Day 75 BackPropagation

反向式傳播簡介

出題教練 陳宇春

知識地圖 深度學習組成概念

CUPOY

倒傳遞

深度神經網路
Supervised LearningDeep Neural Network (DNN)

- 簡介 Introduction
- 套件介紹 Tools: Keras
- 組成概念 Concept
- 訓練技巧 Training Skill
- 應用案例 Application

卷積神經網路
Convolutional Neural Network (CNN)

- 簡介 introduction
- 套件練習 Practice with Keras
- 訓練技巧 Training Skill
- 電腦視覺 Computer Vision

深度學習組成概念
Concept of DNN

感知器概念簡介

本日知識點目標

- 前行網路傳播(ForwardPropagation) /反向式傳播(BackPropagation)的差異
- 反向式傳播BackPropagation的運作

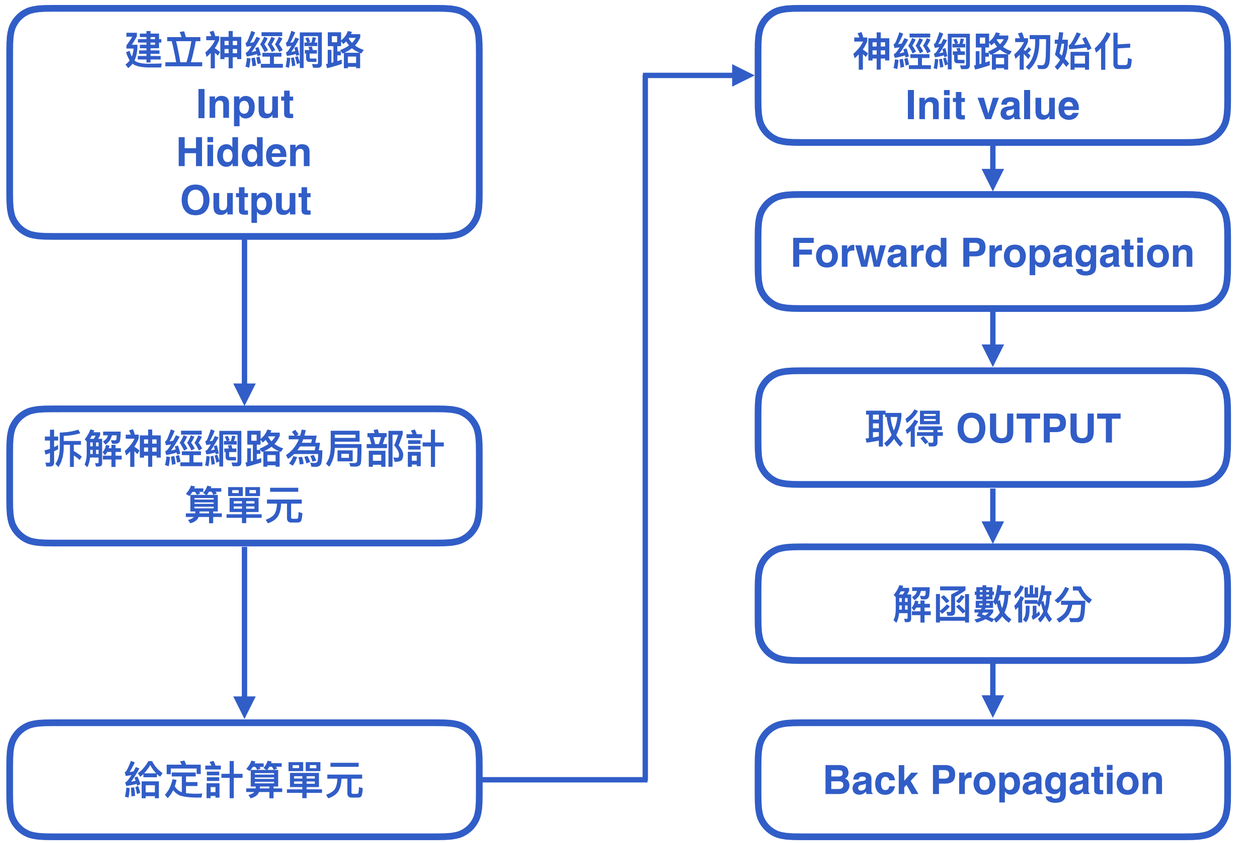
何謂反向傳播

CUPOY

- 反向傳播（BP：Backpropagation）是「誤差反向傳播」的簡稱，是一種與最優化方法（如梯度下降法）結合使用的該方法對網路中所有權重計算損失函數的梯度。這個梯度會反饋給最優化方法，用來更新權值以最小化損失函數。

- 反向傳播要求有對每個輸入值想得到的已知輸出，來計算損失函數梯度。因此，它通常被認為是一種監督式學習方法，可以對每層疊代計算梯度。反向傳播要求人工神經元（或「節點」）的啟動函數可微。

