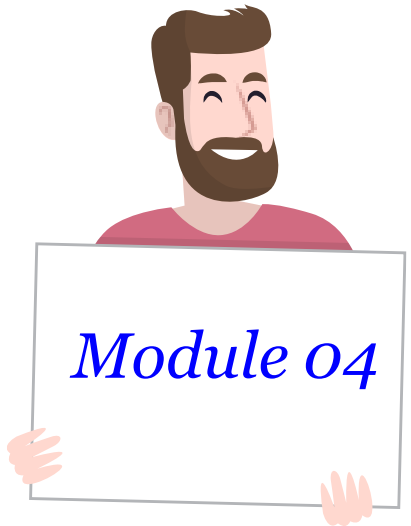
DNN神經網路介紹



designed by freepik

Estimated time:
45 min.

學習目標

- 4-1: 綜觀神經網路
- 4-2: DNN神經網路建構
- 4-3: DNN神經網路範例

4-1:綜觀神經網路

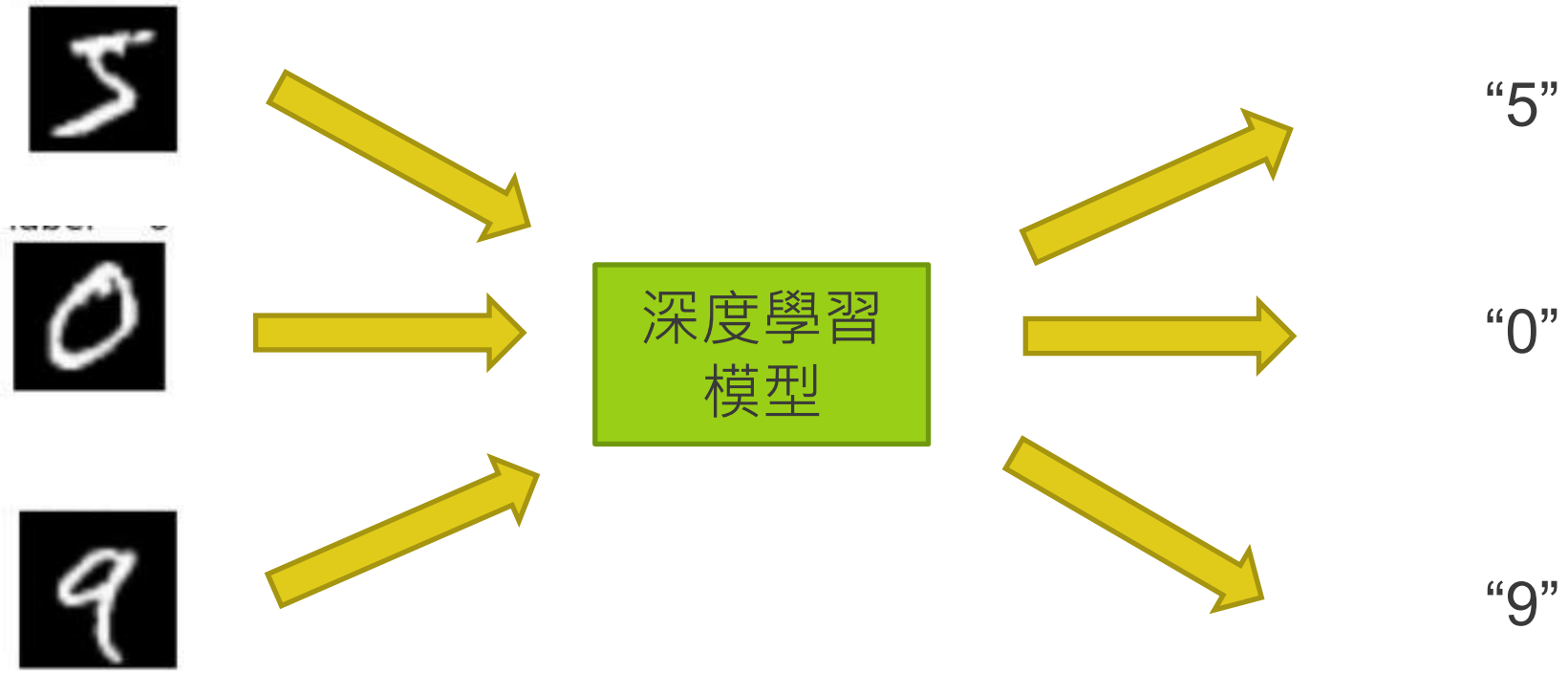
- 深度學習三大步驟
- 神經網路流程

深度學習三大步驟

- 深度學習主要分成三個大步驟
  - 分別是建立模型、定義損失函數、優化
  - 這三個步驟其實就是在解一個最佳化的問題，換句話說，深度學習很多時候都在解最佳化問題

