標記的資料來訓練模型。這種學習方式的目標是讓模型從輸入資料的特徵中，學習其結構或模式。
 - Cluster：將資料透過其相似度分組。
 - 降維：目的是減少資料的維度（即特徵的數量），通常用於資料的視覺化或者是在進行監督式學習之前的資料預處理過程。
 - 異常檢測：在異常檢測中，模型被訓練來識別出資料中與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資料，所以它在處理大量無標籤資料時，尤其有用。

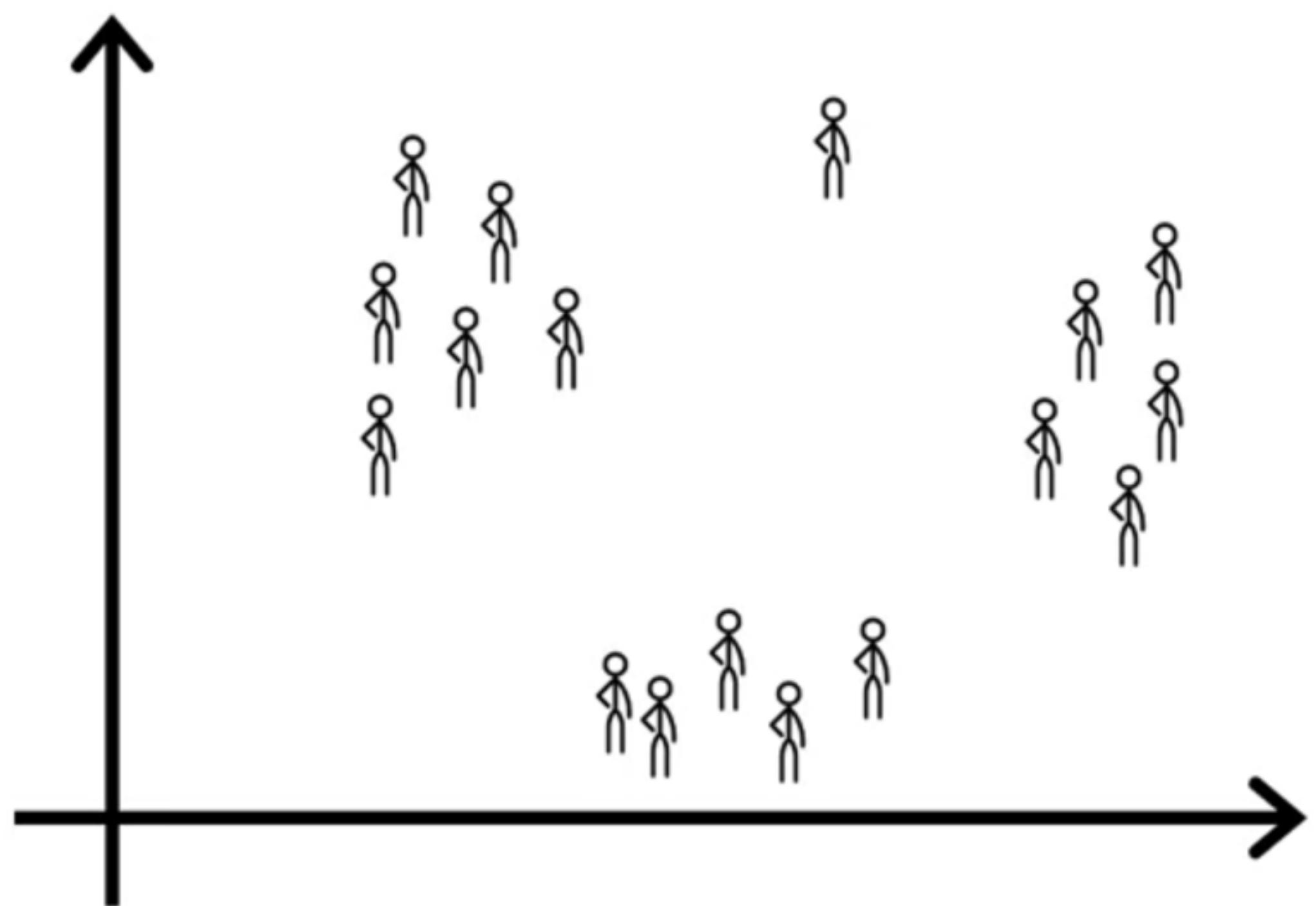
Supervised learning
Learn from data **labeled**
with the “**right answers**”



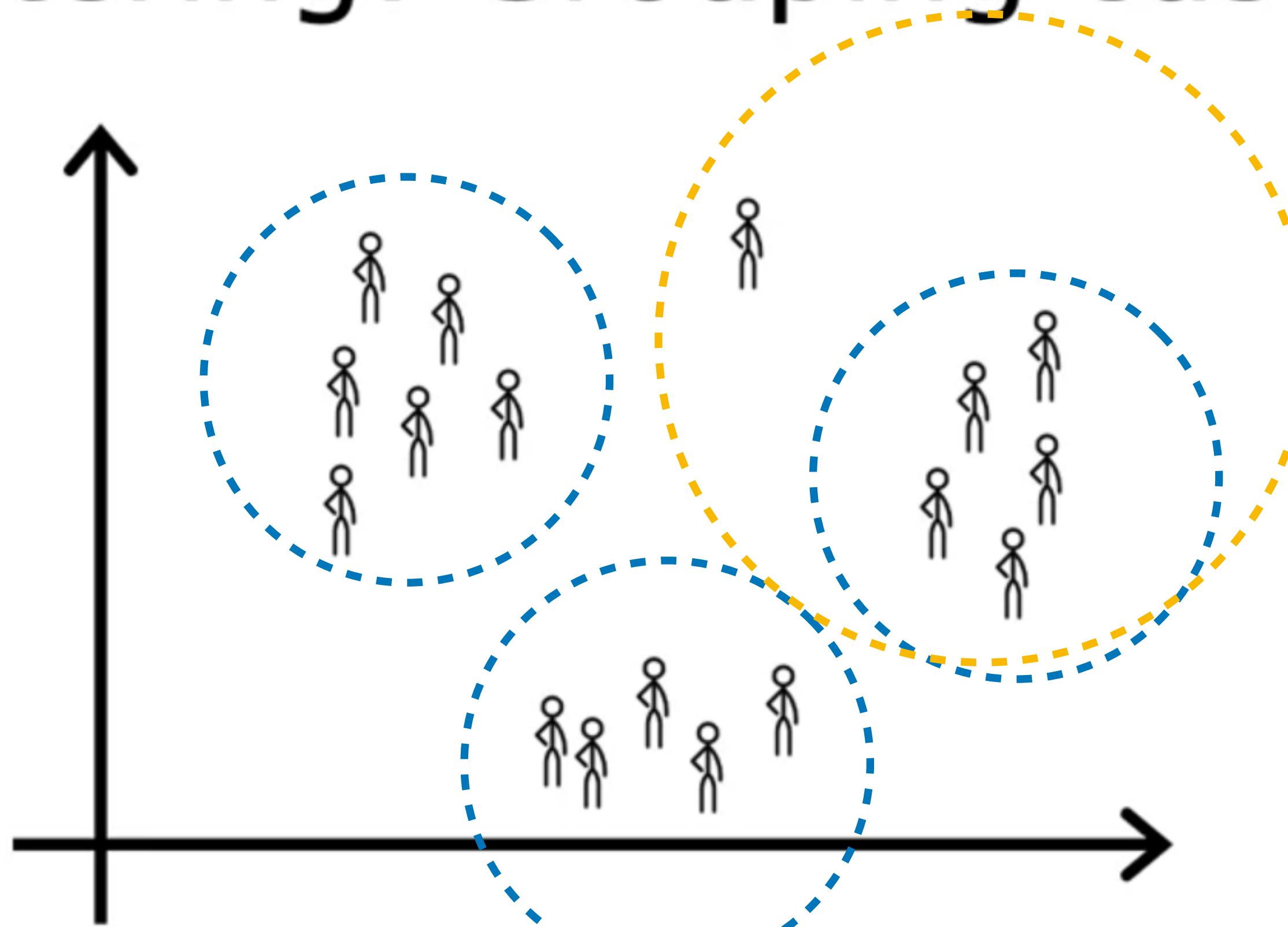
Unsupervised learning



Clustering: Grouping customers

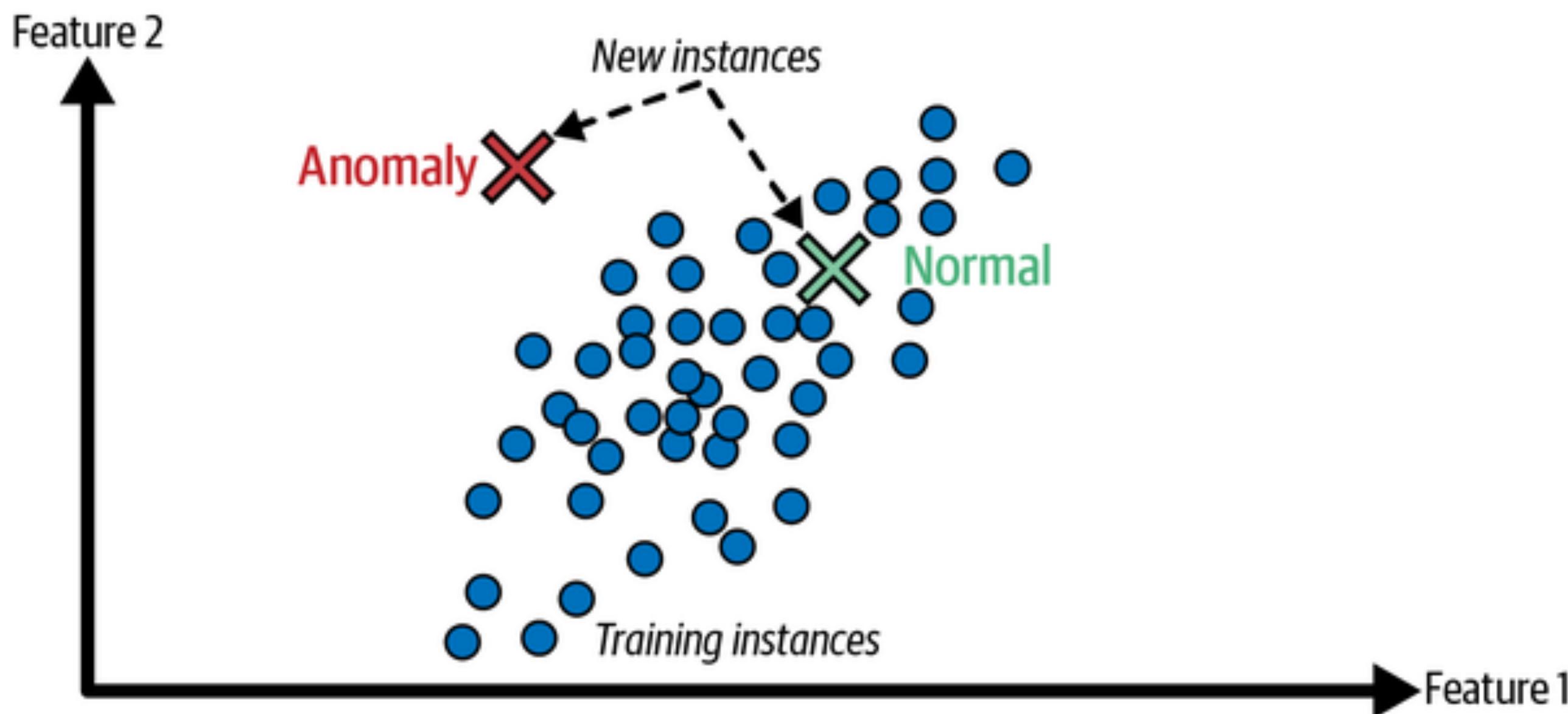


Clustering: Grouping customers



Anomaly Detection

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

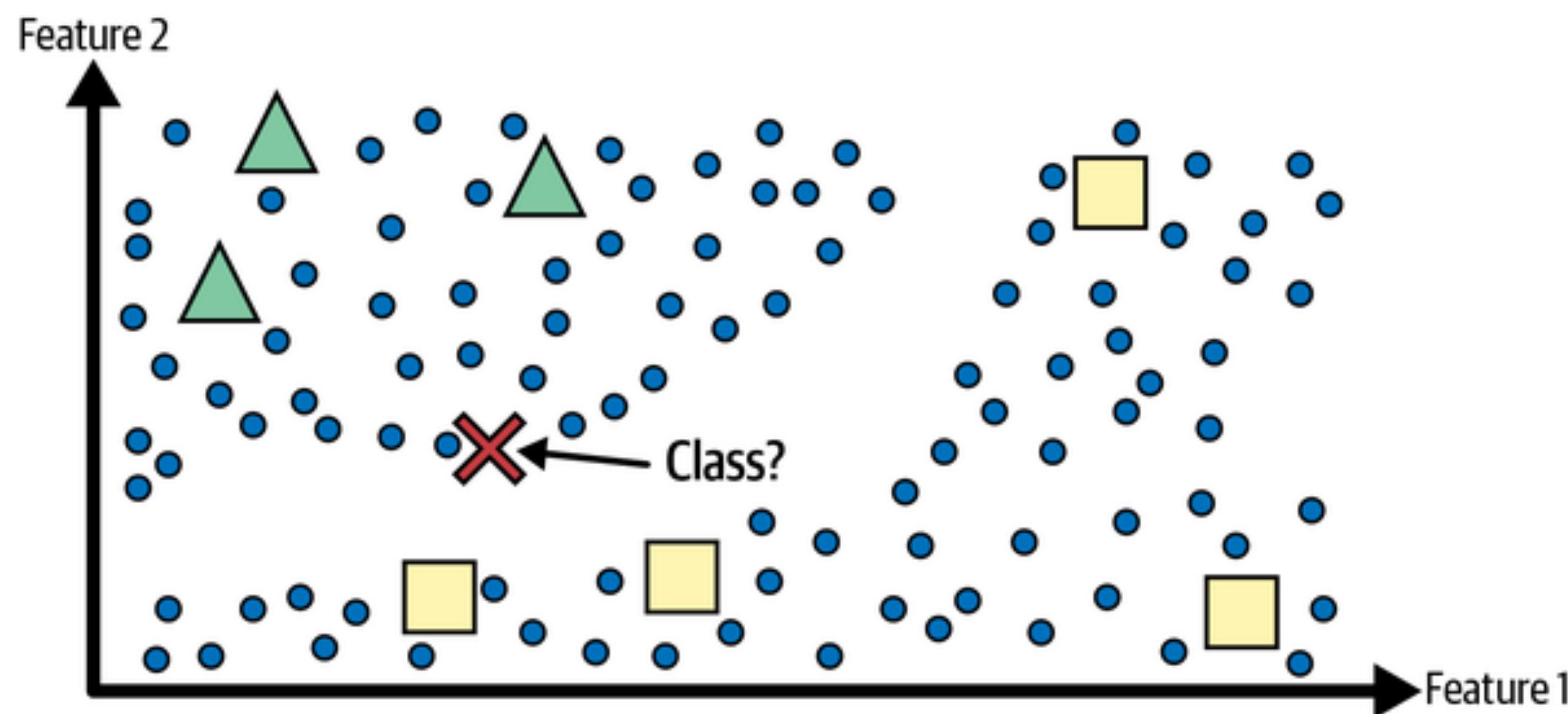


半監督學習 (Semi-supervised Learning)

- 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資料用以建構基礎模型，和大量的未被標籤的資料上用以產生新的資料。
 - 利用未標籤的資料：雖然未標籤的資料不能直接用於訓練或預測，但可用於其資料分布或結構
 - 自我訓練：模型首先在標籤資料上進行訓練，然後使用該模型來產生未標籤資料的預測，這些預測接著被作為新的標籤再次用於訓練。
- 半監督學習尤其適合在有大量未標籤資料和少量標籤資料的情況下使用。
- 由於標籤資料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)

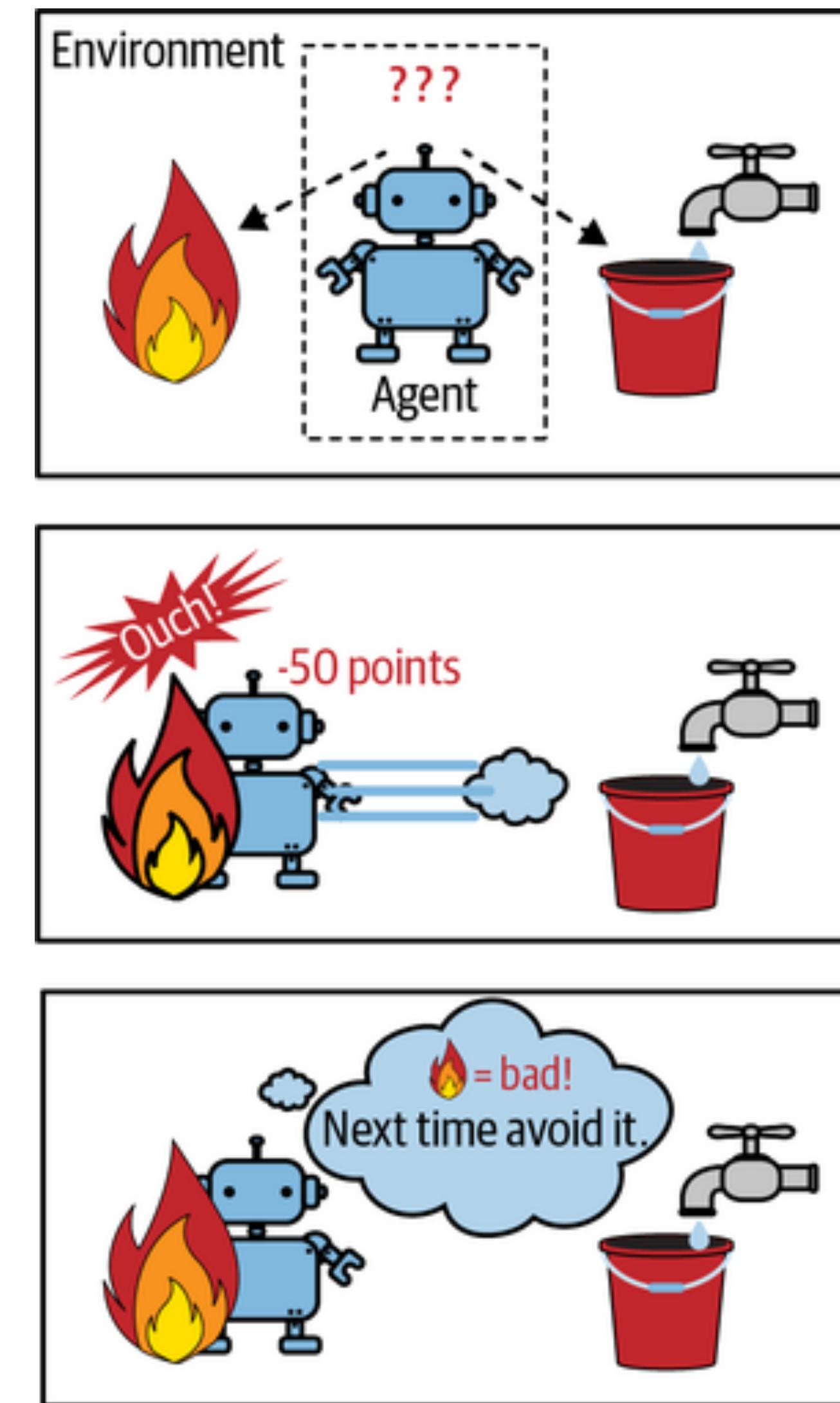
- 一個智能體 (agent) 在與環境 (environment) 的互動中學習如何行動 (action) ，計算其在長期中最大化累計獎勵的一種學習形式 (reward) 。
 - 智能體 (Agent) 、環境 (Environment) 、行動 (Actions) 、狀態 (States) 、獎勵 (Reward) 、政策 (Policy)
- 在強化學習中，智能體的目標是學習一個最佳政策，該政策將在給定狀態下選擇能最大化長期獎勵的行動。
- 這個學習過程通常涉及到探索 (嘗試新的行動) 和利用 (採取已知最佳行動) 之間的平衡，以及處理即時獎勵與長期獎勵之間的權衡。
- 強化學習的範例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。