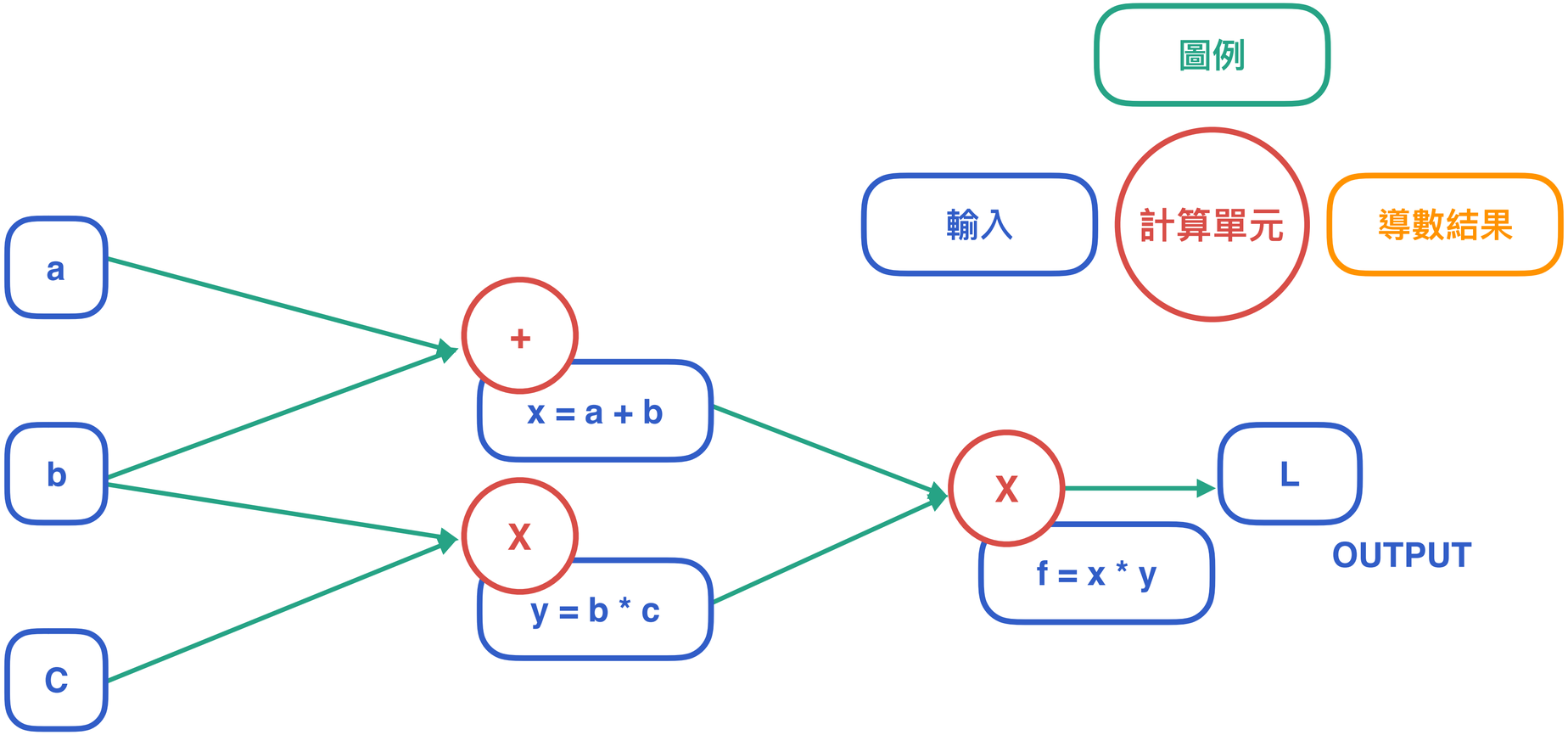
推導流程


建立神經網路
Input
Hidden
Output
拆解神經網路為局部計算單元
給定計算單元
神經網路初始化
Init value
Forward Propagation
取得 OUTPUT
解函數微分
Back Propagation

