TensorFlow+Keras 深度學習人工智慧

基本概念介紹

主講者：王彤云

課程綱要

一、人工智慧、機器學習、深度學習介紹

二、深度學習原理

三、Tensorflow與Keras介紹

一、人工智慧、機器學習、深度學習介紹

人工智慧(Artifical Intelligence)

目標:
希望能讓電腦像
人一般思考與學習

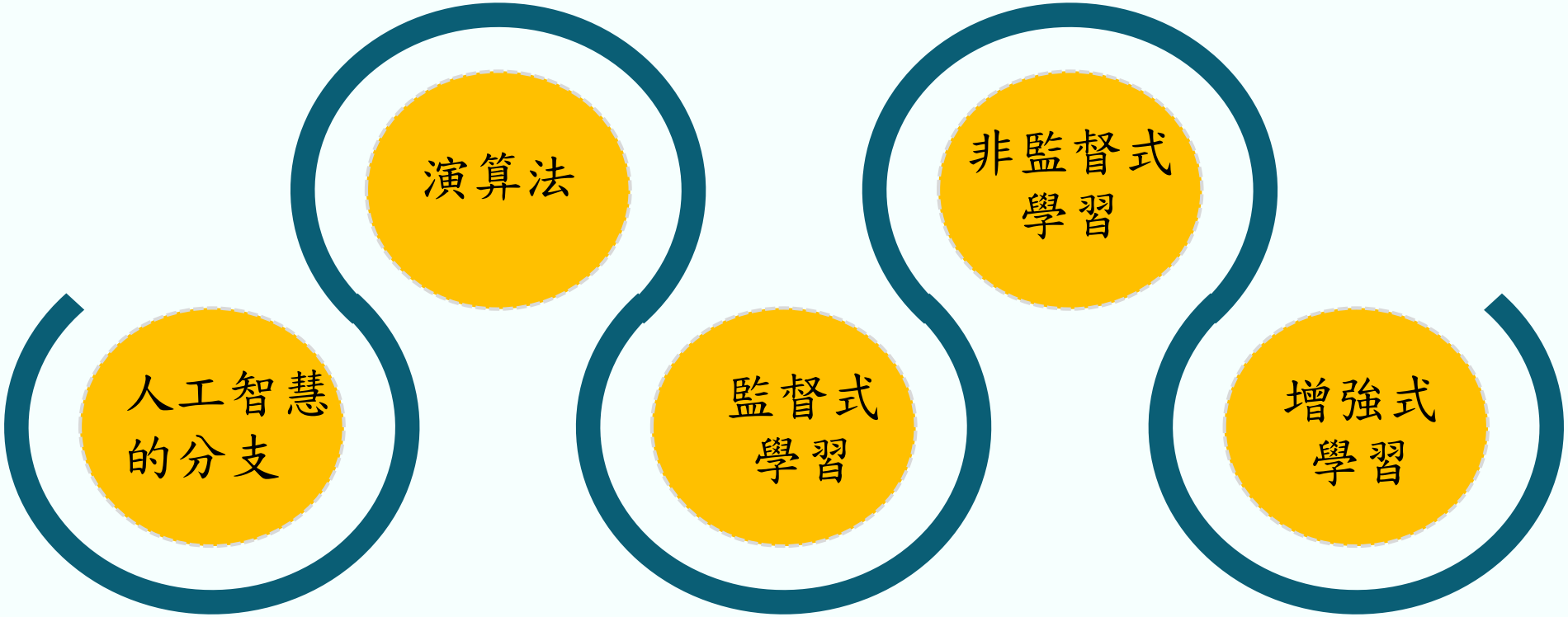
人工智慧的分類
—強人工智慧(Strong AI)
—弱人工智慧(Weak AI)

最早開始於1950年代

圖靈測試
(Turing Testing)

1980 年代

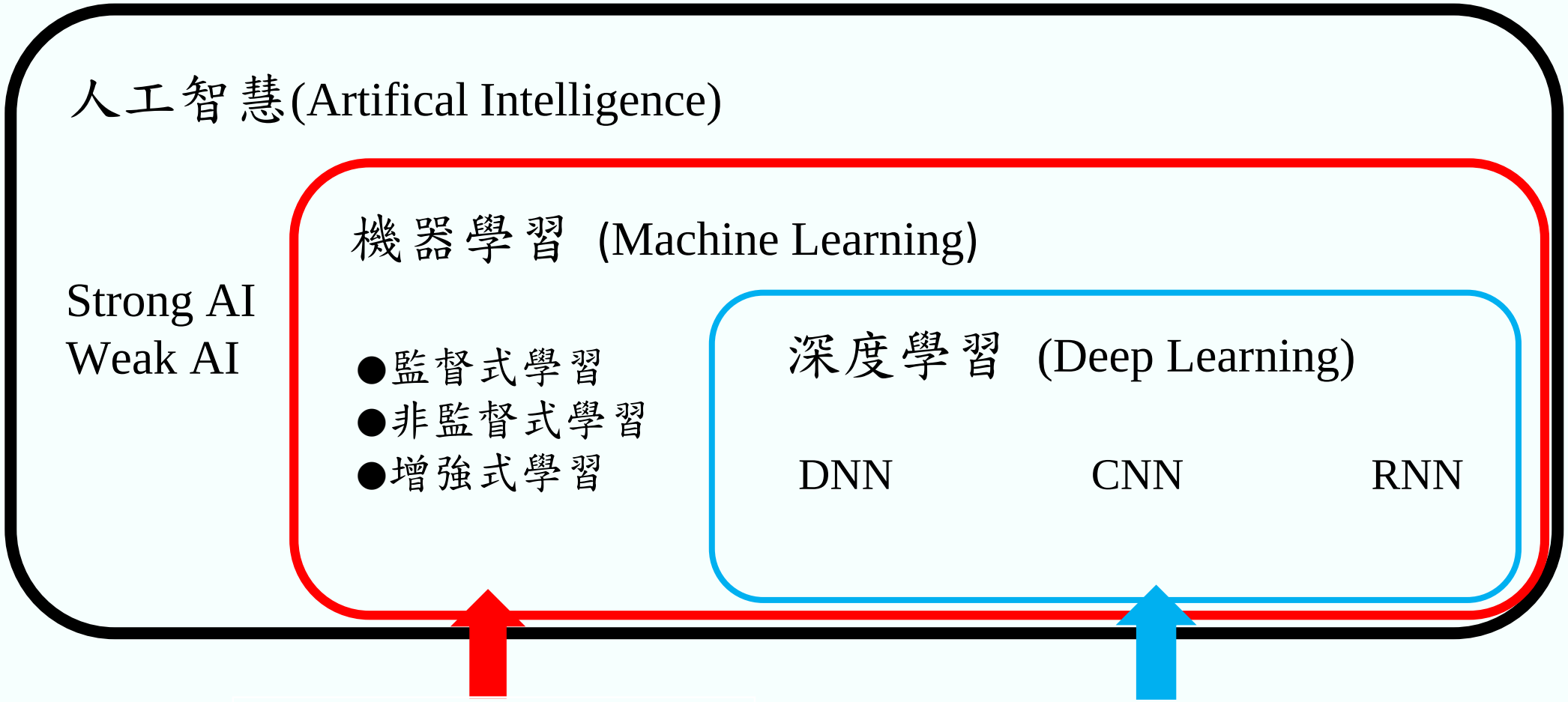
機器學習 (Machine Learning)



深度學習 (Deep Learning)