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

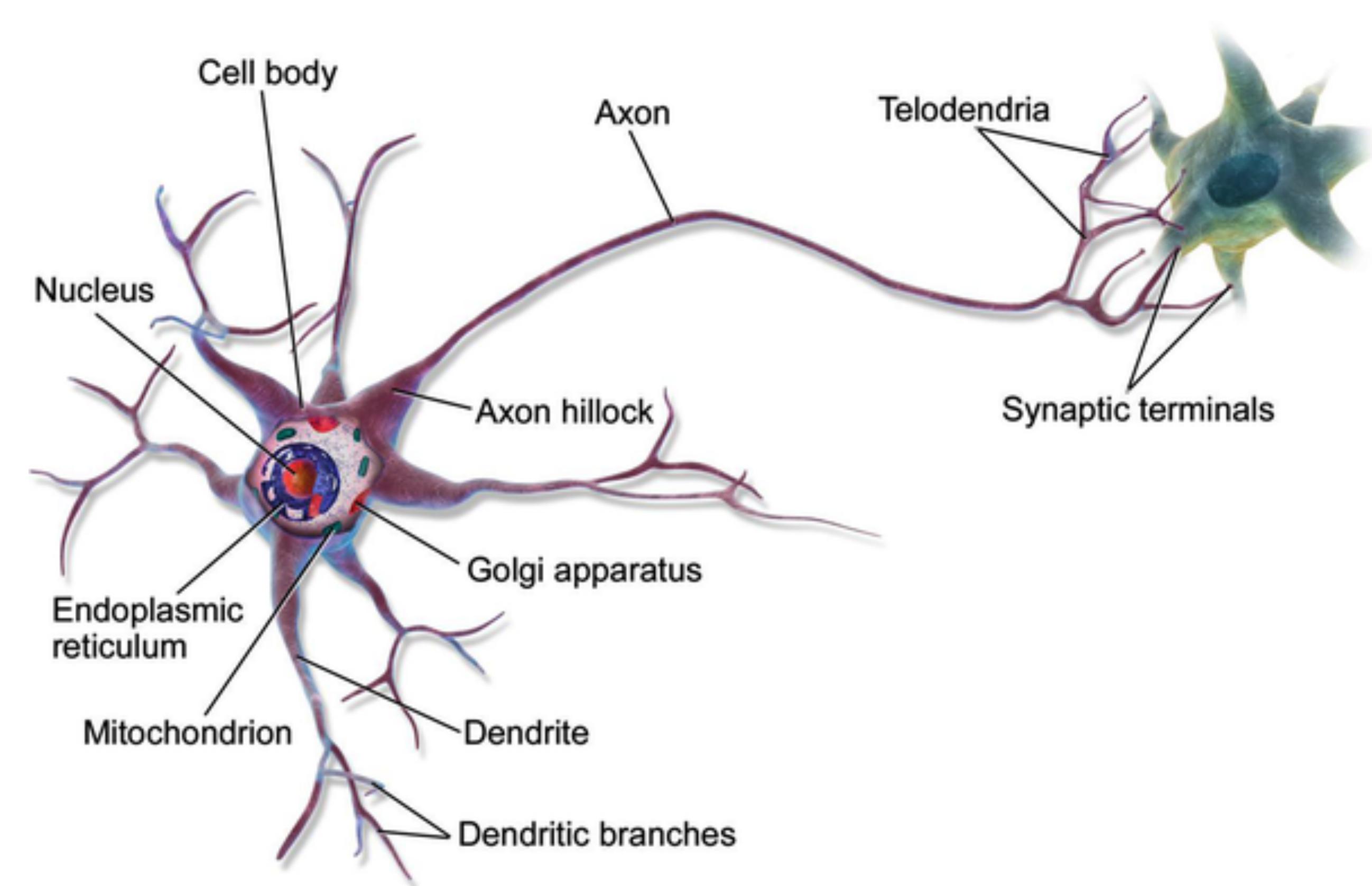
Reinforcement Learning

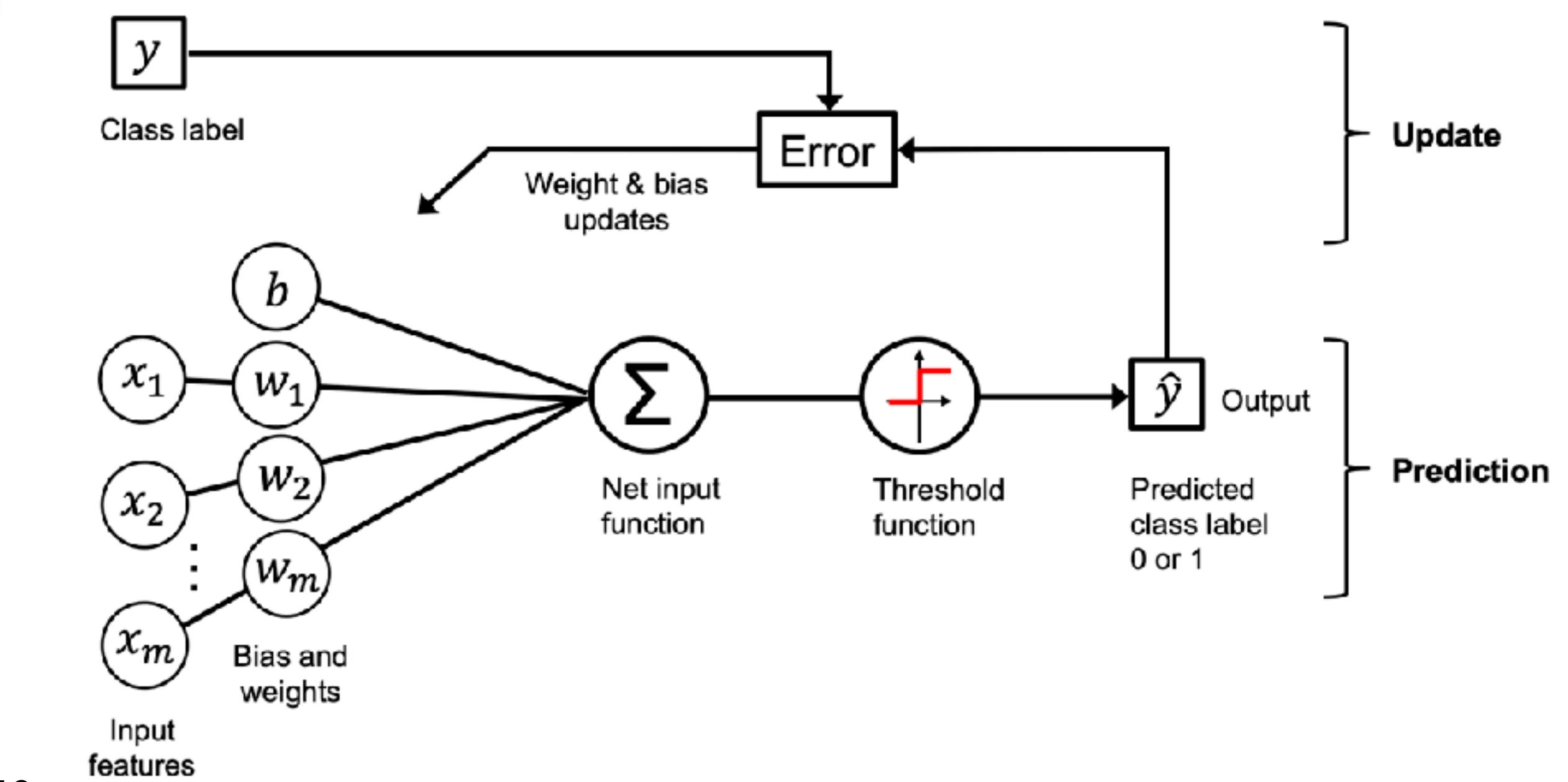
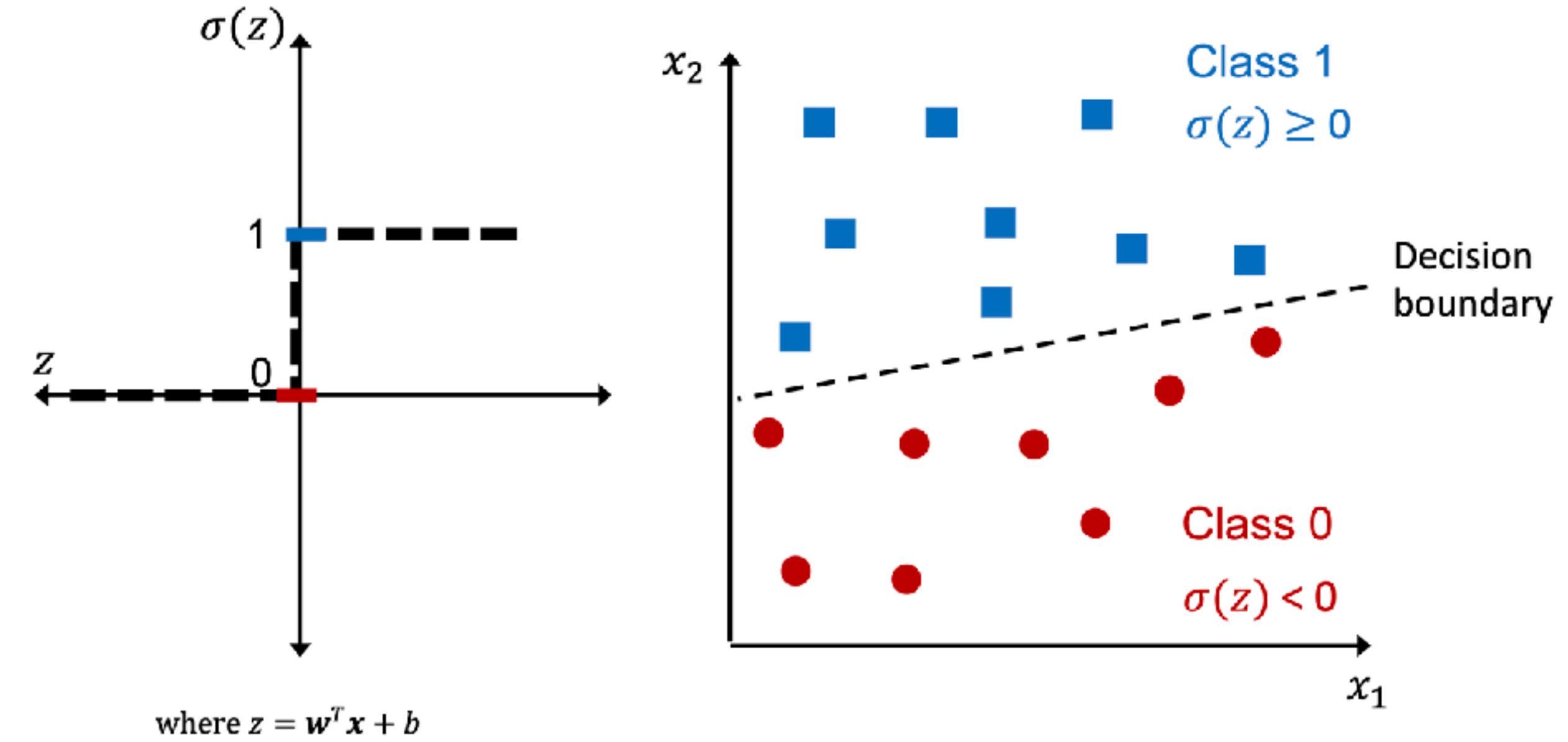
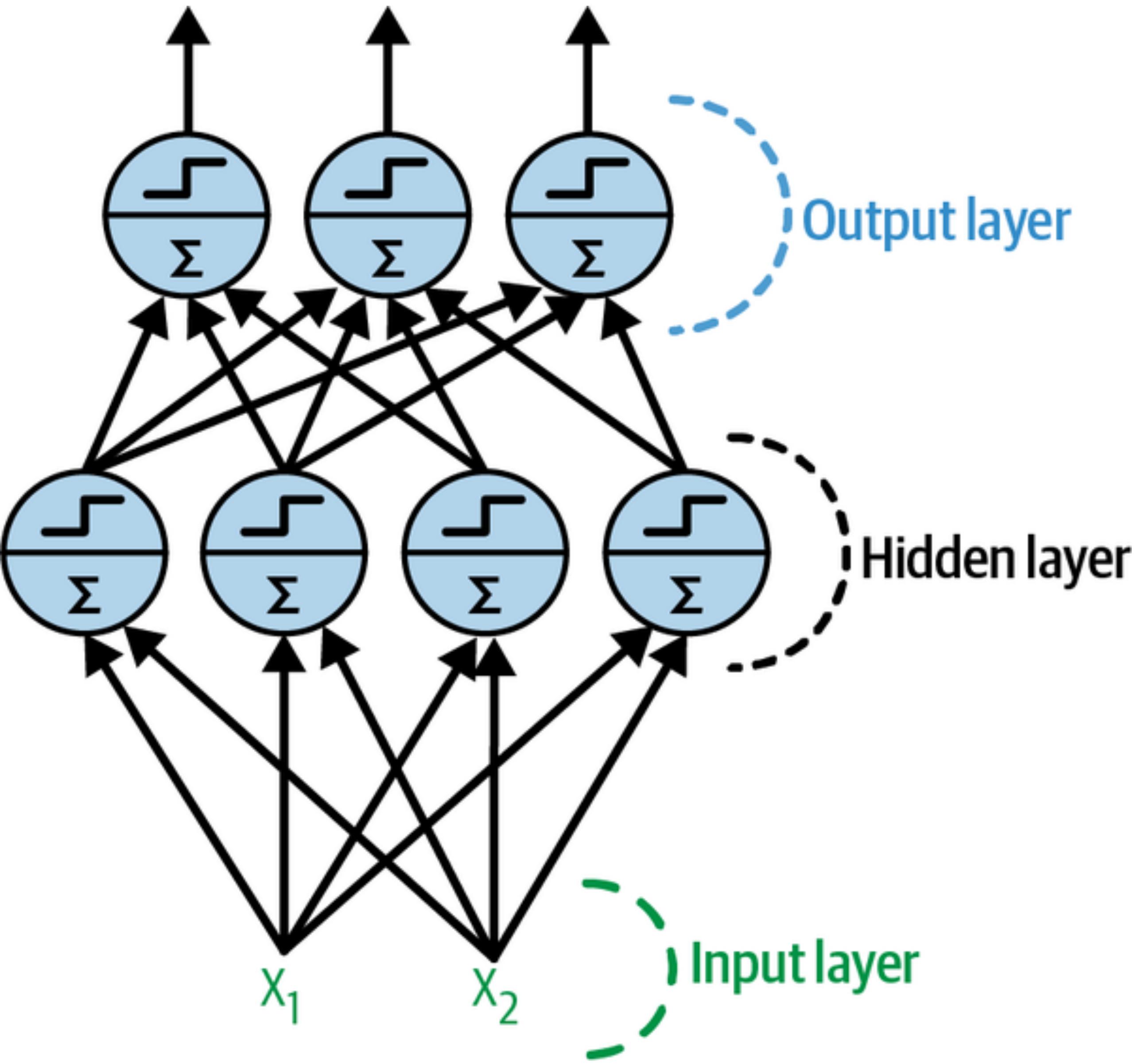
- The learning system, called an agent.
- Observe the environment, select and perform actions, and get rewards in return.
- Learn by itself what is the best strategy, called a policy, to get the most reward over time.
- A policy defines what action the agent should choose when it is in a given situation.

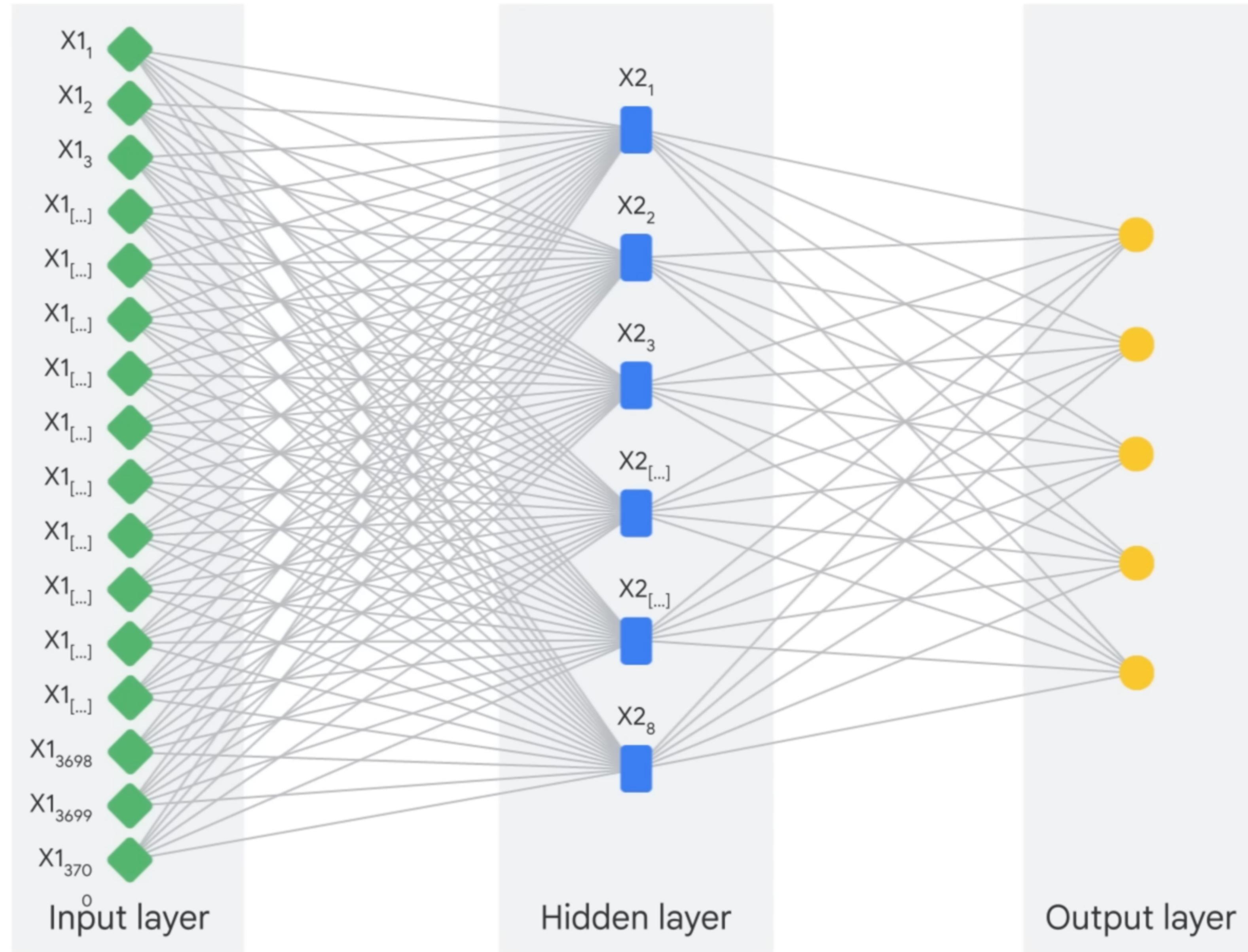


深度學習 (Deep Learning)

