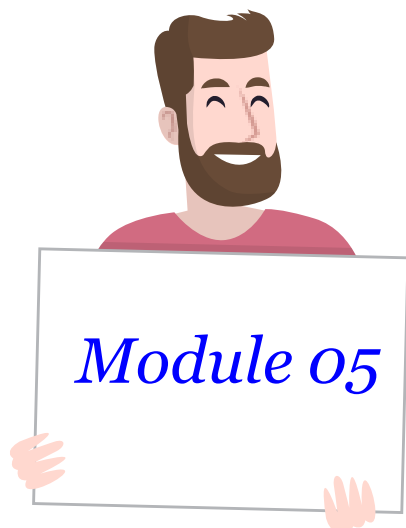




損失函數的定義



designed by  freepik

Estimated time:
45 min.

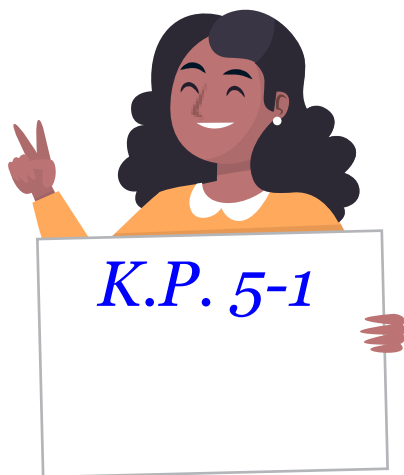
學習目標

- 5-1: 損失函數介紹
- 5-2: MSE與Cross-Entropy
- 5-3: 損失函數範例



5-1: 損失函數介紹

- 損失函數
- 損失函數示意圖



designed by freepik

損失函數

- 損失函數的目的是衡量預測向量及期望向量相似程度的指標
 - 數值越小代表越好
- 損失函數的種類很多
 - Mean Square Error(MSE), Cross-entropy, Hinge loss,

損失函數

- 常見的幾個損失函數
 - **MSE及Cross-Entropy**是兩個常見之損失函數

$$\sum (y^i - \hat{y}^i)^2$$

MSE

$$-\sum y^i \log(\hat{y}^i)$$

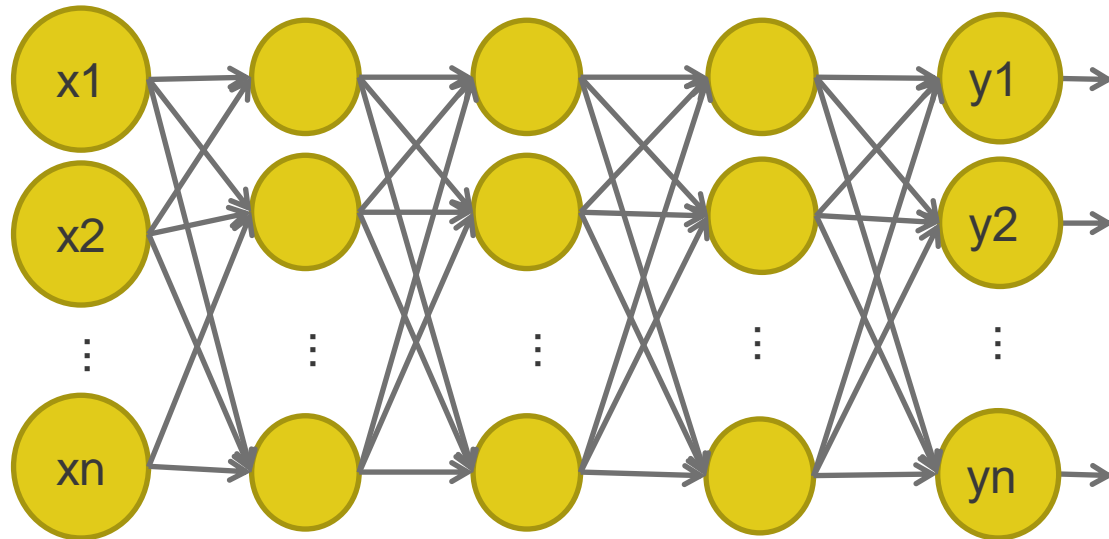
Cross-Entropy

y^i : 預測向量第*i*個元素

\hat{y}^i : 期望向量第*i*個元素

損失函數示意圖

- 之所以需要有損失函數的原因是，我們必須要有一個衡量的指標來衡量預測向量與期望向量落差有多少



$$\begin{bmatrix} 0.15 \\ 0.2 \\ 0.3 \\ \vdots \end{bmatrix} \begin{matrix} \text{"0"} \\ \text{"1"} \\ \text{"2"} \\ \end{matrix}$$

