圖解神經網路－多層感知器(MLP)

吳庭育

tyw429@gmail.com

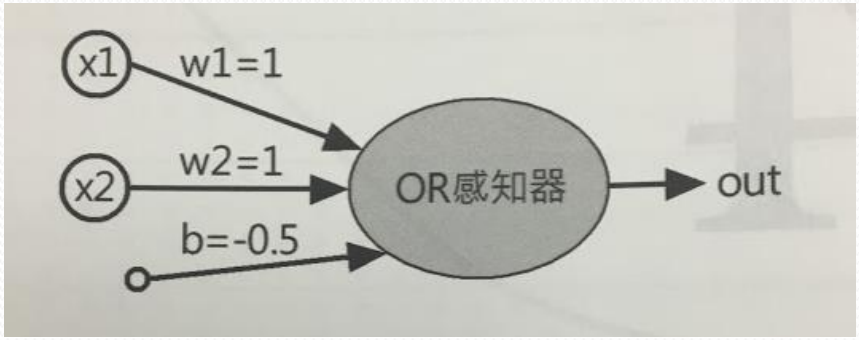
Outline

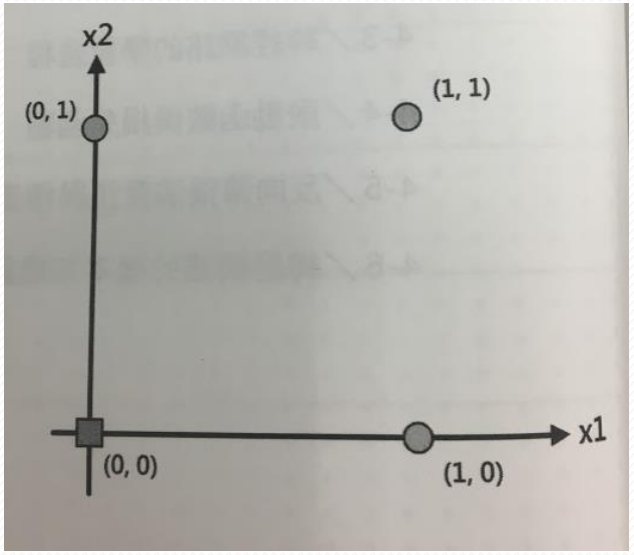
- 線性不可分問題
- 認識多層感知器(MLP)
- 神經網路的學習過程－正向與反向傳播
- 啟動函數與損失函數
- 反向傳播演算法與梯度下降法
- 神經網路的樣本和標籤資料

線性不可分問題

線性不可分問題 (1/2)

- 線性可分的問題，如下圖所示：

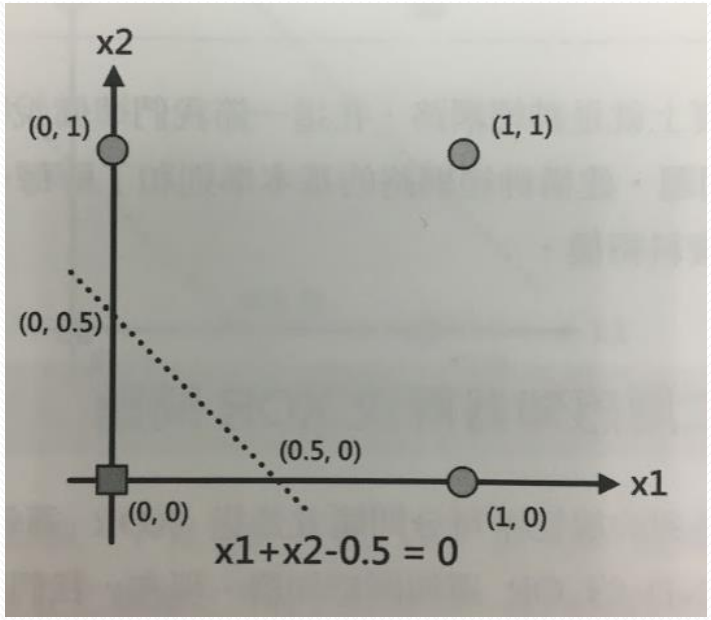




x1	x2	out
0	0	0
0	1	1
1	0	1
1	1	1

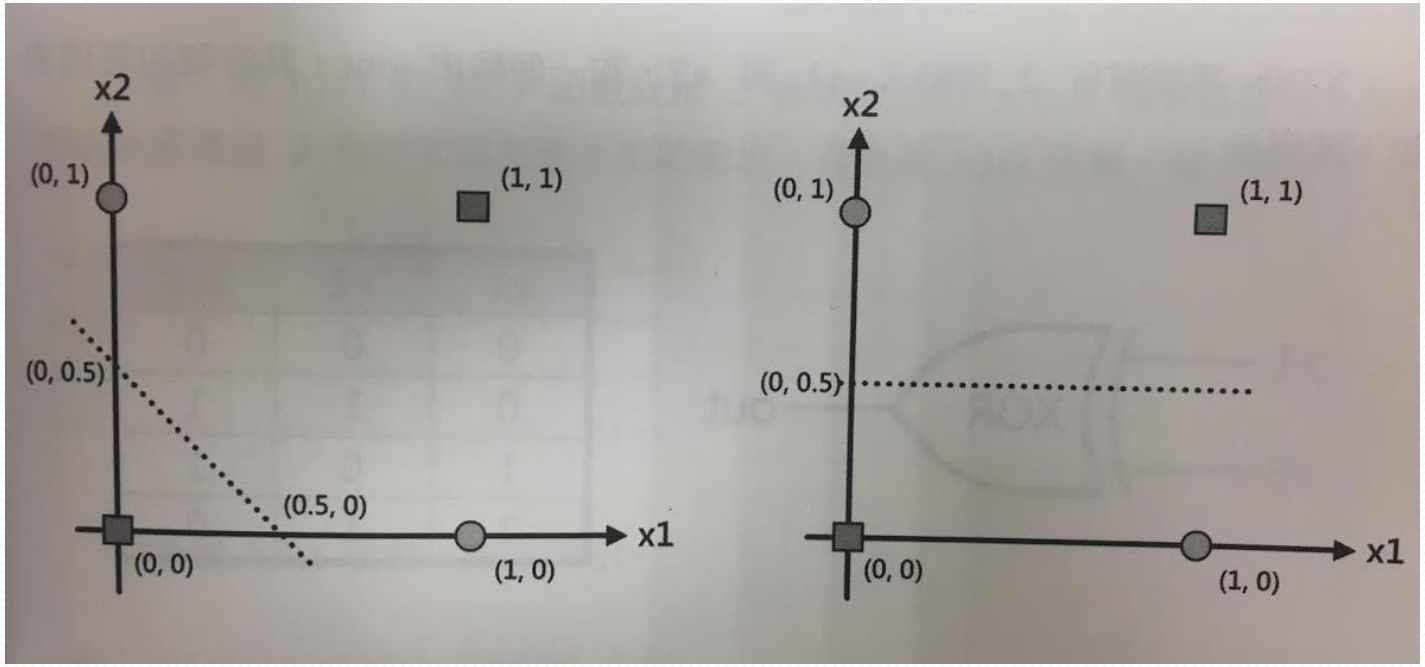
A

B



線性不可分問題 (2/2)