建構並拆解

CUPOY

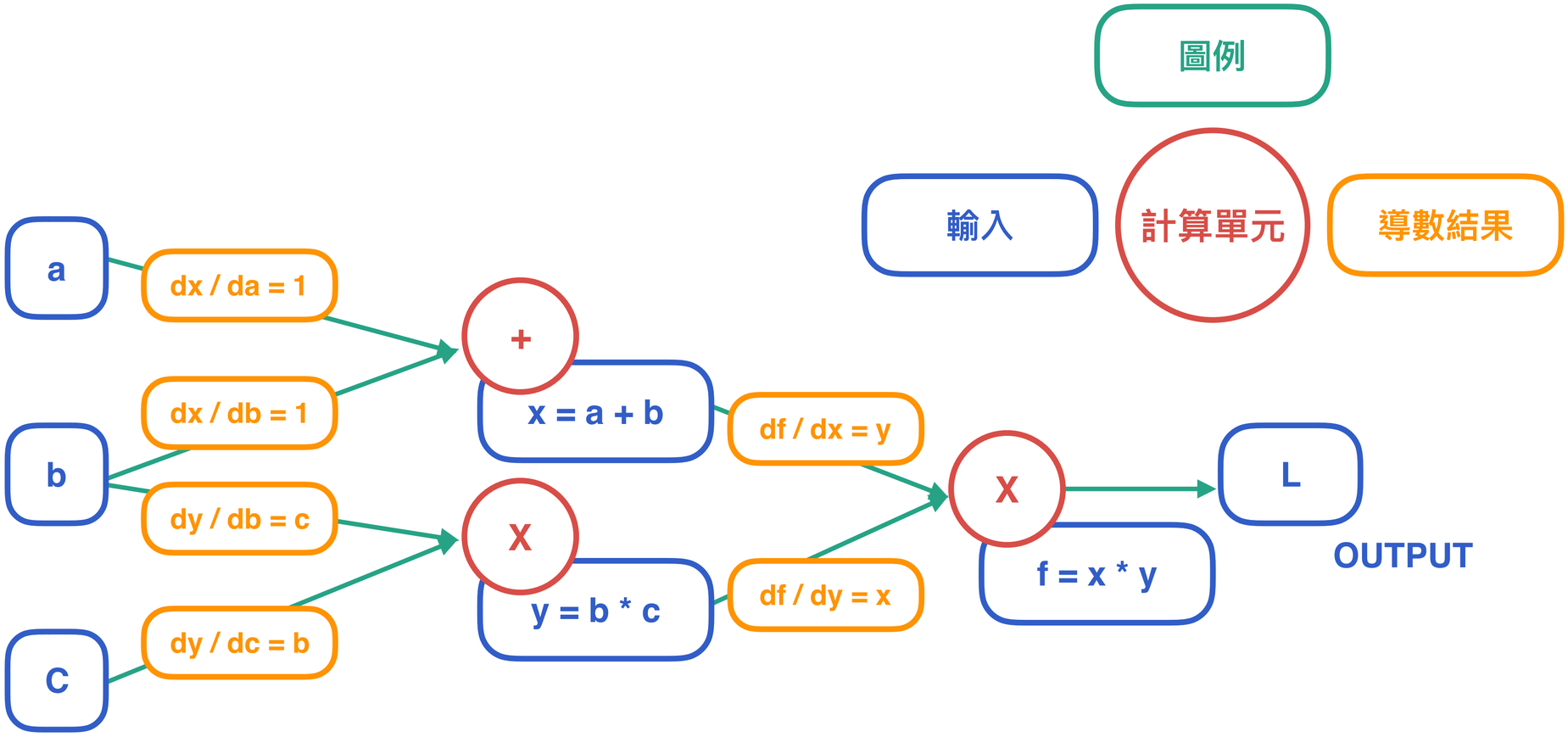
將神經網路的運算拆解為局部單元



BP – Back Propagation

CUPOY

如何解函數微分



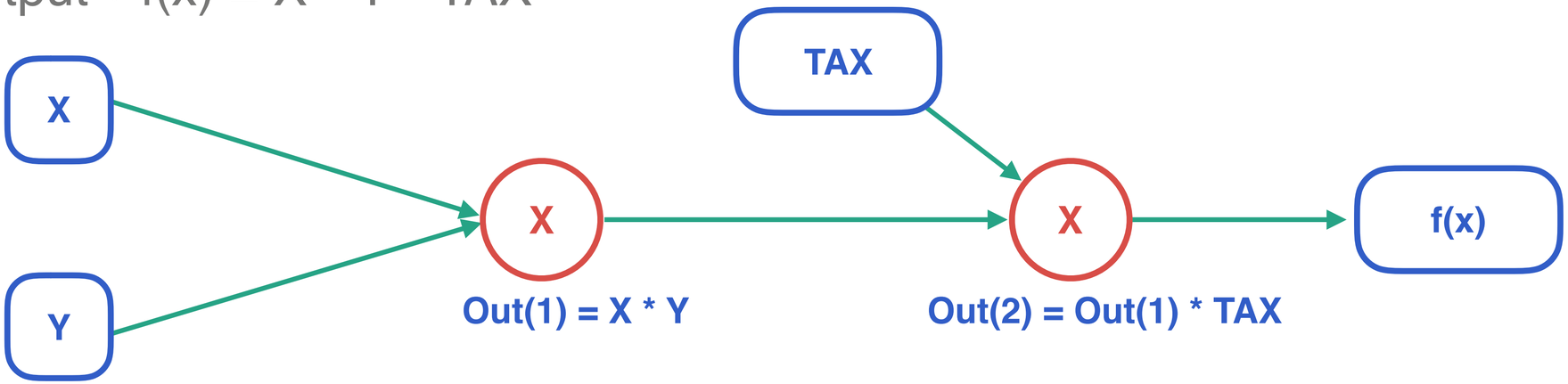
以預測水果銷售為例

CUPOY

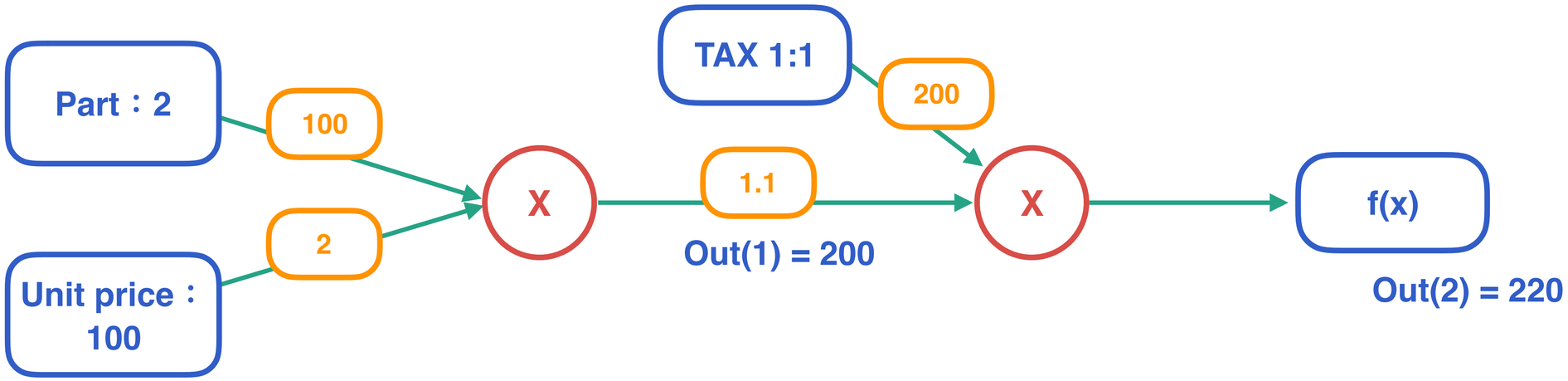
- 水果銷售所應給付的價格決定因子
  - 數量(顆數或是單位重量)
  - 單價
  - 稅金
- 建立運算單元：
  - 稅金是恆定的， 可以當成是 Bias， 給定 TAX
  - Input-1：數量，給定 X
  - Input-2：單價， 給定 Y
  - Output：f(x) = X * Y * TAX

以購買水果為例：

- 付費總價格是根據水果價格，稅金變動而受影響
- 水果價格是根據購買數量與單品價格而變動
- 可以利用每一個cell (cell - 1: 水果價格; cell - 2: 付費總價格)，推導微分的結果