神經網路流程

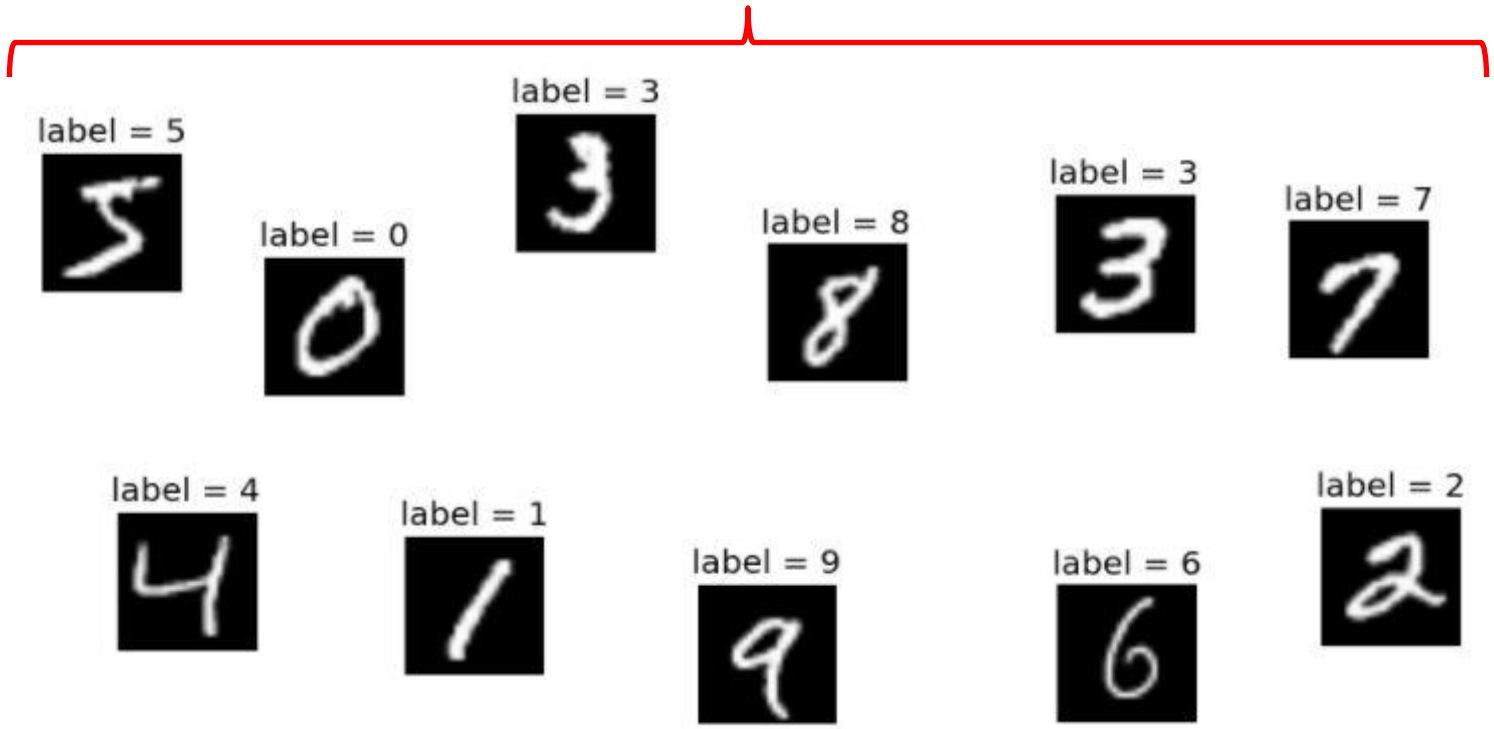
- 假設我們想要讓電腦學會辨識0~9之照片



神經網路流程

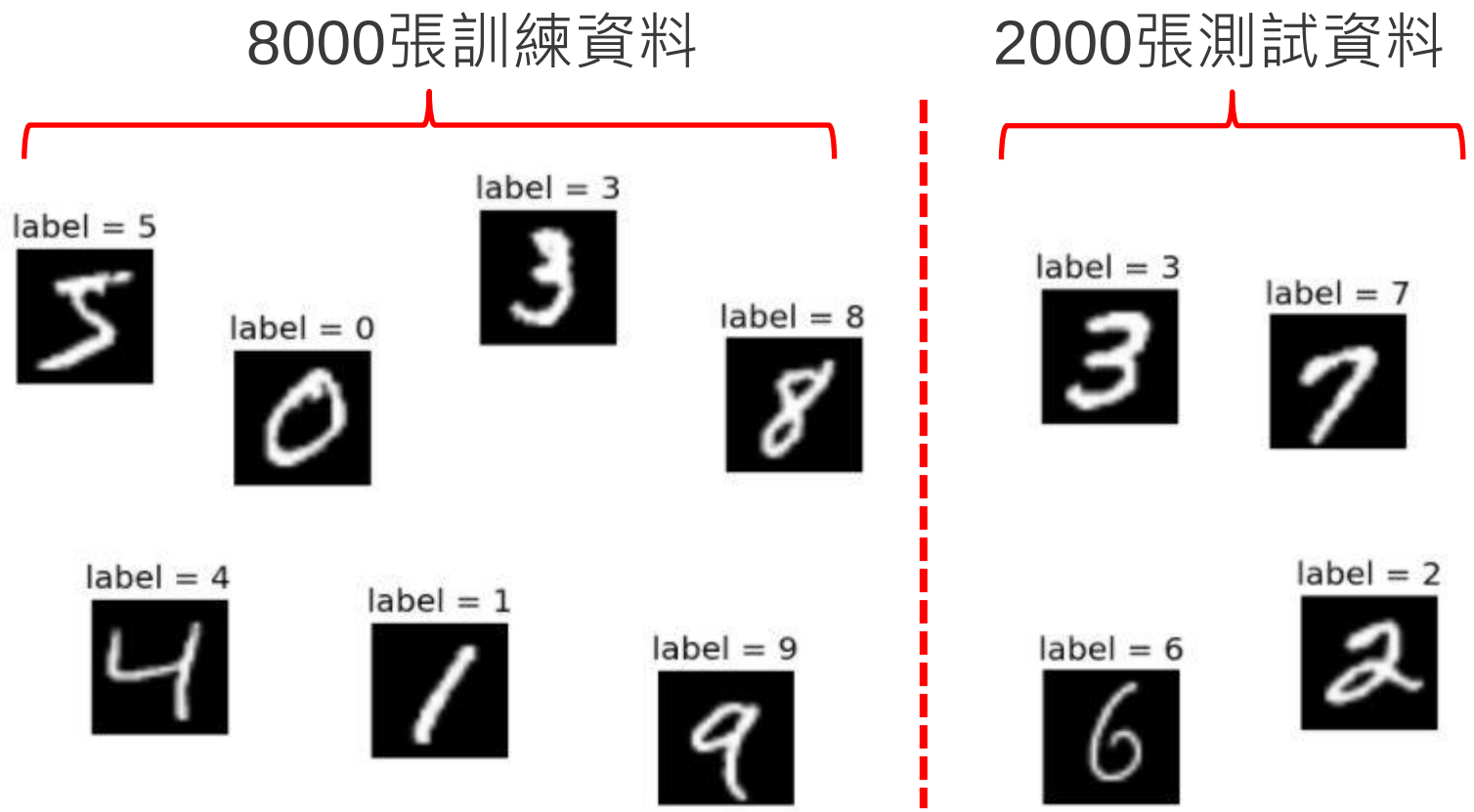
- **假設我們蒐集了10000張照片，並且也將每張照片標註好了**

蒐集了10000張照片以及對應之標籤



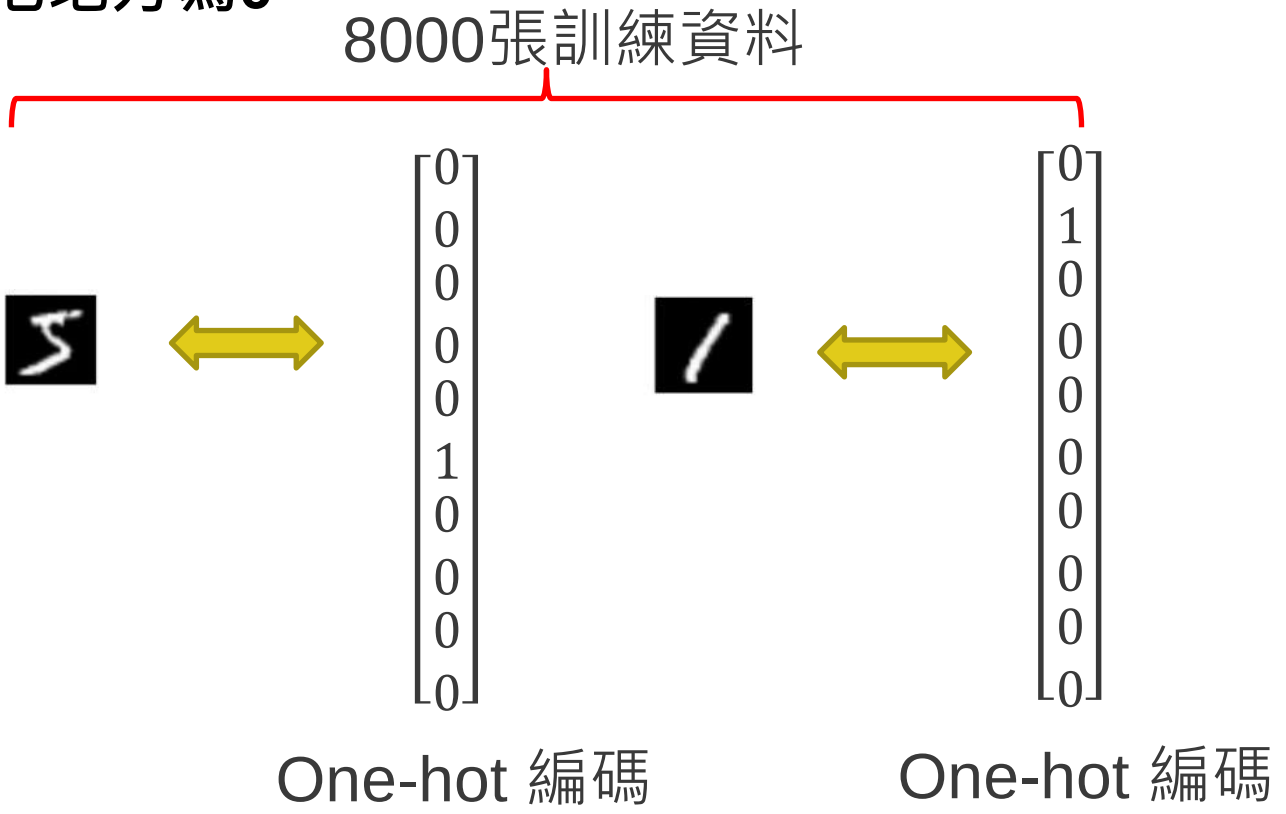
神經網路流程

- 接下來我們將所有資料分成訓練資料及測試資料
  - 常見是8:2分割，即8000張圖片當訓練資料，2000張當測試資料



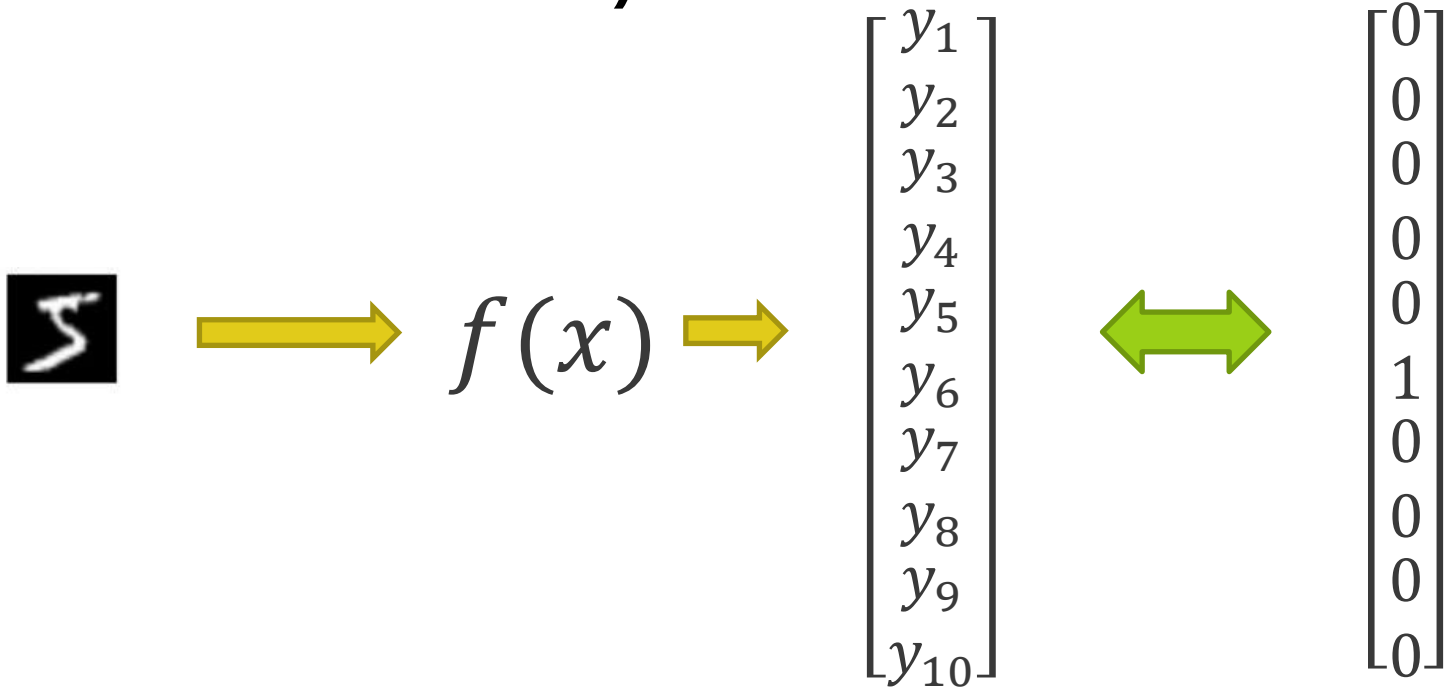
神經網路流程

- **我們將所有訓練資料所對應的標籤做one-hot編碼**