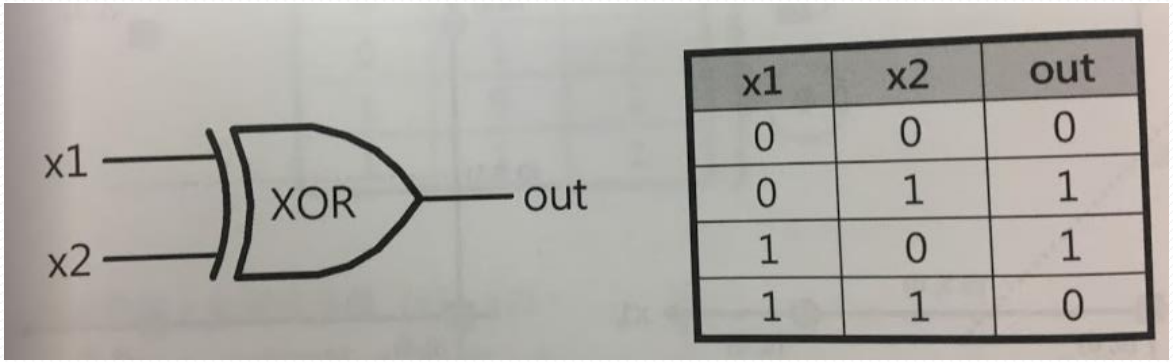
- 線性不可分的問題就是無法解決不能用一條線將資料分成兩類的問題，如下圖：



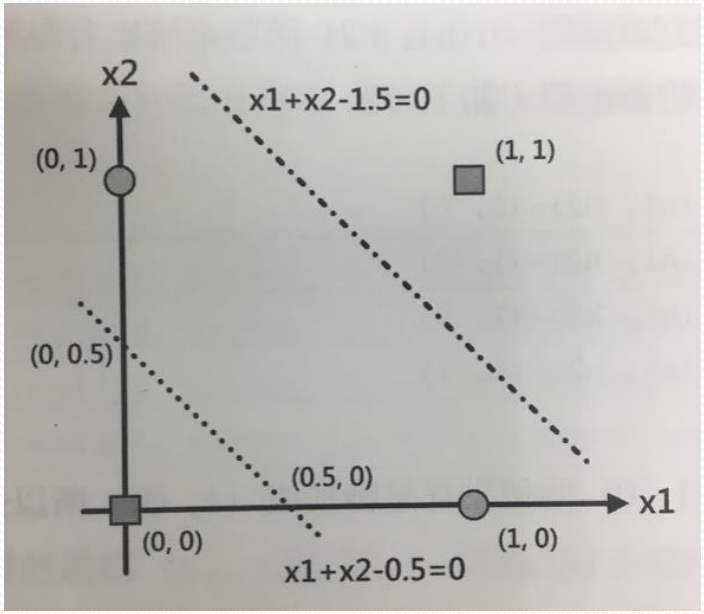
認識多層感知器(MLP)

使用二層感知器解決XOR問題 (1/2)

- 感知器範例：XOR邏輯器



x1	x2	out
0	0	0
0	1	1
1	0	1
1	1	0

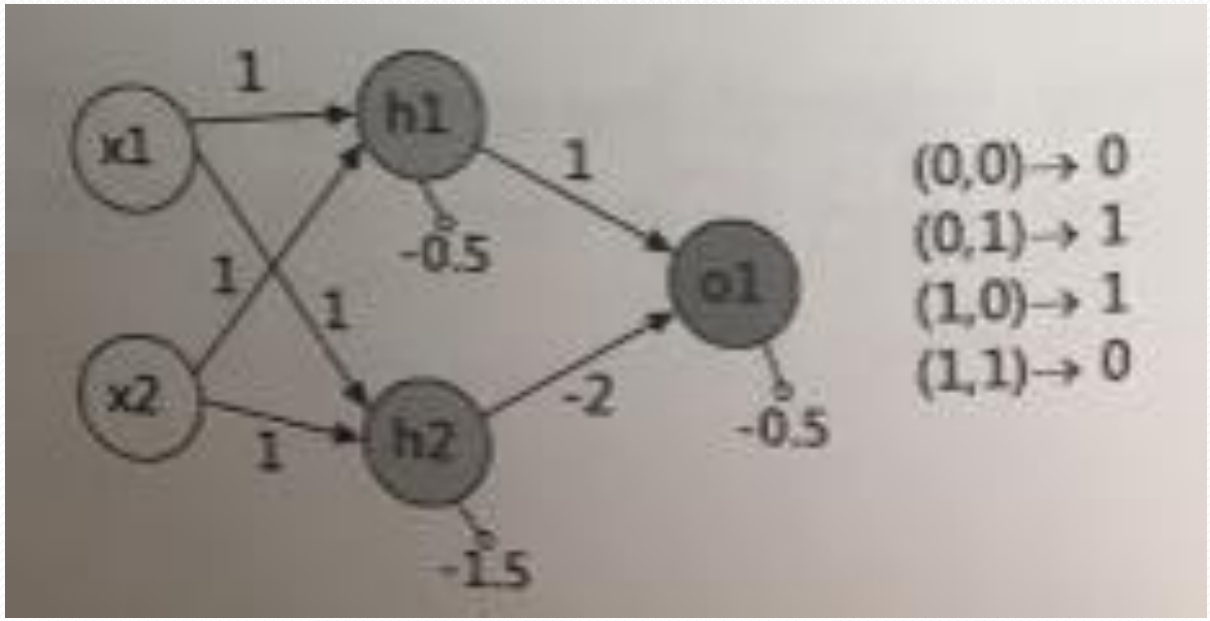


h1(x1, x2)=x1+x2-0.5

h2(x1, x2)=x1+x2-1.5

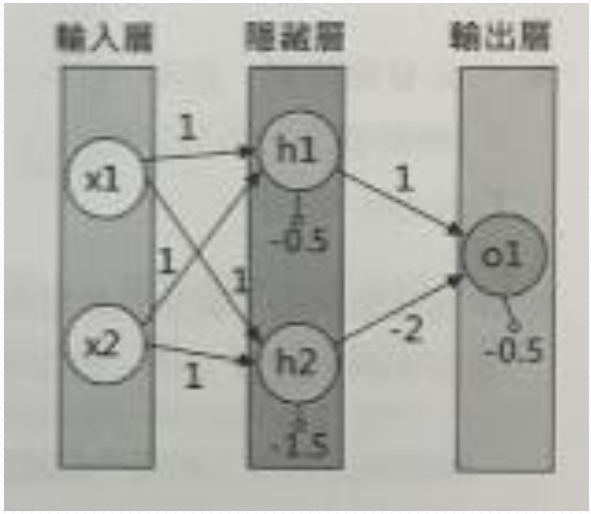
使用二層感知器解決XOR問題 (2/2)