深度神經網路　　卷積神經網路　　遞迴神經網路

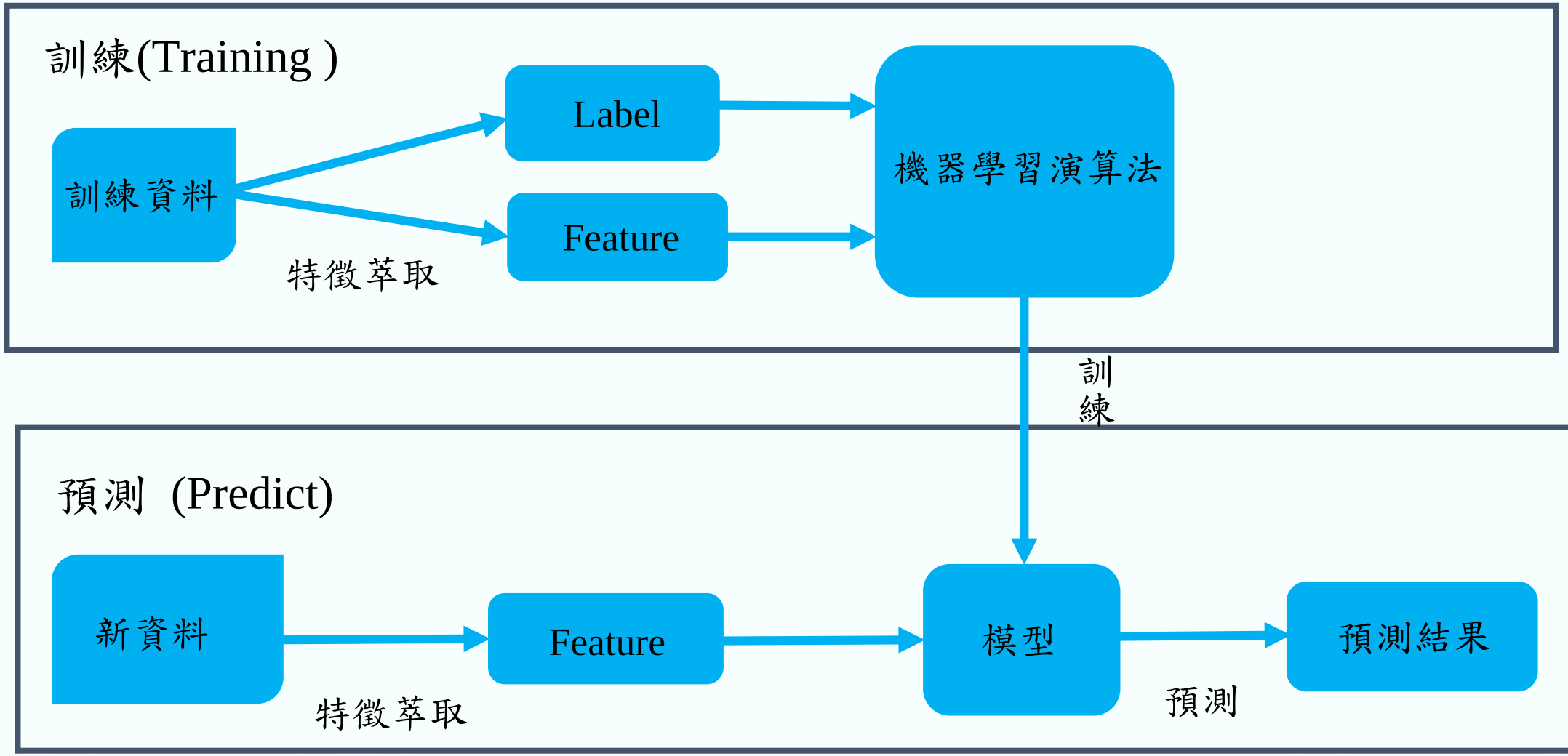
人工智慧、機器學習、深度學習關係



機器學習介紹

1	訓練	透過演算法，訓練歷史資料產生模型
2	Features	資料的特徵 。例如： 濕度、風向、季節、氣壓
3	Label	資料的標籤。例如:天候(1.晴、2.雨、3.陰、4.雪)
4	階段	訓練(Training) & 預測(Predict)

機器學習介紹


訓練(Training)
訓練資料
特徵萃取
Label
Feature
機器學習演算法
訓練
預測 (Predict)
新資料
特徵萃取
Feature
模型
預測
預測結果

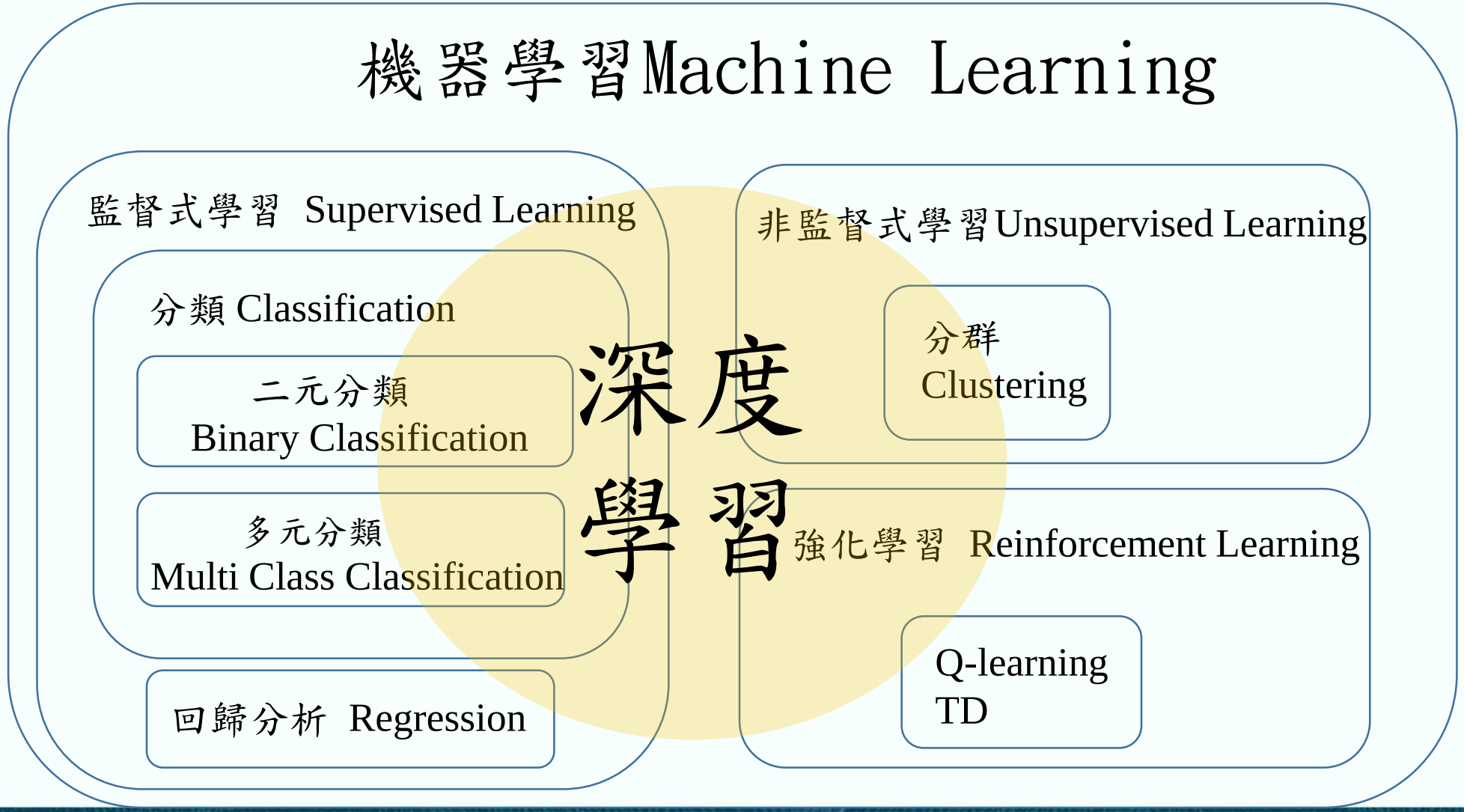
機器學習分類

監督式學習	非監督式學習	強化學習
●二元分類：是非題 ●多元分類：選擇題 ●回歸分析：計算題	cluster 集群演算法	Q-learning、TD、Sarsa