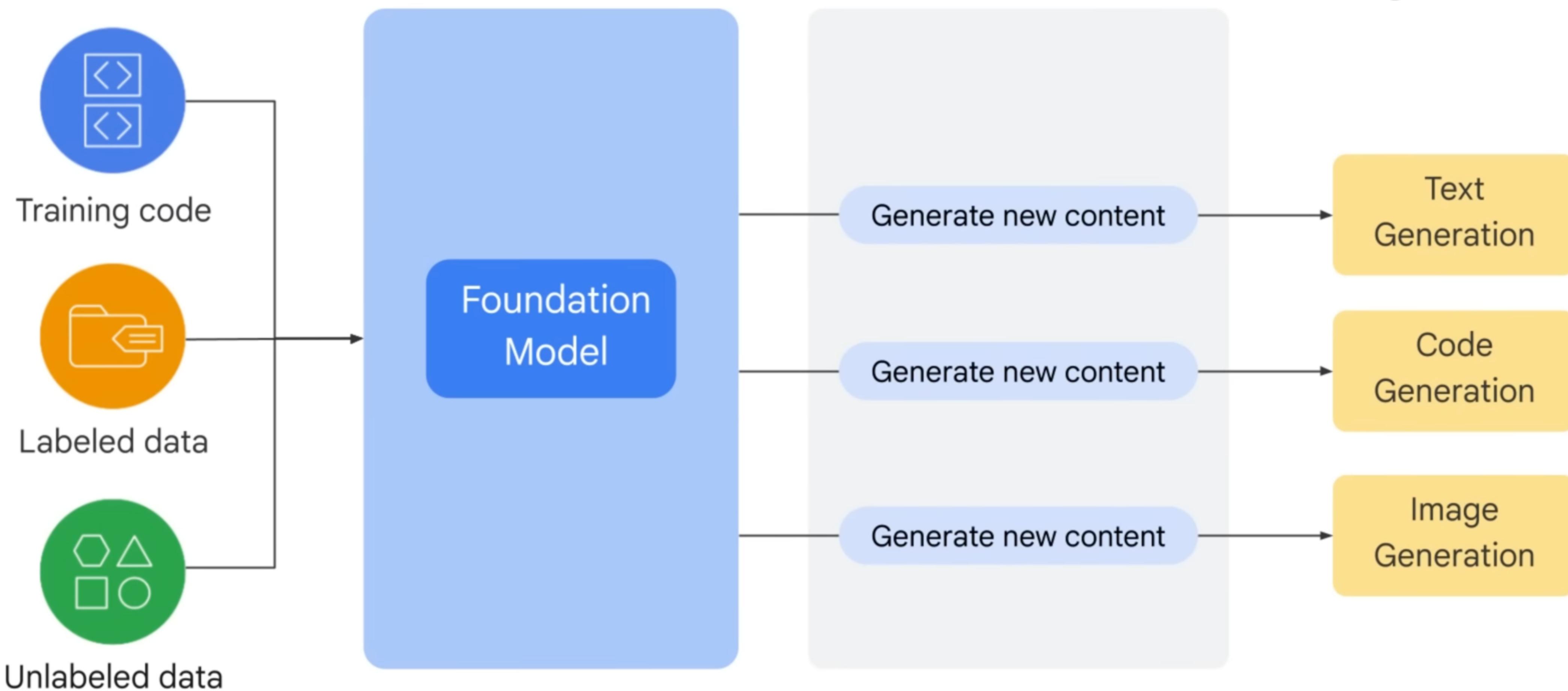
- 專注於利用稱為類神經網路 (Neural Network, NN) 的模型方法來模擬人腦神經元的工作方式。
- "深度"一詞，來自於這種神經網路可以由多個"層"組成，每一層都對應於更高層次的特徵或抽象概念。
- 深度學習模型通過這樣的多層神經網絡來進行學習和預測。



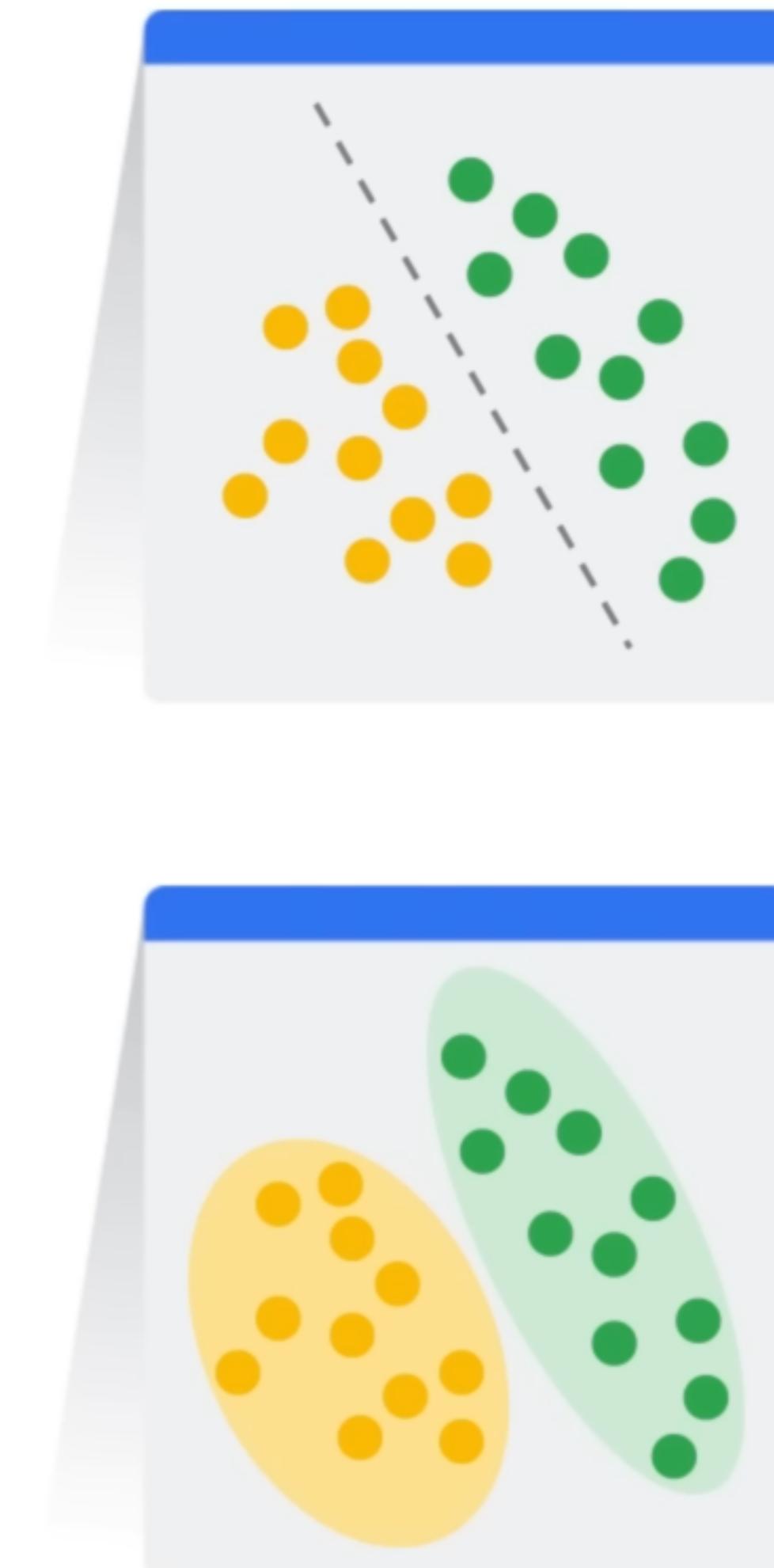




Gen AI Supervised, Semi-Supervised & Unsupervised Learning



Deep Learning Model Types



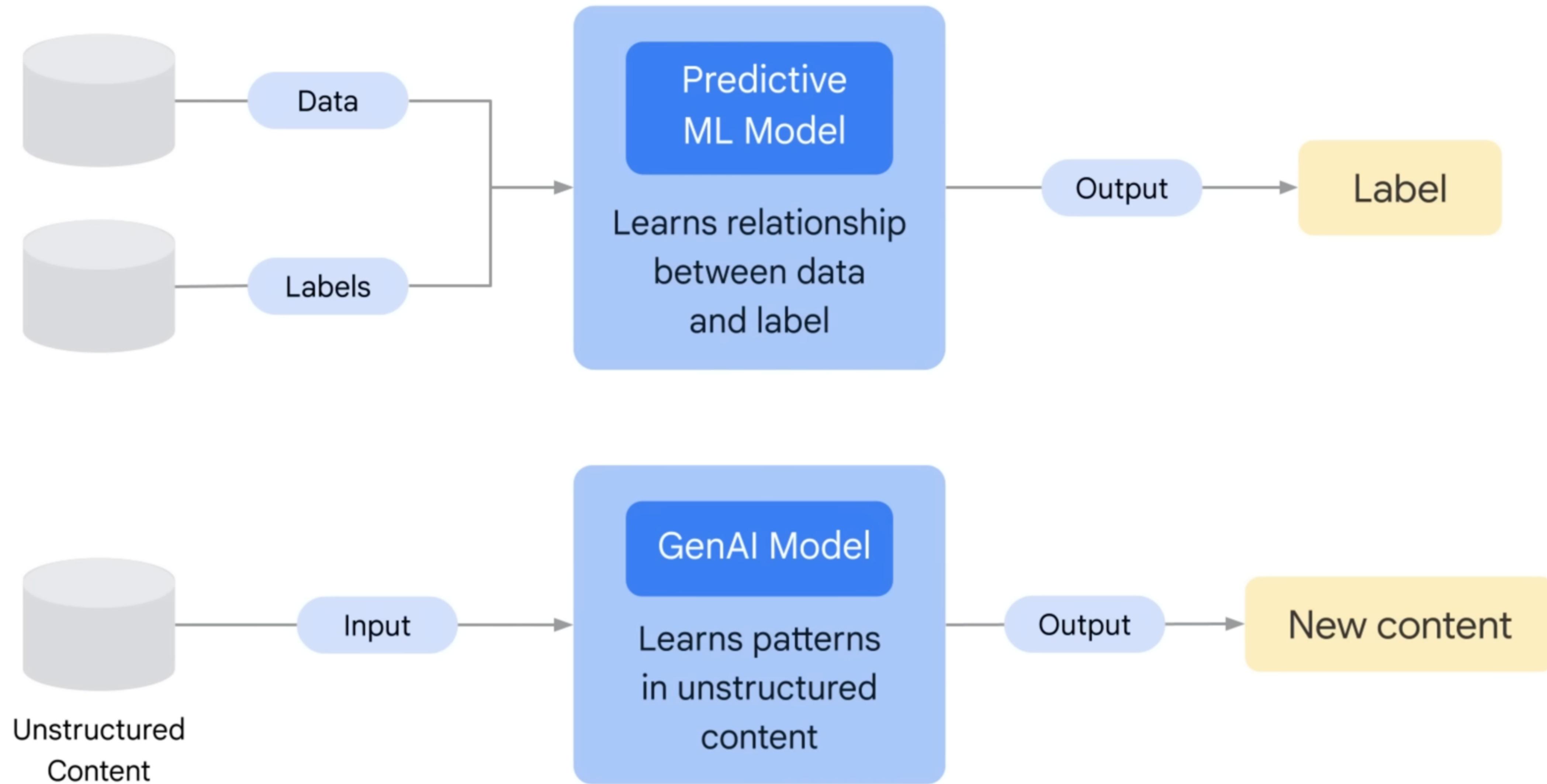
Discriminative

- 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

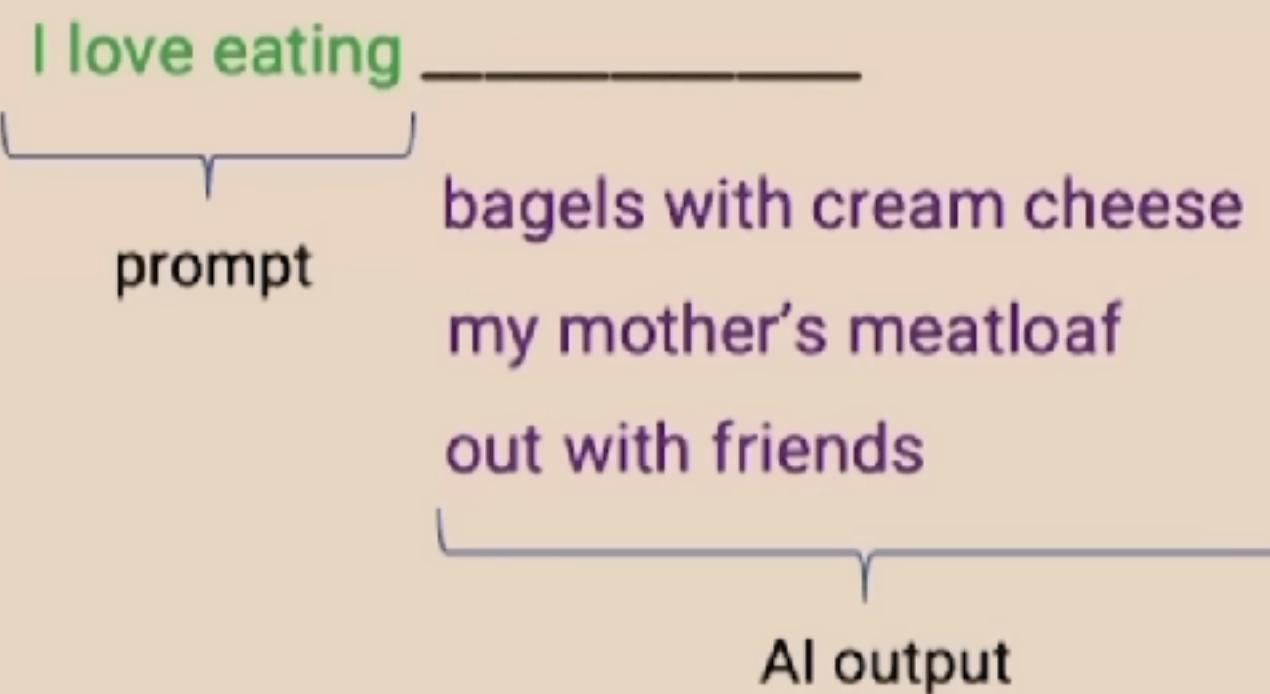




文字生成過程

This decade: Generative AI

Text generation process



How it works

Generative AI is built by using supervised learning ($A \rightarrow B$) to repeatedly predict the next word.

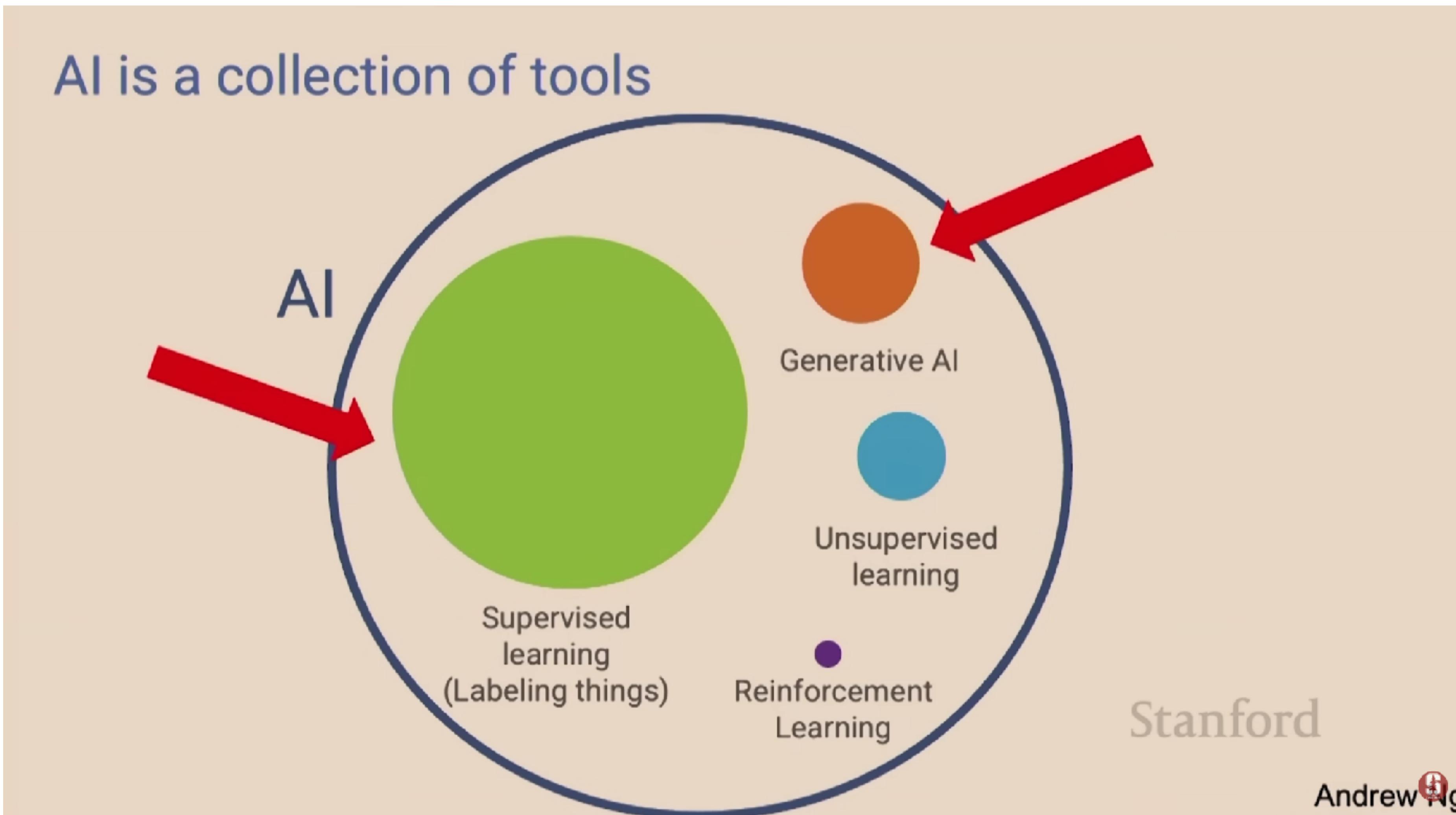
My favorite food is a bagel with cream cheese and lox.