  - **One-hot編碼的意思是產生一個向量，其長度為類別的數量，並在其類別的位置寫1，其他地方寫0**



神經網路流程

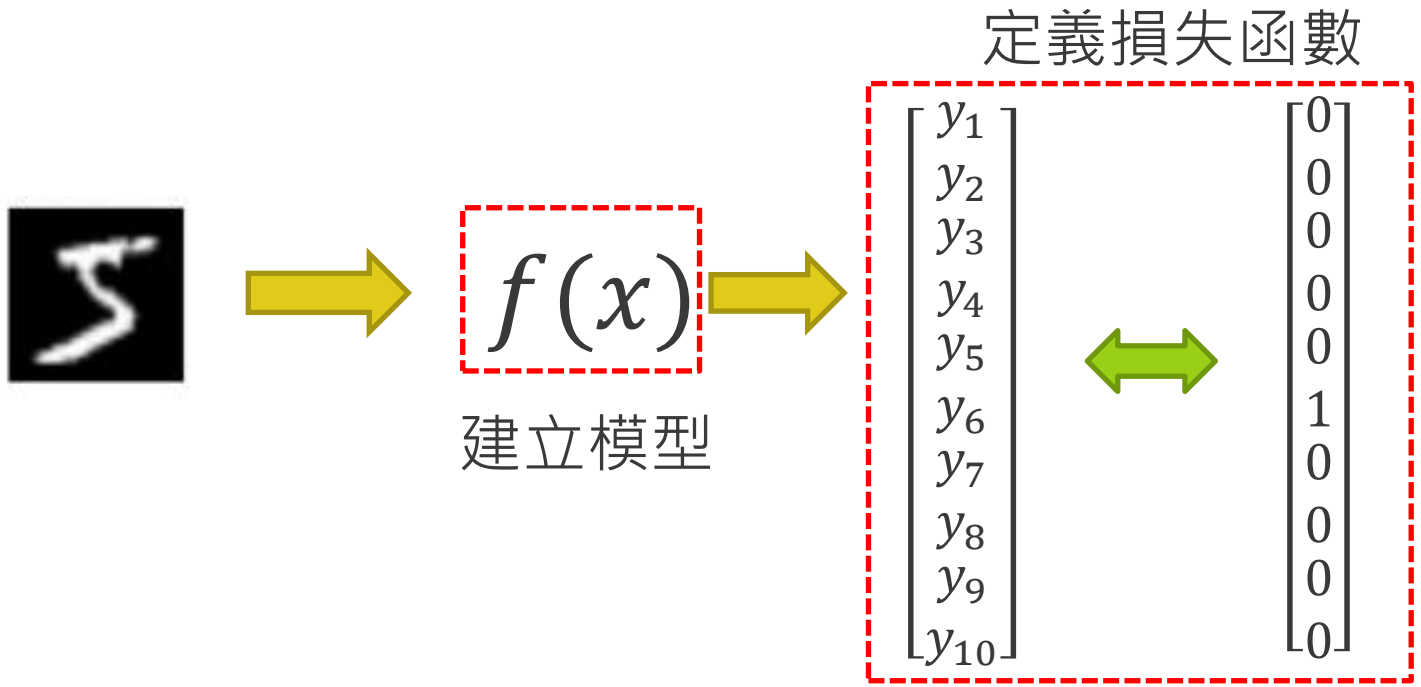
- 我們最終的目的就是期望能找到一個函數
  - 如果我們將某筆資料輸入這個函數，其函數輸出的預測向量會跟期望向量(剛剛one-hot編碼的向量)越接近越好



讓兩個向量越接近越好

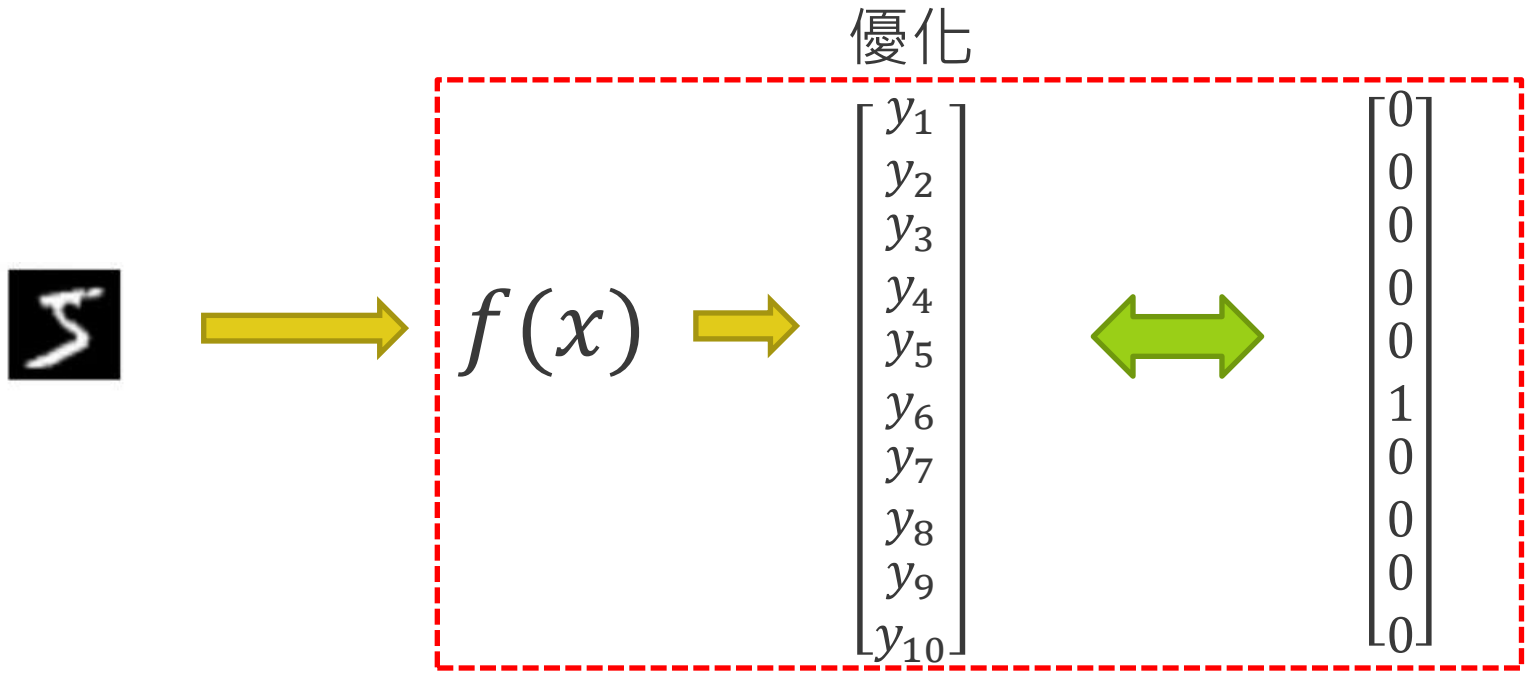
神經網路流程

- 整個流程呼應我們之前提到的三個步驟，建立模型、定義損失函數、優化
  - 建立模型代表要尋找一個f(x)
  - 定義損失函數就是要找一個可以衡量預測向量與期望向量差多遠的函數