- 將2個感知器的輸出作為下一層感知器的輸入，如下圖：

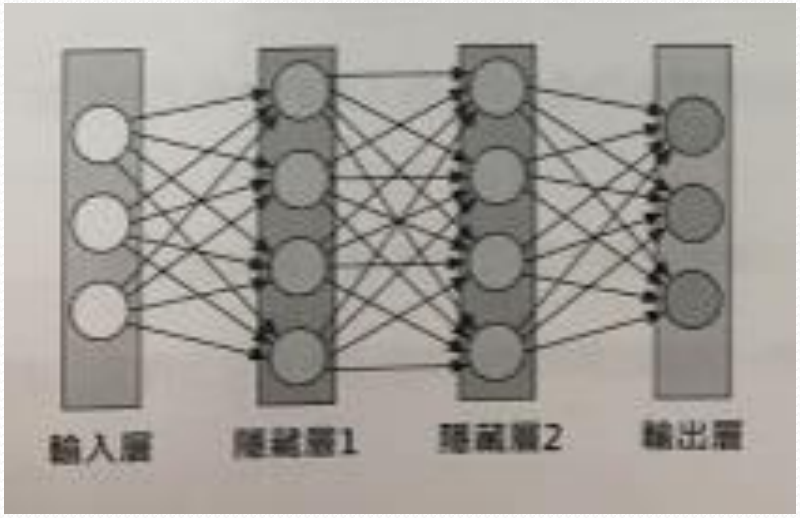


多層感知器就是神經網路

- 3層神經網路如右圖：

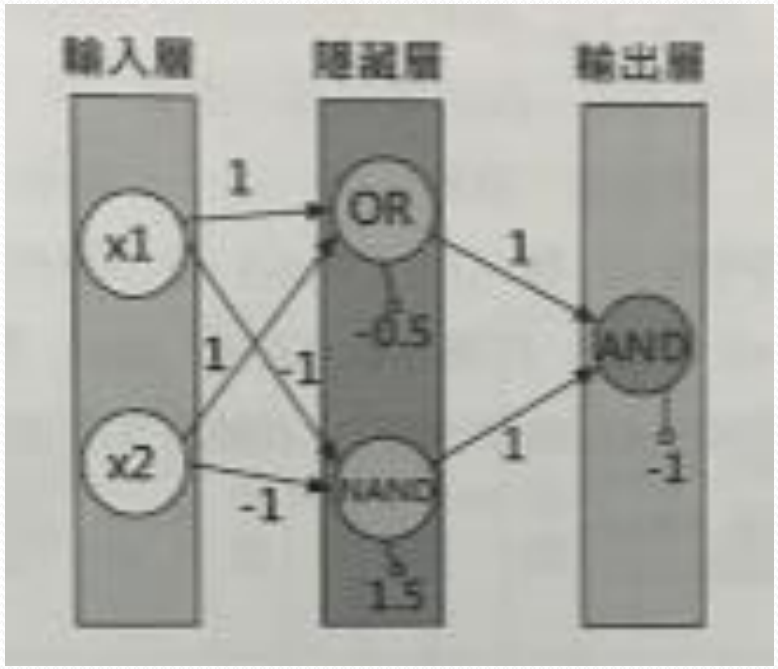


- 深度神經網路
  - 如果多層感知器有2層隱藏層共4層神經網路，就是深度神經網路，即深度學習，如下圖：



神經網路的權重與偏向量

- 結合OR、NAND和AND邏輯匣來建立XOR二層感知器，如下圖：



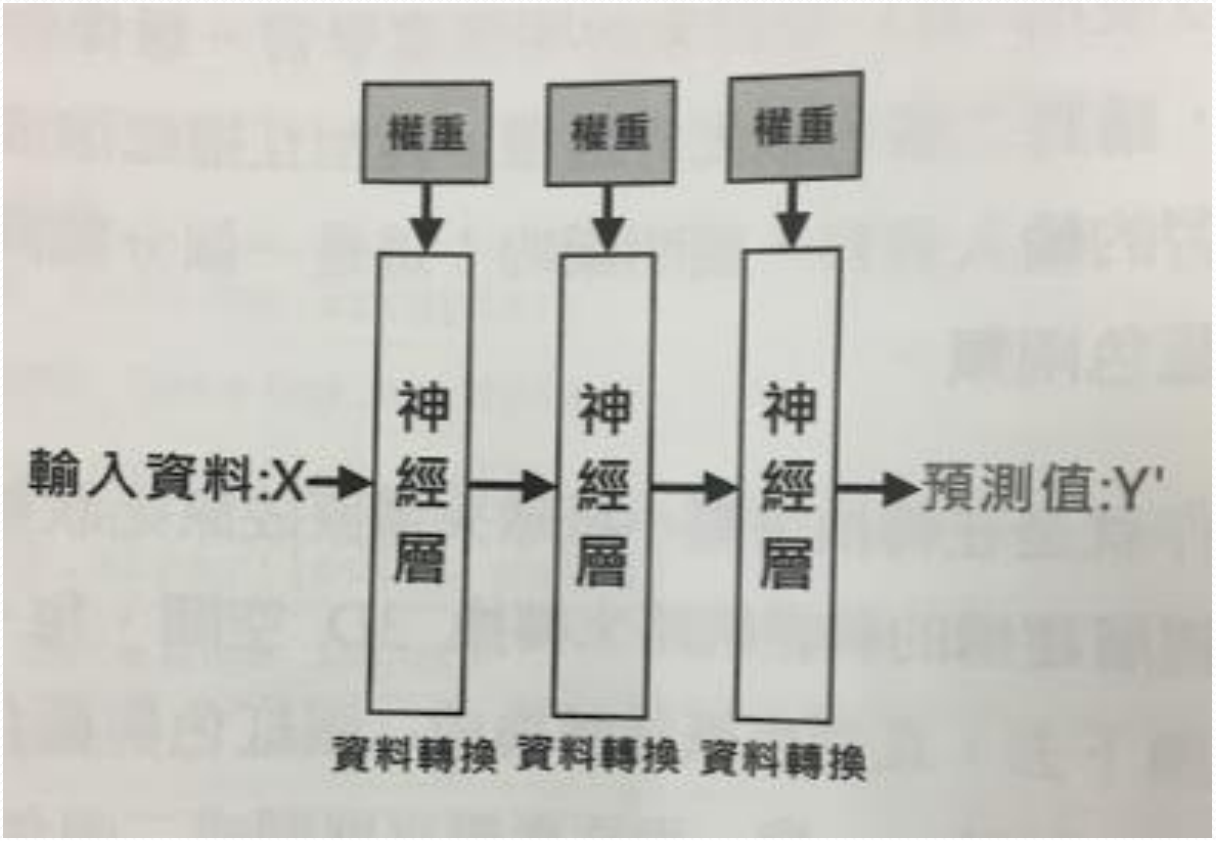
深度學習的幾何解釋

