NVIDIA

DEEP
LEARNING
INSTITUTE

# 深度學習基本原理

## 第 2 部分：神經網路的訓練方式

# 課程安排

- 第 1 部分：深度學習簡介
- 第 2 部分：神經網路的訓練方式
- 第 3 部分：卷積神經網路
- 第 4 部分：資料增強與部署
- 第 5 部分：預先訓練的模型
- 第 6 部分：先進架構

# 課程安排 – 第 2 部分

- 回顧
- 簡化的模型
- 從神經元到網路
- 激活函數
- 過度擬合
- 從神經元到分類

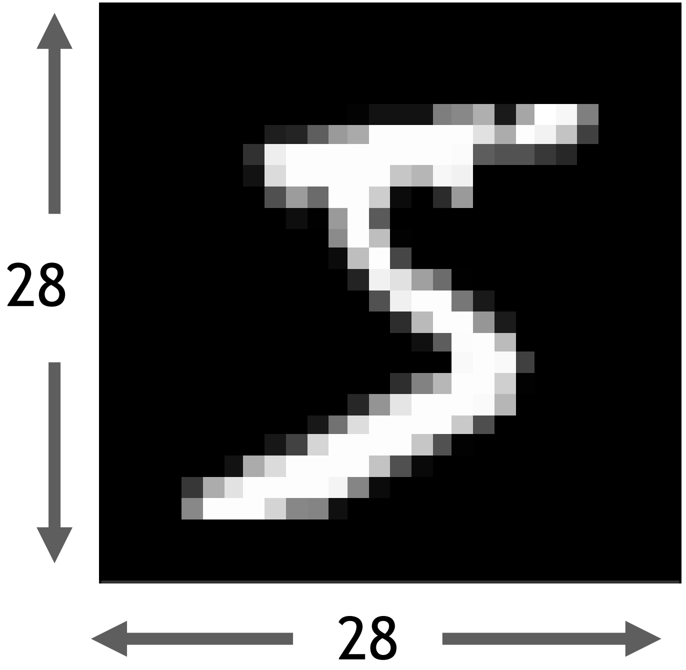
# 實作回顧

## 發生了什麼事？

- 載入並視覺化我們的資料
- 編輯資料 (重新調整、正規化、分類)
- 建立模型
- 編譯模型
- 根據資料訓練模型

# 資料準備

以陣列形式輸入




[0,0,0,24,75,184,185,78,32,55,0,0,0...]

# 資料準備

## 類別目標

0 → [1,0,0,0,0,0,0,0,0,0]