以預測水果銷售為例 – Init & 解微分

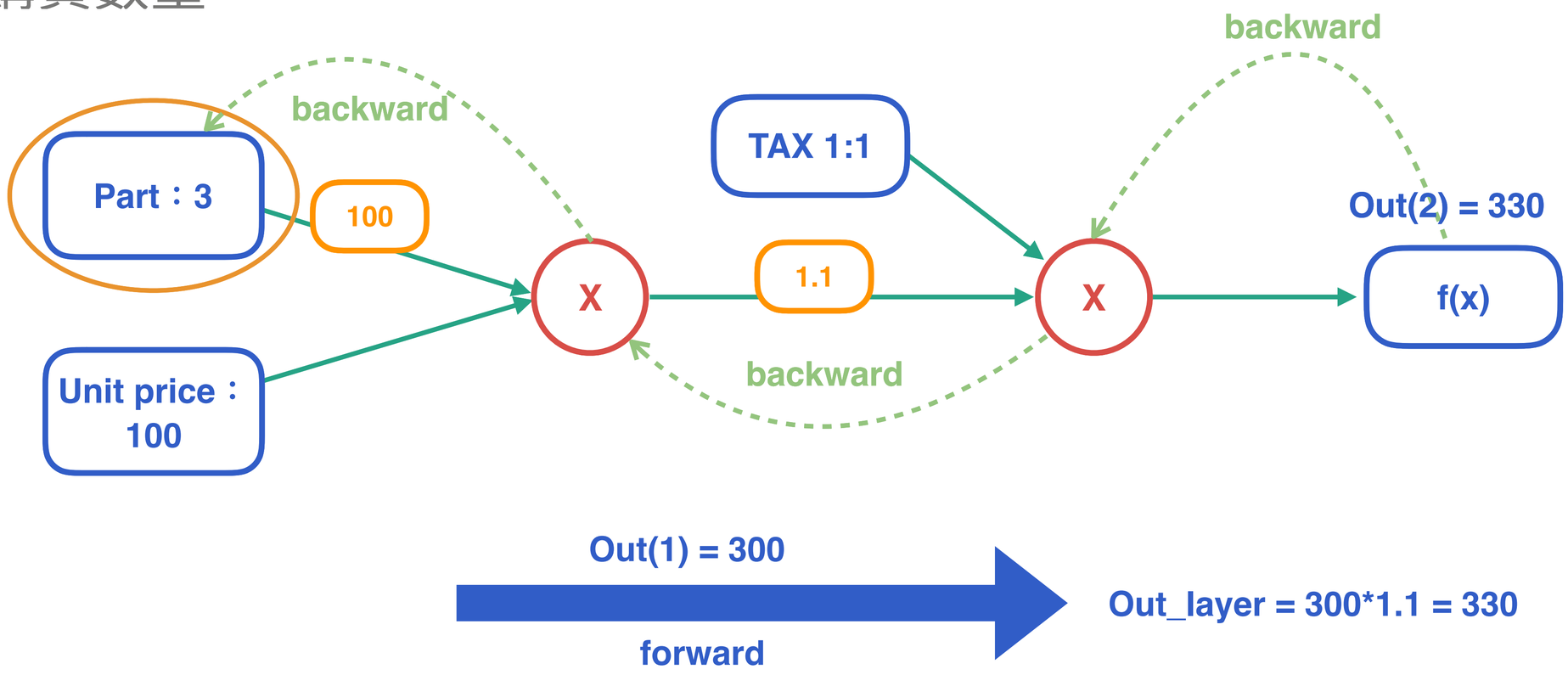


- 要驗證網路模型是否正確?
- 更改 Init Data：
  - 更改購買數量
  - TAX的增加

以預測水果銷售為例 – 更改 Init Data

CUPOY

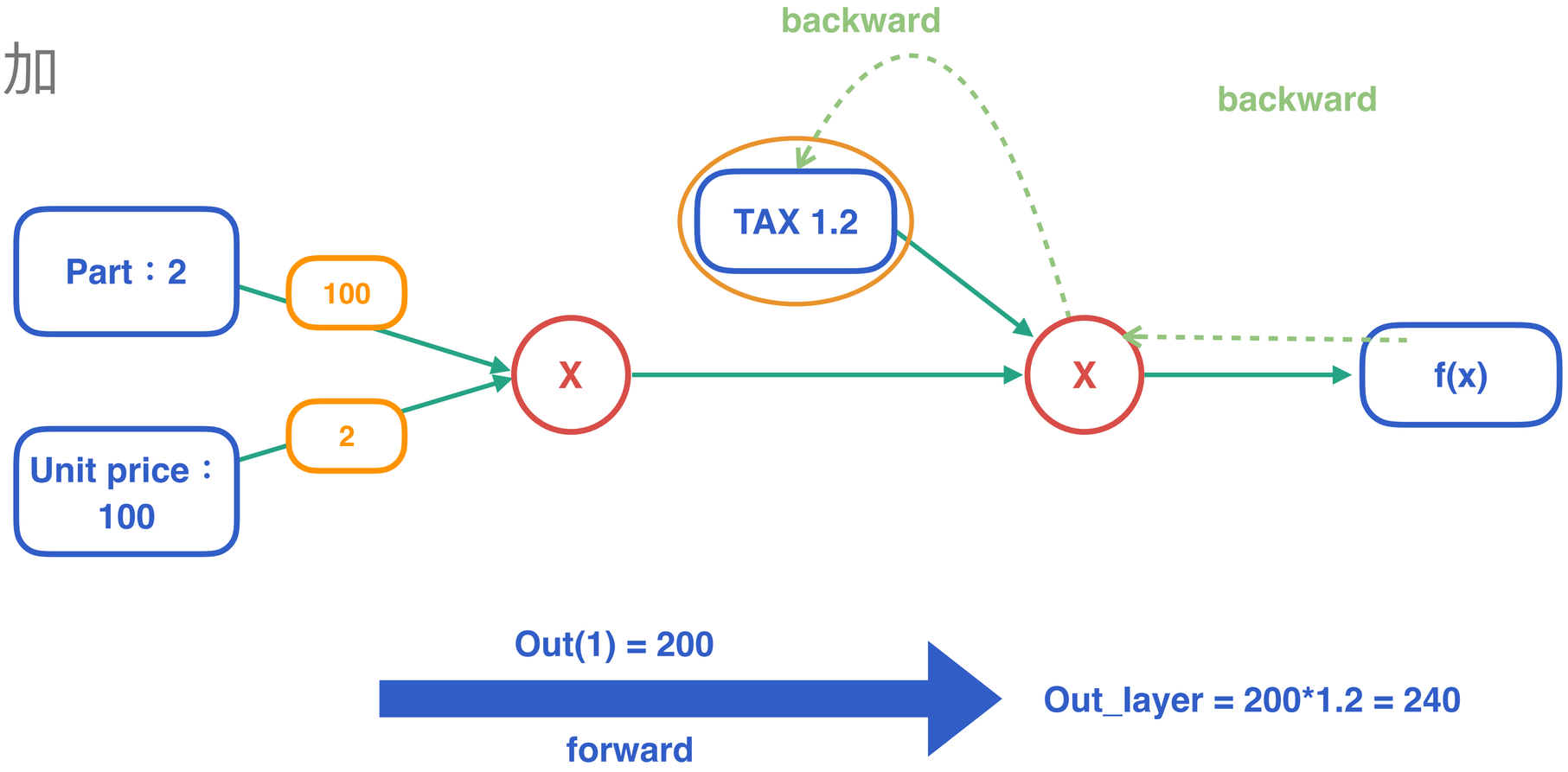
更改購買數量



所以，結帳金額 f(x) 被影響的是 (3-2)x100x1.1 = 110

以預測水果銷售為例 – 更改 Init Data

TAX的增加