- 使用 3D 空間說明深度學習，神經網路的工作就是在轉換這顆小紙球來還原成完使狀態的兩張色紙，如下圖：

神經網路的學習過程－正向與反向傳播

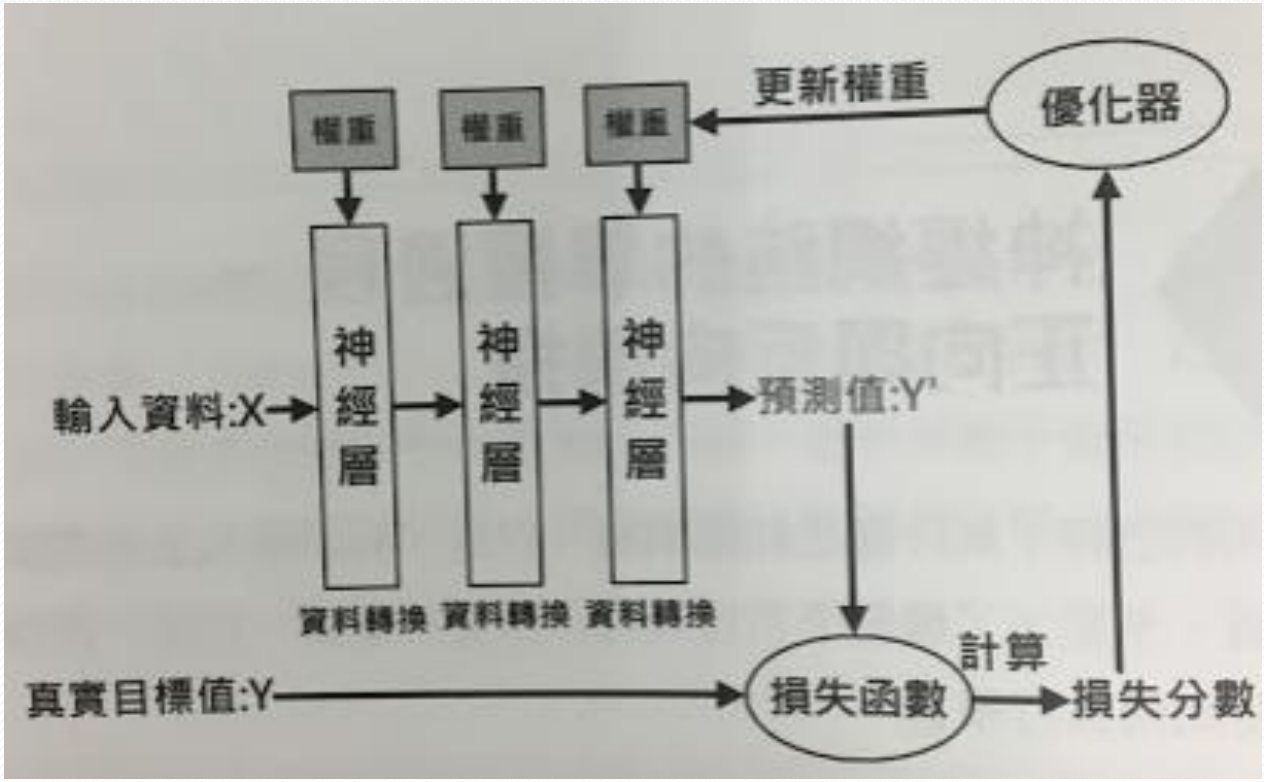
神經網路的學習方式與學習目標 (1/2)

- 神經網路的學習目標就是找出正確的權重值來縮小**損失**，為了方便說明，這裡談到的權重，預設都包含偏向量，這些權重也稱為神經網路的參數，如下圖：



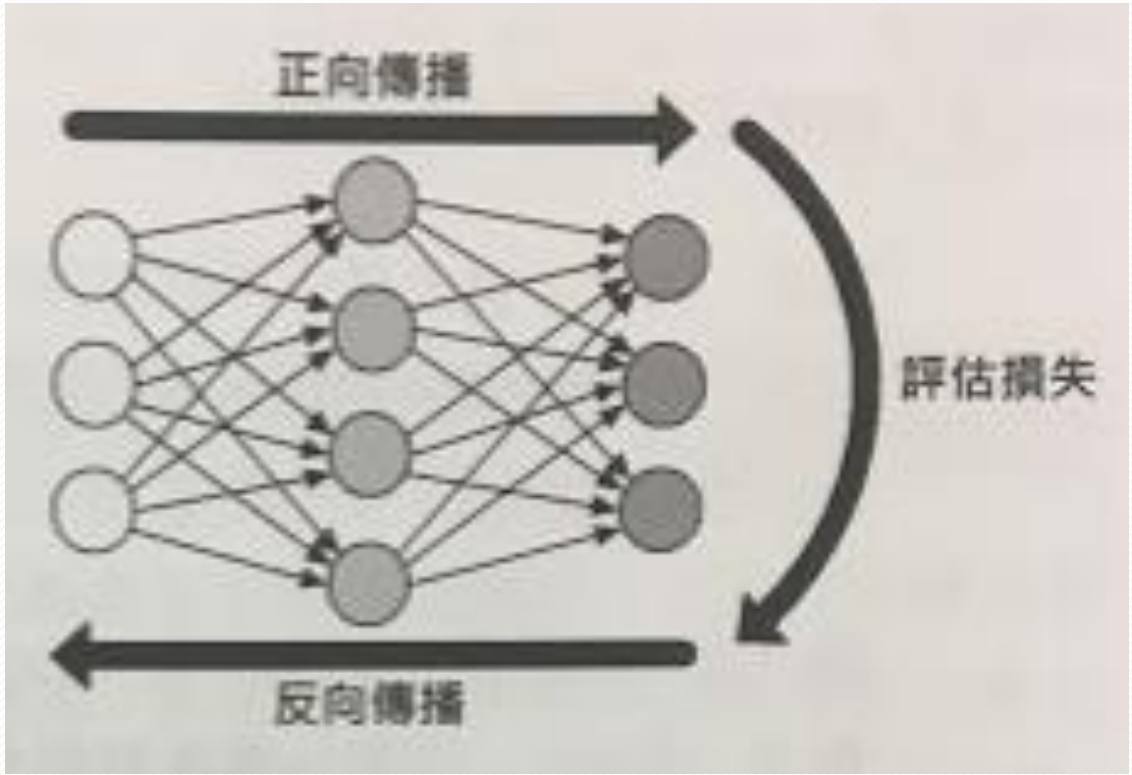
神經網路的學習方式與學習目標 (2/2)

- 下圖就是神經網路的學習方式，學習目標是找出更好的權重來盡量減少損失分數，其最佳結果是讓預測值符合目標值



神經網路的訓練迴圈 (1/2)