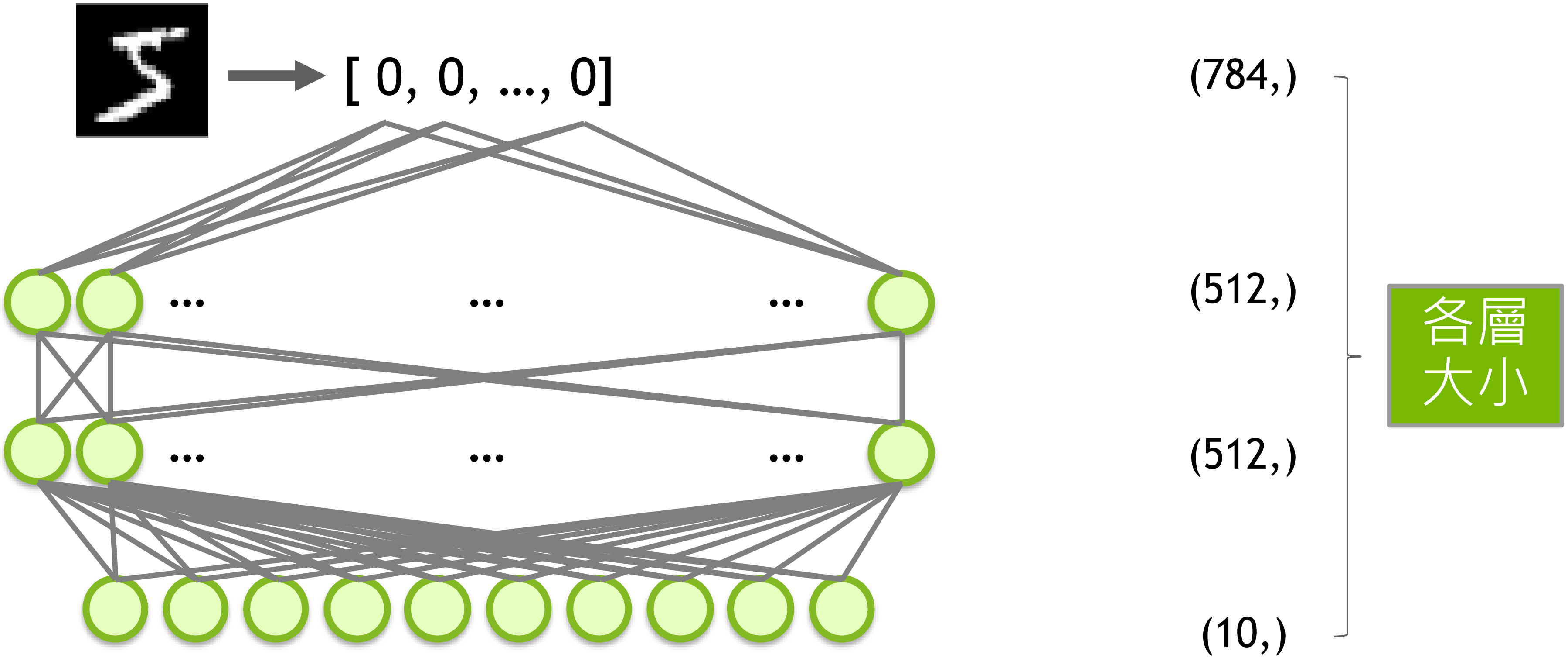
1 → [0,1,0,0,0,0,0,0,0,0]

2 → [0,0,1,0,0,0,0,0,0,0]

3 → [0,0,0,1,0,0,0,0,0,0]