神經網路流程

- 整個流程呼應我們之前提到的三個步驟，建立模型、定義損失函數、優化
  - 優化就是要去調整函數裡面的參數，使得預測向量與期望向量更接近



神經網路流程

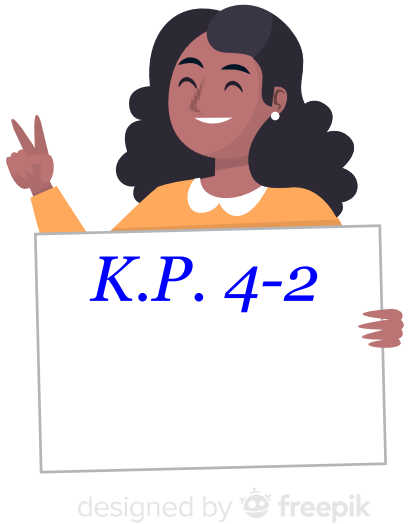
- 當我們將三個步驟完成了以後，我們便可以得到一個優化過後的函數
  - 我們可以拿這個函數去預測測試資料(新的資料)，如果此函數預測出的向量，在某個位置數字特別大，我們邊猜測其為那個類別

$$f^*(x) \rightarrow \begin{bmatrix} 0 \\ 0.2 \\ 0 \\ 0.1 \\ 0 \\ 0.8 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

預測為”5”

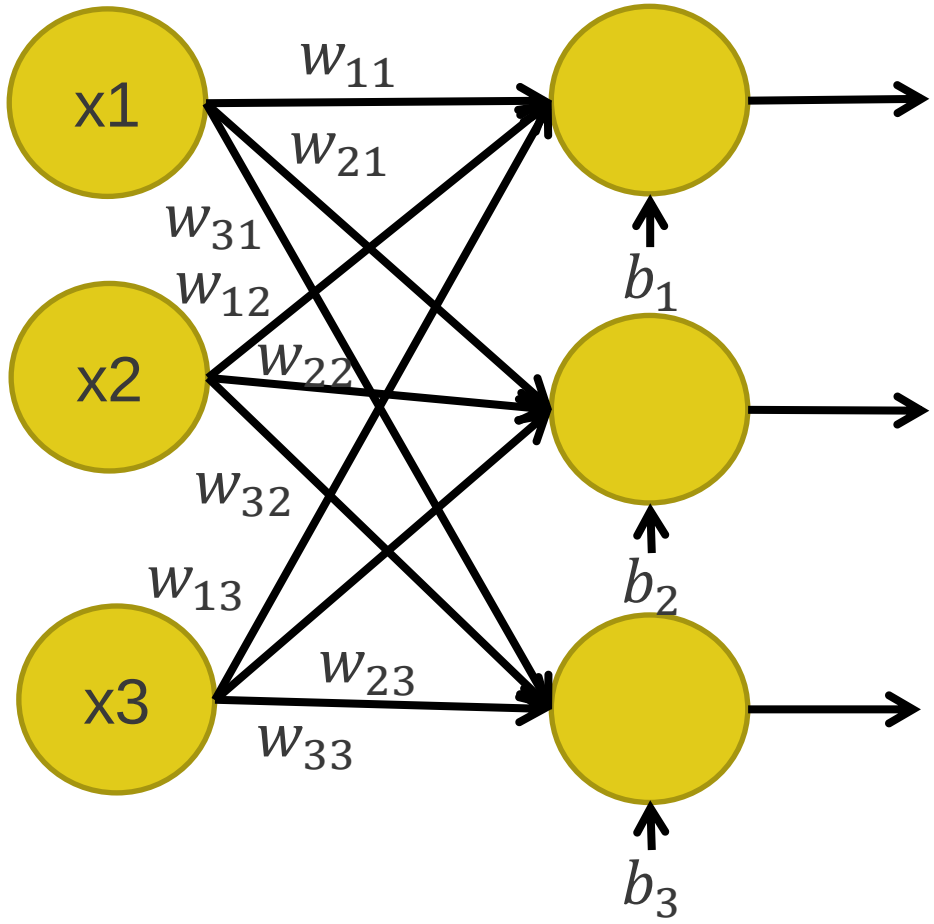
4-2: DNN神經網路建構

- 神經元運算複習
- DNN神經網路建構
- 激活函數



神經元運算複習

- 神經元的運算我們在之前的課程有提到，如果假設將三個神經元並聯在一起，其運算方法如下



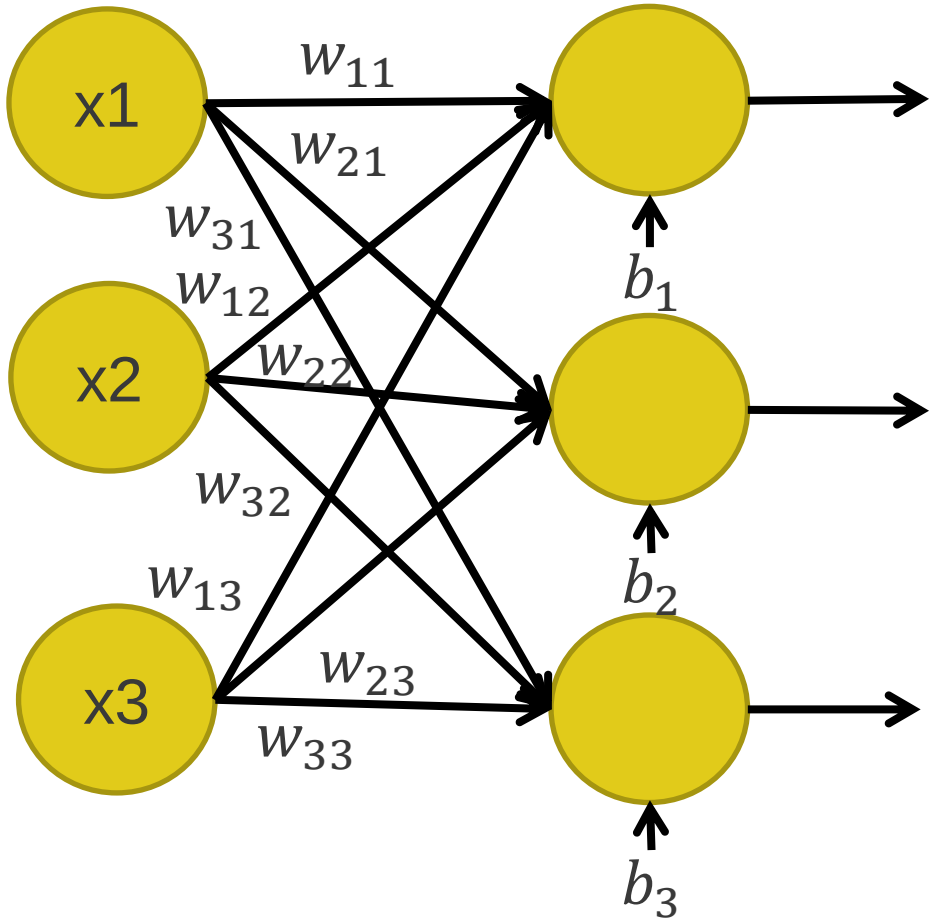
$$\sigma(x1 * w_{11} + x2 * w_{12} + x3 * w_{13} + b_1)$$