- 神經網路可以自行使用資料來自我訓練神經網路，這個訓練步驟不是只跑一次，而是一個**訓練迴圈**，其需要重複輸入資料來訓練很多次，也稱為迭代，如下圖：

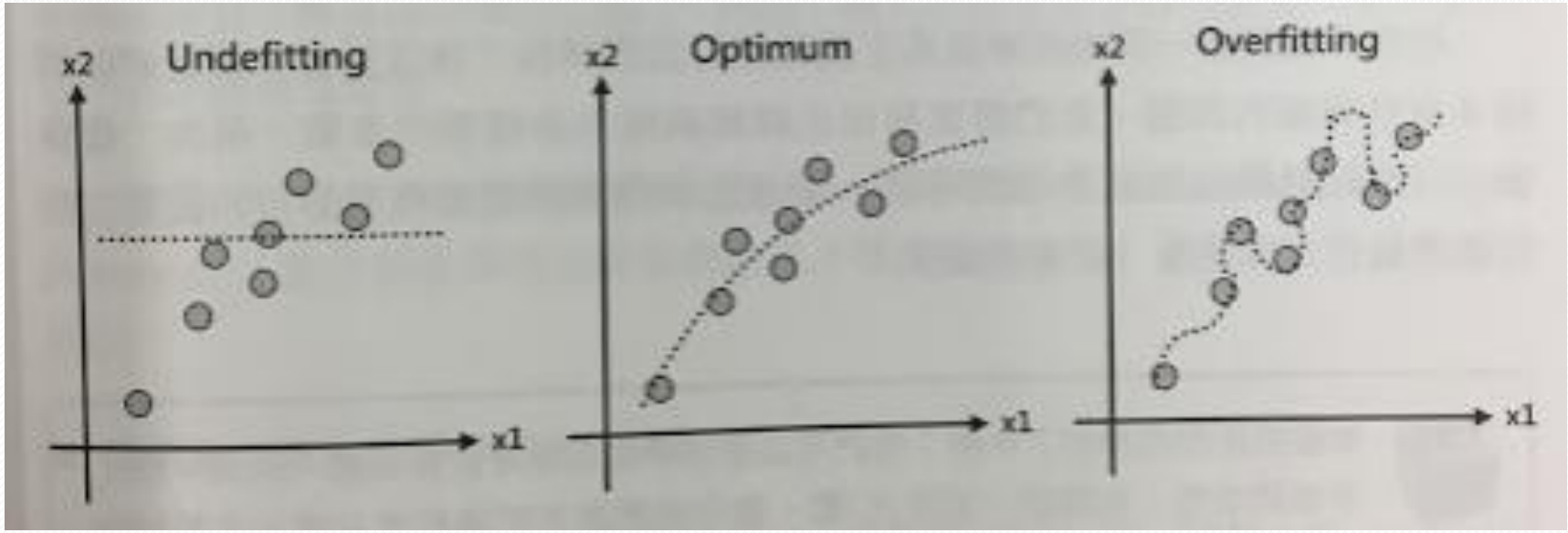


神經網路的訓練迴圈 (2/2)

- 整個訓練迴圈的步驟都是圍繞著權重的初始、使用和更新操作，如下圖：

神經網路到底學到了什麼

- 在我們訓練神經網路時，並不是跑越多次訓練迴圈就能訓練出更加的預測模型，隨著訓練迴圈次數增加，神經網路更新權重的數量和次數也會增加，整個神經網路學習曲線會從「低度擬合」、「最佳化」，最後到「過度擬合」，如下圖：



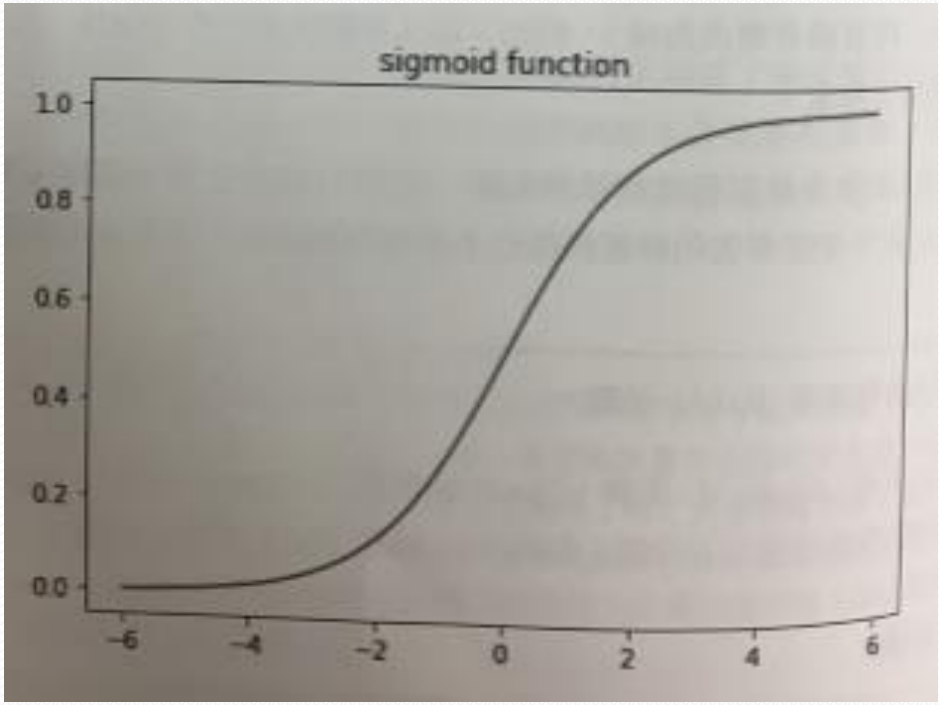
啟動函數與損失函數

啟動函數 Activation Function (1/)

- 傳統感知器其啟動函數是使用梯度函數(Step Function)，當今深度學習的神經網路已不再使用梯度函數，常用的啟動函數如下：
  - 隱藏層：最常使用ReLU函數。
  - 輸出層：使用Sigmoid 和 Tanh 函數和Softmax函數。