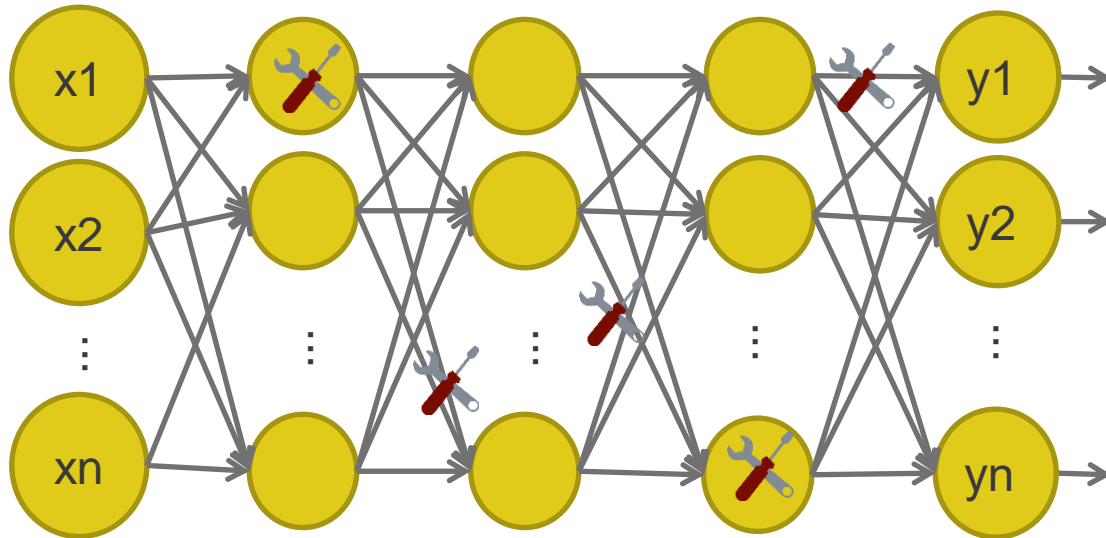
預測向量
(神經網路之輸出)

$$\begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \end{bmatrix} \begin{matrix} \text{"0"} \\ \text{"1"} \\ \text{"2"} \\ \end{matrix}$$

期望向量
(one-hot編碼過的標籤)

損失函數示意圖

- 隨著網路參數的修正，理論上損失函數的值要越來越小
 - 這也是“優化”這個步驟所要做的

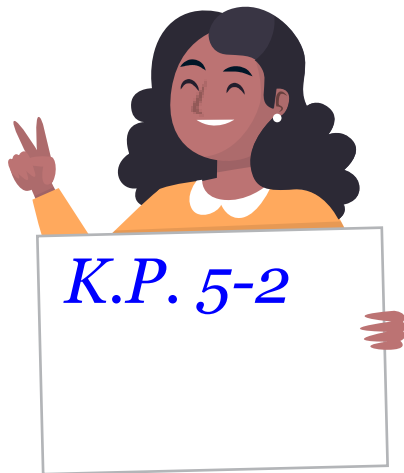


$$\begin{bmatrix} 0.15 \\ 0.2 \\ 0.3 \\ \vdots \end{bmatrix} \longleftrightarrow \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \end{bmatrix}$$

修正網路裡的參數使得預測向量與期望向量越來越接近

5-2: MSE與Cross-Entropy

- MSE介紹
- Cross-Entropy介紹



designed by freepik

MSE介紹

- 最基礎的損失函數
 - 代表兩個向量之間的直線距離

$$\sum (y^i - \hat{y}^i)^2$$




Diagram illustrating the components of the MSE formula:

Left vector (labeled y^i):

$$\begin{bmatrix} 0.15 \\ 0.2 \\ 0.3 \\ \vdots \end{bmatrix} \begin{matrix} \text{"0"} \\ \text{"1"} \\ \text{"2"} \\ \end{matrix}$$

Right vector (labeled \hat{y}^i):

$$\begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \end{bmatrix} \begin{matrix} \text{"0"} \\ \text{"1"} \\ \text{"2"} \\ \end{matrix}$$

MSE介紹

- 假設我們現在有兩個向量，其MSE的計算式子如下
 - 用幾何的觀點看是在計算兩個向量空間中的距離

$$\begin{bmatrix} 0.15 \\ 0.2 \\ 0.3 \\ \vdots \end{bmatrix} \begin{matrix} \text{“0”} \\ \text{“1”} \\ \text{“2”} \\ \end{matrix} \qquad \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \end{bmatrix} \begin{matrix} \text{“0”} \\ \text{“1”} \\ \text{“2”} \\ \end{matrix}$$

$$\sqrt{(0.15 - 0)^2 + (0.2 - 1)^2 + (0.3 - 0)^2 + \dots}$$

Cross-Entropy介紹

- **Cross-Entropy**是一種衡量兩個機率分布是否相似的損失函數
 - 在深度學習中，使用**Cross-Entropy**時，前面常會加上**Softmax**層

$$-\sum y^i \log(\hat{y}^i)$$

Cross-Entropy

Cross-Entropy 介紹

- 在通訊理論理，資訊含量(information)被定義如下：

$\log\left(\frac{1}{p^i}\right)$ ，其中 p^i 為某一個事件所發生之機率

太陽明天從東方升起

明天台北會下雨

以直覺來看，那一句話比較有”資訊含量”

Cross-Entropy介紹

- **Entropy**
 - 資訊含量的期望值
 - 數值越大代表越混亂、越不確定
- **Cross-Entropy**
 - 衡量兩個機率分布是否相似
 - 數值越大代表越不相似、越小代表越相似