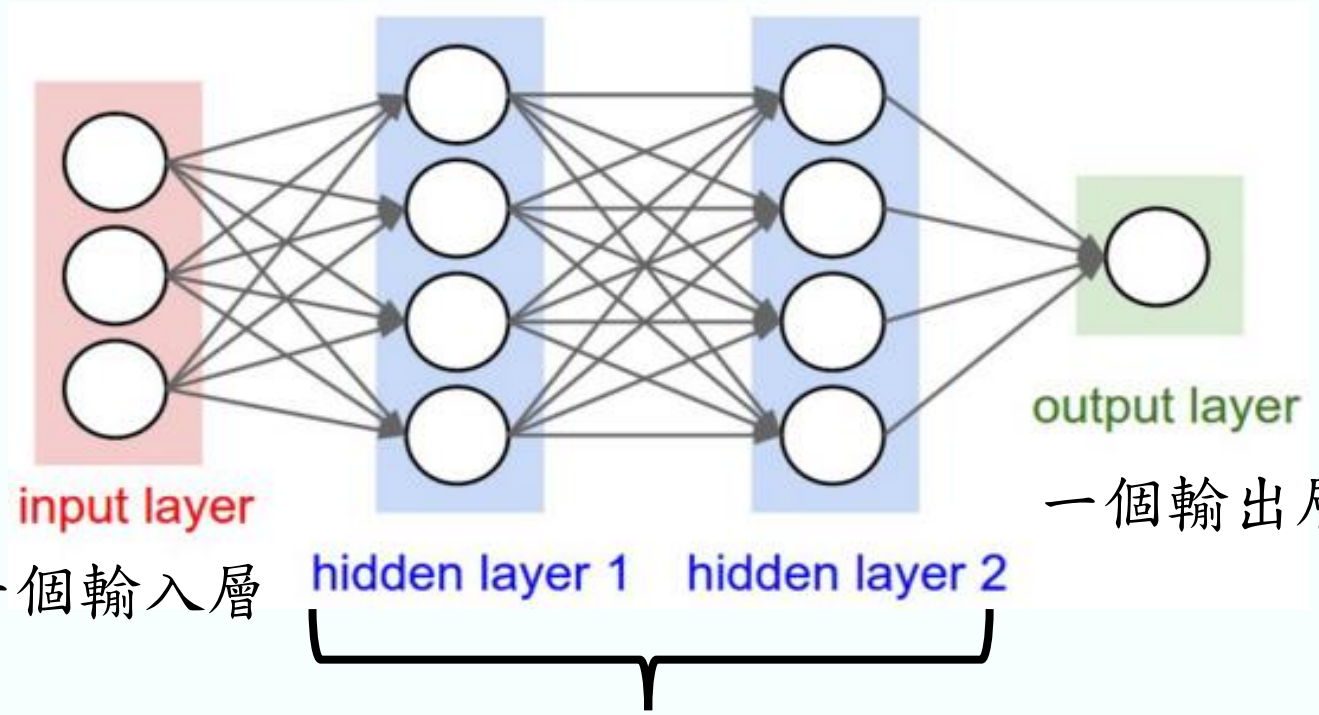
機器學習分類整理

分類	細分類	Features(特徵)	Label(預測目標)
監督式學習	Binary Classification 二元分類	濕度、風向、風速、季節、氣壓……	只有0與1選項(是非題) 0:不會下雨、1:會下雨
監督式學習	Multi-Class Classification 多元分類	濕度、風向、風速、季節、氣壓……	有多個選項(選擇題) 1:晴 2: 雨 3: 陰 4: 雪
監督式學習	Regression 回歸分析	濕度、風向、風速、季節、氣壓……	值是數值(計算題) 溫度可能是 -50~50 度的範圍
非監督式學習	Clustering 群集	濕度、風向、風速、季節、氣壓……	無label。目的: 將資料依照特徵分成幾個相異性最大的群組，而群組內的相似程度最高
強化學習	Q-learning、TD (Temporal Difference)	濕度、風向、風速、季節、氣壓……	不段訓練機器循序漸進，學會執行某項任務的演算法

機器學習分類整理


機器學習Machine Learning
監督式學習 Supervised Learning
分類 Classification
二元分類
Binary Classification
多元分類
Multi Class Classification
回歸分析 Regression
深度學習
非監督式學習Unsupervised Learning
分群
Clustering
強化學習 Reinforcement Learning
Q-learning
TD