⋮

# 未經訓練的模型

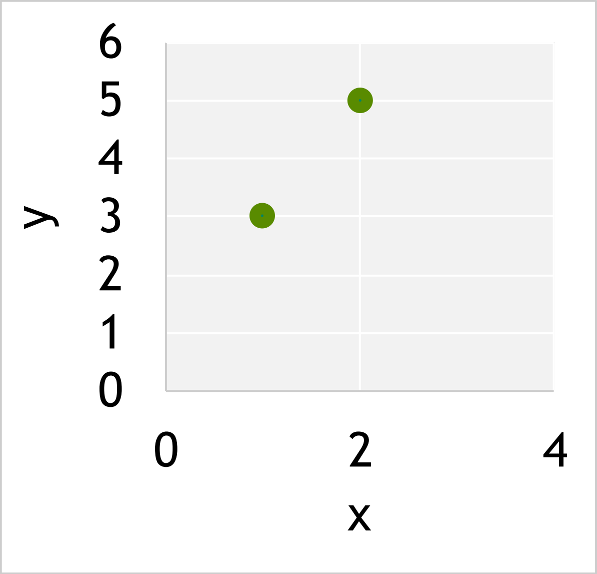


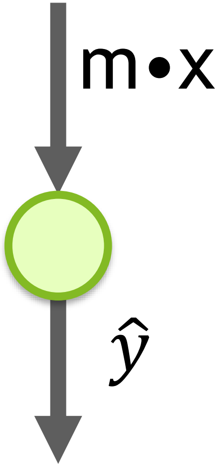
# 簡化的模型

# 簡化的模型

$$y = mx + b$$

| x | y |
|---|---|
| 1 | 3 |
| 2 | 5 |



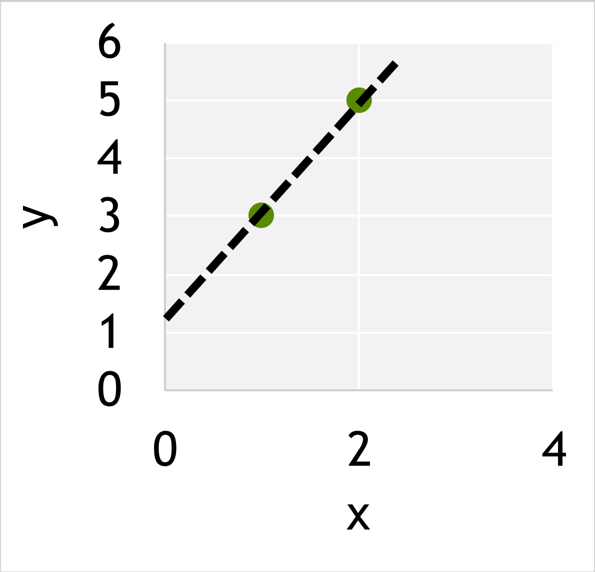

$$m = ?$$

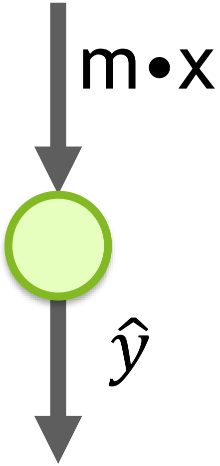
$$b = ?$$

# 簡化的模型

$$y = mx + b$$

| x | y |
|---|---|
| 1 | 3 |
| 2 | 5 |



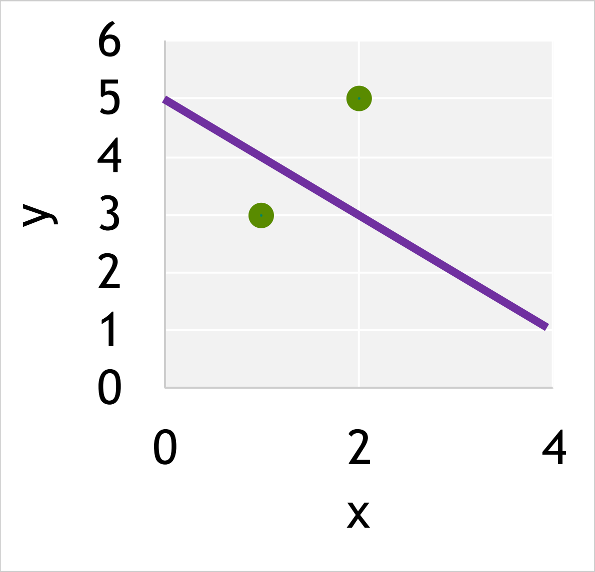

$m = ?$

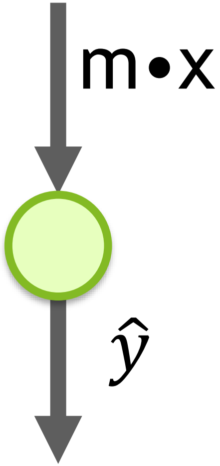
$b = ?$

# 簡化的模型

$$y = mx + b$$

| x | y | $\hat{y}$ |
|---|---|---|
| 1 | 3 | 4 |
| 2 | 5 | 3 |




Start Random