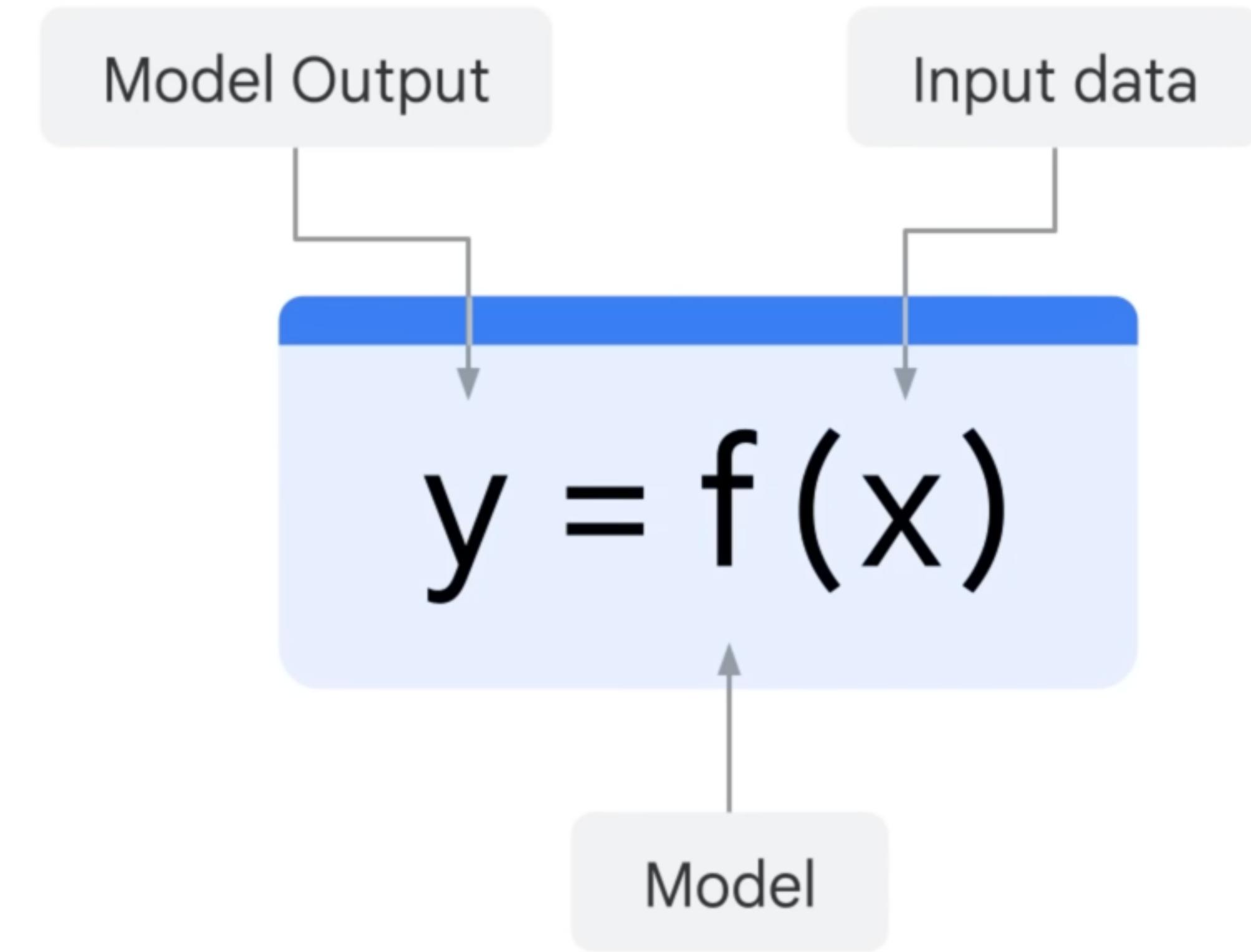
Input (A)	Output (B)
My favorite food is a	bagel
My favorite food is a bagel	with
My favorite food is a bagel with	cream

Stanford

Andrew Ng

AI 是一系列方法的集合





Not GenAI when y is a:

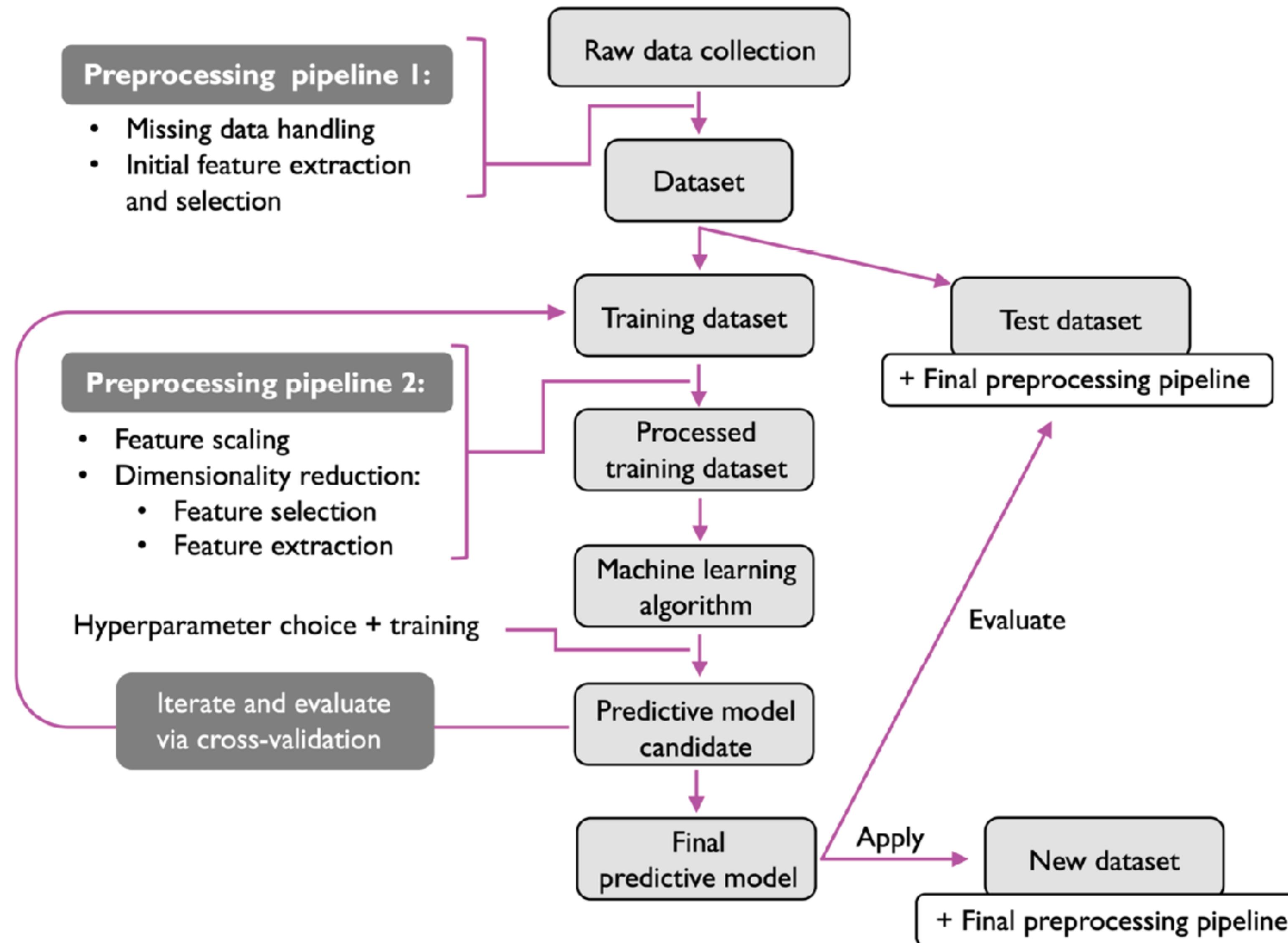
- Number
- Discrete
- Class
- Probability

Is GenAI when y is:

- Natural language
- Image
- Audio



機器學習系統建立流程圖

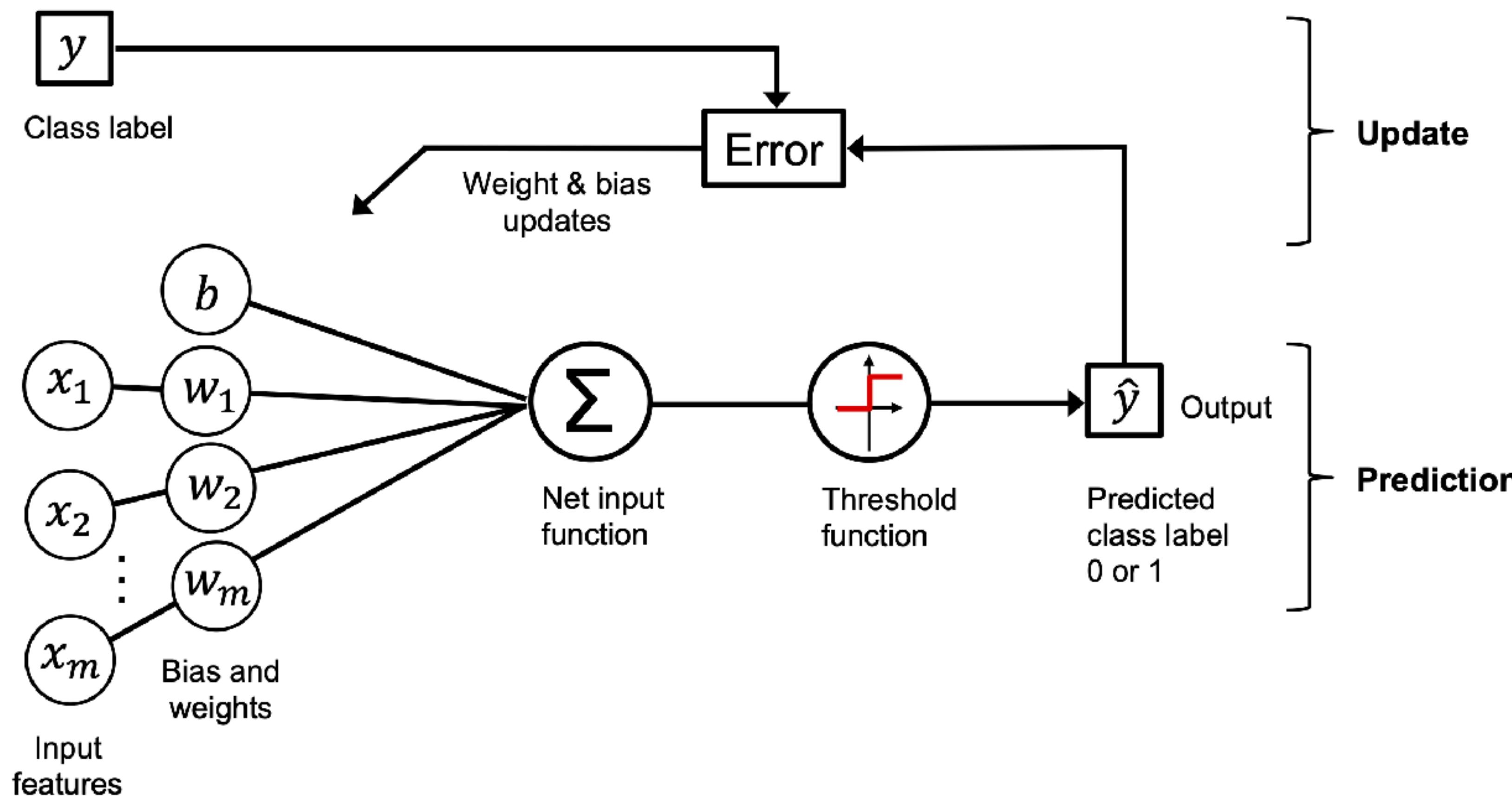


人工智慧的演算法

- 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)

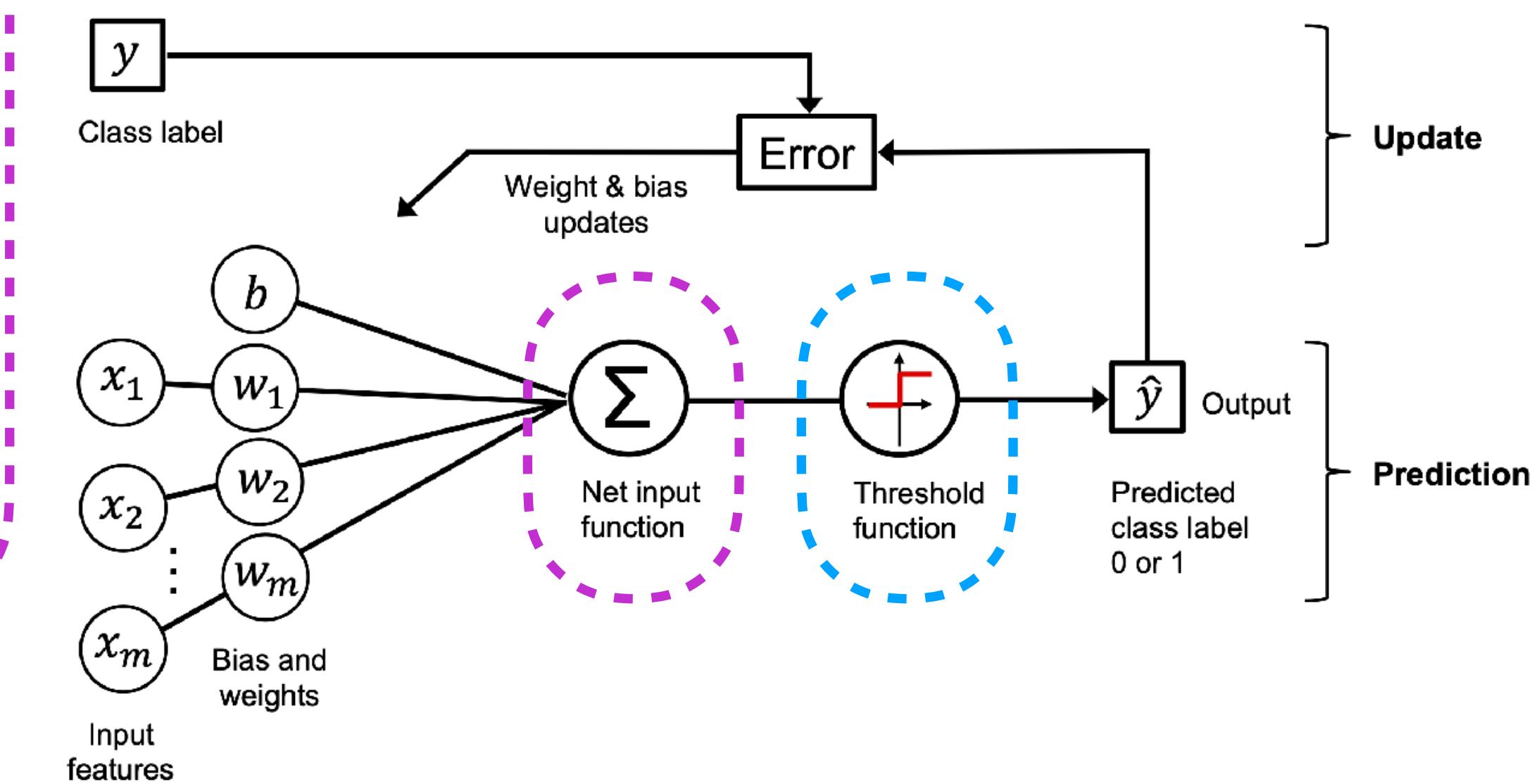


Perceptron

- 用於監督式學習二元分類，透過學習輸入特徵和權重的線性組合來決定輸出是0還是1
 - 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 (w_i) 和常數 (b) 為小的隨機值或零。
 2. 對於每個訓練樣本，進行以下操作：
 - 計算輸出：對訓練樣本的預測輸出。
 - 更新權重：如預測結果錯誤，則更新權重。
 3. 重複第二步，直到所有預測都正確，或者達到預設的迭代次數。
- 如果訓練資料是線性可分的，感知器保證能夠找到一組解決方案。
- 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。

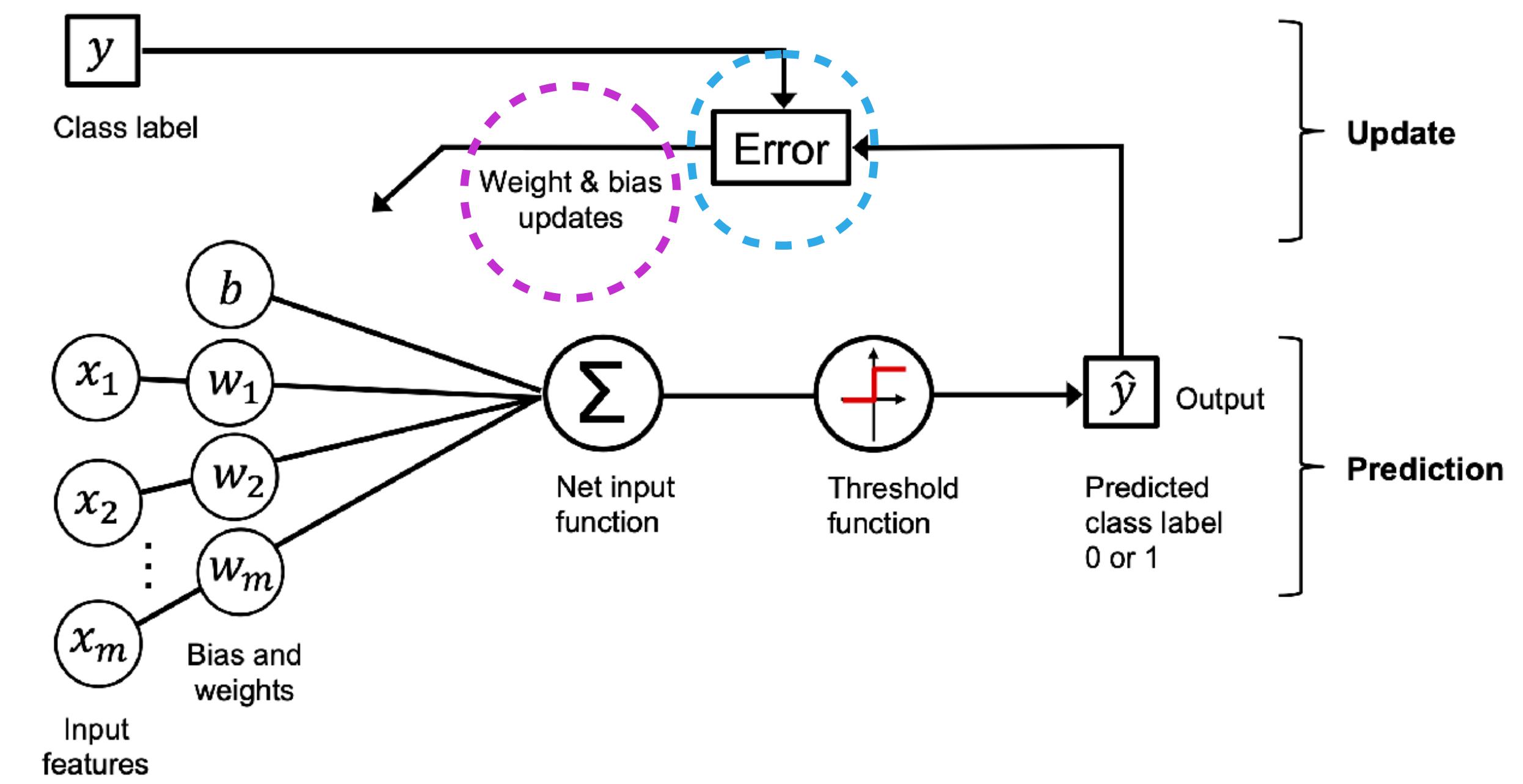
Perceptron

- $z = w_1x_1 + w_2x_2 + \dots + w_mx_m + b$
- $w = \begin{bmatrix} w_1 \\ \vdots \\ w_m \end{bmatrix}, x = \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix}$
- x_i 是第 i 個輸入特徵， w_i 是第 i 個輸入特徵的權重， b 為常數
- Activation function (激勵函數) $\sigma(z)$
- $\sigma(z) = \begin{cases} 1 & \text{if } z \geq \theta \\ 0 & \text{otherwise} \end{cases}$