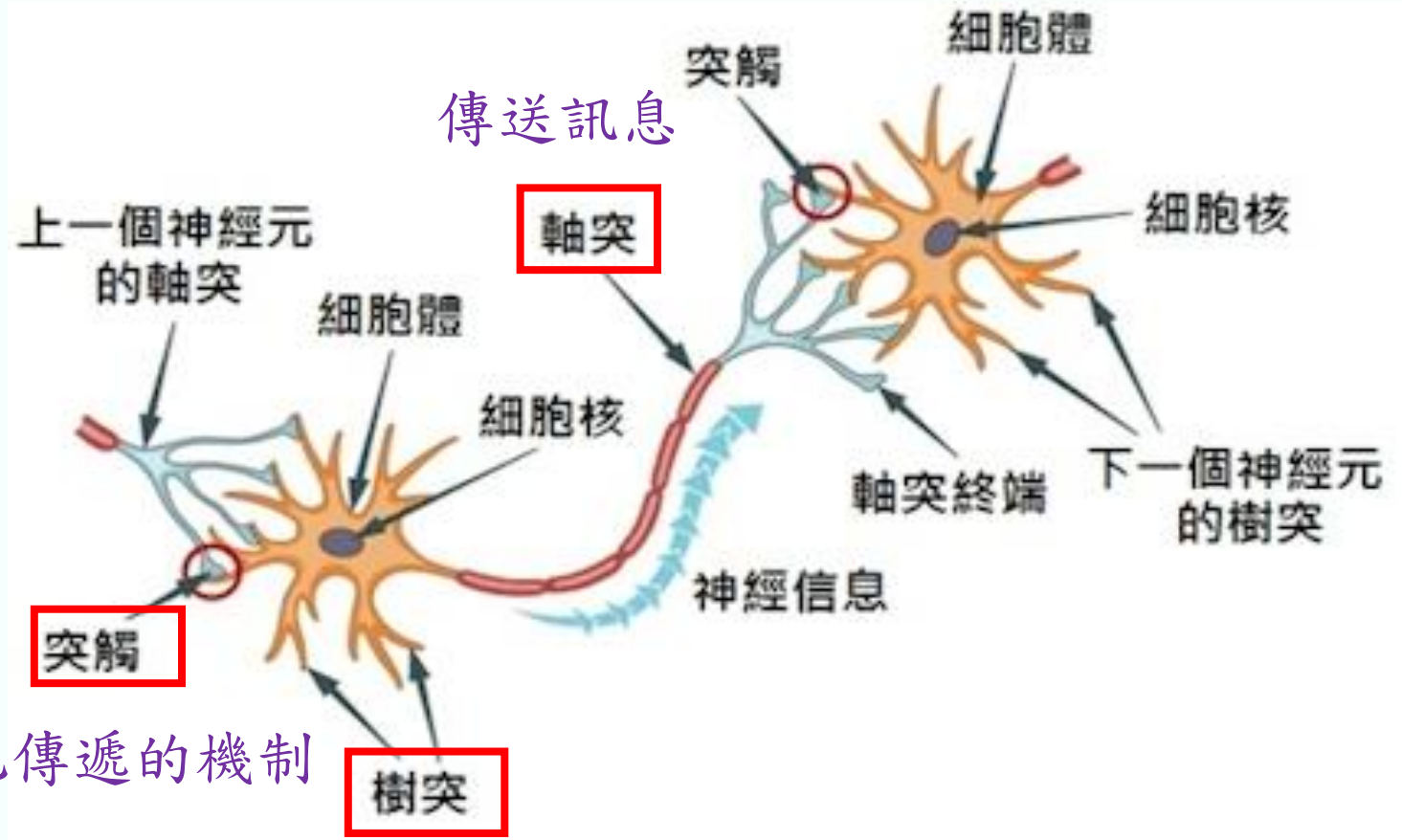
深度學習簡介



隱藏層可以非常多層，所以稱為深度學習

二、深度學習原理

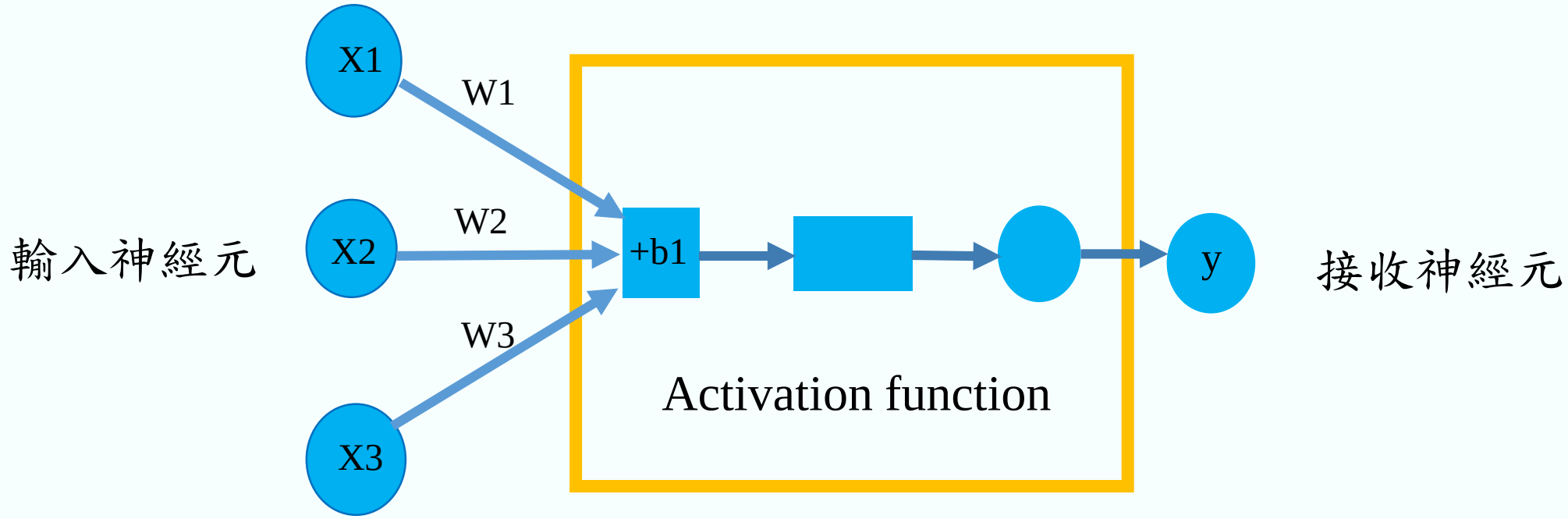
神經傳導原理介紹



輸入與輸出的神經元傳遞的機制

負責接收訊息

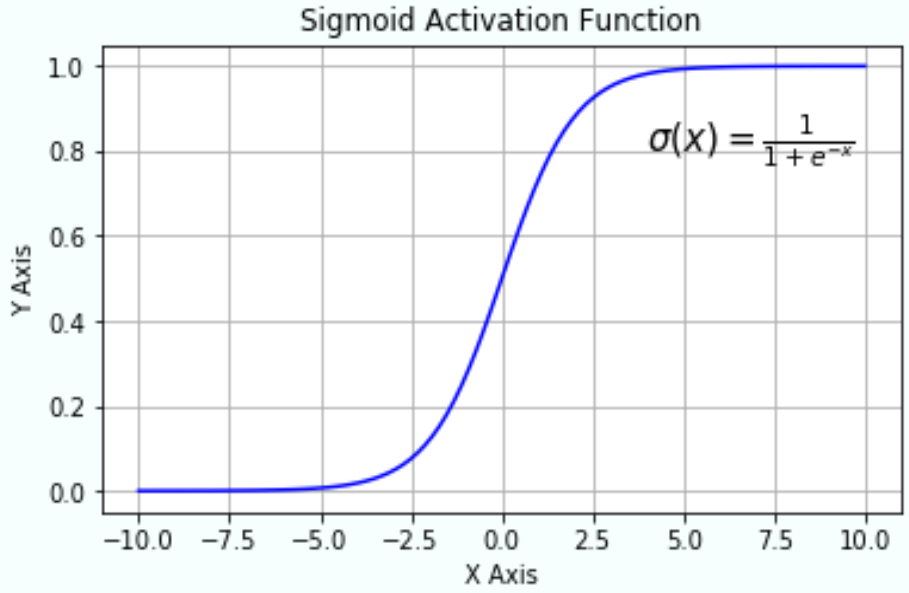
以數學公式模擬神經元



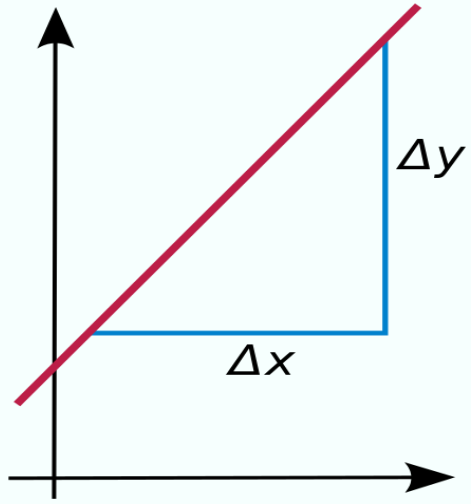
Activation function

模擬神經傳導的運作方式

通常為非線性函數

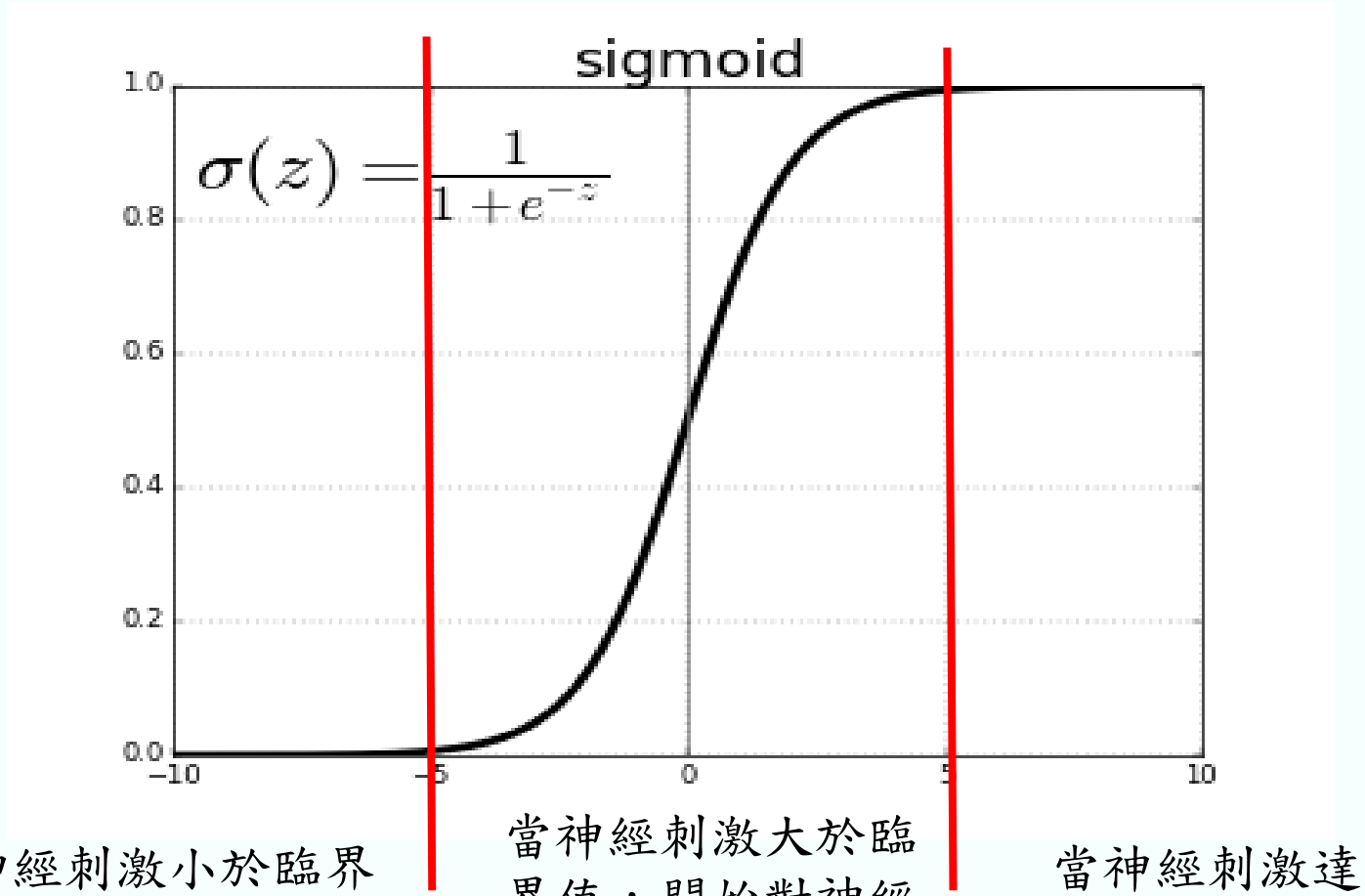


非線性函數



線性函數

Sigmoid 激活函數

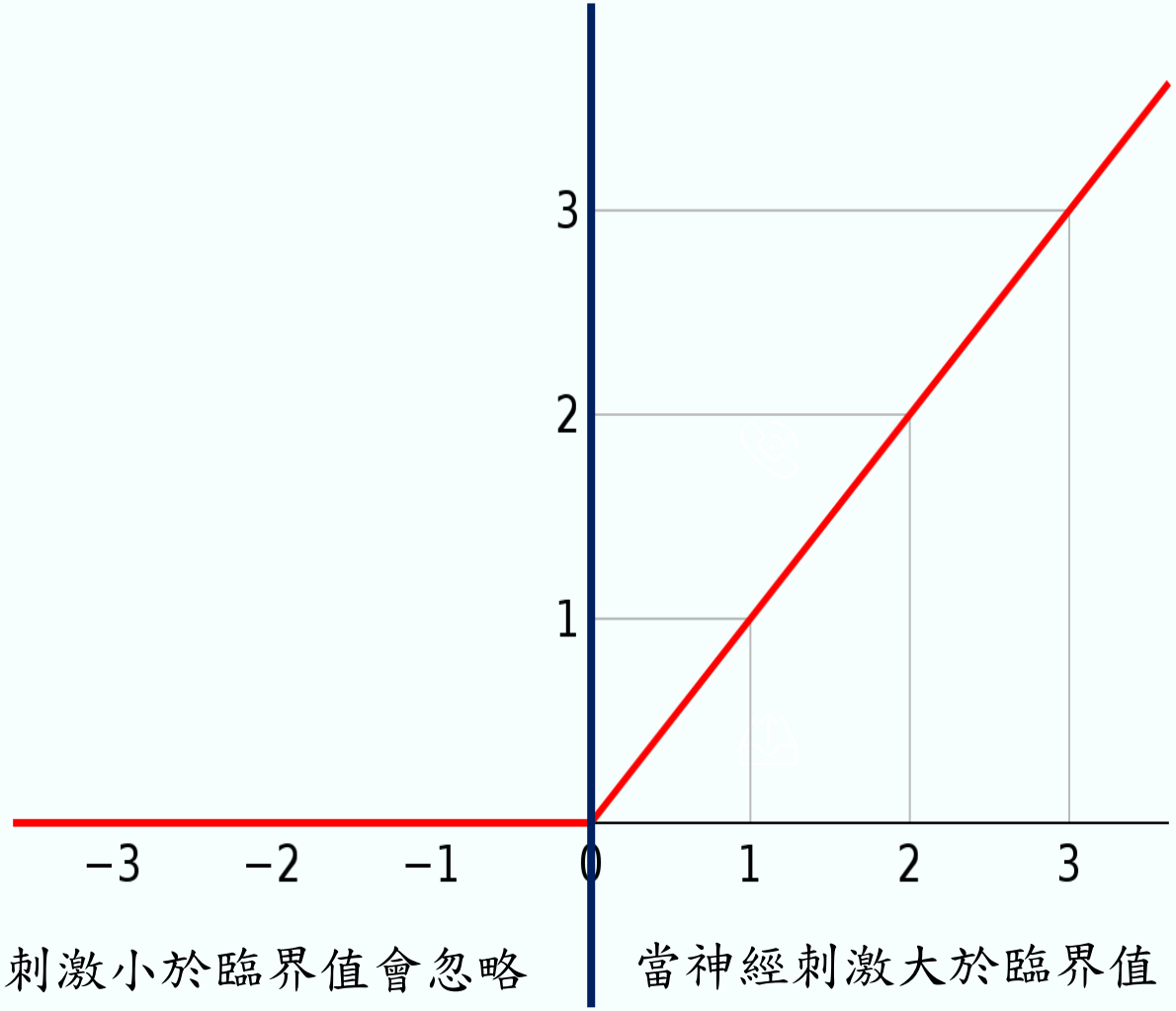


當神經刺激小於臨界值會忽略

當神經刺激大於臨界值，開始對神經刺激有反應

當神經刺激達到一定程度，感覺會開始鈍化

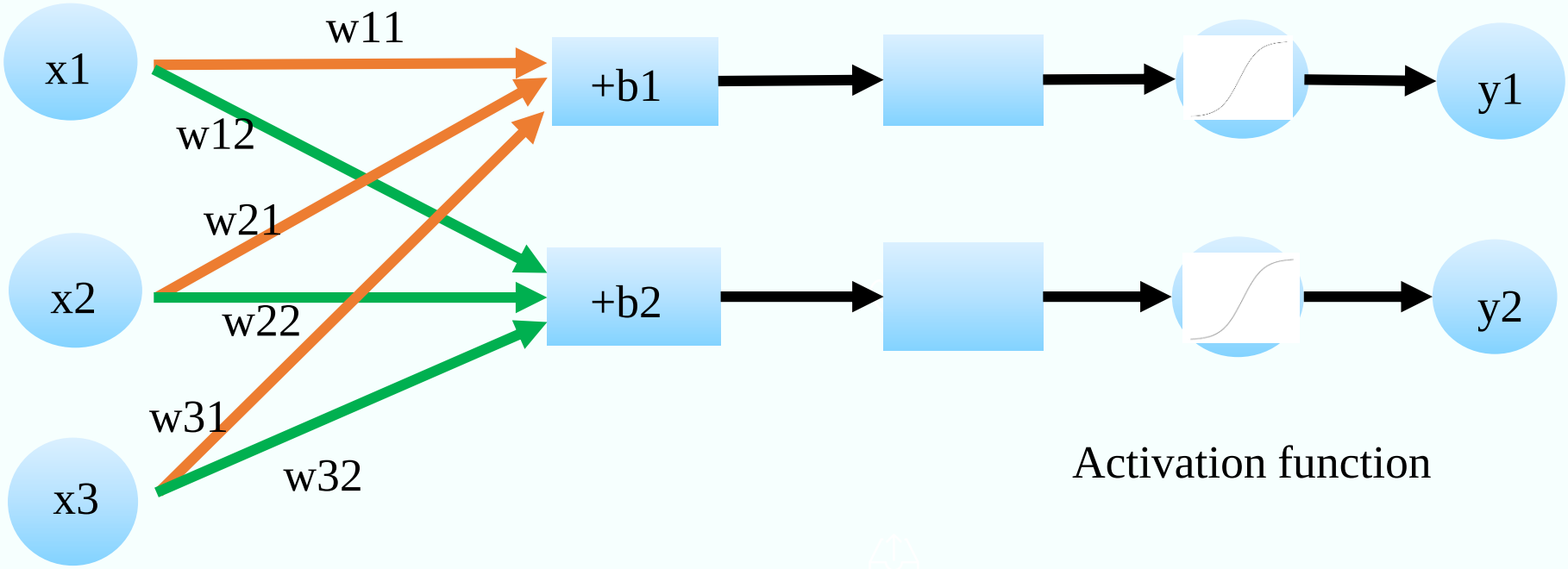
Relu 函數



當神經刺激小於臨界值會忽略

當神經刺激大於臨界值，開始對神經刺激有反應

以矩陣運算模擬神經網路