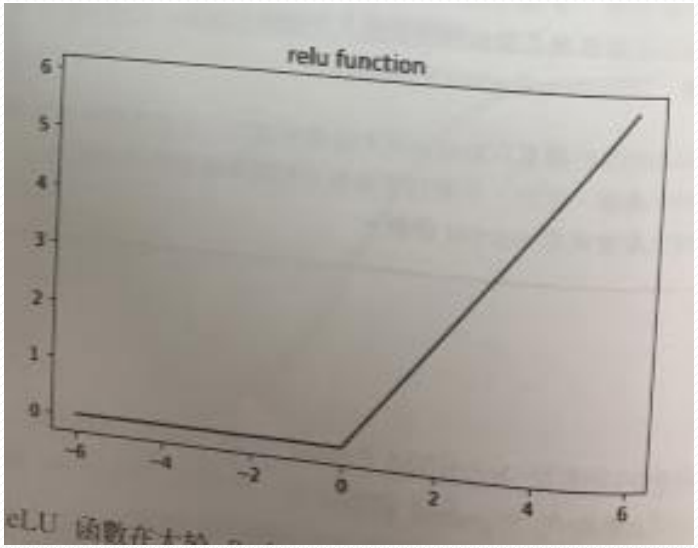
- Sigmoid函數

$$f(x)=\frac{1}{(1+e^{-x})}$$



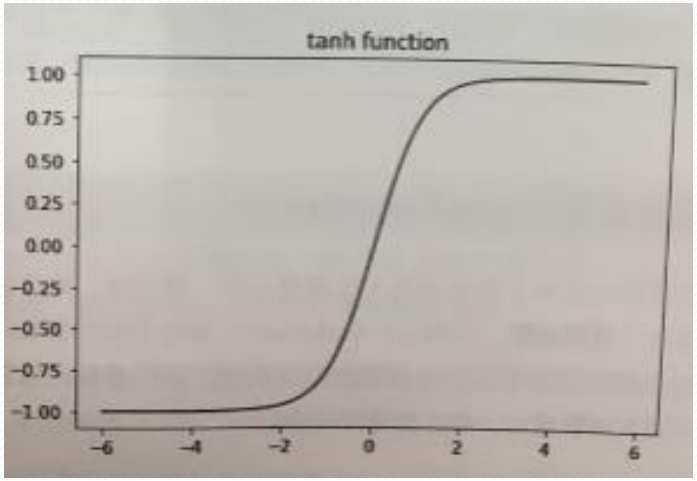
啟動函數 Activation Function (2/)

- ReLU函數
  - 公式： f(x) = max(0,x)



- Tanh 函數
  - 公式：

$$f(x) = \frac{\sinh(x)}{\cosh(x)}$$



損失函數 Loss Function (1/)

- 均方誤差
  - 公式如右圖：

$$E = \frac{1}{2}\sum_{n}(y_n - t_n)^2$$

- 資訊量：資訊的量化值，其值的大小和事件發生機率有關，公式如下：

$$H(X_i) = -\log_2 P$$

- 資訊熵：主要目的是量化資訊的混亂程度，公式如下：

$$H(X) = -\sum_{x} P(x)\log_2 P(x)$$

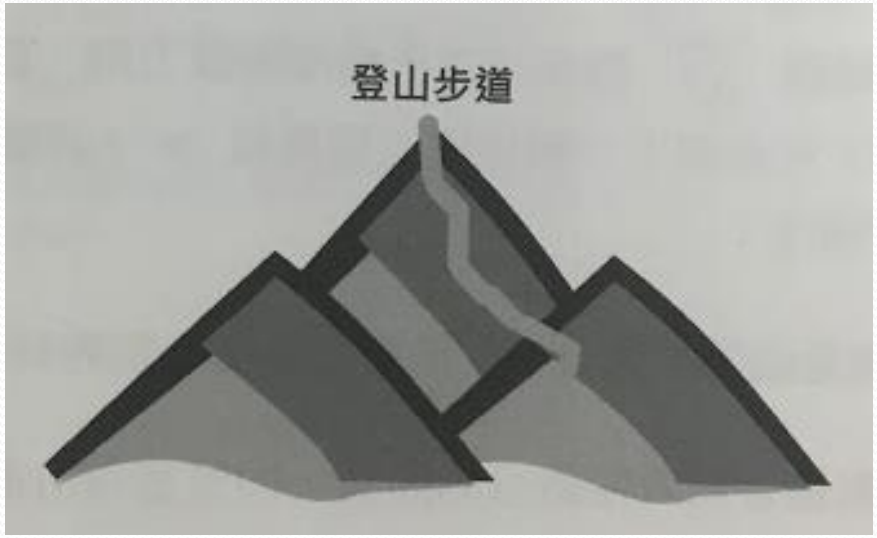
- 交叉熵：是使用資訊熵來評估2組機率向量之間的差異程度，當交叉熵越小，就表示2組機率向量越接近，公式如下：

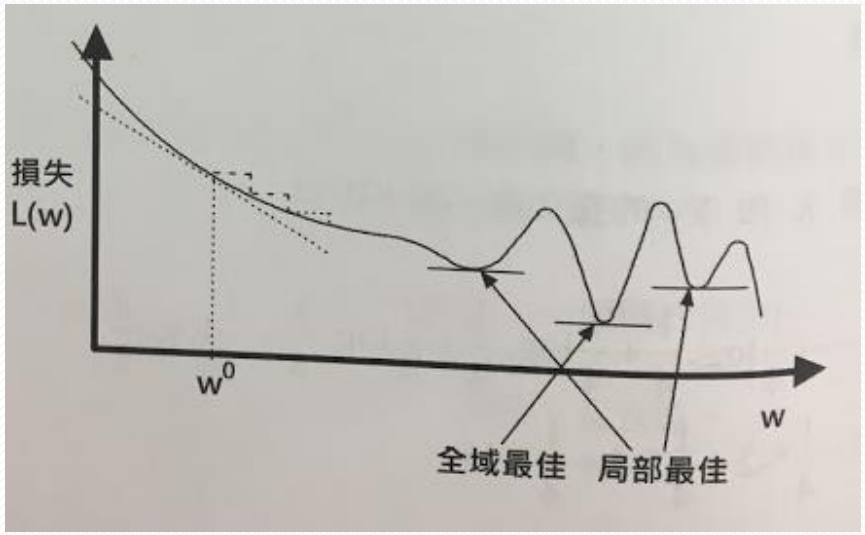
$$H(X,Y) = -\sum_{i=1}^{n} P(x_i)\log_2 P(y_i)$$