Perceptron

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



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

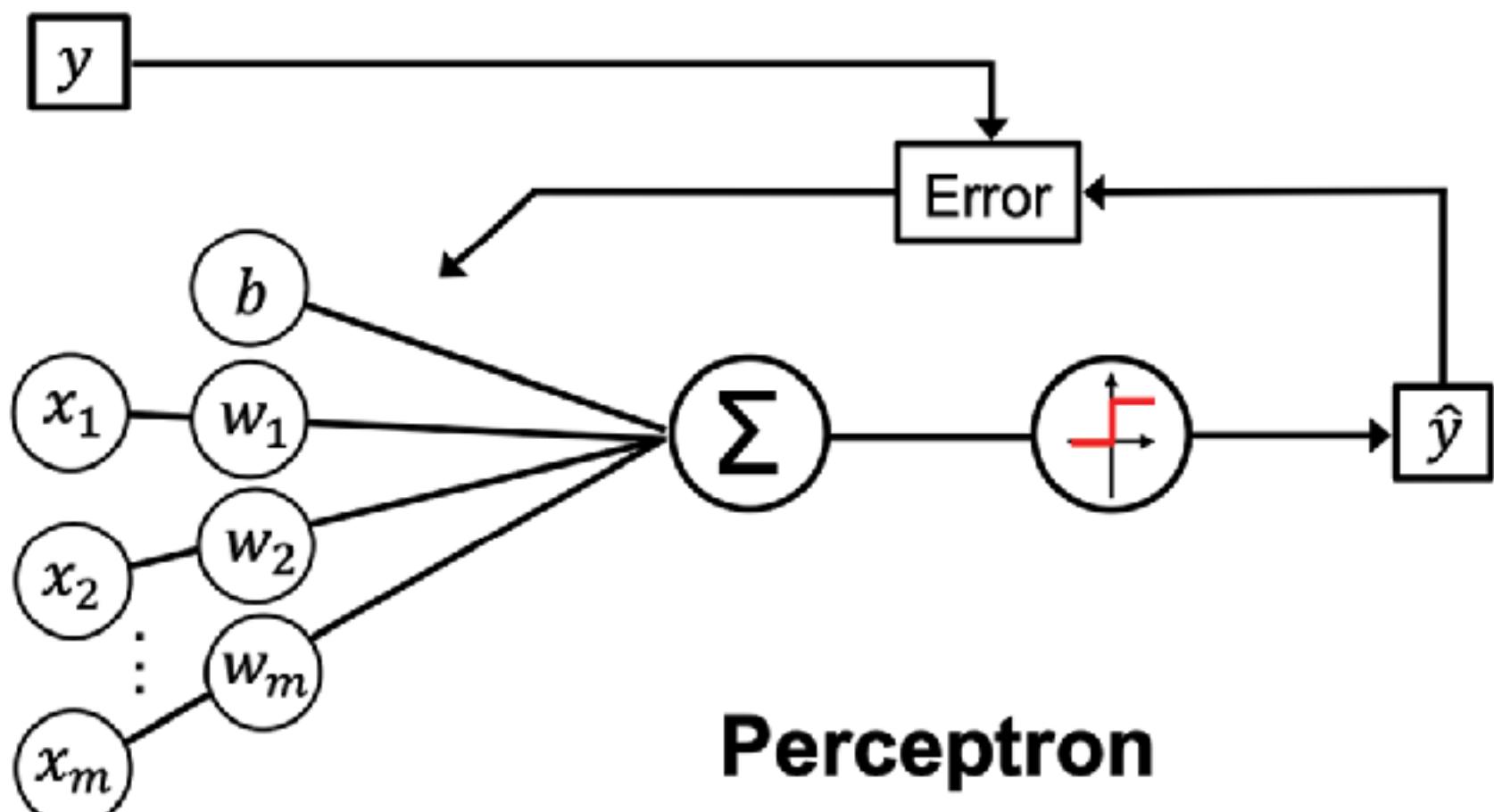
$$y^{(i)} = 1, \quad \hat{y}^{(i)} = 1, \quad \Delta w_j = \eta(1 - 1)x_i^{(i)} = 0, \quad \Delta b = \eta(1 - 1) = 0$$

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

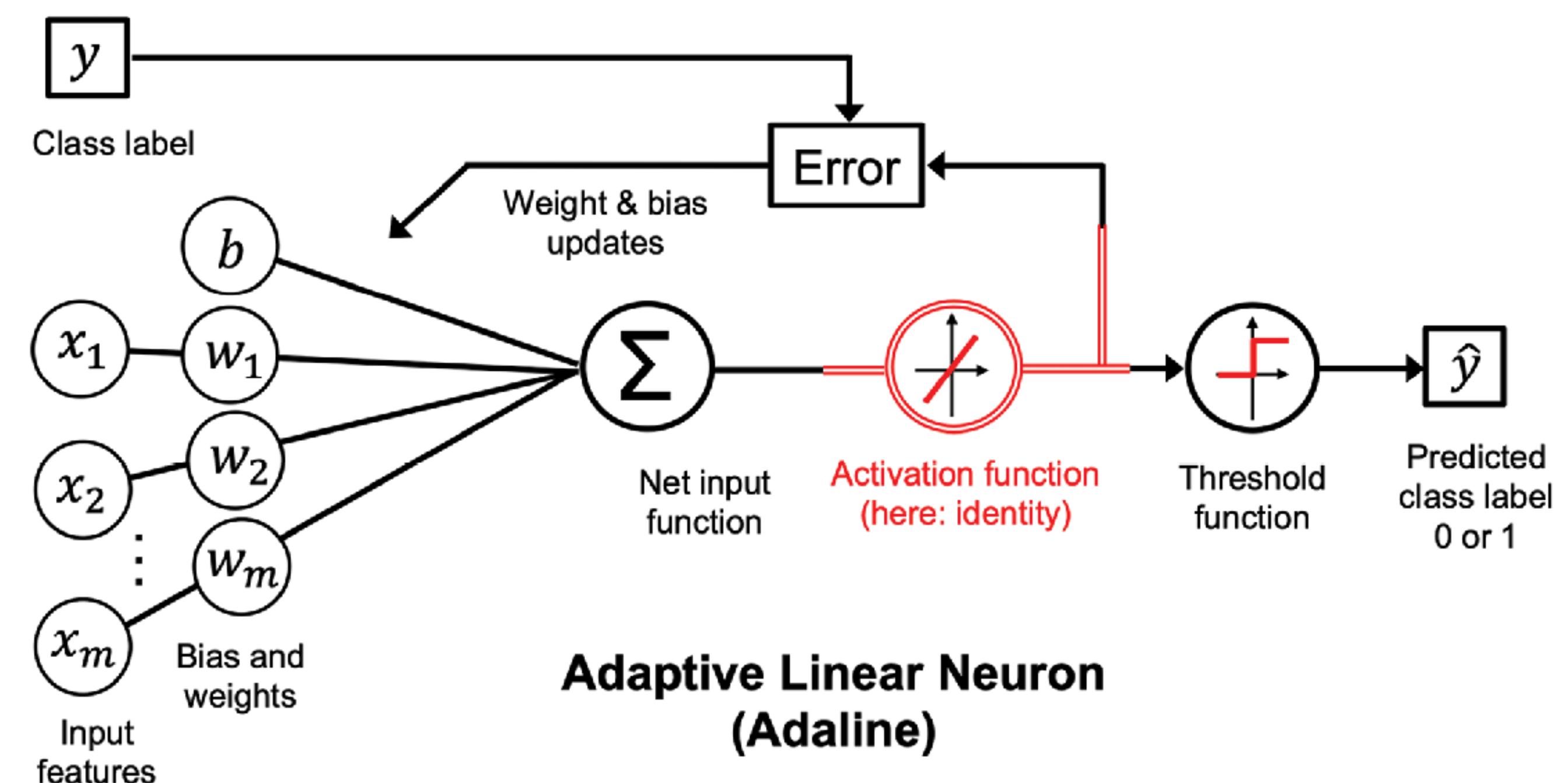
$$y^{(i)} = 0, \quad \hat{y}^{(i)} = 1, \quad \Delta w_j = \eta(0 - 1)x_j^{(i)} = -\eta x_j^{(i)}, \quad \Delta b = \eta(0 - 1) = -\eta$$

Adaptive Linear Neurons

Adaline VS Perceptron



Perceptron



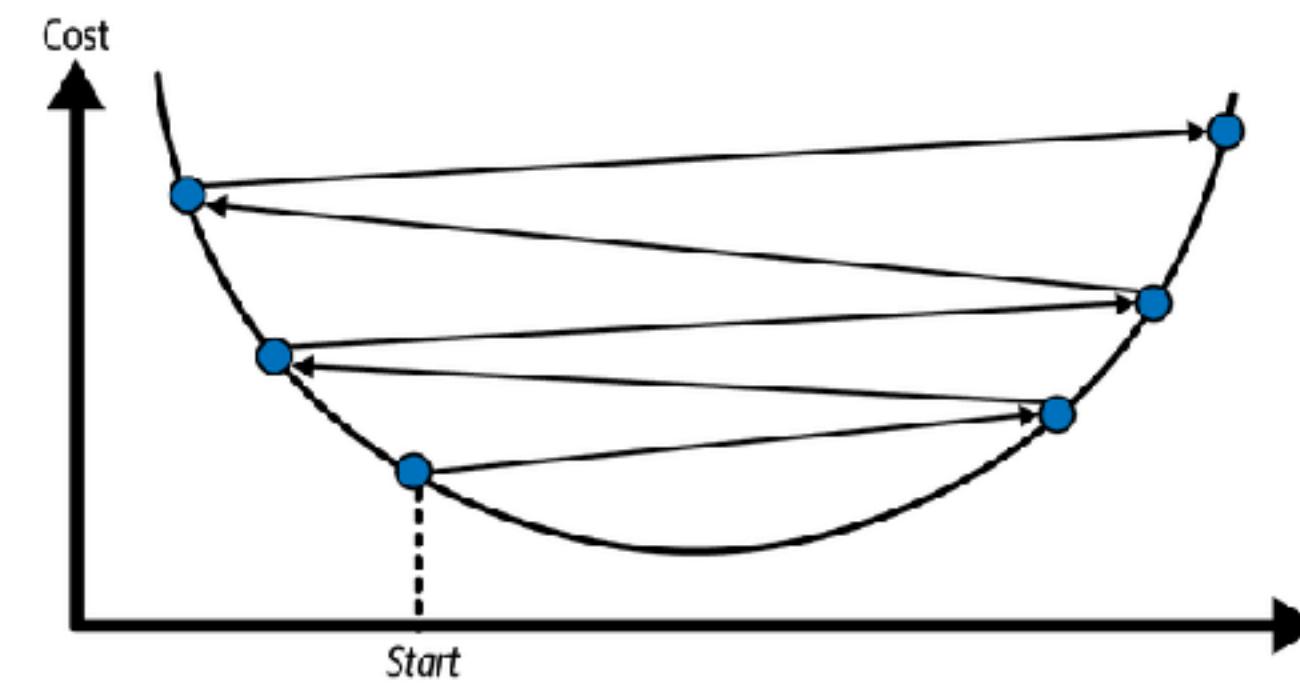
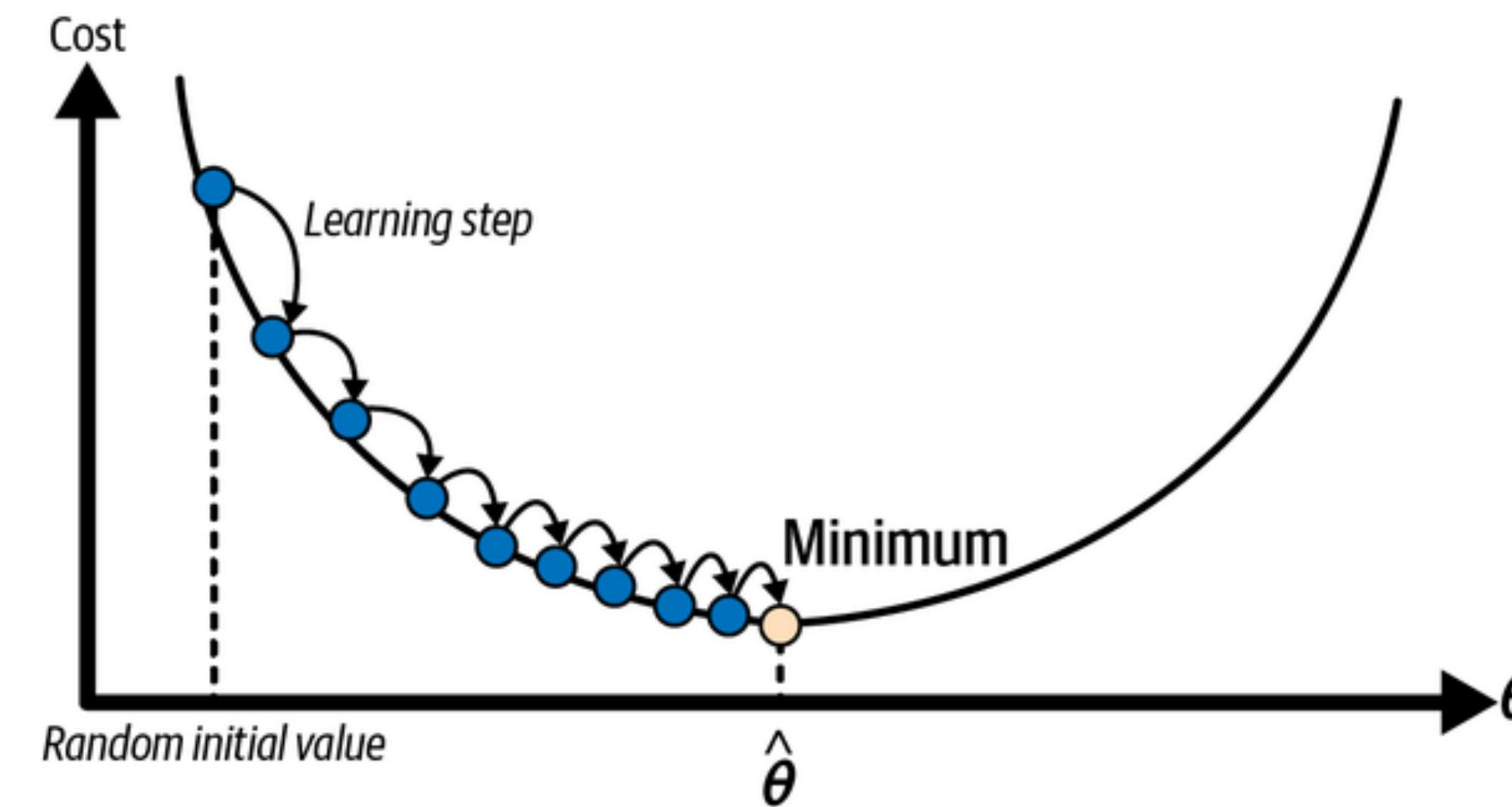
**Adaptive Linear Neuron
(Adaline)**

Adaptive Linear Neurons

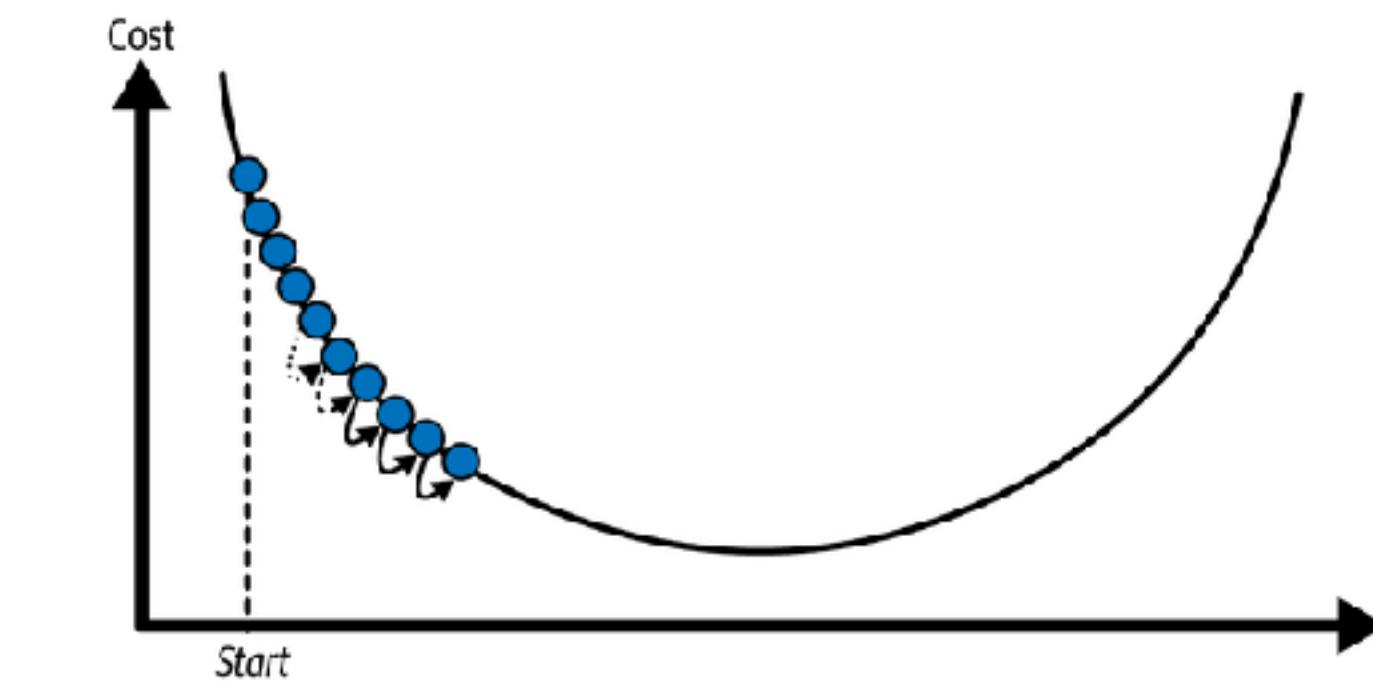
- 權重更新的方式的不同：
 - Perceptron: 權重的更新是基於實際輸出和預期輸出的差異。如果預測是正確的，則權重保持不變。如果預測是錯誤的，權重就會更新。
 - Adaline 模型中，權重的更新是基於損失函數 (loss function，或稱成本函數 cost function) 的梯度下降。
 - 不僅考慮是否錯誤，還考慮錯誤的程度。
 - 使用的是梯度下降 (gradient descent) 來最小化連續的損失函數。
 - 在 Adaline 中，目標最小化錯誤的損失函數 L ，可將其定義為 mean squared error(MSE) 這個函數的值越小，代表模型的預測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

$$L(\mathbf{w}, b) = \frac{1}{n} \sum_{i=1}^n \left(y^{(i)} - \sigma(z^{(i)}) \right)^2$$