$$H(y) = - \sum y^i \log(y^i)$$

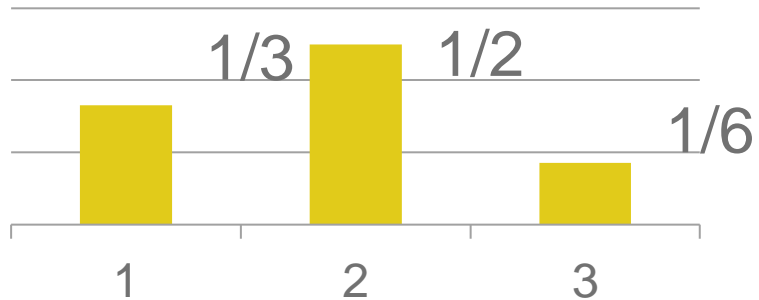
Entropy

$$H(y, \hat{y}) = - \sum y^i \log(\hat{y}^i)$$

Cross-Entropy

Cross-Entropy 介紹

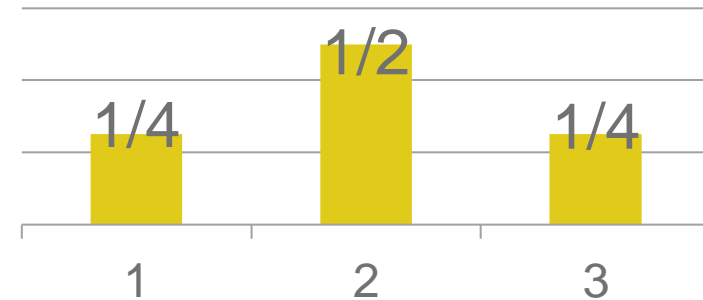
機率分布1



Entropy(機率分布1) =

$$1/3 * \log(3) + 1/2 * \log(2) + 1/6 * \log(6)$$

機率分布2



Entropy(機率分布2) =

$$1/4 * \log(4) + 1/2 * \log(2) + 1/4 * \log(4)$$

Cross-entropy(機率分布1, 機率分布2) =

$$1/3 * \log(4) + 1/2 * \log(2) + 1/6 * \log(4)$$

Cross-entropy(機率分布2, 機率分布1) =

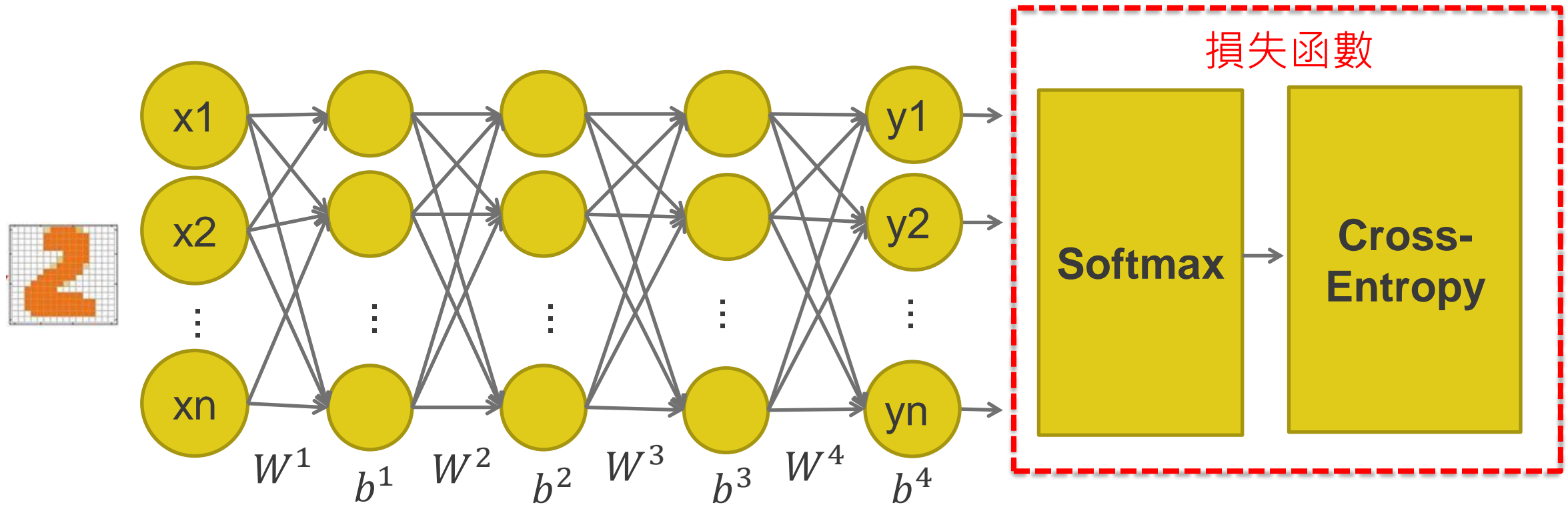
$$1/4 * \log(3) + 1/2 * \log(2) + 1/4 * \log(6)$$

Cross-Entropy 介紹

- 根據數學推導，**Cross-Entropy**會大於等於**Entropy**
 - **Cross-Entropy(機率分布1, 機率分布2) \geq Entropy(機率分布1)**
 - 等好成立於當機率分布1與機率分布2一樣時，**Cross-Entropy**有最小值
 - 深度學習就是根據此特性來衡量預測向量以及期望向量之相似程度