y1=activation function (x1 * w11 + x2 * w21 + x3 * w31+ b1)

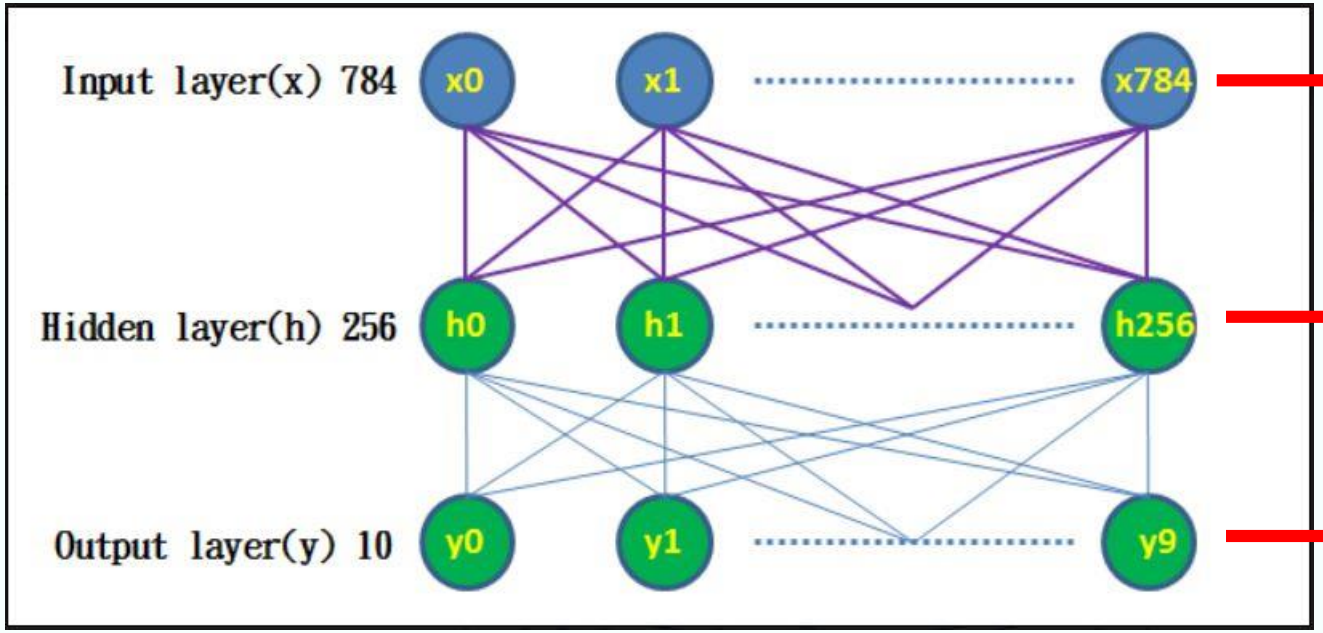
y2=activation function (x1 * w12 + x2 * w22 + x3 * w32 + b2)

以矩陣運算模擬神經網路

- y1=activation function (x1 * w11 + x2 * w21 + x3 * w31+ b1)
- y2=activation function (x1 * w12 + x2 * w22 + x3 * w32 + b2)
- 合併 y1 + y2，整合成一個矩陣運算公式

多層感知器模型

以多層感知器模型辨識 Minst手寫數字影像



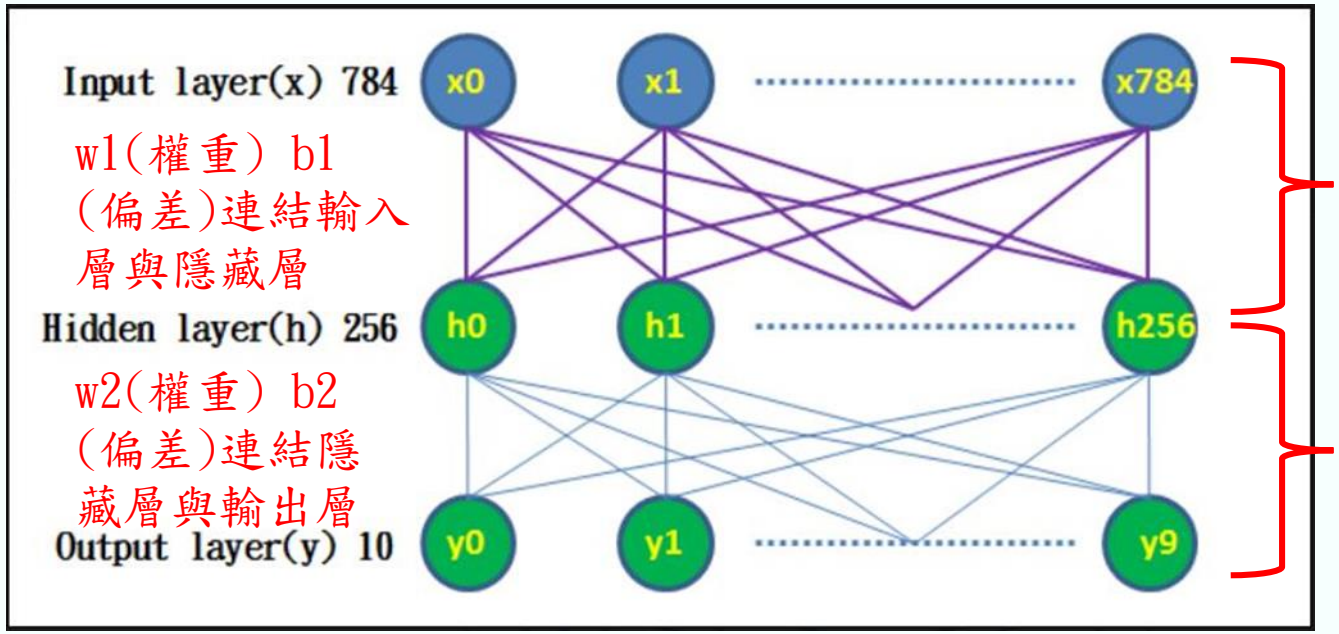
輸入層 ： 784 個輸入神經元，接收外界訊號

隱藏層 ： 模擬內部神經元，共有256個隱藏神經元

輸出層 ： 10個輸出神經元，就是預測結果。對應到我們希望預測的數字 0-9共有10個結果

多層感知器模型

以矩陣公式模擬多層感知器模型運作



建立輸入層與隱藏層，
公式 ： h = relu (x * w1 + b1)