反向傳播演算法與梯度下降法

梯度下降法 Gradient Descent

- 神經網路是使用優化器來更新神經網路的權重，優化器是使用反向傳播計算出每層權重需要分擔損失的梯度，然後再使用梯度下降法更新神經網路每一個神經層的權重。

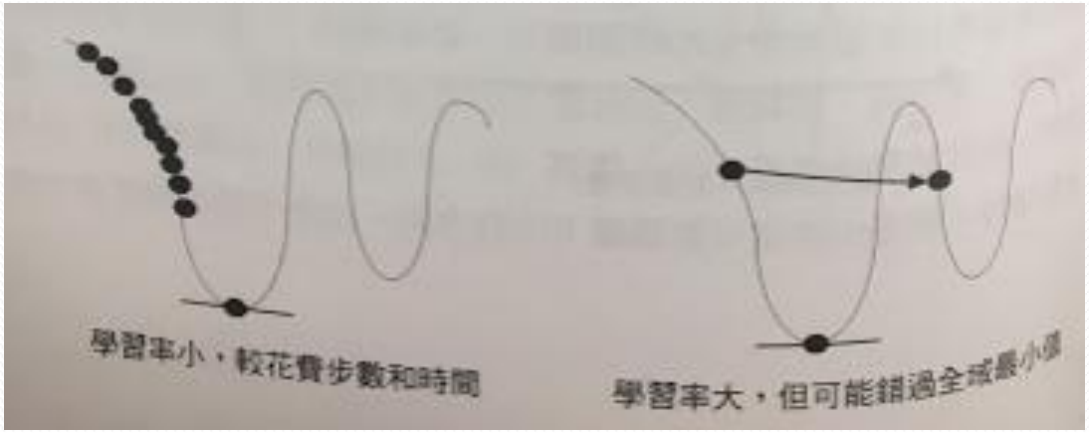




梯度下降法的數學公式

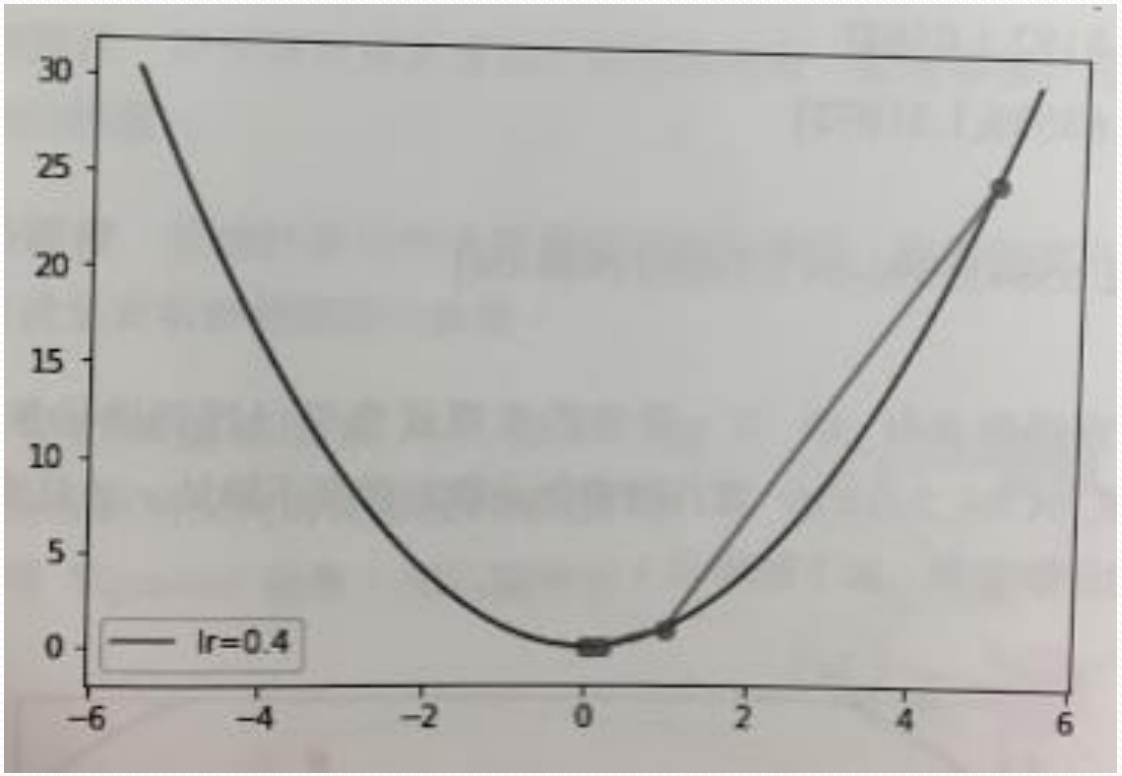
- 梯度下降法公式

$$w^1 = w^0 - \alpha \frac{\partial L(w)}{\partial w^0}$$

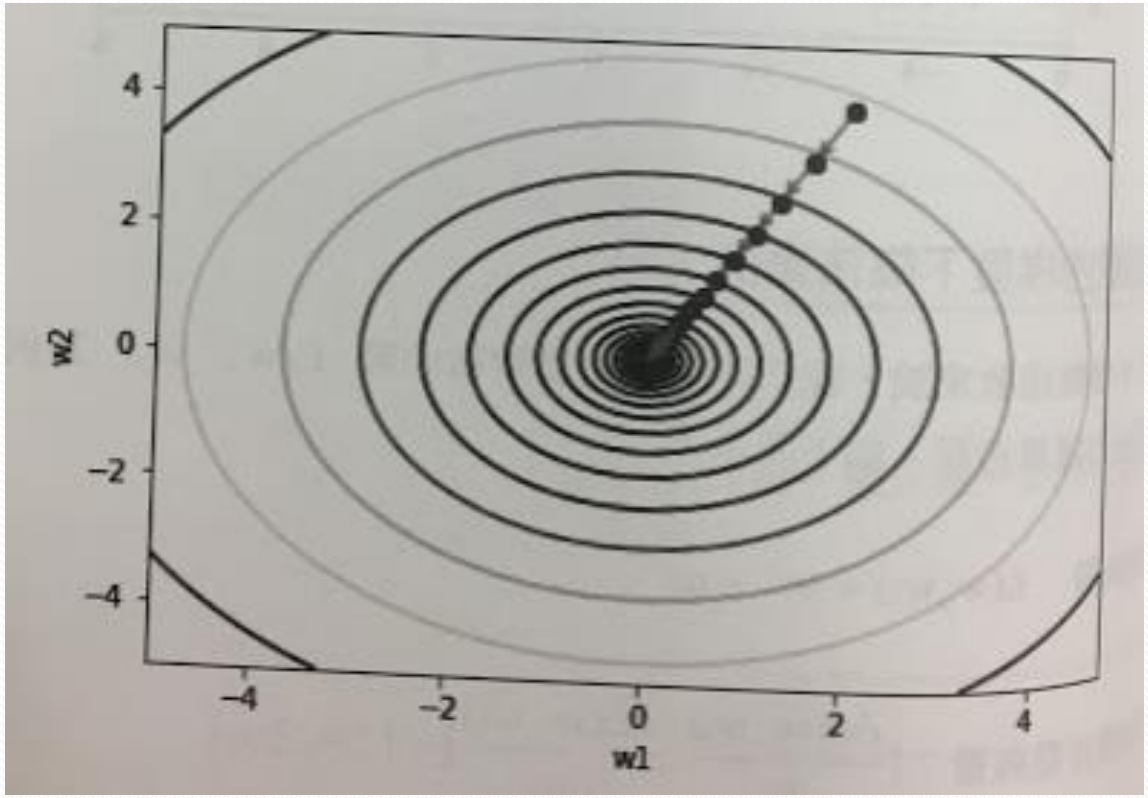


單變數函數的梯度下降法實例

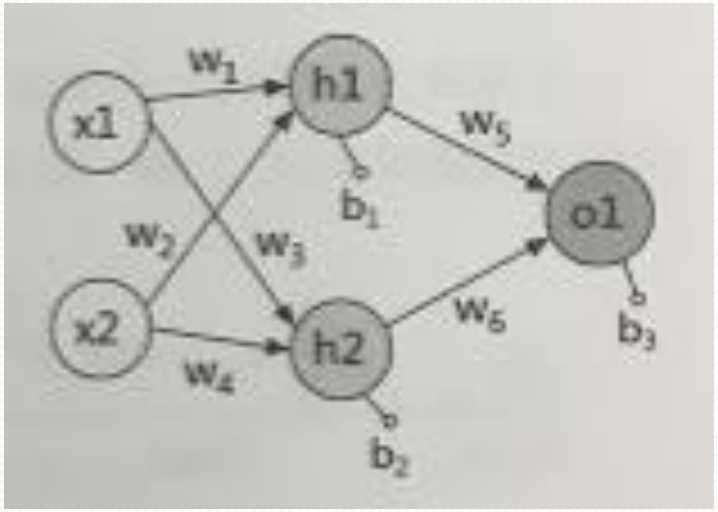
- 單變數函數：$L(w) = w^2$