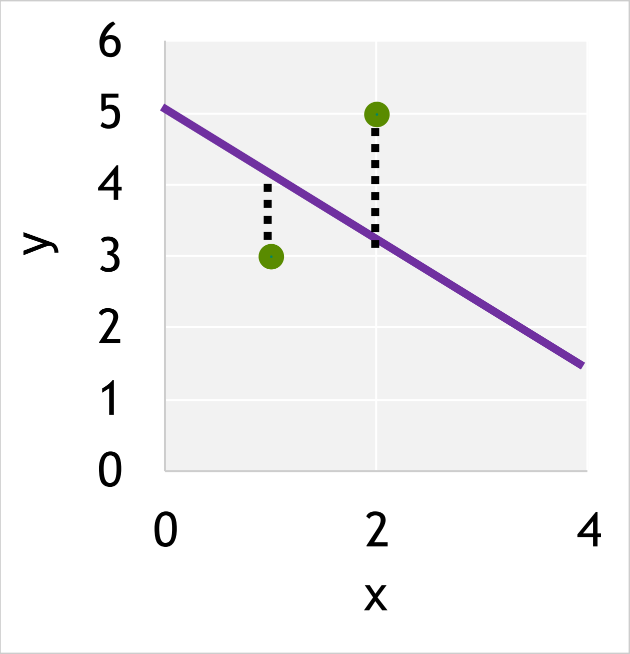
$$m = -1$$

$$b = 5$$

# 簡化的模型

$$y = mx + b$$

| x | y | $\hat{y}$ | $err^2$ |
|---|---|---|---|
| 1 | 3 | 4 | 1 |
| 2 | 5 | 3 | 4 |
| | | MSE = | 2.5 |
| | | RMSE = | 1.6 |



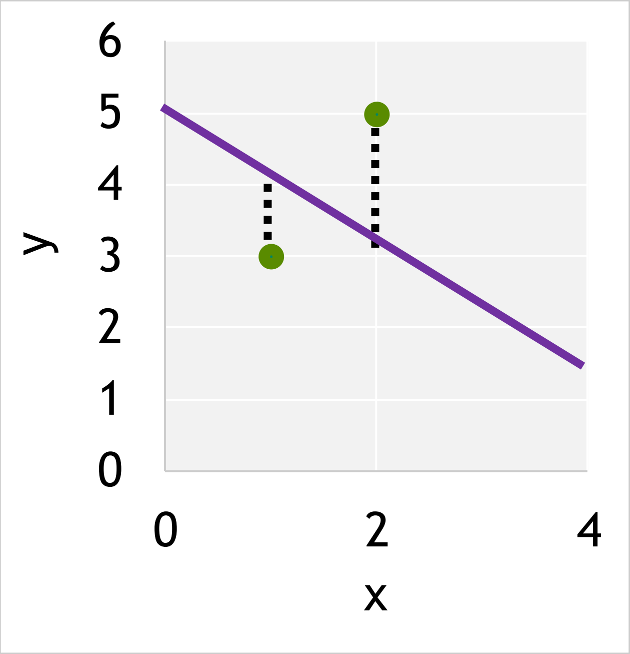

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y-\hat{y})^2$$

$$RMSE = \frac{1}{n}\sqrt{\sum_{i=1}^{n}(y-\hat{y})^2}$$

# 簡化的模型

$$y = mx + b$$

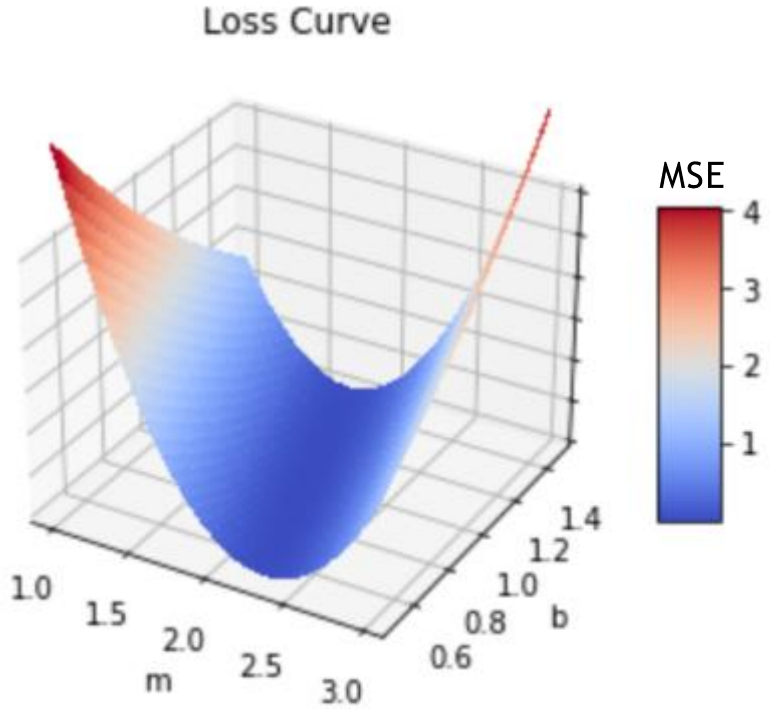
| x | y | $\hat{y}$ | $err^2$ |
|---|---|---|---|
| 1 | 3 | 4 | 1 |
| 2 | 5 | 3 | 4 |
| | | MSE = | 2.5 |
| | | RMSE = | 1.6 |