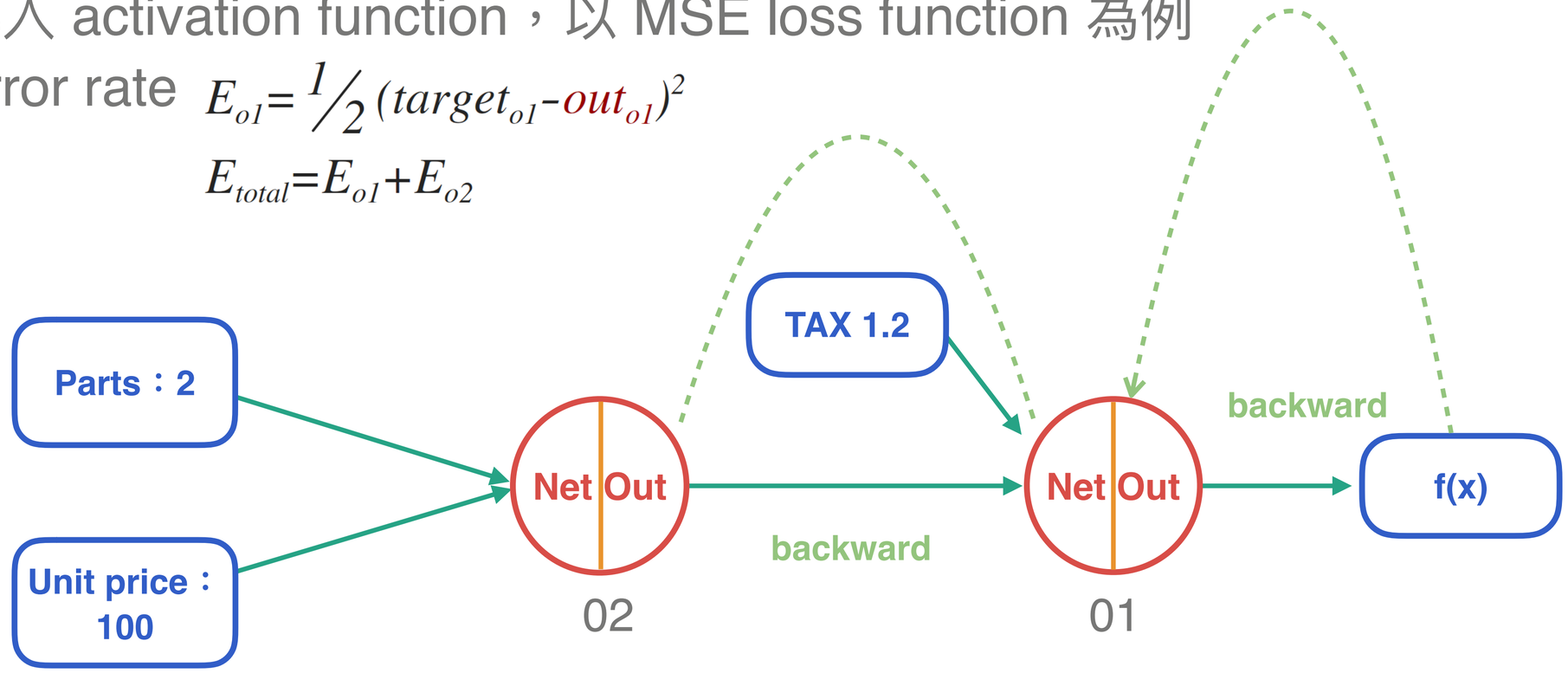
所以，結帳金額 f(x) 被影響的是 2x100x(1.2-1.1) = 20

進一步說明

更改 init data，輸出會有變動，模型的執行結果跟預期有落差也是變動，這個落差就是 error rate

- Error rate = (Target 輸出)–(實際輸出)
- 導入 activation function，以 MSE loss function 為例
- Error rate $E_{o1}=\frac{1}{2}(target_{o1}-out_{o1})^2$