建立隱藏層與輸出層，
公式 ： y = softmax (h1 * w2 + b2)

Back Propagation 反向傳播演算法

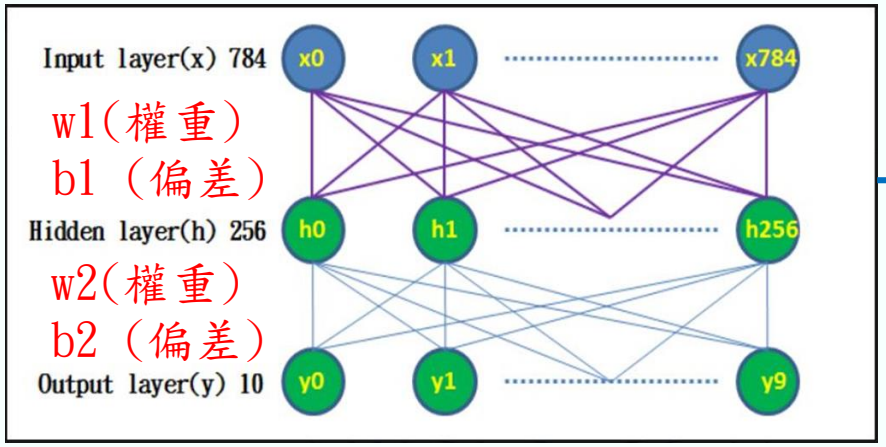
- 訓練人工神經網路的常見方法，並且與最優化方法 Optimizer (例如: 梯度下降法)結合使用
- 監督學習方法，必須輸入 features(特徵值)，與 label（真實的值）
- 簡單說就是從錯誤中學習

Mnist 資料集

預處理

1. 資料預處理features(數字影像的特徵值)

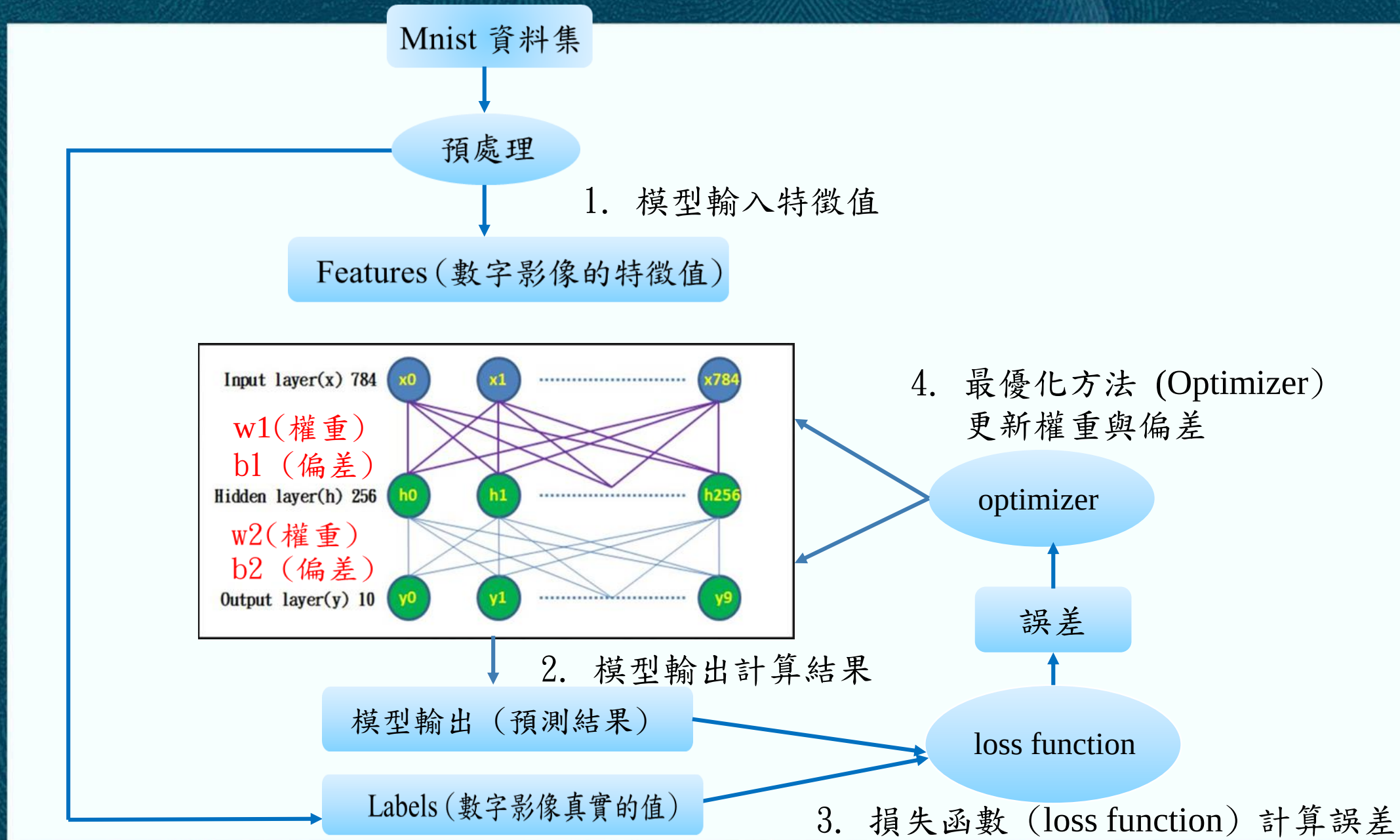
Features (數字影像的特徵值)



2. 建立模型

Labels (數字影像真實的值)

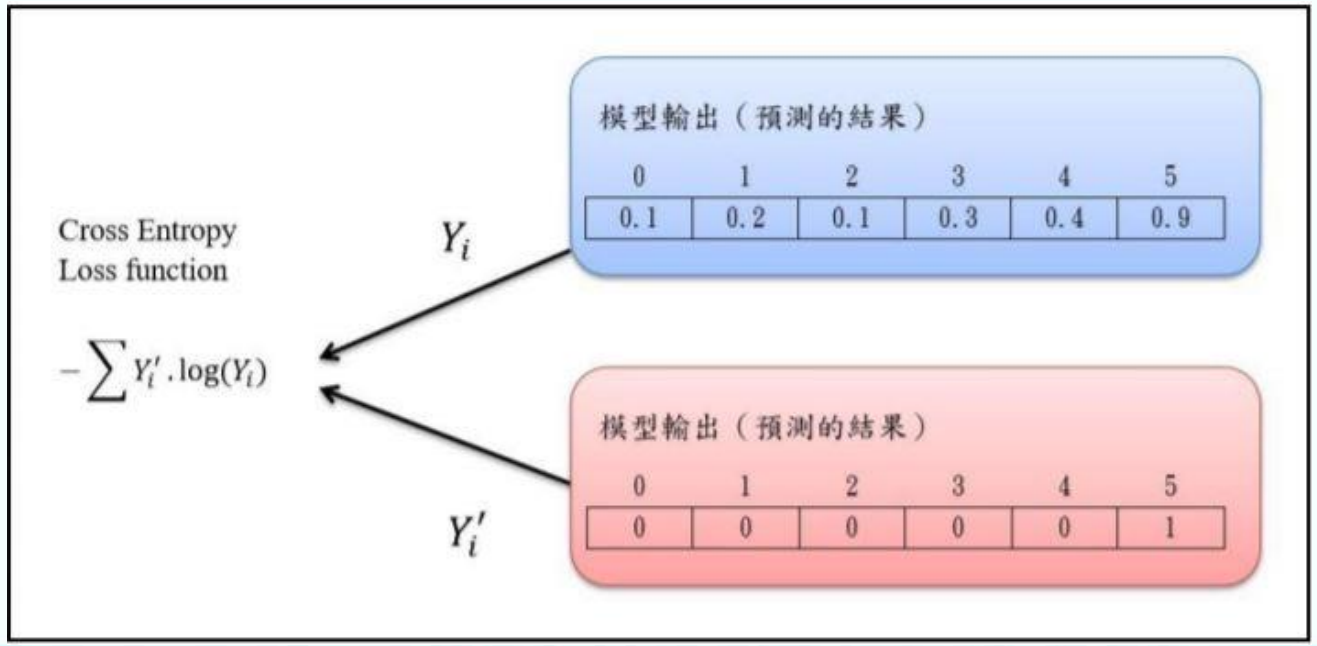
1. 資料預處理labels (數字影像真實的值)