$$\sigma(\ x1 * w_{21} + x2 * w_{22} + x3 * w_{23} + b_2)$$

$$\sigma(x1 * w_{31} + x2 * w_{32} + x3 * w_{33} + b_3)$$

神經元運算複習

- 也有人將三個神經元並聯的運算方法表示成矩陣形式
  - 這樣表示比較精簡，也是現在主流文獻上所表示的式子



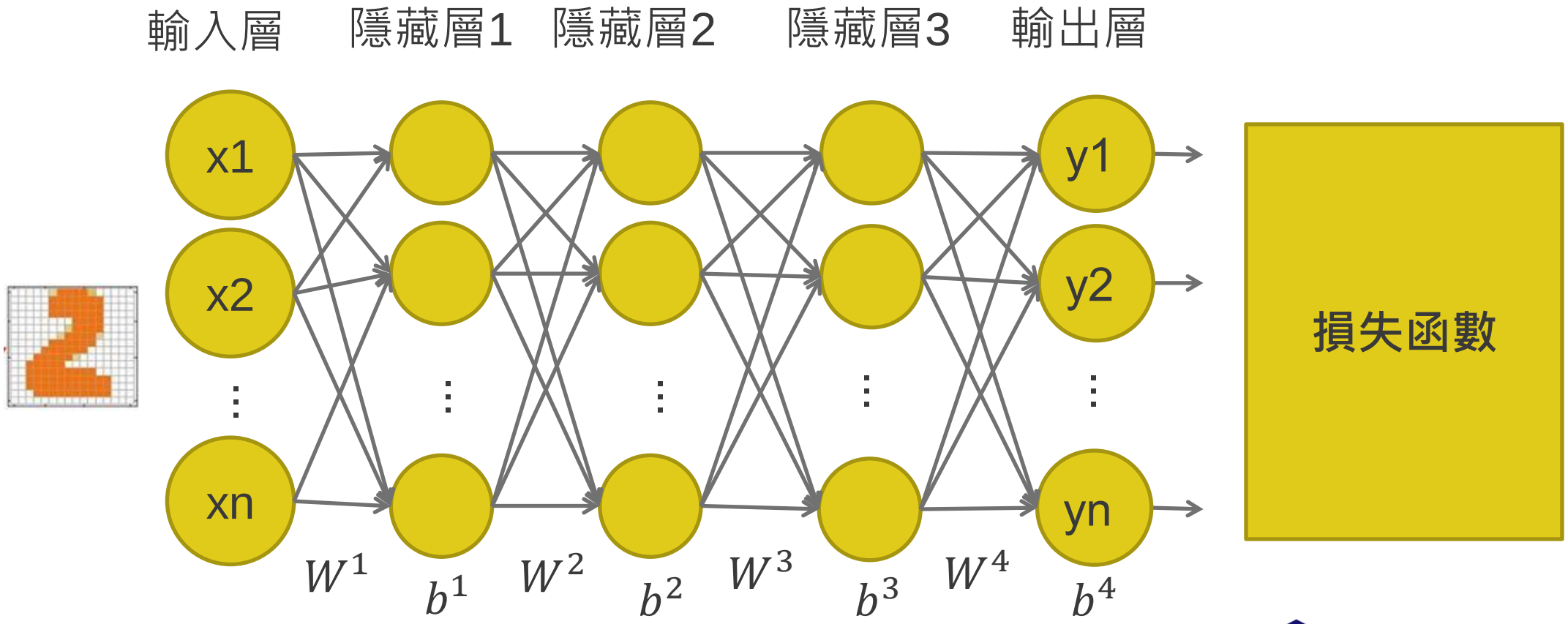
$$\sigma(W * X + b)$$

$$X = \begin{bmatrix} x1 \\ x2 \\ x3 \end{bmatrix} \quad W = \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \\ w_{31} & w_{32} & w_{33} \end{bmatrix}$$

$$b = \begin{bmatrix} b1 \\ b2 \\ b3 \end{bmatrix}$$

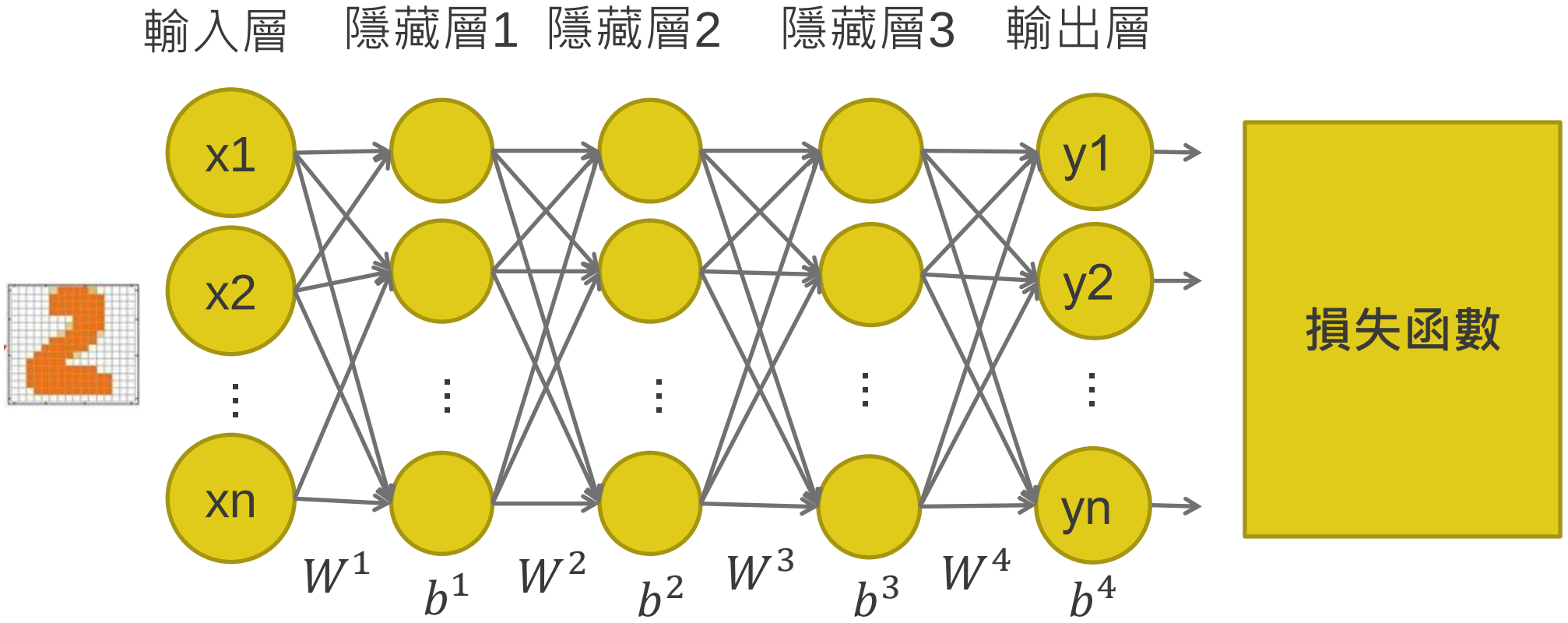
DNN神經網路建構

- 如果不斷重複一層又一層的神經元，我們就成其為DNN神經網路
  - DNN神經網路有時候又稱為”多重感知器”



DNN神經網路建構

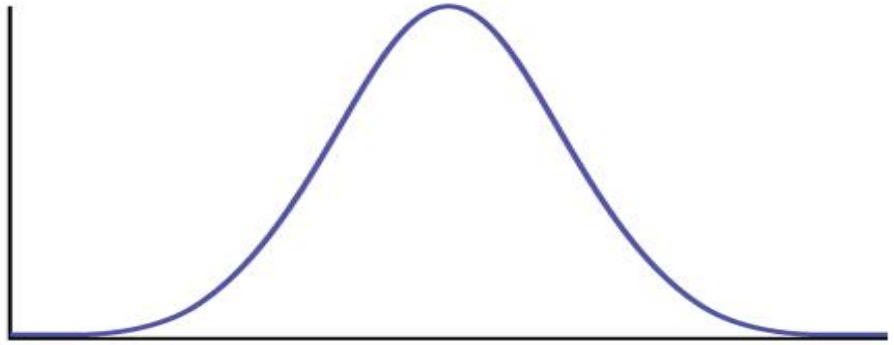
- **期望網路內的參數為W1, b1, W2, b2, ...**
  - **網路運算方式如下，其中W是Weight的縮寫、b是Bias的縮寫**