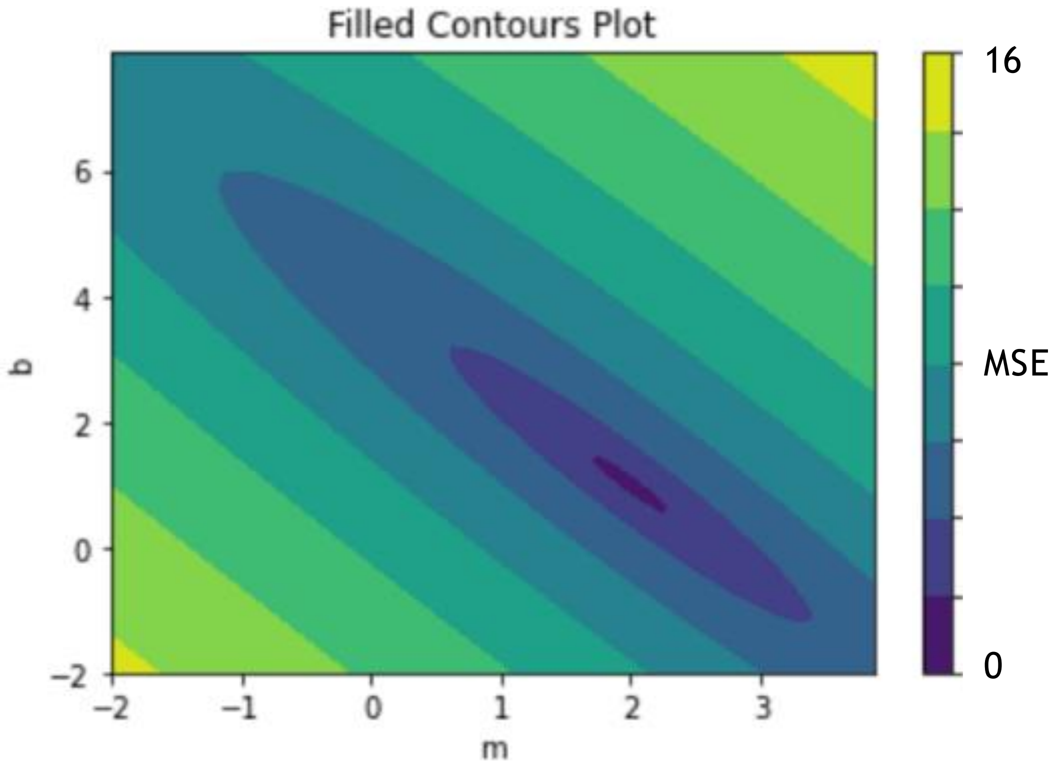

```python
data = [(1, 3), (2, 5)]
m = -1
b = 5

def get_mse(data, m, b):
    """Calculates Mean Square Error"""
    n = len(data)
    squared_error = 0
    for x, y in data:
        # Find predicted y
        y_hat = m*x+b
        # Square difference between
        # prediction and true value
        squared_error += (
            y - y_hat)**2
    # Get average squared difference
    mse = squared_error / n
    return mse
```