Cross-Entropy介紹

- 在深度學習裡，當使用Cross-Entropy為損失函數時，前面常會加一層Softmax層



Softmax函數

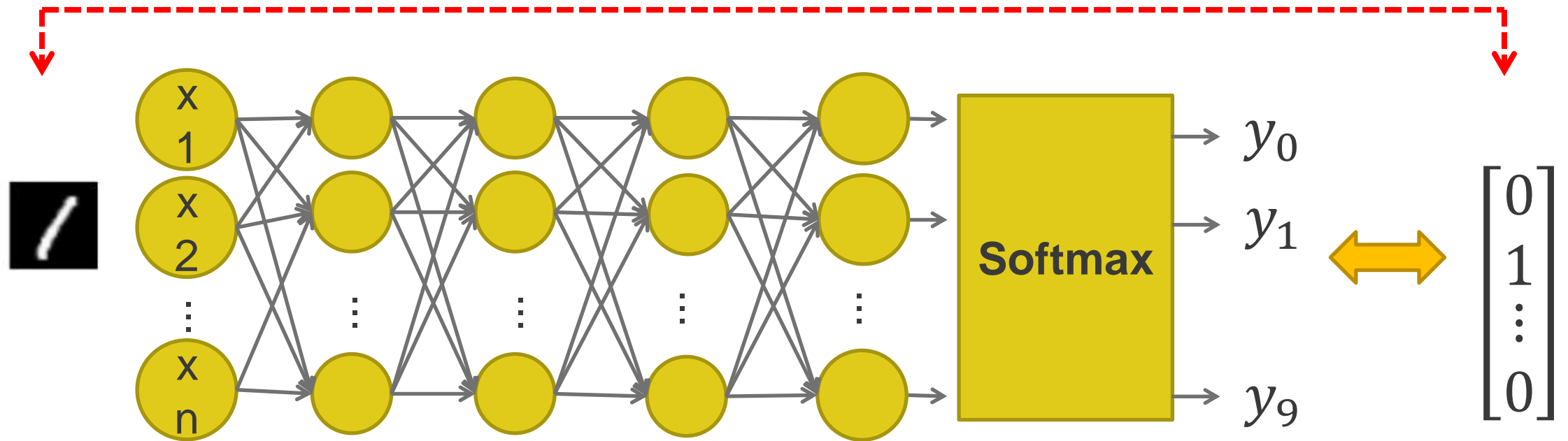
- softmax層主要用途是將某一向量轉換成機率分佈
 - 將一個向量裡面的所有元素壓縮到0~1之間

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^K e^{z_k}}$$

$\begin{bmatrix} 0.9 \\ 1.2 \\ 0.4 \end{bmatrix} \rightarrow \text{Softmax} \rightarrow \begin{bmatrix} \frac{e^{0.9}}{e^{0.9} + e^{1.2} + e^{0.4}} \\ \frac{e^{1.2}}{e^{0.9} + e^{1.2} + e^{0.4}} \\ \frac{e^{0.4}}{e^{0.9} + e^{1.2} + e^{0.4}} \end{bmatrix} = \begin{bmatrix} 0.34 \\ 0.46 \\ 0.2 \end{bmatrix}$

Cross-Entropy介紹

- 假設我們將照片”1”輸入神經網路，並使用Cross-Entropy為損失函數，其示意圖如下



$$\begin{aligned} &\text{Cross-Entropy(預測向量, 期望向量)} \\ &= -(0 * \log(y_0) + 1 * \log(y_1) + 0 * \log(y_2) + \dots + 0 * \log(y_9)) \end{aligned}$$

5-2 Demo

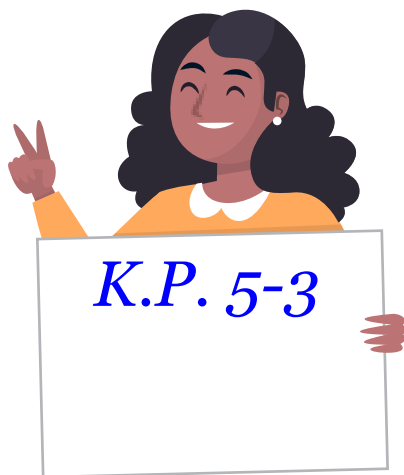
- 開啟Demo_5-2.ipynb
- TensorFlow MSE demo
- TensorFlow Cross-Entropy demo