Mnist 資料集
預處理
1. 模型輸入特徵值
Features（數字影像的特徵值）
Input layer(x) 784
x0
x1
x784
w1(權重)
b1 (偏差)
Hidden layer(h) 256
h0
h1
h256
w2(權重)
b2 (偏差)
Output layer(y) 10
y0
y1
y9
4. 最優化方法 (Optimizer)
更新權重與偏差
optimizer
誤差
2. 模型輸出計算結果
模型輸出（預測結果）
loss function
Labels（數字影像真實的值）
3. 損失函數（loss function）計算誤差

使用Back Propagation 反向傳播演算法進行訓練

- 損失函數 ： 計算誤差

- Cross Entropy 是深度學習常用的損失函數（loss function）



使用Back Propagation 反向傳播演算法進行訓練

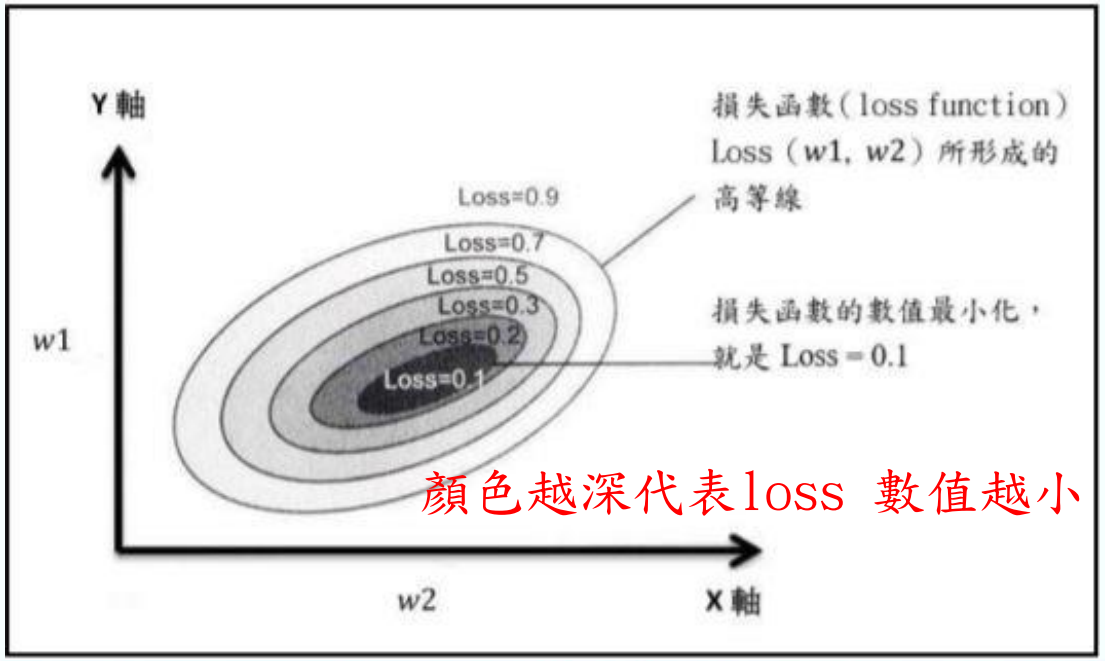
- 最佳化方法（Optimizer）：使用某種數值方法，在不斷的批次訓練中，不斷更新權重(weight)與偏差(bias)，使損失函數的誤差值最小化。最終找出誤差值最小的「權重與偏差的組合」