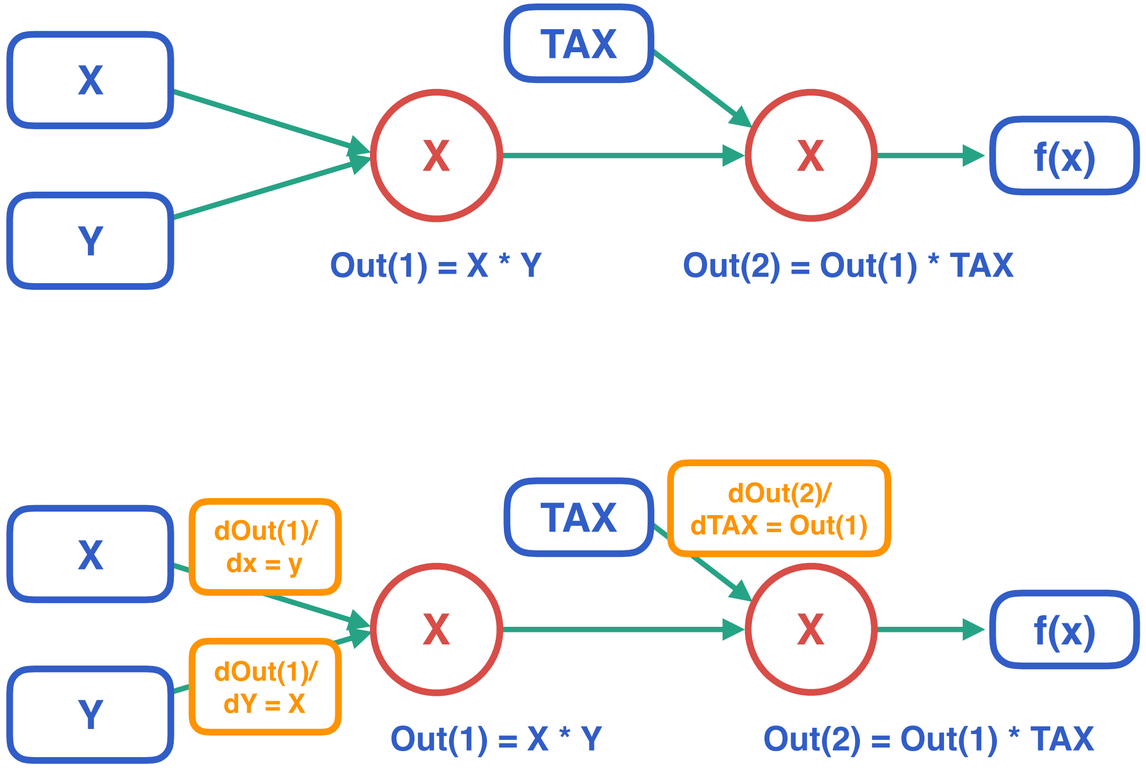
$$E_{total}=E_{o1}+E_{o2}$$



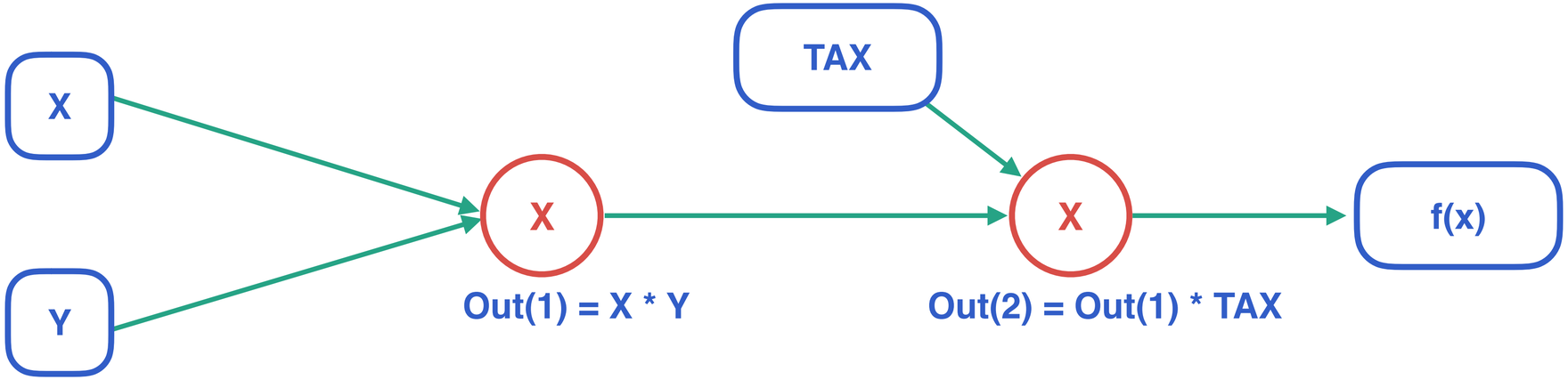
建立 Forward & Backward

CUPOY

```
class mul_layer():
    def _ini_(self):
        self.x = None
        self.y = None
    def forward(self, x, y):
        self.x = x
        self.y = y
        out = x*y
        return out
    def backward(self, dout):
        dx = dout * self.y
        dy = dout * self.x
        return dx, dy
```