designed by freepik

5-3: 損失函數範例

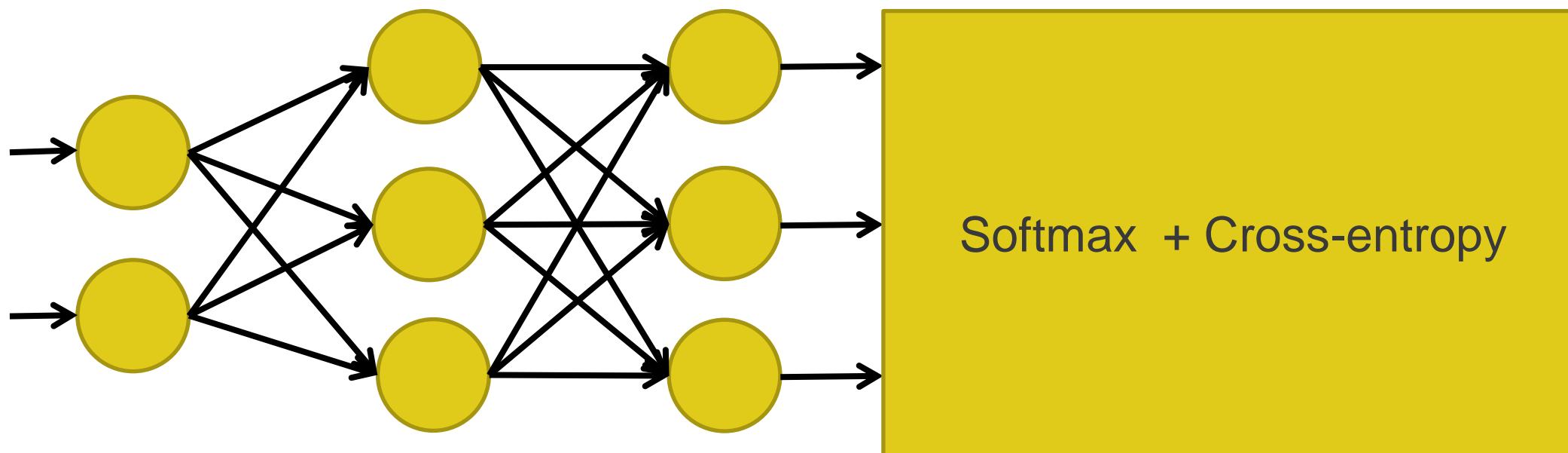
- 損失函數數值範例



designed by freepik

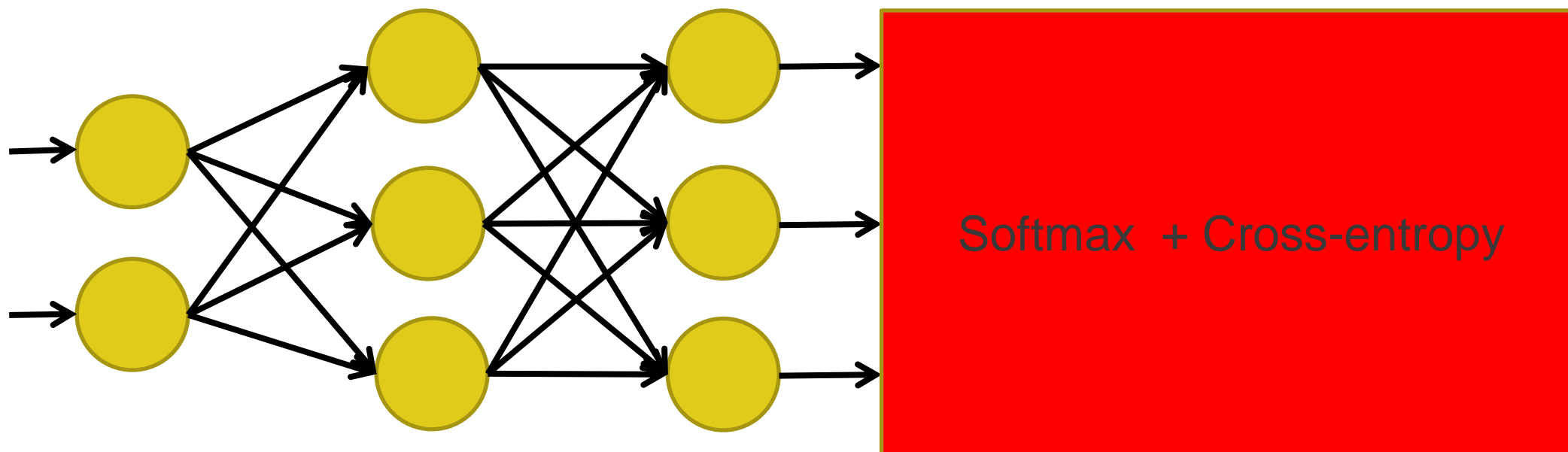
損失函數數值範例

- 接續之前神經網路的數值範例，假設我們使用**Cross-Entropy**來當損失函數



損失函數數值範例

- 神經網路的輸出為[0, 0.006, 0.12]，我們先經由softmax層的轉換，得到一個機率分布

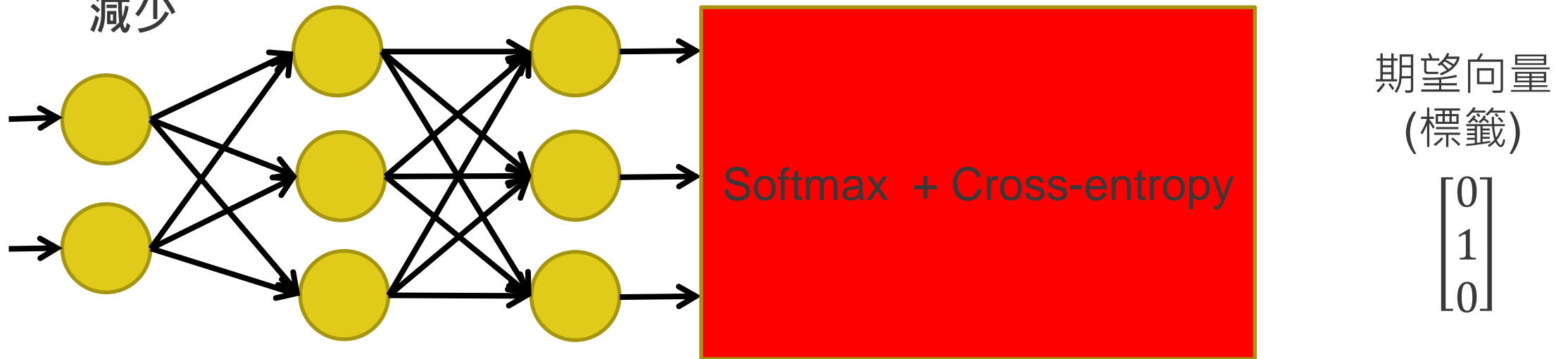


$$\text{Softmax}\left(\begin{bmatrix} 0 \\ 0.006 \\ 0.12 \end{bmatrix}\right) = \begin{bmatrix} 0.319 \\ 0.321 \\ 0.36 \end{bmatrix}$$

損失函數數值範例

- 假設我們期望的向量為[0, 1, 0]，則我們可以去計算其**Cross-Entropy**如下

- 訓練初期時，損失函數數值會比較高，隨著訓練的過程當中，數值要慢慢減少

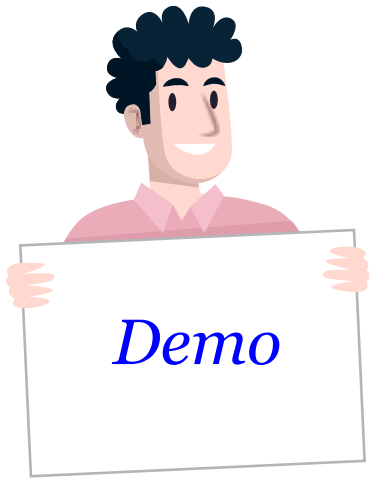


$$\begin{aligned} & - 0 * \ln(0.319) - 1 * \ln(0.321) - 0 * \ln(0.36) \\ & = 1.1363 \end{aligned}$$

$$H(y, \hat{y}) = - \sum y^i \log(\hat{y}^i)$$

5-3 Demo

- 開啟Demo_5-3.ipynb
- 建立神經網路模型並加上損失函數



designed by freepik

線上Corelab

- **題目1：計算MNIST資料集的MSE**
 - 取出MNIST的前2筆資料，計算兩張圖片矩陣的MSE值
- **題目2：計算MNIST資料集的Cross-Entropy**
 - 取出MNIST的前2筆資料，計算兩張圖片矩陣的Cross-Entropy值
- **題目3：建立5層的DNN並套用Cross-Entropy**
 - 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

- 損失函數的功用
- **Means Square Error(MSE)**
- **Cross-Entropy**
- **Softmax層**的功用



Lab: 損失函數介紹

- **Lab01: MSE損失函數**
- **Lab02: Cross-Entropy損失函數**
- **Lab03: 建立神經網路損失函數**

Estimated time:

20 minutes