Minimizing Loss Functions with Gradient Descent



when learning rate is too large



when learning rate is too small

Minimizing Loss Functions with Batch Gradient Descent

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

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

↓

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

when learning rate is too large

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

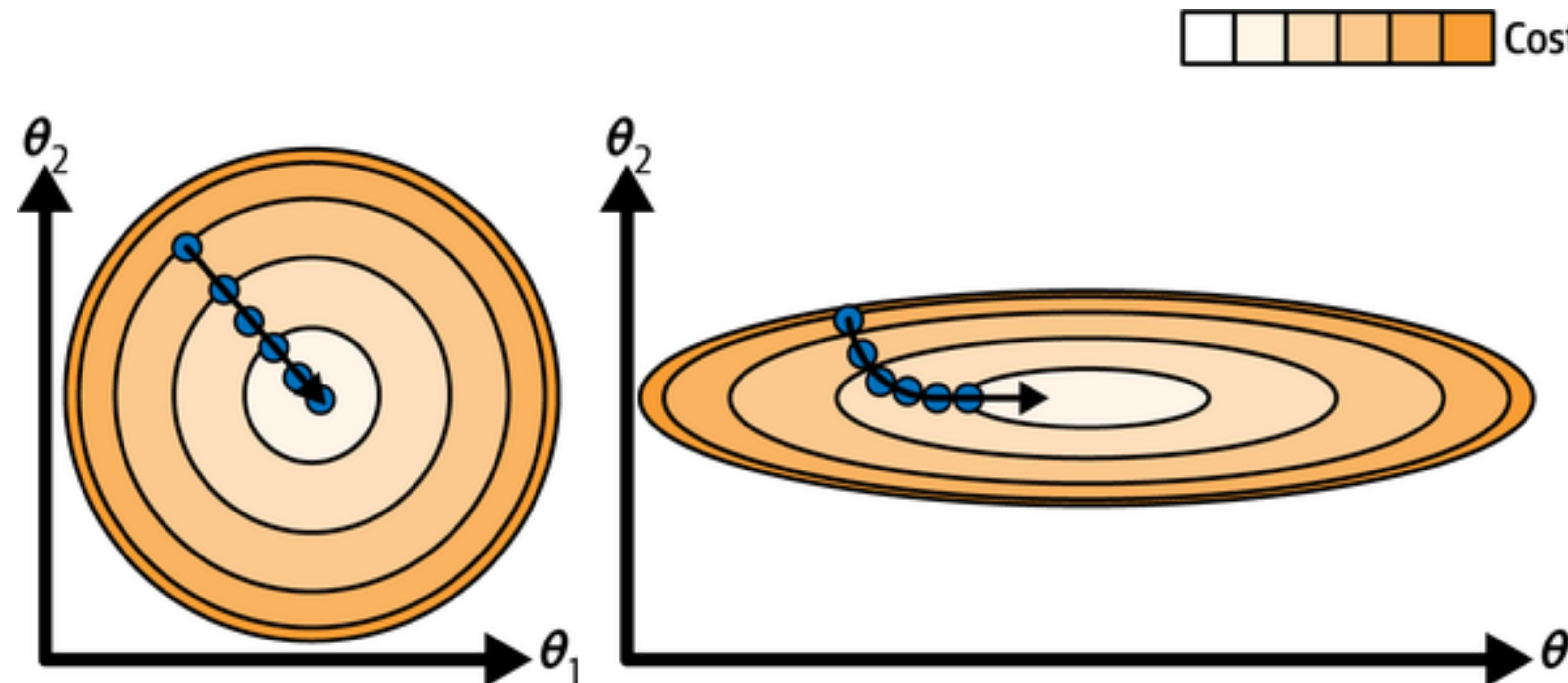
$$\frac{\partial L}{\partial b} = -\frac{2}{n} \sum_i (y^{(i)} - \sigma(z^{(i)}))$$

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

Improving Gradient Descent through Feature Scaling

- 特徵縮放 (Feature Scaling) 是調整資料的尺度，確保所有特徵都在相似的範圍。
 - 正規劃 (Normalization) 將特徵依照資料與最小值跟最大最小值差距關係，調整到 0 跟 1 之間。
 - 標準化 (Standardization) ，將特徵調整為平均為 0，標準差為 1 的標準常態分佈。
- 加快收斂速度
 - 特徵的範圍差異很大時，梯度下降會在不同特徵方向上以不同的速度進行收斂，會導致收斂速度變慢，甚至可能無法收斂。
- 防止特徵支配
 - 某些特徵的範圍遠大於其他特徵，這些特徵會在梯度計算中佔據主導地位，從而對模型的學習過程產生不公平的影響。
- 改善損失函數的形狀
 - 未經縮放的特徵會導致損失函數的曲率變高，這會使得梯度下降需要更多的步驟來達到最小值。

Improving Gradient Descent through Feature Scaling



如何選擇不同的演算法

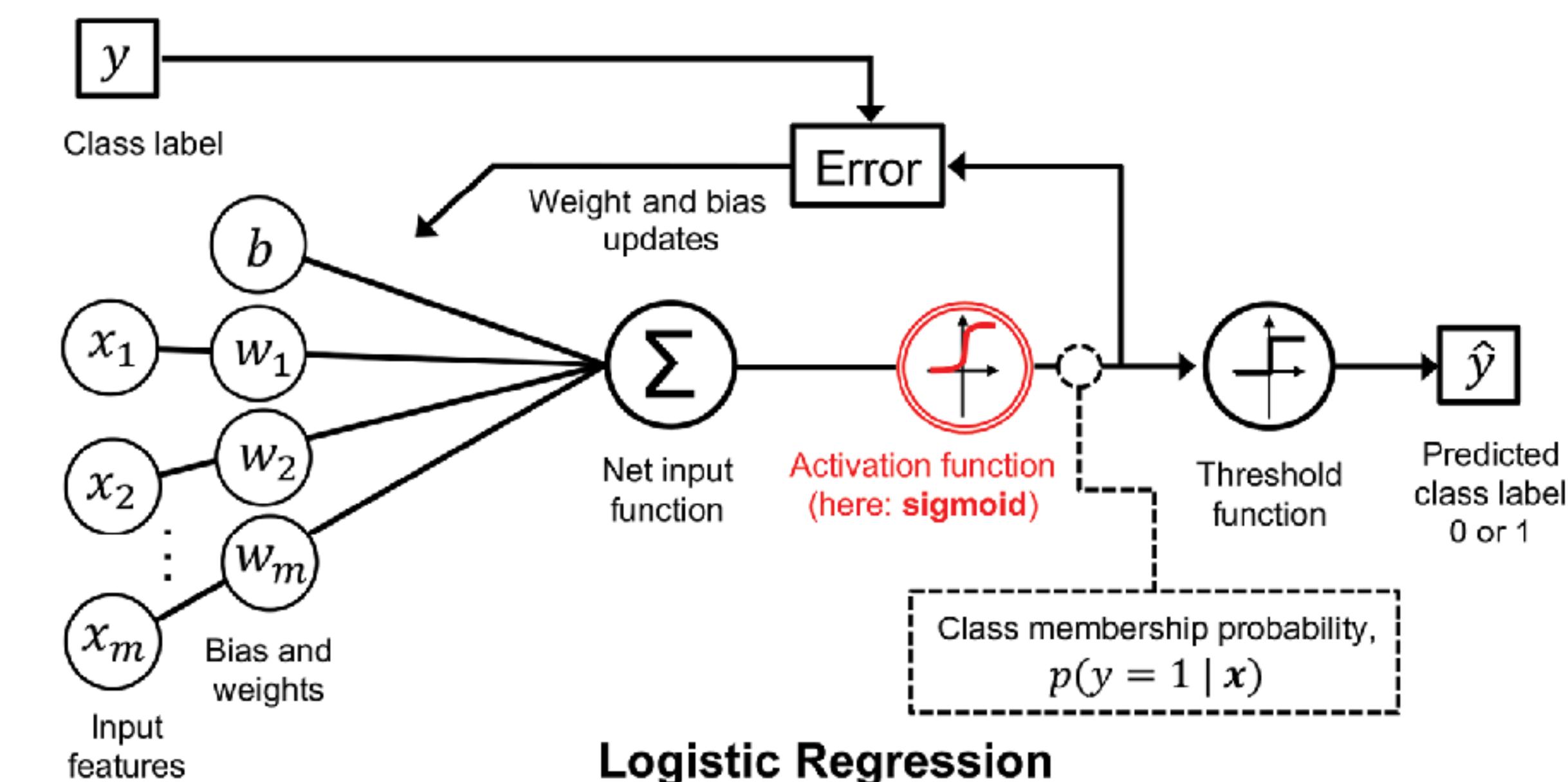
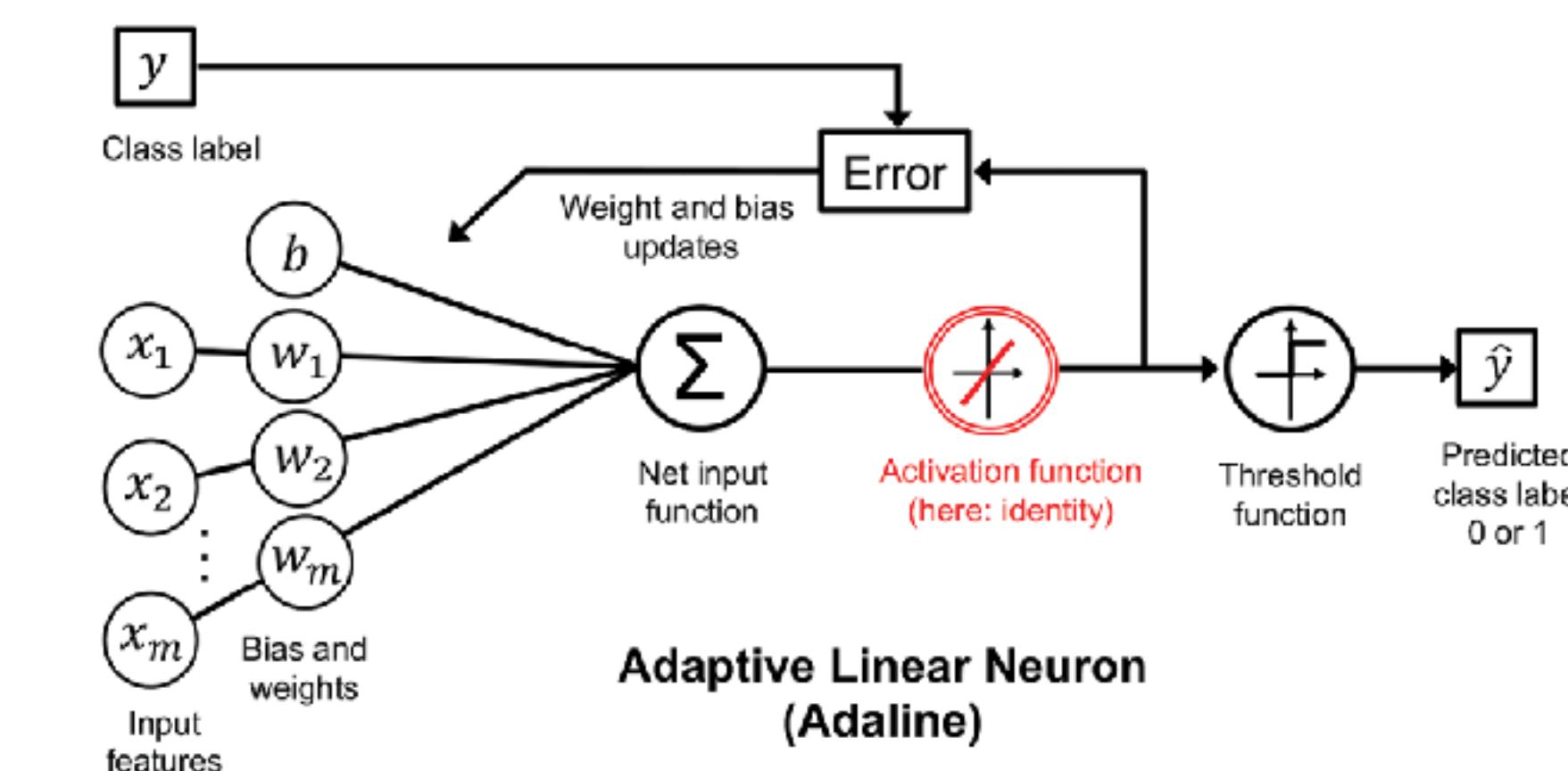
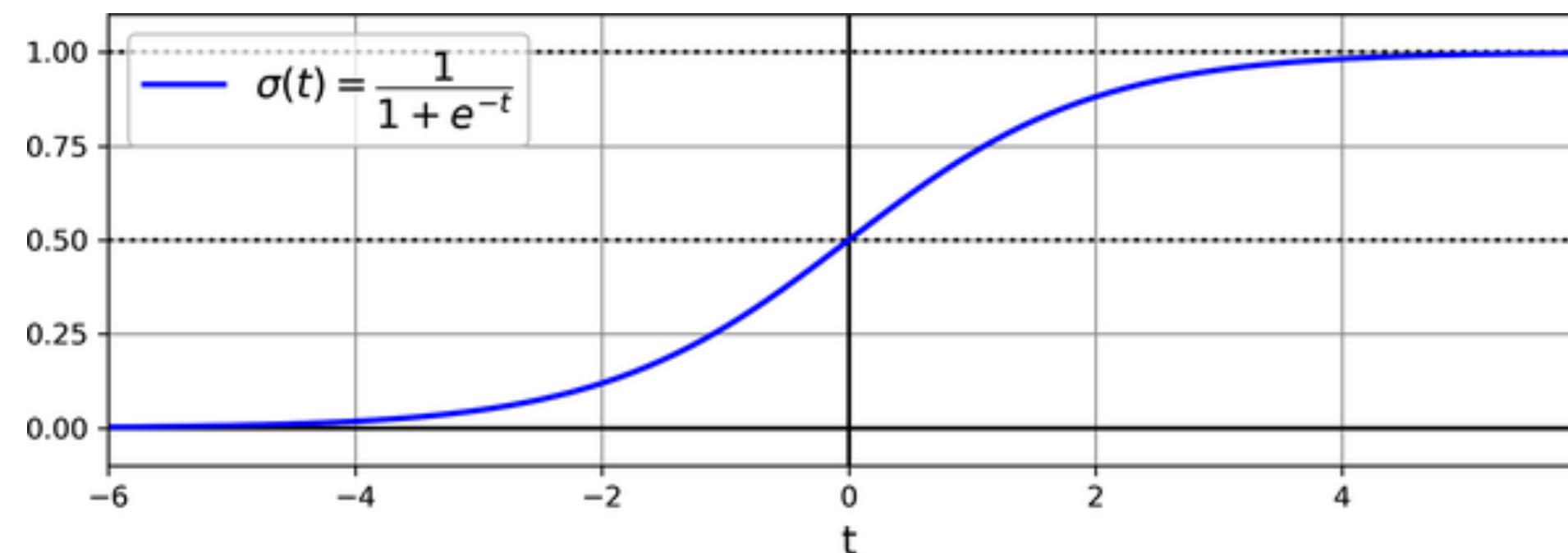
- 選擇特徵並收集帶有標籤的訓練樣本
- 選擇性能評估指標
- 選擇學習算法並訓練模型
- 評估模型的性能
- 調整算法的設定並調整模型

Logistic Regression

- 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。

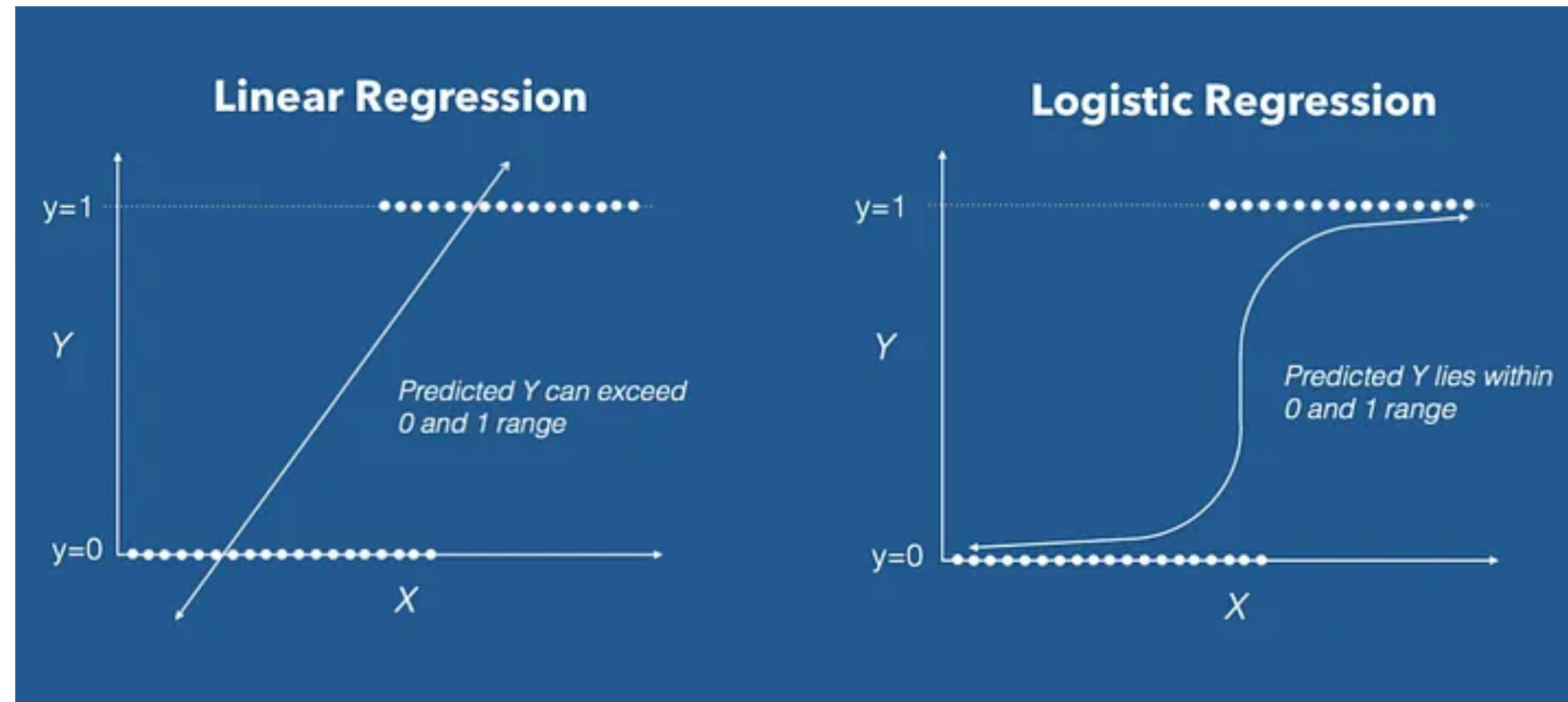
$$\sigma(z) = \frac{1}{1 + e^{-z}}, \quad z = w^T x + b$$

$$\hat{y} = \begin{cases} 1 & \text{if } \sigma(z) \geq 0.5 \\ 0 & \text{otherwise} \end{cases}$$