- 函數的微分：$\frac{\partial L(w)}{\partial w} = 2w$

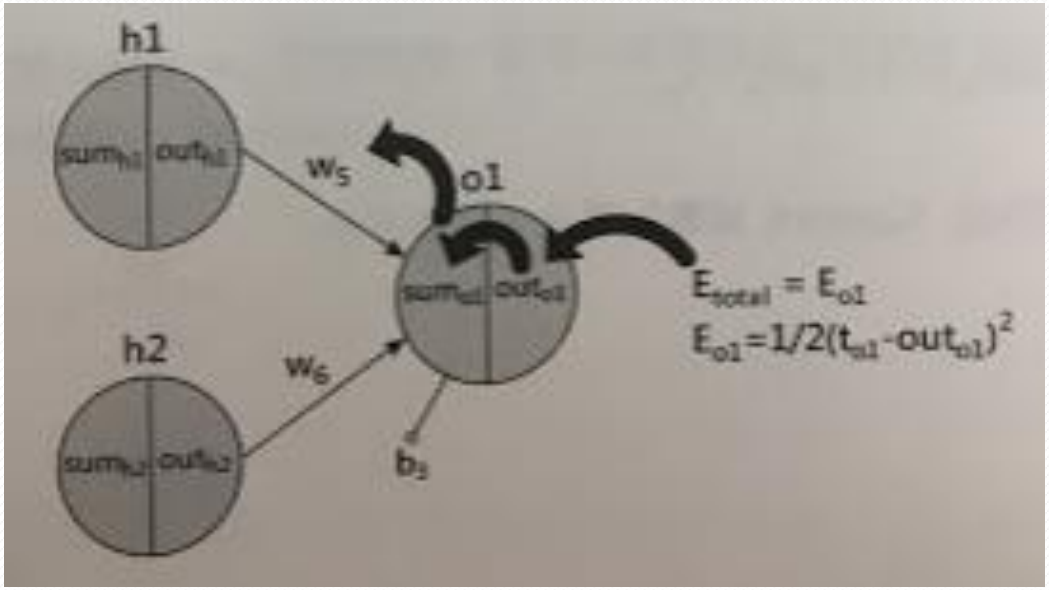


多變數函數的梯度下降法實例



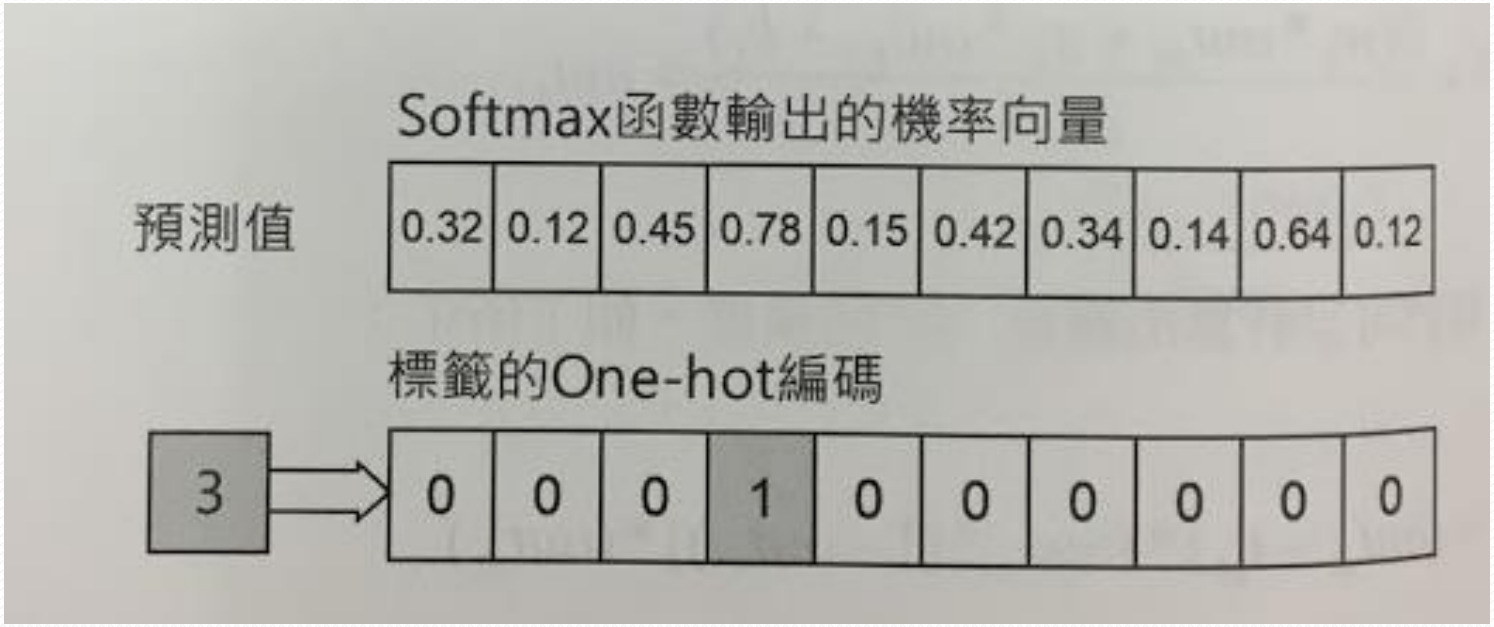
反向傳播演算法 Backpropagation





神經網路的樣本和標籤資料

標籤資料 – One-hot 編碼



標籤資料－特徵標準化

- 正規化 Normalization：也稱最小最大值縮放，這是一種特徵標準化，可以將樣本資料集的特徵按比例縮放，讓資料落在一個特定區間。

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

- 標準化 Standardization：也稱 Z 分數，可以位移資料分布成為平均值是0，標準差是1的資料分布。

$$X_{z分數} = \frac{X - 平均值}{標準差}$$

End !