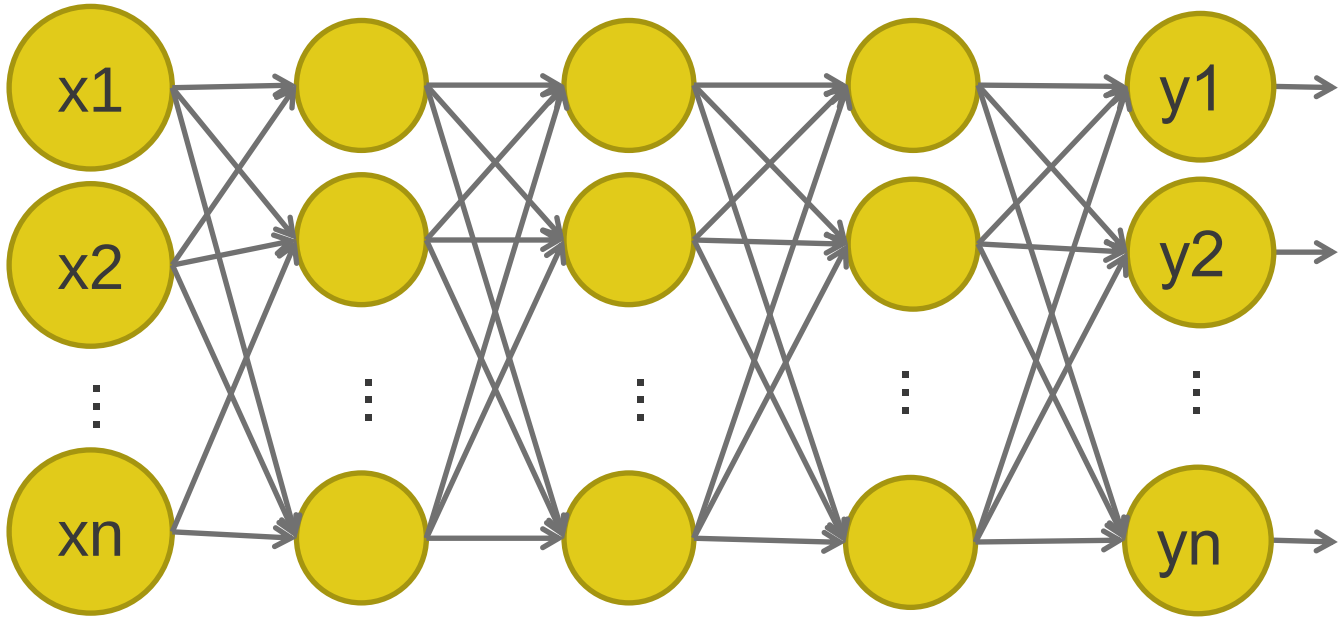


$$y = f(\theta) = \sigma(W^4 \ldots \sigma(W^2\sigma(W^1X + b^1) + b^2) \ldots + b^4)$$

DNN參數初始化

- 將DNN神經網路建立好後，需要先對所有參數進行初始化
  - 最簡單的方法為在常態分佈函數上，以0為平均、很小的標準差來隨機產生網路參數之初始值

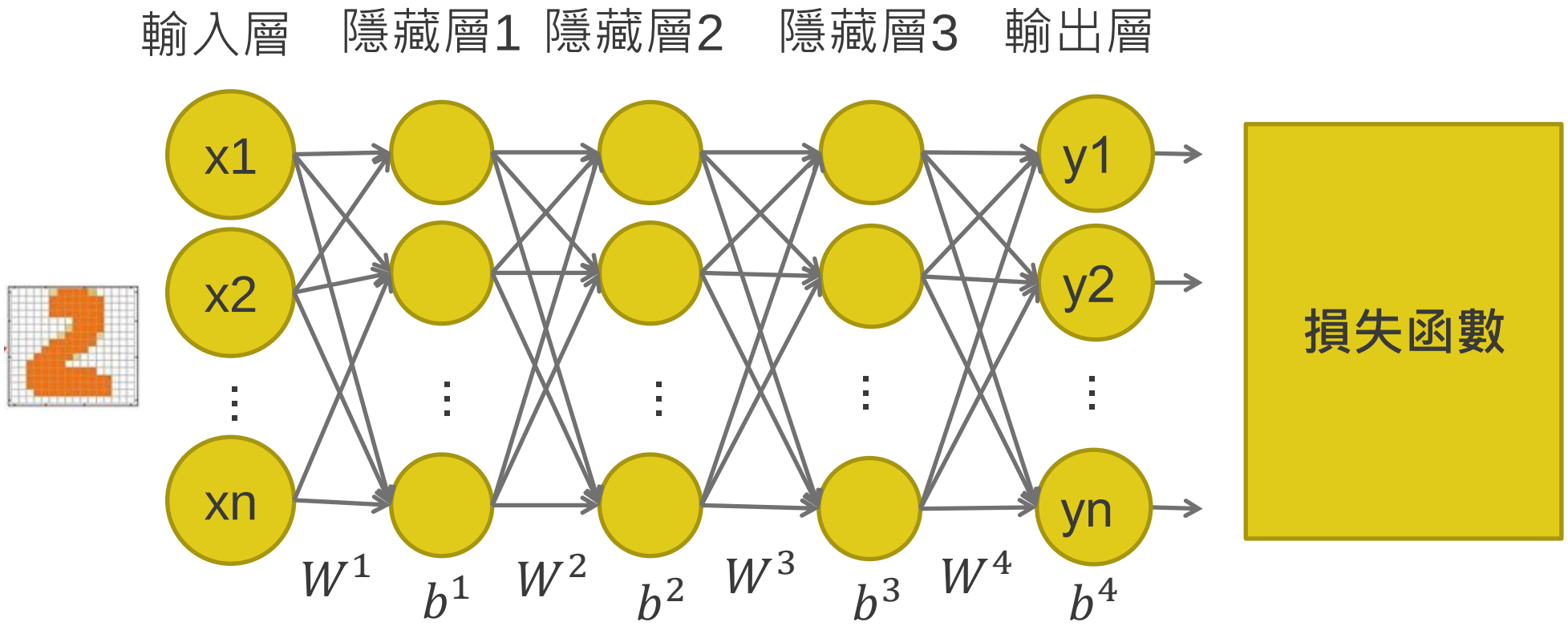




網路參數$\theta = \{W^1, b^1, W^2, b^2, \ldots\ldots\}$

激活函數

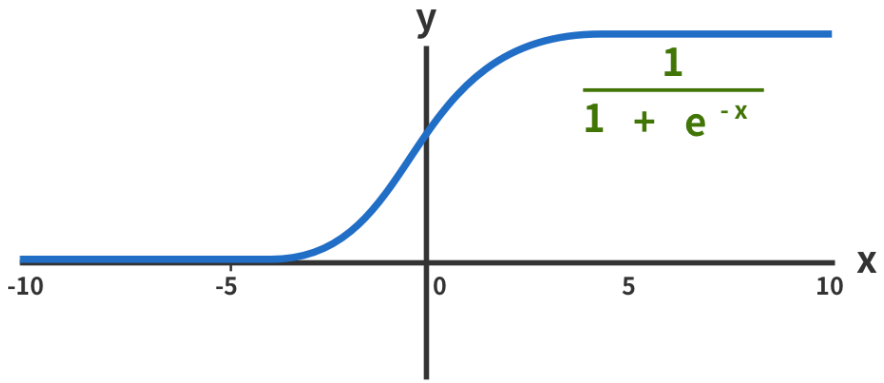
- 激活函數σ賦予神經網路非線性的能力
  - 可以視為是”開”或”關”



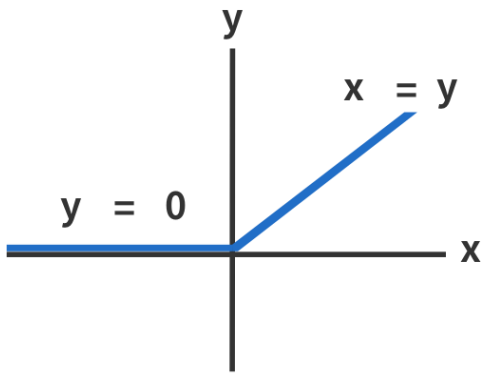
$$y = f(\theta) = \sigma(W^4 \ldots \sigma(W^2\sigma(W^1X + b^1) + b^2) \ldots + b^4)$$