# 損失曲線

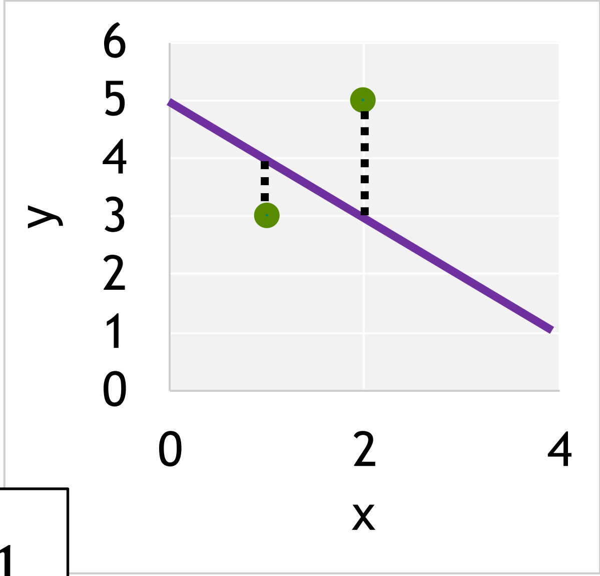


Loss Curve
MSE
4
3
2
1
1.0
1.5
2.0
2.5
3.0
m
0.6
0.8
1.0
1.2
1.4
b

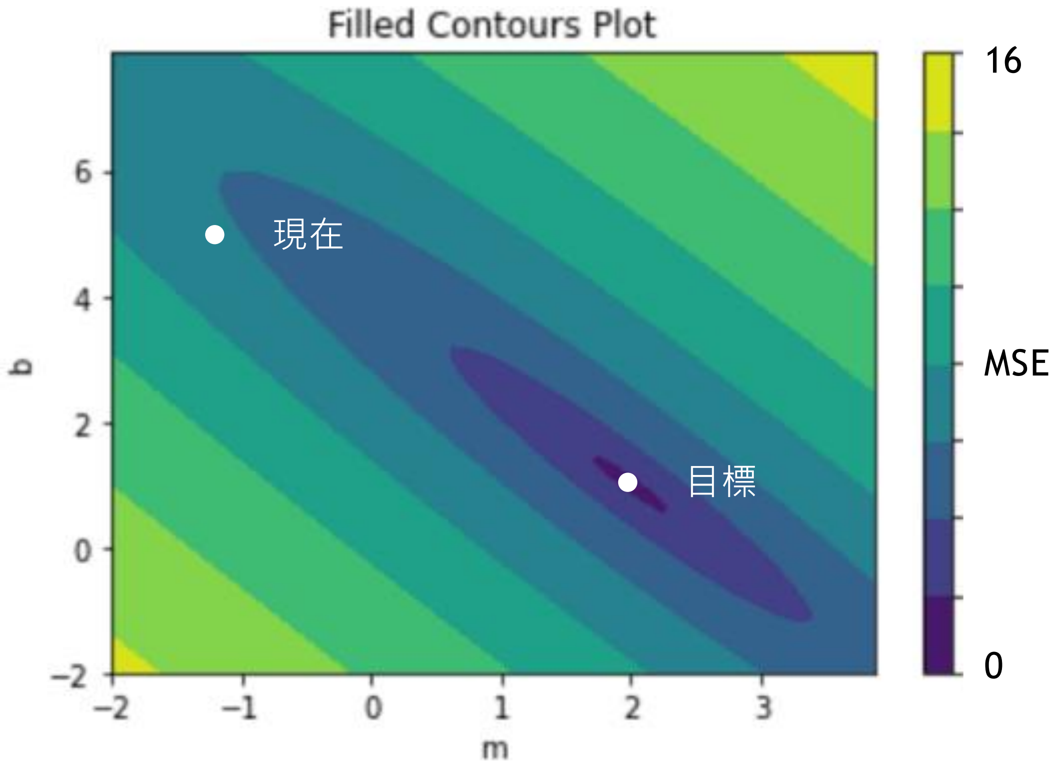
Filled Contours Plot
16
MSE
0
6
4
2
0
−2
b
−2
−1
0
1
2
3
m