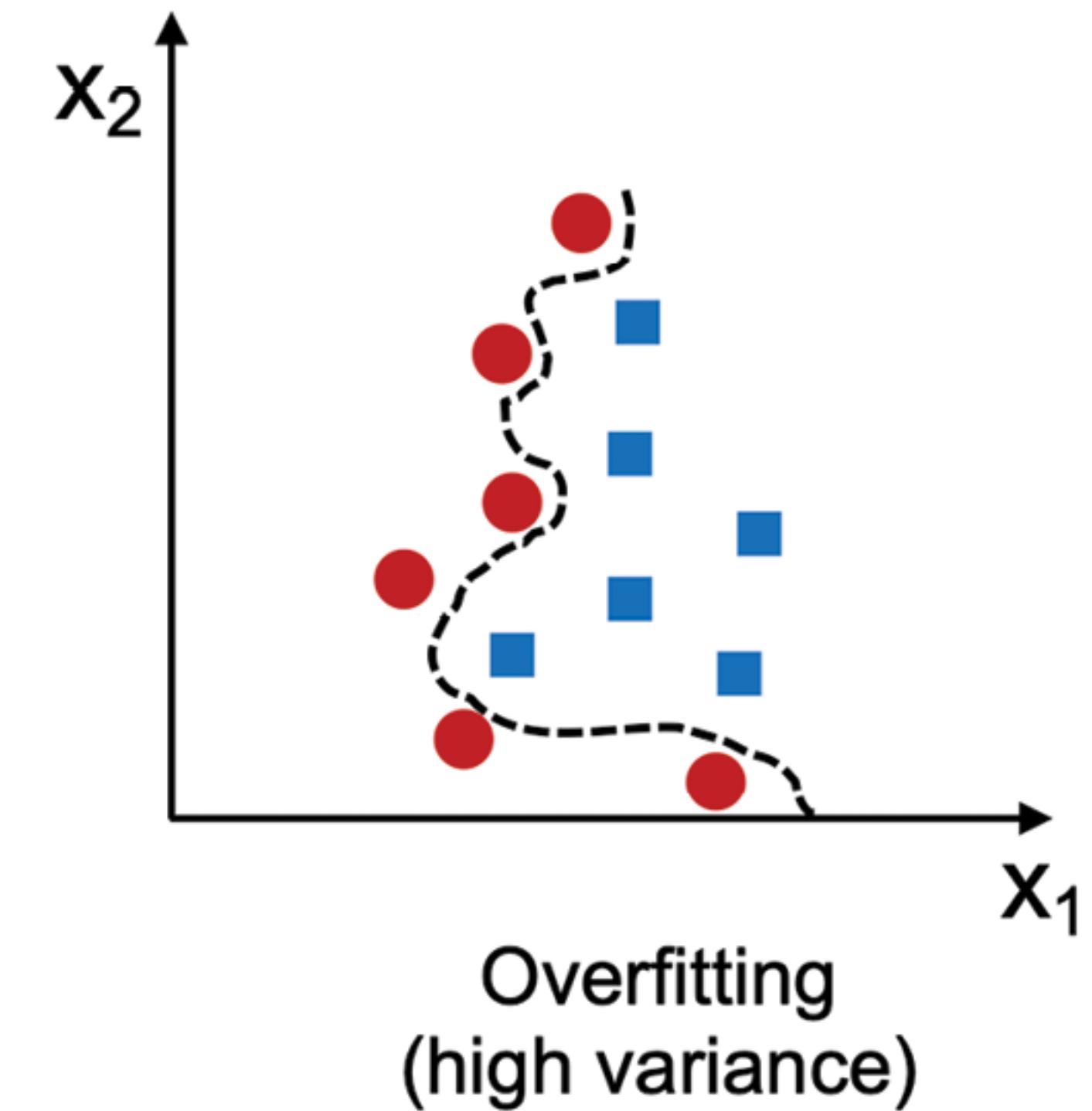
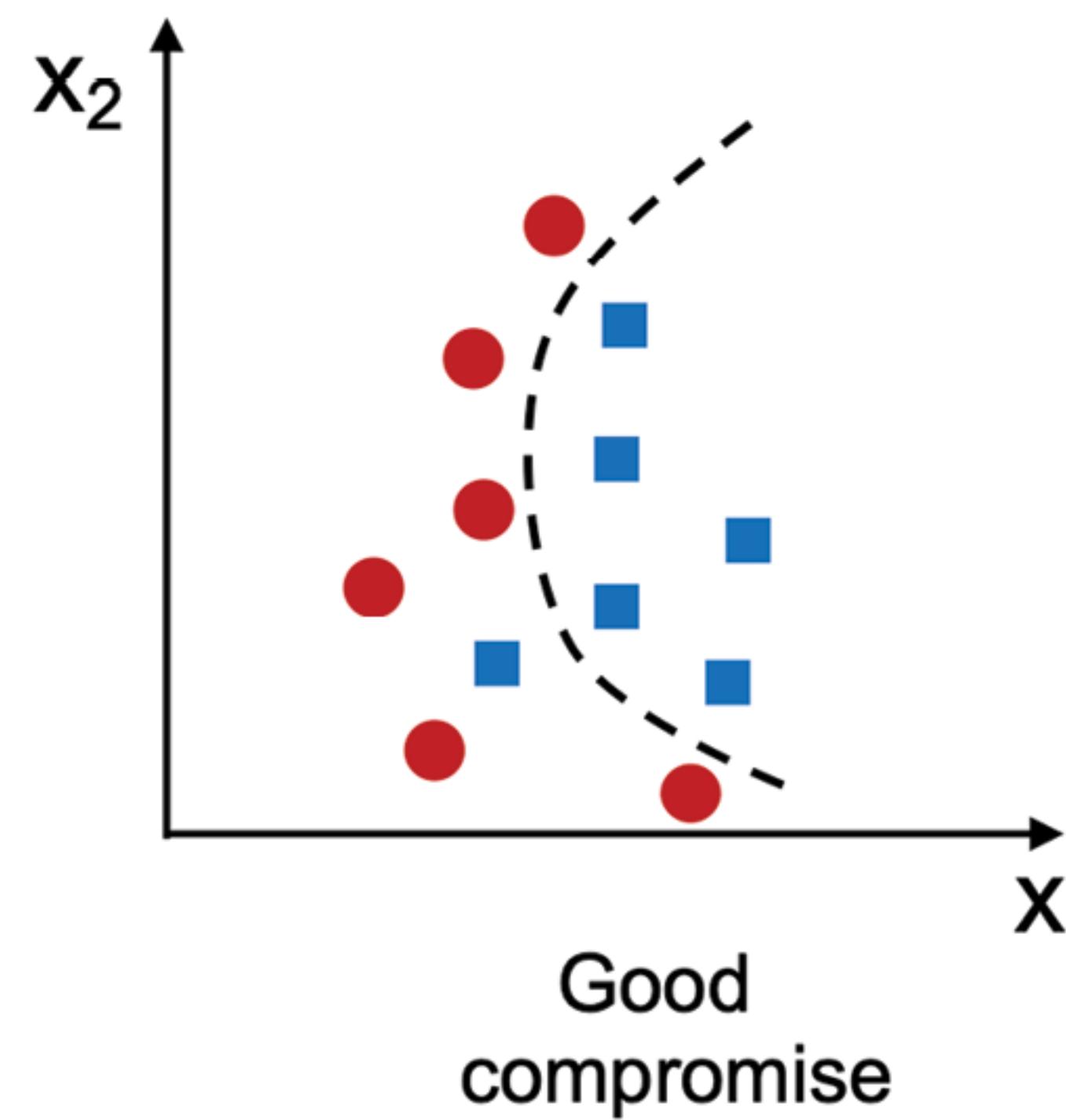
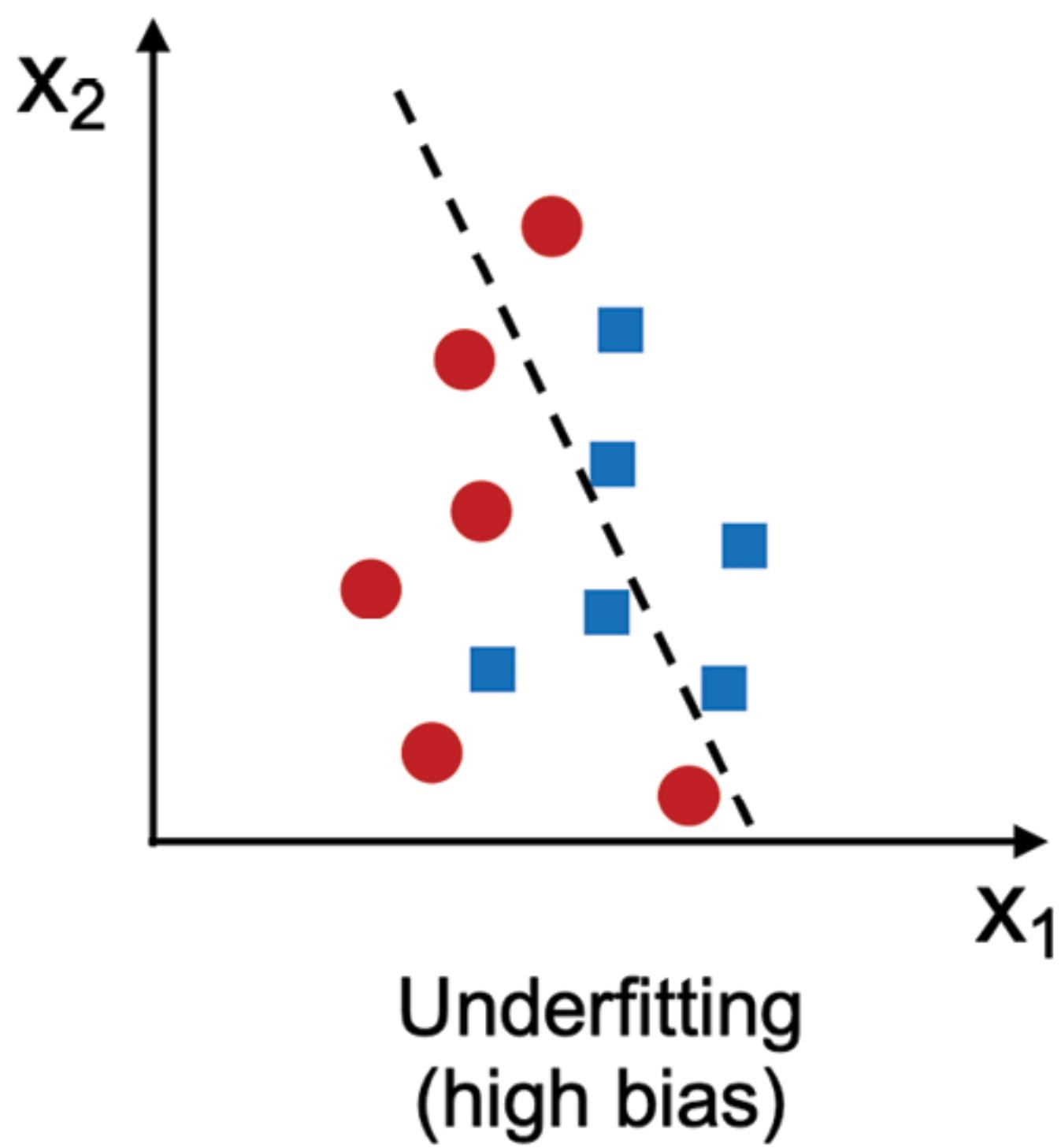
Logistic Regression

- 使用 $L(y, \hat{y}) = -[y \log(\hat{y}) + (1 - y)\log(1 - \hat{y})]$ 做為損失函數



Regularization (正規化)

- 防止過擬合



Regularization (正規化)

- L1 正規化 (Lasso) : 添加係數絕對值之和的懲罰項，可以產生稀疏解。

$$\bullet L(\beta) = \text{Loss}(\beta) + \lambda \sum_{i=1}^n |\beta_i|$$

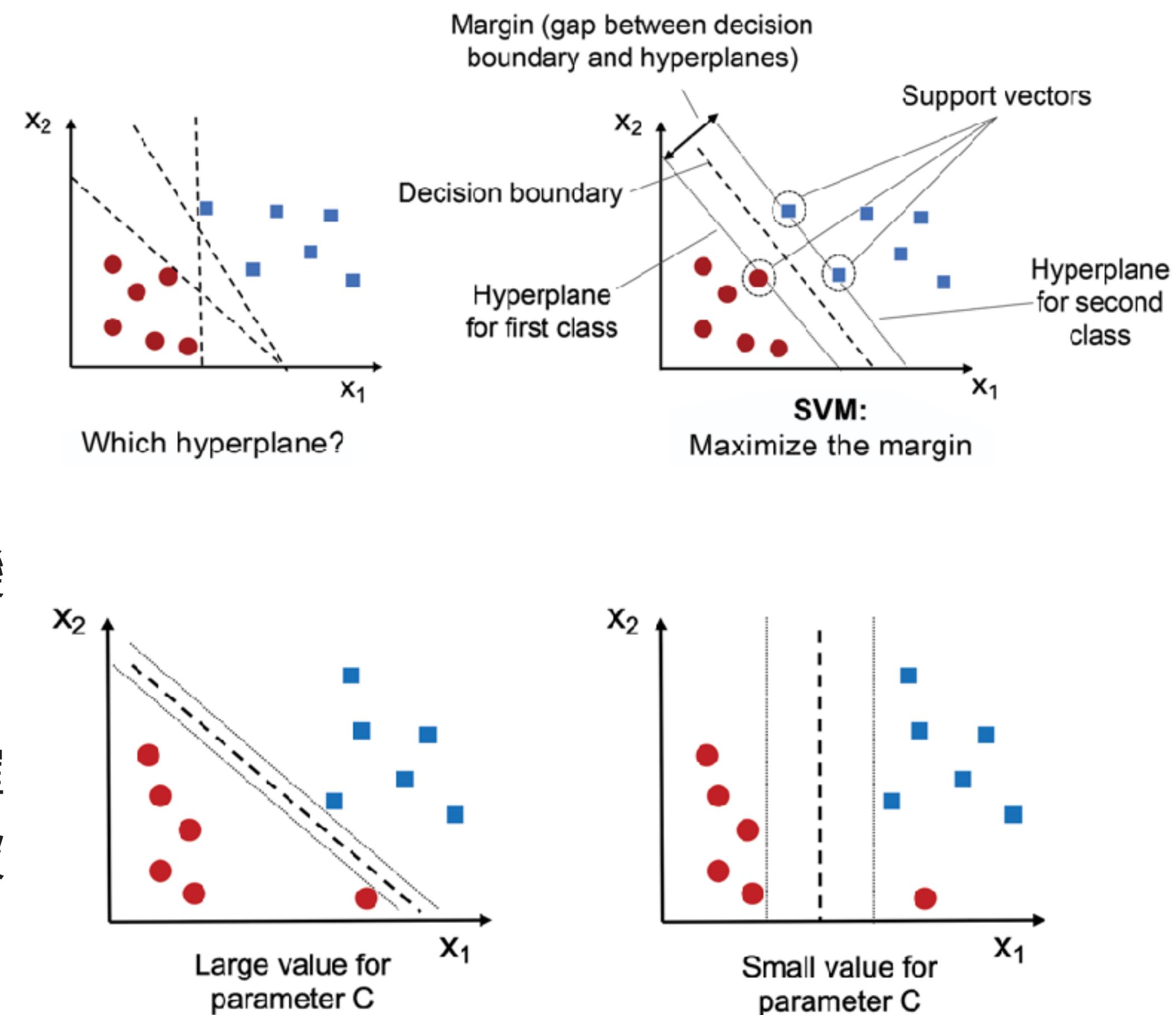
- 傾向於產生稀疏解，即將一些特徵的係數壓縮為零。
- 具有自動特徵選擇的能力。
- L2 正規化 (Ridge) : 添加係數平方和的懲罰項，可以平滑地縮小所有係數。

$$\bullet L(\beta) = \text{Loss}(\beta) + \lambda \sum_{i=1}^n \beta_i^2$$

- 傾向於平滑地縮小所有係數，而不是將它們設為零。
- 這意味著它保留了所有特徵，但減少了它們的影響。

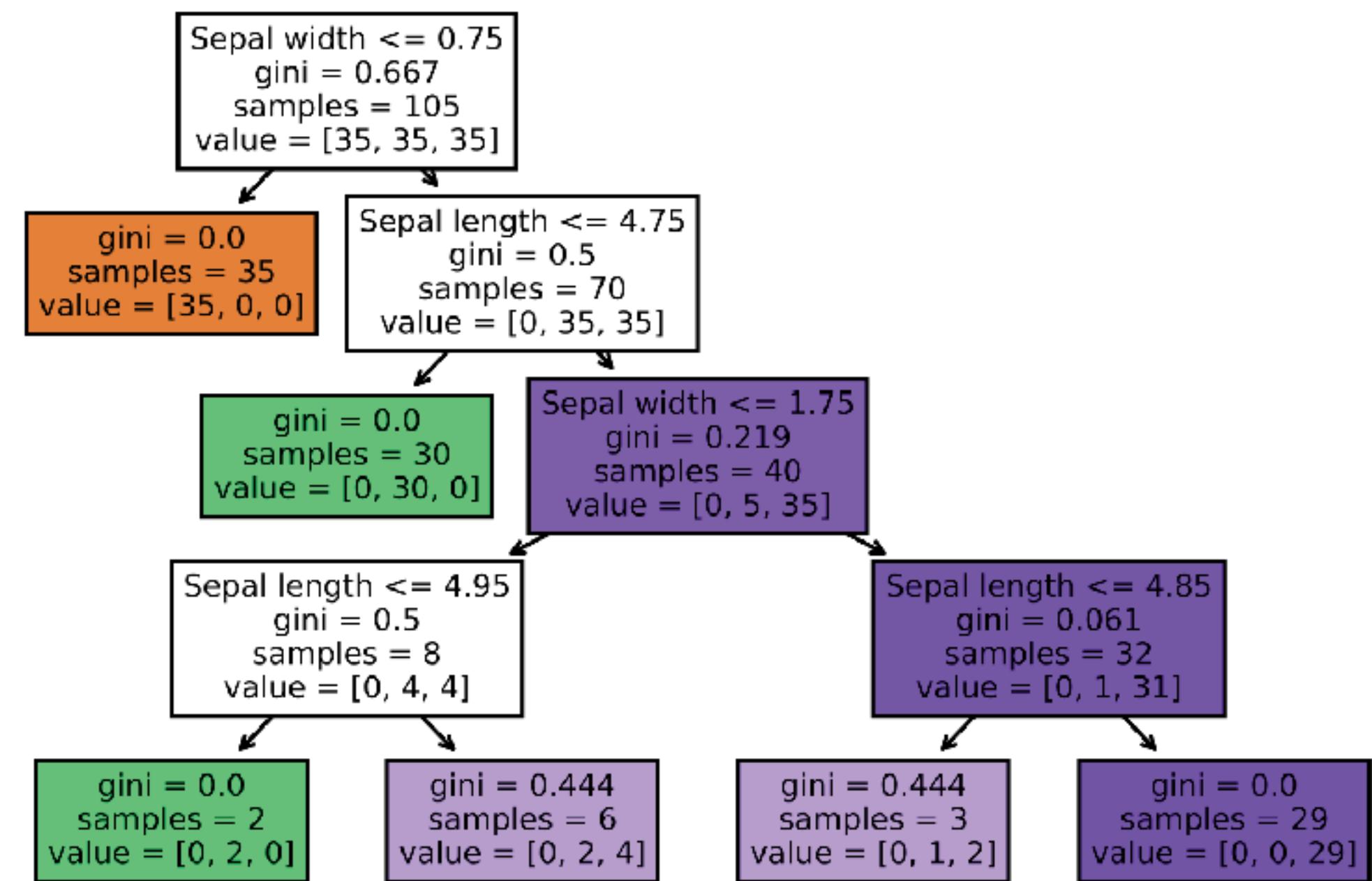
Support Vector Machines

- 最大邊際 (margin) 分類器：找到一個最好的超平面 (Hyperplane) ，將不同類別的數據點分隔開來。
- 這個超平面在高維空間中將數據點分成兩個類別，並且最大化兩類數據點之間的間隔 (Margin)
- 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，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：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。內部節點代表一個特徵或屬性，分支代表一個決策規則，而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）或基尼不純度（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：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，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。



決策樹的生成過程

- 選擇最佳分割特徵：根據某種評估標準（如信息增益、增益比率、基尼指數）選擇最能區分數據的特徵。
- 分割數據集：根據選定的特徵將數據集分割成子集。
- 遞歸地生成子樹：對每個子集重複上述步驟，直到滿足停止條件（如所有數據點屬於同一類別，或達到最大樹深）。
- 生成葉節點：當停止條件滿足時，生成葉節點並賦予類別標籤或預測值。

Random Forests

- 隨機森林屬於整體學習（Ensemble Learning）的範疇，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。

Random Forests

- 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)

- 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。

K-Nearest Neighbors (KNN)

- 距離度量：KNN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，根據數據特性來決定。

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

- Manhattan distance
- 特性縮放：基於距離來進行學習的，如果特徵的尺度（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。

Confusion Matrix (混淆矩陣)

- True Positive (TP) : 模型正確預測為正類的數量。
- True Negative (TN) : 模型正確預測為負類的數量。
- False Positive (FP) : 模型錯誤地將負類預測為正類的數量 (Type I Error)。
- False Negative (FN) : 模型錯誤地將正類預測為負類的數量 (Type II Error)。

		True Class	
		Positive	Negative
Predicted Class	Positive	TP	FP
	Negative	FN	TN

Image: DataCamp

Confusion Matrix (混淆矩陣)