激活函數

- 激活函數有非常多的選擇
  - 常見的有relu, sigmoid, elu, ……
  - 目前大家比較喜歡使用relu而非sigmoid函數



Sigmoid function



RELU

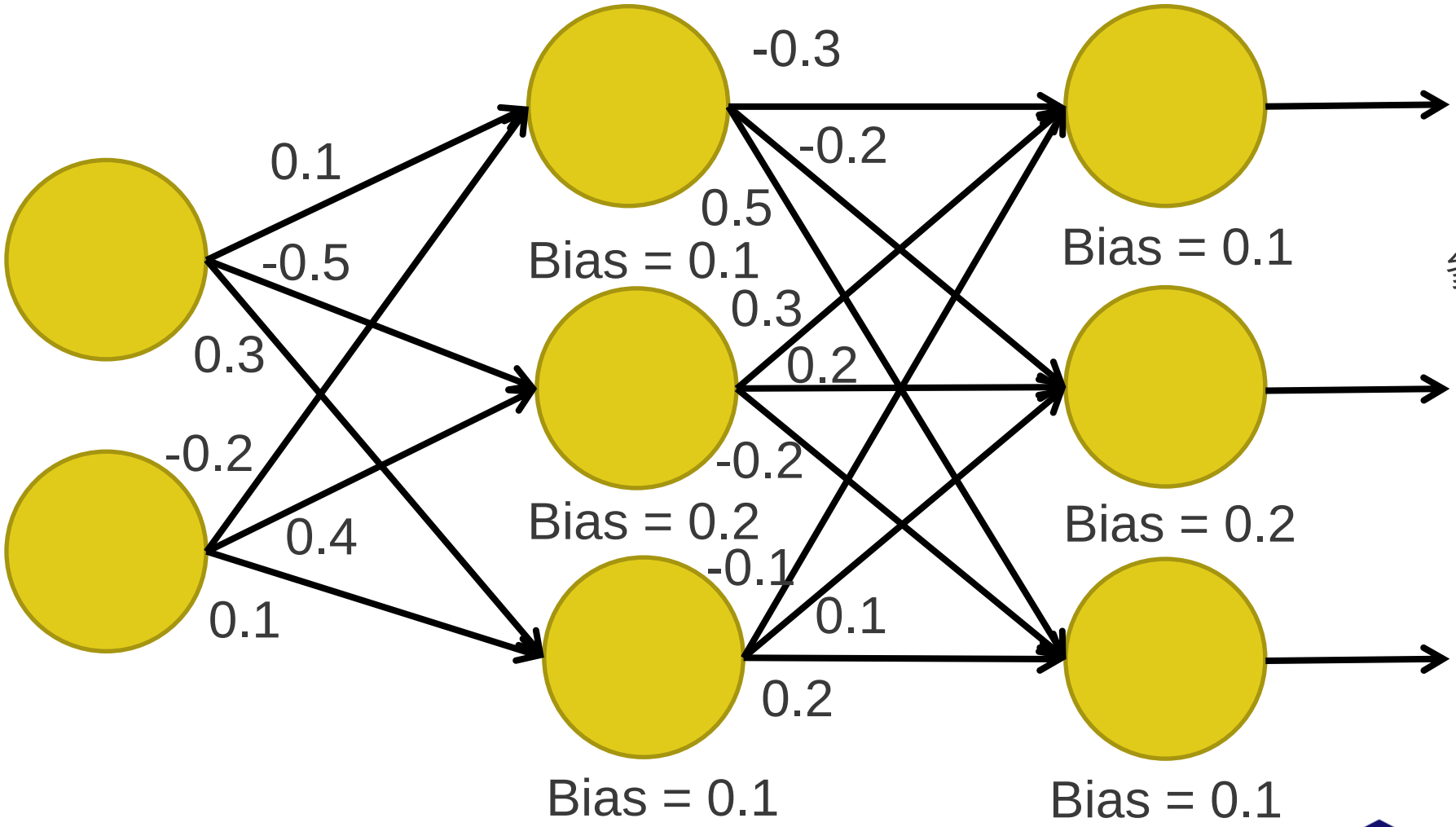
我們現在比較常用

4-3: DNN神經網路範例

- DNN神經網路數值範例

DNN神經網路數值範例

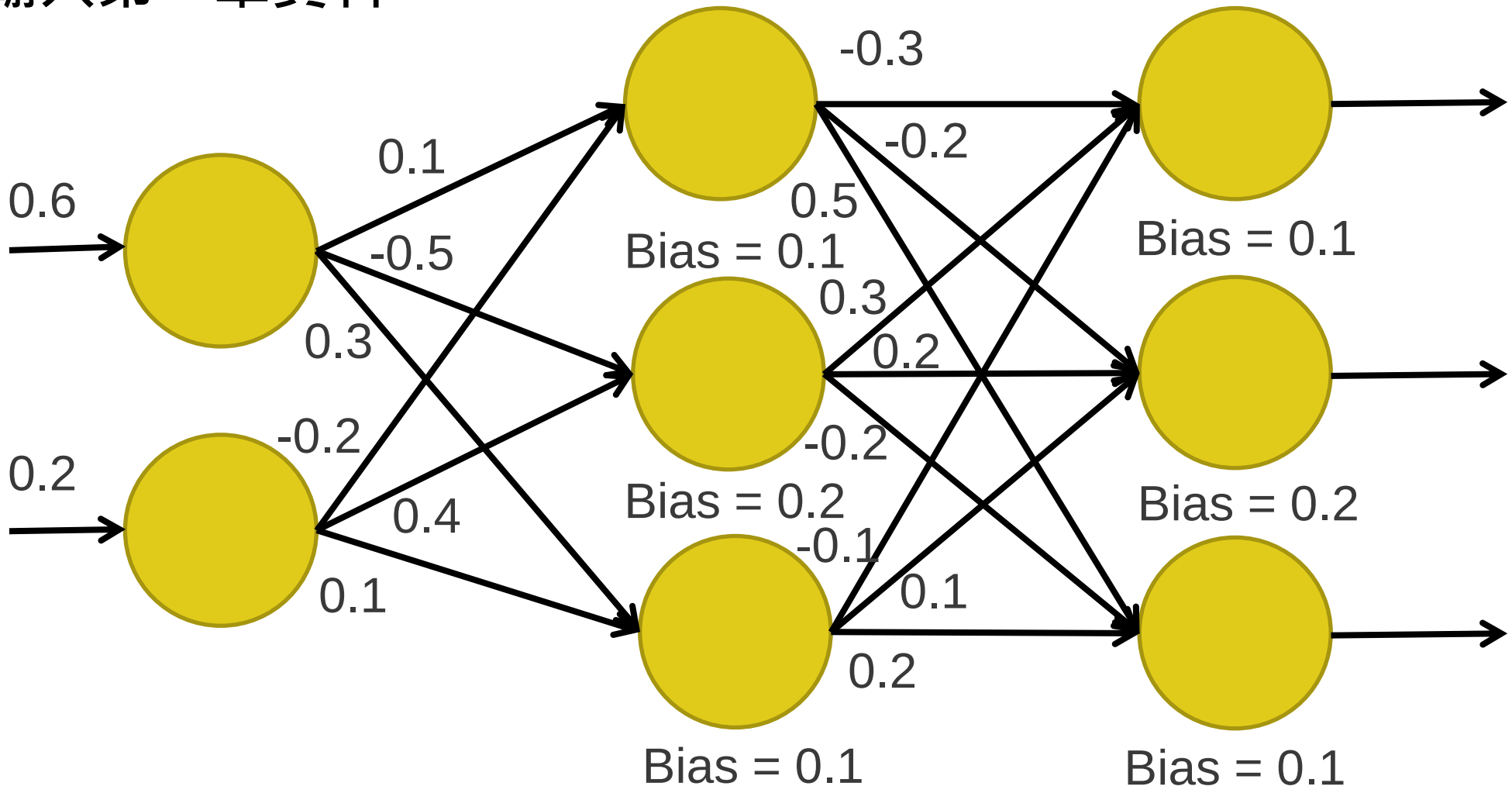
- 假設我們建立一個DNN神經網路並隨機初始化如下



參數個數 = 21

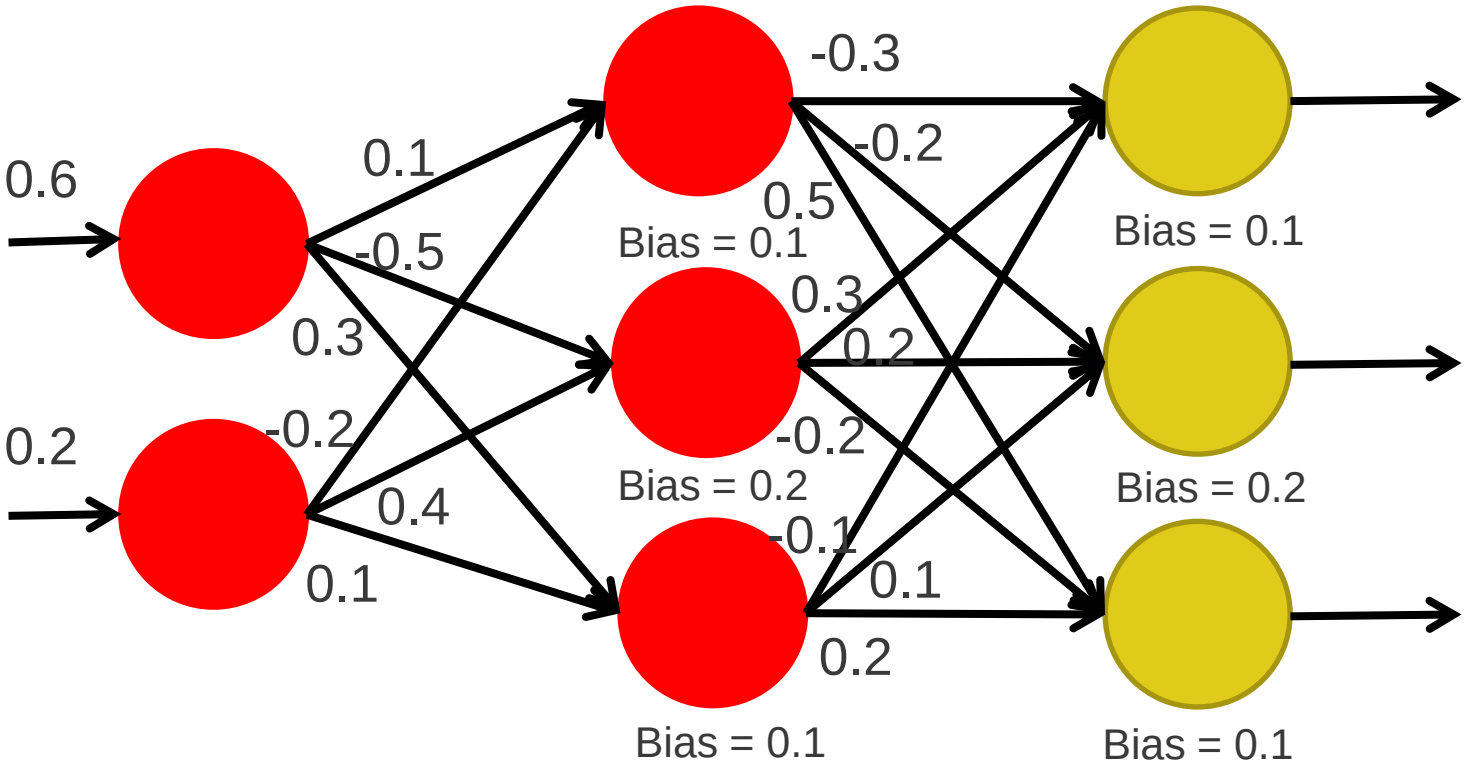
DNN神經網路數值範例

- 輸入第一筆資料



DNN神經網路數值範例

- 計算輸入層到隱藏層1(假設我們使用的激活函數為relu)



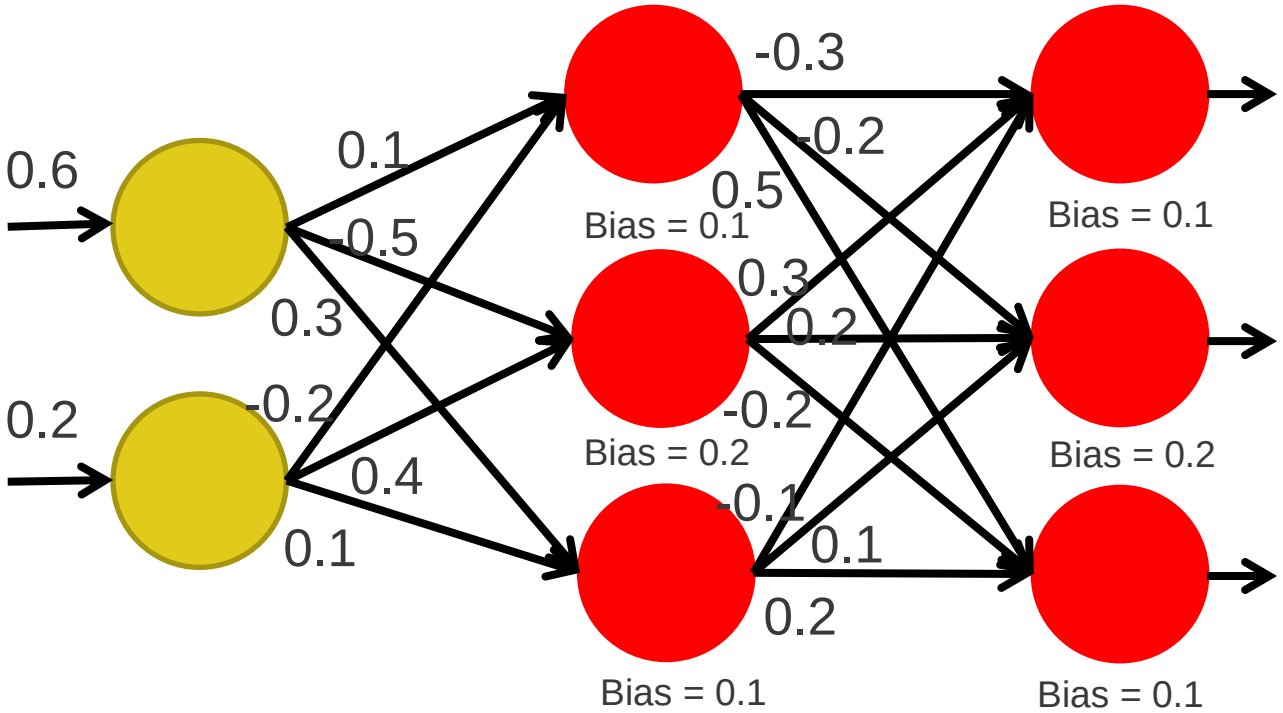
$$relu\left(\begin{bmatrix} 0.1 & -0.2 \\ -0.5 & 0.4 \\ 0.3 & 0.1 \end{bmatrix}\begin{bmatrix} 0.6 \\ 0.2 \end{bmatrix} + \begin{bmatrix} 0.1 \\ 0.2 \\ 0.1 \end{bmatrix}\right)$$

$$= relu\left(\begin{bmatrix} 0.12 \\ -0.02 \\ 0.3 \end{bmatrix}\right)$$

$$= \begin{bmatrix} 0.12 \\ 0 \\ 0.3 \end{bmatrix}$$