- 隨機梯度下降法（Stochastic gradient descent , SGD）

- 想成：在所有「權重與偏差的組合」所組成的高維度空間中，每個訓練批次，沿著每個維度下降的方向走一小步，經過多次步驟，就可以找到最佳化的「權重與偏差的組合」

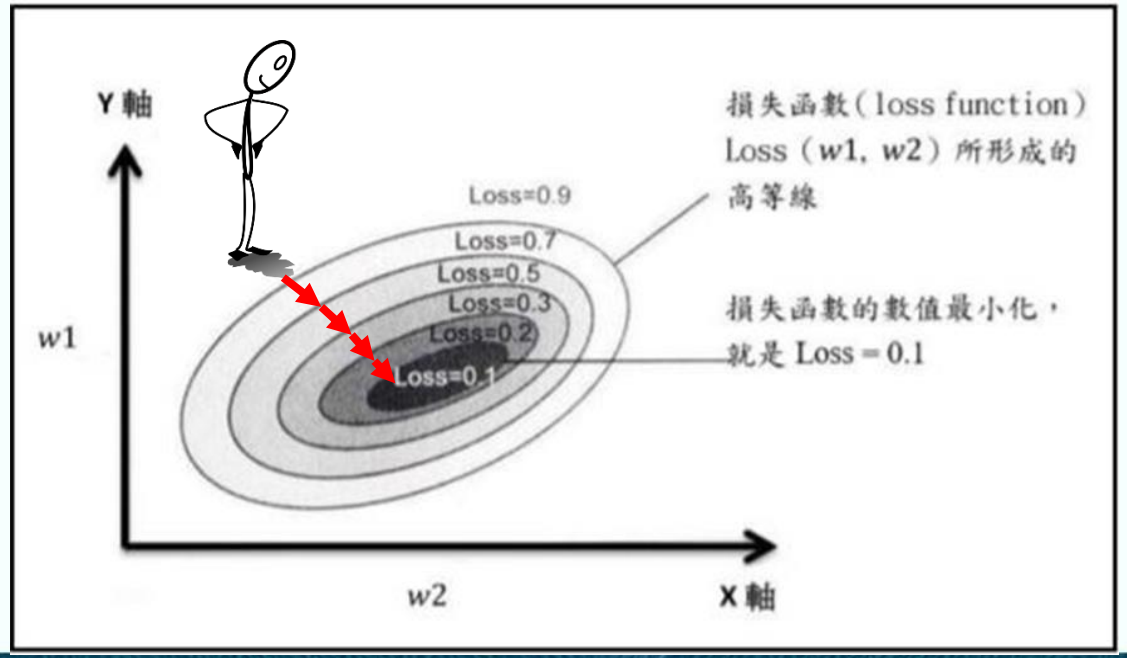
使用Back Propagation 反向傳播演算法進行訓練

- 二維權重與損失函數
- 真實深度學習中權重與偏差數量很多，會形成非常多維度的空間



使用Back Propagation 反向傳播演算法進行訓練

- SGD 梯度下降法
- 每次沿著 LOSS 下降的方向，每次訓練批次走一小步，經過許多次訓練步驟，就能夠下降到 Loss=0.1，損失函數的誤差最小化



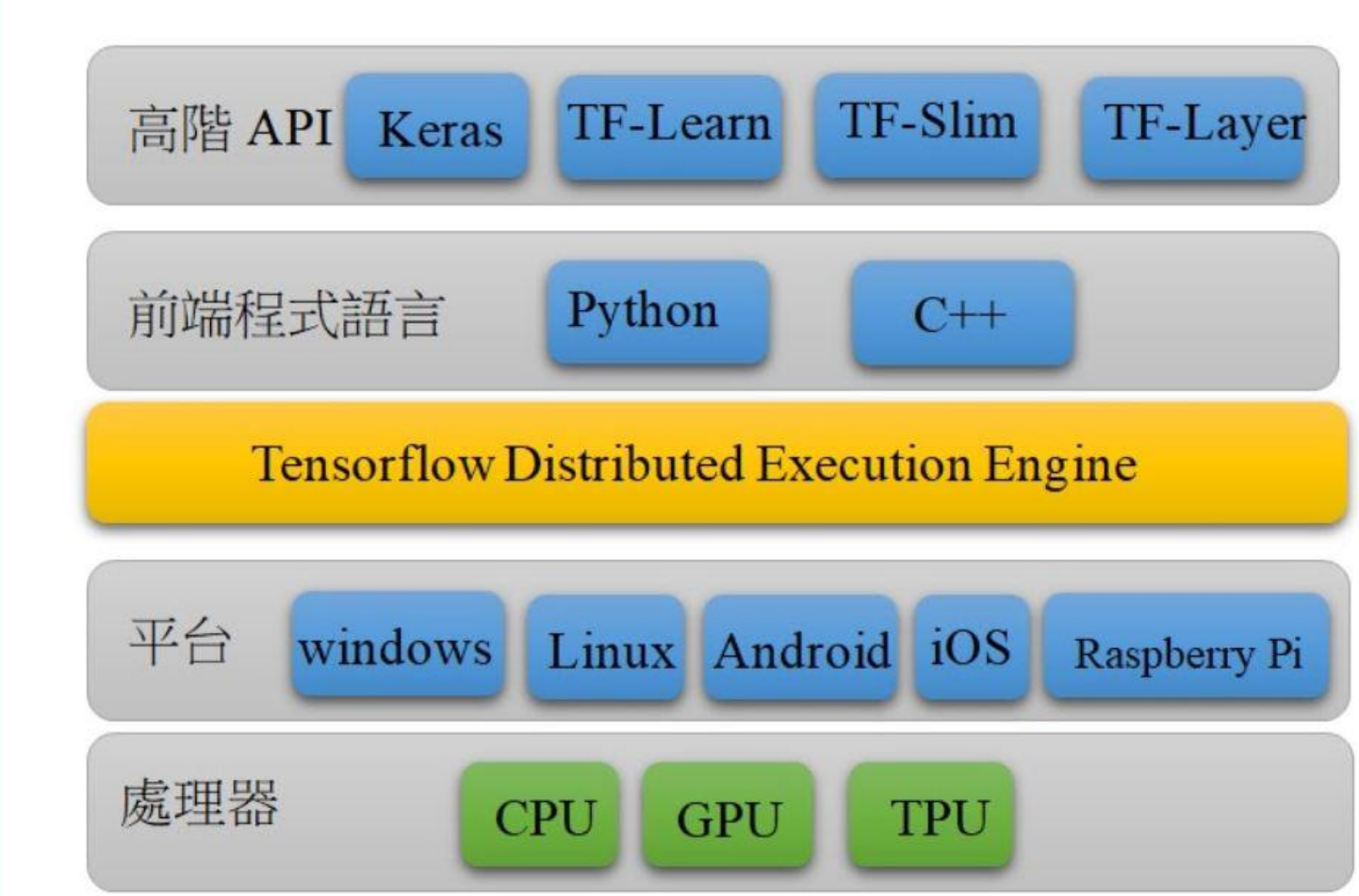
三、TensorFlow 與 Keras 介紹

TensorFlow vs Keras

- Tensorflow：功能強大、執行效率高、支援各種平台、低階的深度學習程式庫，學習門檻高

- Keras：高階的深度學習程式庫、學習門檻低

Tensorflow 架構圖



Tensorflow簡介

- Tensorflow 是由 Tensor 與 Flow 所組成

- Tensor 張量 ： 是一種幾何實體，或是廣義上的「數量」，在此所謂的「數量」包含「純量、向量或矩陣」

- 0 維的張量是純量
- 1 維的張量是向量
- 2 維以上的張量是矩陣

Tensorflow簡介

- Flow 資料流程
- 例子 ：
  假設:您到了陌生的地方，不會當地的語言，為了到達目的地？
  解決方法：畫張地圖，告訴當地的司機，司機依照地圖，載您到目的地
- 為了讓Tensorflow可以**支援不同的程式語言介面**，並且讓Tensorflow程式**可以在各種平台執行**
- 動態顯示計算圖

Tensorflow 程式設計模式

- 核心是「計算圖」(computational graph)，可分為 2 部分 ： 建立「計算圖」與執行 「計算圖」

1. 建立「計算圖」： 使用Tensorflow提供的模組，建立「計算圖」。設計張量運算流程，並且建構各種深度學習或機器學習模型

2. 執行「計算圖」：
   - 建立「Session（原意是對談)」執行「計算圖」。
   - 目的:是在用戶端和執行裝置之間建立連結。可以將「計算圖」在各種不同裝置中執行。
   - 資料的傳遞也必須透過Session，最後取得執行後的結果

Keras介紹

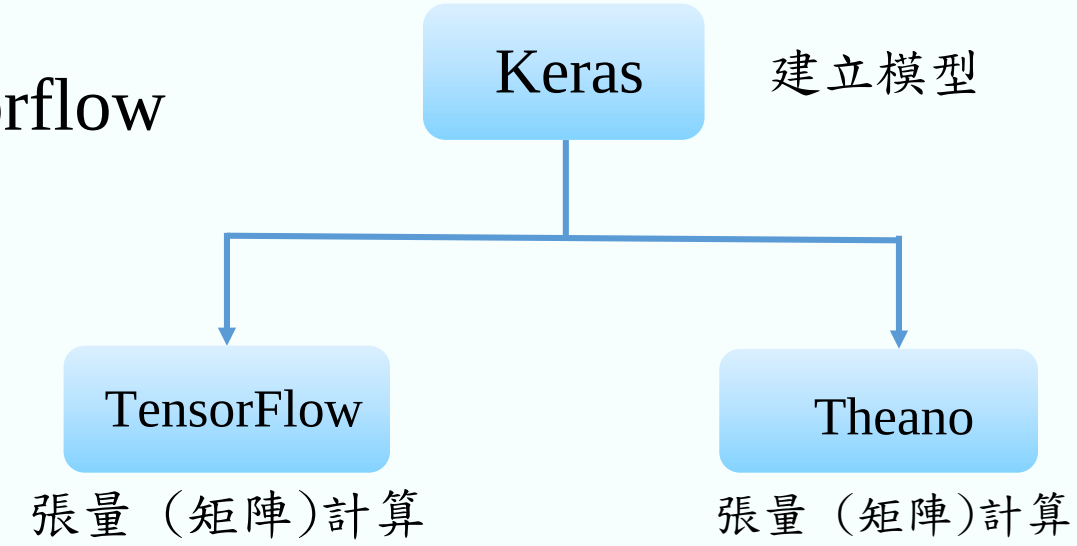
- 一個開放原始碼，高階深度學習程式庫，使用python編寫，能夠運行在TensorFlow 或 Theano 之上

為何要使用Keras?

- 最少的程式碼，花費最少的時間，就可建立深度學習模型，並進行訓練、評估準確率，進行預測。

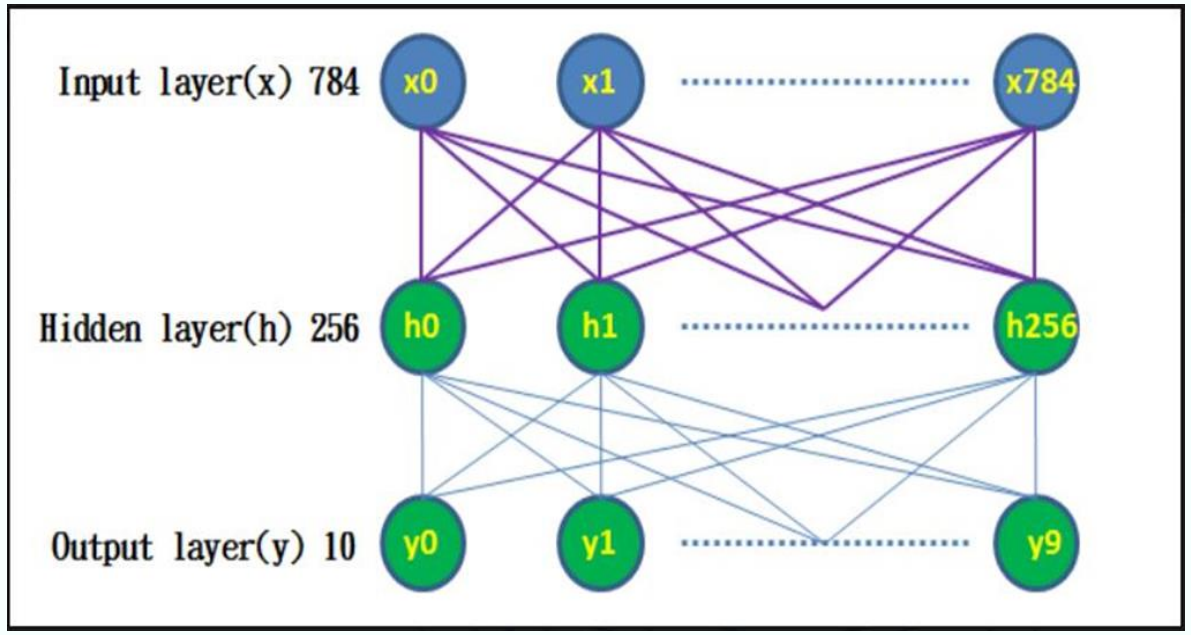
Keras的運作方式

- Keras 是一個(model-level)模型及的深度學習程式庫
- Keras只處理模型的建立、訓練、預測等功能。底層運作(張量)運算，Keras必須配合「後端引擎」(backend engine)進行運算。
- 兩種後端引擎:Theano 與 Tensorflow



Keras程式設計模式

- 像做一個多層蛋糕



Keras 建立多層感知器 Multilayer perceptron模型

- 1. 建立 Sequential 模型

  model = Sequential ()
- 2. 加入「輸入層」與「隱形層」至模型 ： Keras 已經內建各式神經網路層，例如: Dense層、Conv2d層等，只要在之前建立的模型上，加入選擇的神經網路層就可以了（蛋糕架加入蛋糕層）

  model.add (Dense (unit = 256 , input_dim = 784 , kernel_initializer = 'normal', activation = 'relu'))
- 3. 加入「輸出層」至模型 ： 就像在加入一層蛋糕

  model.add(Dense(unit = 10 , kernel_initializer = 'normal', activation = 'softmax'))

Keras 與 TensorFlow 比較

	Keras	TensorFlow
學習難易度	簡單	比較困難
使用彈性	中等	高
開發生產力	高	中等
執行效能	高	高
適合使用者	初學者	進階使用者
張量（矩陣）運算	不需自行設計	需自行設計

繼續努力