- 準確率 (Accuracy) : 表示模型預測正確的比例。

$$\text{• Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

- 精確率 (Precision) : 表示模型預測為正類的樣本中實際為正類的比例。

$$\text{• Precision} = \frac{TP}{TP + FP}$$

- 召回率 (Recall) or 敏感度 (Sensitivity) : 表示實際為正類的樣本中被正確預測為正類的比例。

$$\text{• Recall} = \frac{TP}{TP + FN}$$

- F1-Score : 是精確率和召回率的調和平均，綜合考慮了兩者的平衡。

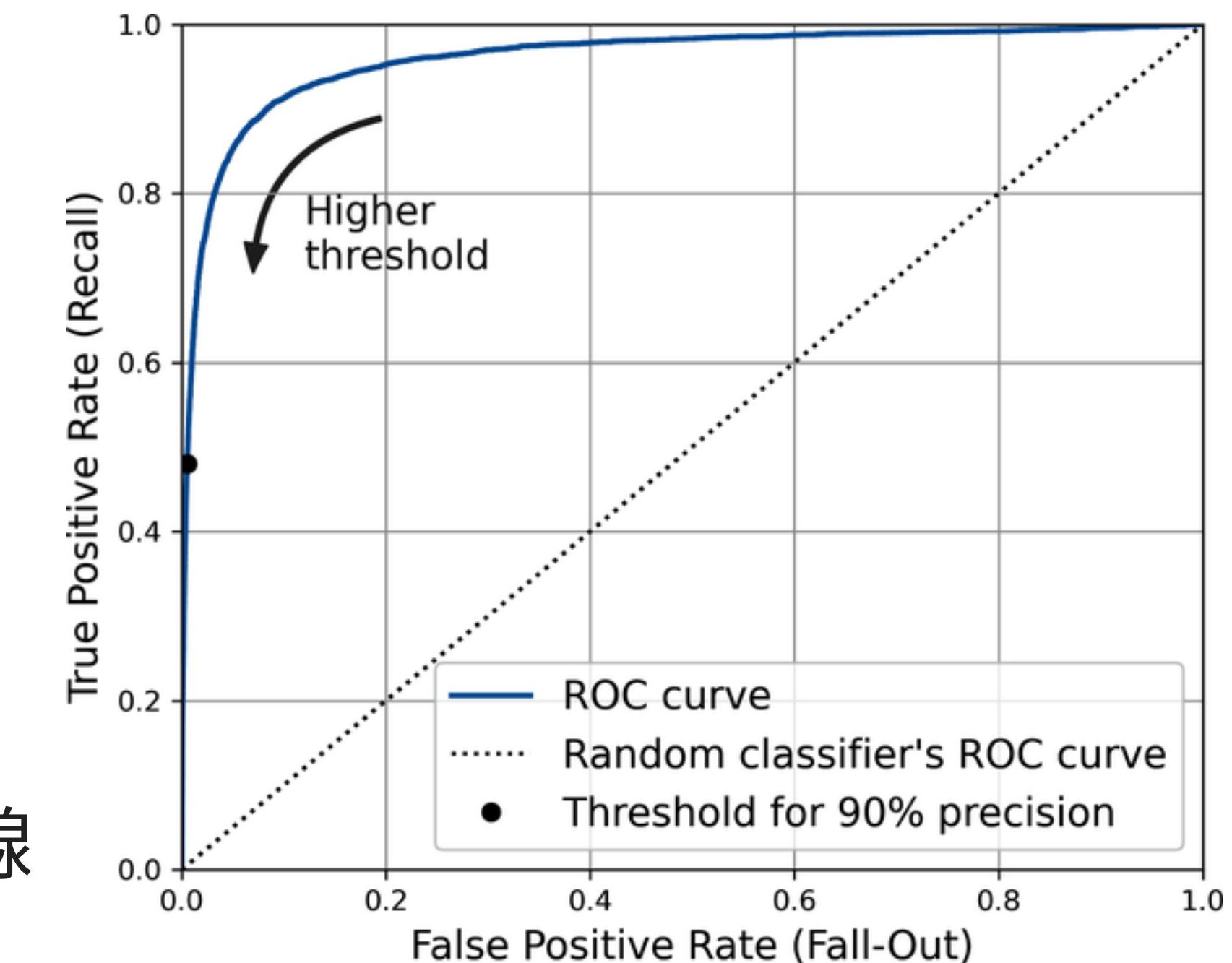
$$\text{• F1-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

- 特異性 (Specificity) : 表示實際為負類的樣本中被正確預測為負類的比例。

$$\text{• Specificity} = \frac{TN}{TN + FP}$$

ROC (Receiver Operating Characteristic)

- ROC曲線是以假陽性率 (False Positive Rate, FPR) 為橫軸，真陽性率 (True Positive Rate, TPR) 為縱軸的曲線圖。
- 對每個可能的分類閾值：
 - 計算TPR和FPR
 - 在圖上繪製對應的點
- 連接所有點，形成ROC曲線
- 完美分類器的ROC曲線會通過點 $(0,1)$ ，即 $FPR=0$ ， $TPR=1$
- 隨機猜測的分類器的ROC曲線是一條從 $(0,0)$ 到 $(1,1)$ 的對角線
- 實際的分類器的ROC曲線通常位於隨機猜測線之上



ROC (Receiver Operating Characteristic)

- 優點
 - 不受樣本不平衡影響：ROC曲線對正負樣本比例不敏感
 - 直觀比較不同模型：可以在同一圖上繪製多個模型的ROC曲線進行比較
 - 闕值無關：不需要預先設定分類闕值
- 應用
 - 模型選擇：比較不同模型的性能
 - 闕值選擇：根據特定需求（如平衡TPR和FPR）選擇最佳闕值
 - 異常檢測：評估異常檢測算法的性能
 - 醫學診斷：評估診斷測試的準確性

AUC (Area Under the Curve)

- AUC是ROC曲線下的面積，用於量化模型的整體性能：
 - $AUC = 1$ ：完美分類器
 - $0.5 < AUC < 1$ ：優於隨機猜測
 - $AUC = 0.5$ ：等同於隨機猜測
 - $AUC < 0.5$ ：劣於隨機猜測

Natural Language Processing , NLP

- 自然語言處理（Natural Language Processing, NLP）是一門讓電腦能夠理解、解釋、生成和操作人類自然語言的學科。這項技術的核心目標是使機器能夠像人類一樣處理語言，從而實現更自然的人機互動。
- 語言模型（language modeling）：語言模型試圖對語言（字詞的序列）的機率分布進行建模。
- 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。

Natural Language Processing , NLP

- 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。

語言模型

- 語言模型 (Language Models) 是用於預測語言序列機率的模型。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。
 3. Neural Network Language Models:
 - A. Recurrent Neural Networks (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): 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): 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**: 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整體訊息。

大型語言模型

- 大型語言模型（Large Language Models, LLMs）為生成式AI的一種主要的應用。利用大量的文本進行預訓練，能夠理解語法語意，進而讀寫語言。可用於文本生成、翻譯、問答系統等。
- 可透過微調（Fine-tuning）、檢索增強生成（RAG）等方式改善回答品質
 - 微調：透過給予一定數量問題與答案的集合，改變LLMs的理解
 - 檢索增強生成（RAG）：針對提供資料產生答案
- 透過提示（prompt）進行互動

LLMs is More Likely Reasoning Engines

- Search Engines
 - 擴展收尋相關內容
 - 不會進一步優化收尋內容
 - 不瞭解收尋本身
- Reasoning Engines
 - 瞭解人類語言
 - 提供直接相關的回覆
 - 瞭解內容

LLMs NLP

- 智能客服
 - RAG or Fine-Tuning，哪一個比較好
 - 如何 Fine-Tuning
- App 評價自動回覆
 - RAG
- 內部檢索系統
 - Agentic design

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。

BERT

- 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，但是透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。

GPT

- 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。

OpenAI API & ChatGPT Models

GPT-4o

最快又實惠的版本

- 文字影像輸入
- 文字輸出
- 128K tokens len
- Input: \$5
- Output: \$15

GPT-4 Turbo

過往最聰明的版本

- 文字影像輸入
- 文字輸出
- 128K tokens len
- Input: \$10
- Output: \$30

GPT-3.5 Turbo

簡單工作，快又最便宜的版本

- 文字影像輸入
- 文字輸出
- 16K tokens len
- Input: \$0.5
- Output: \$1.5

*價格皆為每百萬個 tokens

提示工程的原則

- 了解你的目標：定義目標，並瞭解透過提示取得資訊、創意還是解決問題？
- 保持清晰簡潔：避免過於複雜或模糊的提示。清晰的提示能有更準確的回應。
- 重視上下文：提供足夠的背景資訊以便 LLM 理解情境，但避免不必要的資訊。
- 嘗試和迭代：根據你得到的回應，嘗試不斷的改善提示。迭代是找到最有效措辭的關鍵。
- 考慮你的受眾：根據會與 AI 回應互動或受眾來調整你的提示。
- 評估和調整：不斷評估提示的有效性，並準備隨時進行調整。

Prompts Engineering Principles

- 兩個prompt的原則
 - 寫出清楚明確的指示
 - 紿模型"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，e.g., JSON, HTML
 - 要求模型檢查條件是否滿足
- 紿模型思考的時間
 - 指定完成任務所需的步驟
 - 指定回答格式
 - 指示模型製定出自己的解決方案，在模型匆忙下結論之前

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。

金融資訊爬蟲

- 社群情緒
 - PTT stock crawler
 - 盤中推文
 - 一般文章
- 個股重大訊息
 - TWSE 公開資訊觀測站

DEMO

1. 請分析以下股市討論的情緒，並以「使用者的角度」去「仔細」分析且必須回答該文情緒分析的結果是 "positive"、"negative" 或 "neutral"。
2. Your answer is a single word, either "positive" or "negative" or "neutral".
如果討論中沒有提及股票代碼或代號，請辨認情緒為 "neutral"。
否則，請將該討論指涉的股票代碼或代號列在答案中。如果沒有提及股票代碼或代號，請將答案設為空字串 ""，並儘量分析文章隱含的情感。
3. Provide above answers in just JSON format which could convert to Pandas DataFrame directly with keys for the series chats separated by angle brackets, respectively: Sentiment, StockTarget.
Follow the format:

```
{  
  {"Sentiment": ..., "StockTarget": ...}  
}
```

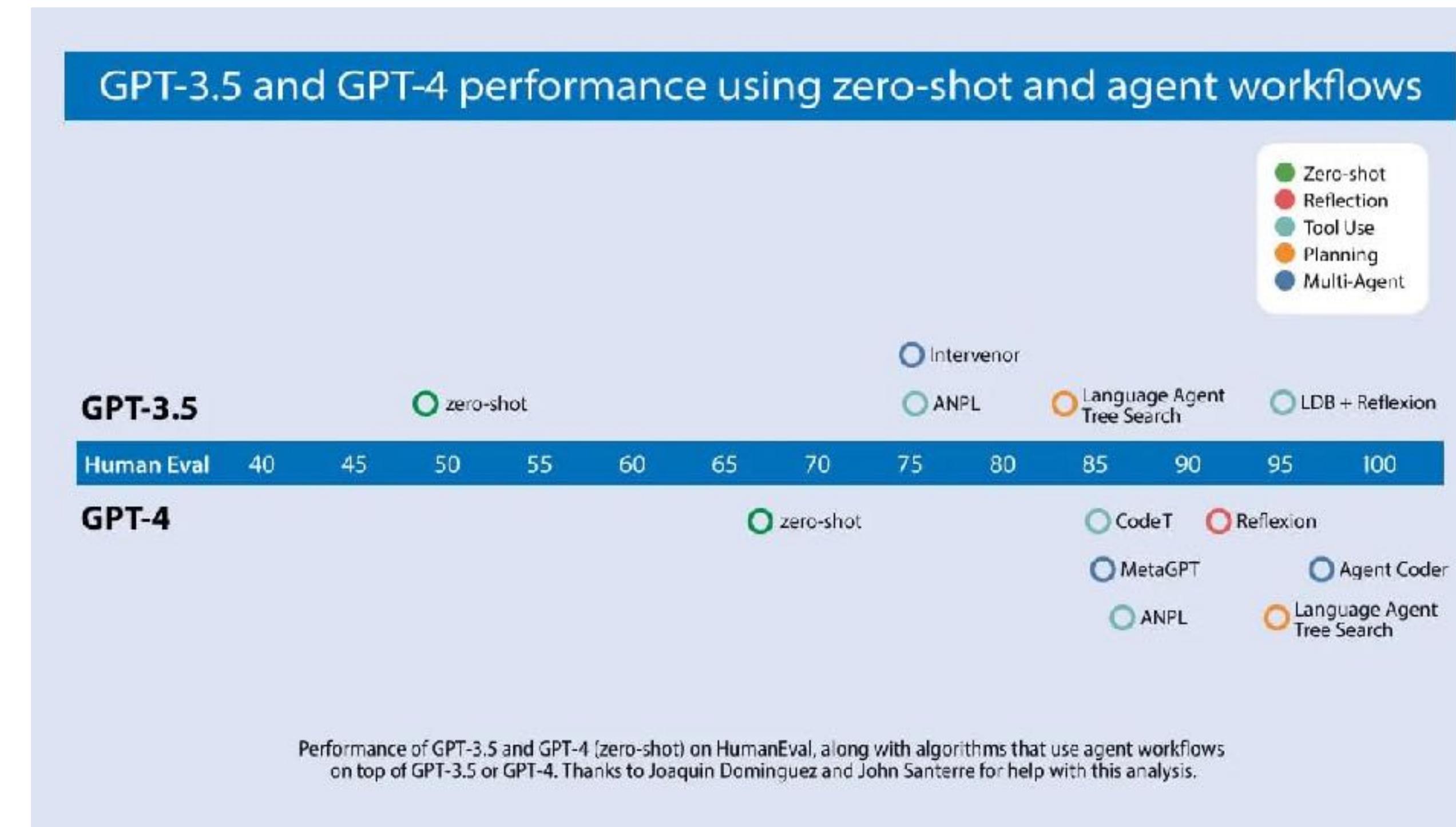
Chat text: {stock_market_chat}

請注意，請基於「文本的整體意思和情感」來回答，而不僅僅是依賴特定詞語。

Please note, it is essential to adhere to the rules specified by the JSON formats and values.

Agentic Patterns Prompting

- Reflection
 - LLM 自己審視自己回覆的結果並改善
- Tool Use
 - 讓 LLM 擁有像是網路收尋、程式執行或其他可以取得資料、資料分析或是採取行動
- Planning
 - LLM 採取更細部的步驟，有計畫的一步一步去達成目標
- Multi-agent
 - 許多的 agent 一起合作，互相討論或是互相辯論後產生出結果



Building an LLM application



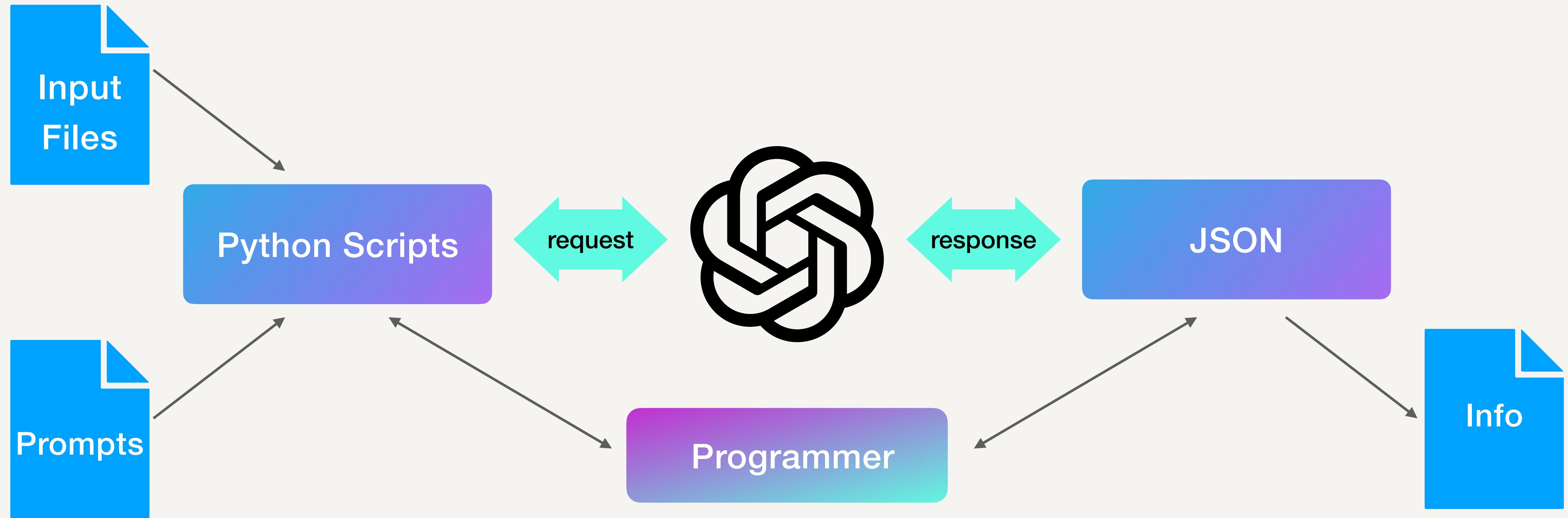
Building an LLM application

- 定義問題
 - 單一且夠大的問題
- 模型選擇
 - 商用 licensed
 - 模型大小
 - 模型效能

Building an LLM application

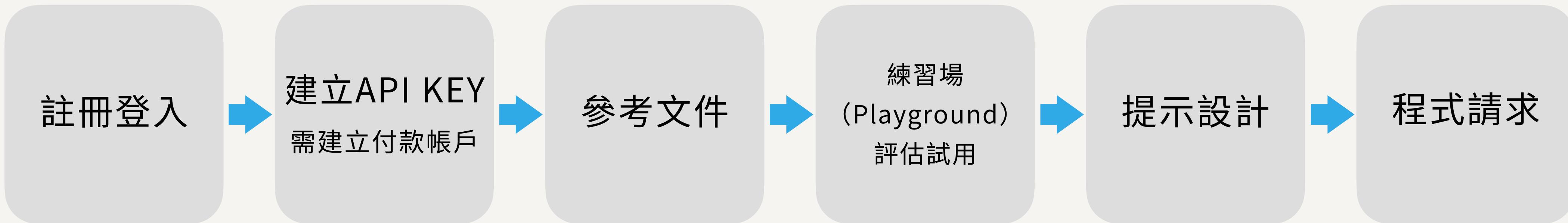
- 客製化模型
 - 提示工程 prompt engineering
 - Agentic design patterns
 - RLHF
 - Fine-tuning
 - RAG
 - 建構環境
 - 使用者介面
 - 回應快取與過濾

OpenAI API Usage Flow



ChaGPT logo, 丁志仁, CC BY-SA 4.0 <<https://creativecommons.org/licenses/by-sa/4.0>>, via Wikimedia Commons

OpenAI API Use Steps



跟著 ChatGPT 學程式

- 利用提示工程與模版
- 策略：
 - 增進現有程式，要求他寫得更 Pythonic
 - 簡化程式
 - 撰寫測試程式
 - 增加程式效率
 - 協助偵錯
 - 重複測試

你是一個程式專家，能夠寫出精準、明確且乾淨的程式碼。
請寫一個股票計算移動平均的類別。
每一行程式碼都加上註解。
你的程式碼：

參考資料

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. 人工智能, viewed June 2023, wiki, <<https://zh.wikipedia.org/zh-tw/人工智能>>
7. 人工智能史, viewed June 2023, wiki, <<https://zh.wikipedia.org/zh-tw/人工智能史>>

參考資料

- 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.
- <https://github.com/rasbt/machine-learning-book>