Init Network data




```
# Init Data
n_X = 2
price_Y = 100
b_TAX = 1.1

# Build _Network
mul_fruit_layer = mul_layer()
Mul_tax_layer = mul_layer()
```

Forward & Backward operation

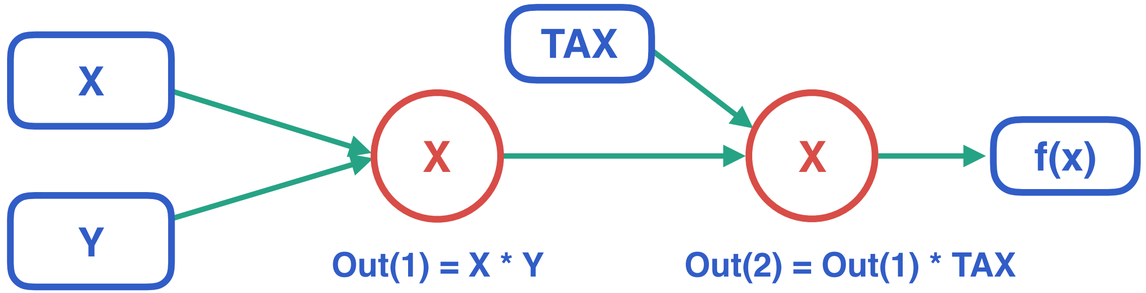
CUPOY

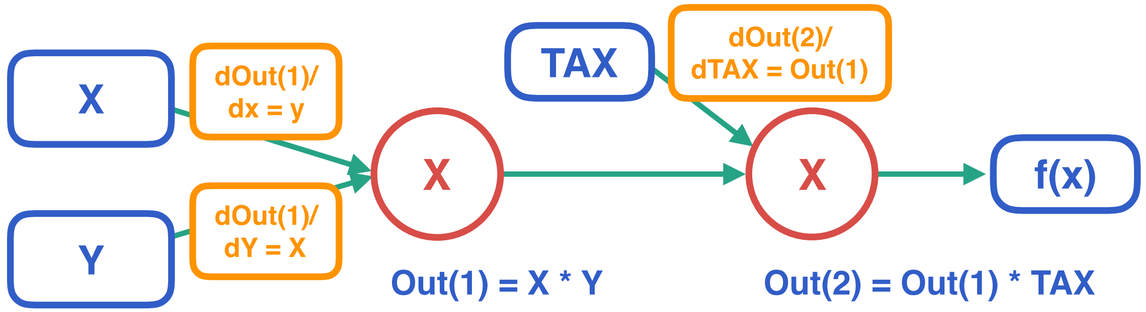
```
#forward
fruit_price = mul_fruit_layer.forward(price_Y, n_X)
total_price = mul_tax_layer.forward(fruit_price, b_TAX)

#backward
dtotal_price = 1 #this is linear function, which y=x, dy/dx=1
d_fruit_price, d_b_TAX = mul_tax_layer.backward(dtotal_price)
d_price_Y, d_n_X =  mul_tax_layer.backward(d_fruit_price)
```

```
#result
print("fruit price: %i"%fruit_price)
print("針對所有水果價格微分, 得到 TAX: %2f" %d_fruit_price)
```

```
fruit price: 200
針對所有水果價格微分, 得到 TAX: 1.100000
```





重要知識點複習：

- BP 神經網路是一種按照逆向傳播算法訓練的多層前饋神經網路
- 優點：具有任意複雜的模式分類能力和優良的多維函數映射能力，解決了簡單感知器不能解決的異或或者一些其他的問題。
  - 從結構上講，BP 神經網路具有輸入層、隱含層和輸出層。
  - 從本質上講，BP 算法就是以網路誤差平方目標函數、採用梯度下降法來計算目標函數的最小值。
- 缺點：
  ①學習速度慢，即使是一個簡單的過程，也需要幾百次甚至上千次的學習才能收斂。②容易陷入局部極小值。
  ③網路層數、神經元個數的選擇沒有相應的理論指導。
  ④網路推廣能力有限。
- 應用：①函數逼近。②模式識別。③分類。④數據壓縮

重要知識點複習：

CUPOY

- 第1階段：解函數微分
  - 每次疊代中的傳播環節包含兩步：
  - （前向傳播階段）將訓練輸入送入網路以獲得啟動響應；
  - （反向傳播階段）將啟動響應同訓練輸入對應的目標輸出求差，從而獲得輸出層和隱藏層的響應誤差。

- 第2階段：權重更新
  - Follow Gradient Descent
  - 第 1 和第 2 階段可以反覆循環疊代，直到網路對輸入的響應達到滿意的預定的目標範圍為止。

重要知識點複習：

CUPOY

在課程的範例程式：

- BP Neural Network
  - 實現 forward network，解函數微分求 Loss rate
    - Linear： Error rate = (target_out – real_out)
  - Weights refresh per iteration
  - Training and update
  - 得出 Loss rate

解題時間
It’s Your Turn

請跳出PDF至官網Sample Code＆作業
開始解題