DNN神經網路數值範例

- 計算隱藏層1到輸出層(假設我們使用的激活函數為relu)



$$relu\left(\begin{bmatrix}-0.3 & 0.3 & -0.1\\ -0.2 & 0.2 & 0.1\\ 0.5 & -0.2 & 0.2\end{bmatrix}\begin{bmatrix}0.12\\ 0\\ 0.3\end{bmatrix}+\begin{bmatrix}0.1\\ 0.2\\ 0.1\end{bmatrix}\right)$$

$$= relu\left(\begin{bmatrix}-0.066\\ 0.006\\ 0.12\end{bmatrix}\right)$$

$$= \begin{bmatrix}0\\ 0.006\\ 0.12\end{bmatrix}$$

Demo 4-3

- 開啟Demo_4-3.ipynb
- MNIST資料集介紹
- 建立簡單模型
- 建立DNN網路模型

線上Corelab

- 題目1：請依照以下提示建立兩個tf.constant的矩陣，並輸出相乘結果
- 題目2：請將MNIST資料集讀取進來，取出第一筆資料印出，並以簡單判斷式判斷該資料為哪個數字
- 題目3：請建立一個5層的DNN網路，各層的神經元數量請自行給定，並將MNIST代入

本章重點精華回顧

- 深度學習三大步驟
- DNN神經網路的運算方法
- DNN網路的參數
- 激活函數介紹

Lab:DNN神經網路介紹

- **Lab01:MNIST資料集讀取**
- **Lab02:建立簡單模型**
- **Lab03:建立神經網路模型**

Estimated time:
20 minutes


THANK YOU