# 損失曲線

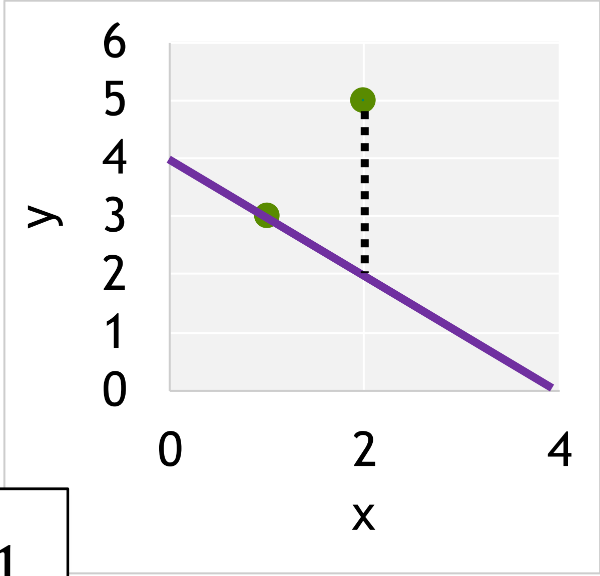



$m = -1$

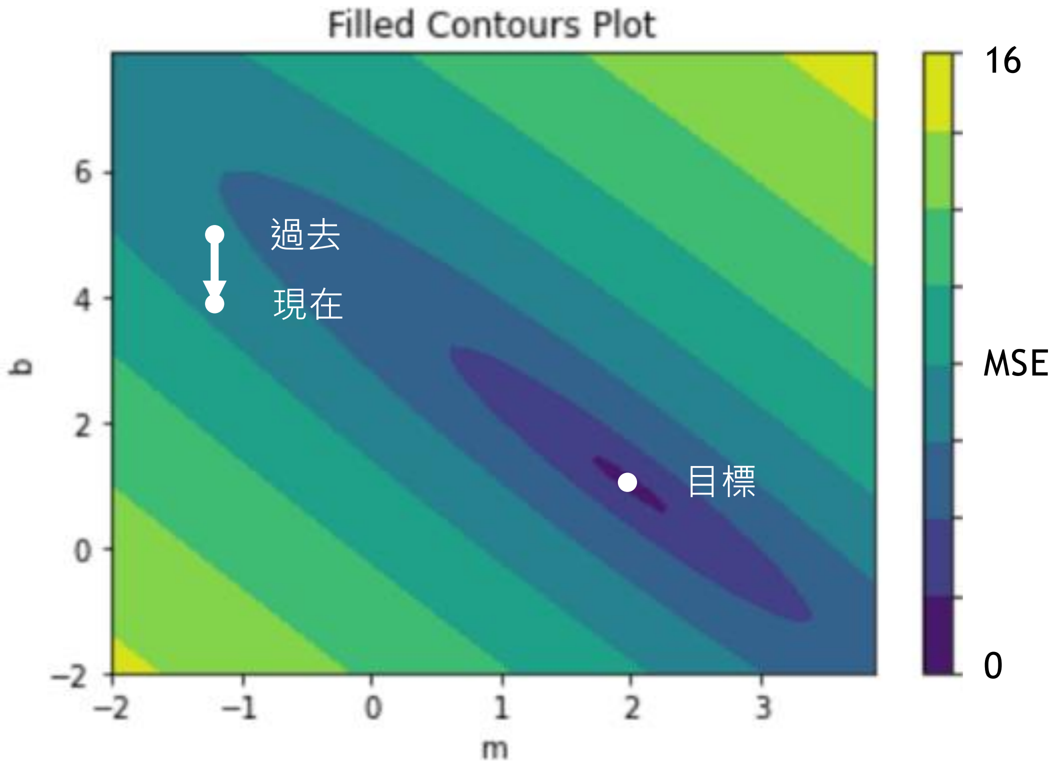
$b = 5$



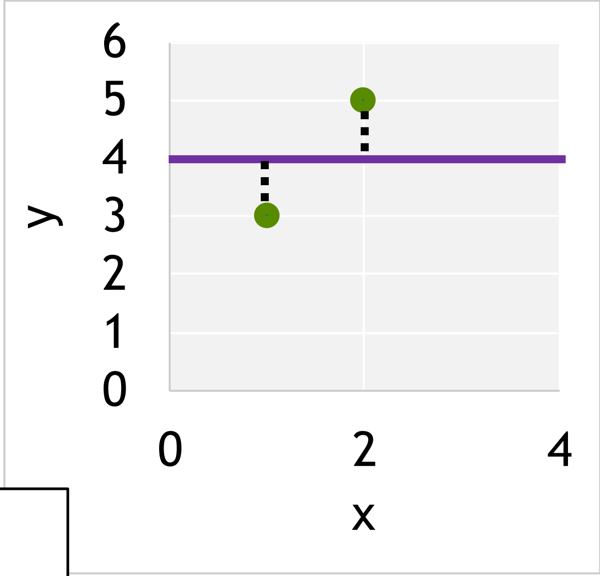
# 損失曲線




$$m = -1$$
$$\mathrm{b} = 4$$

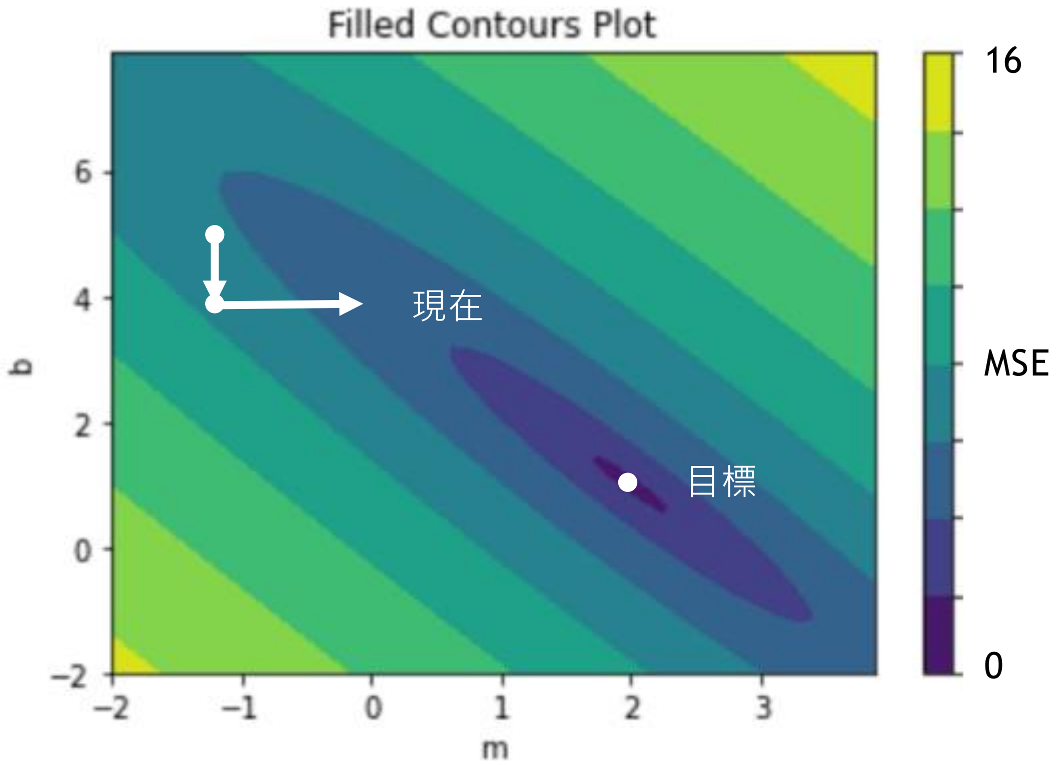


# 損失曲線


6
5
4
3
2
1
0
y
0
2
4
x

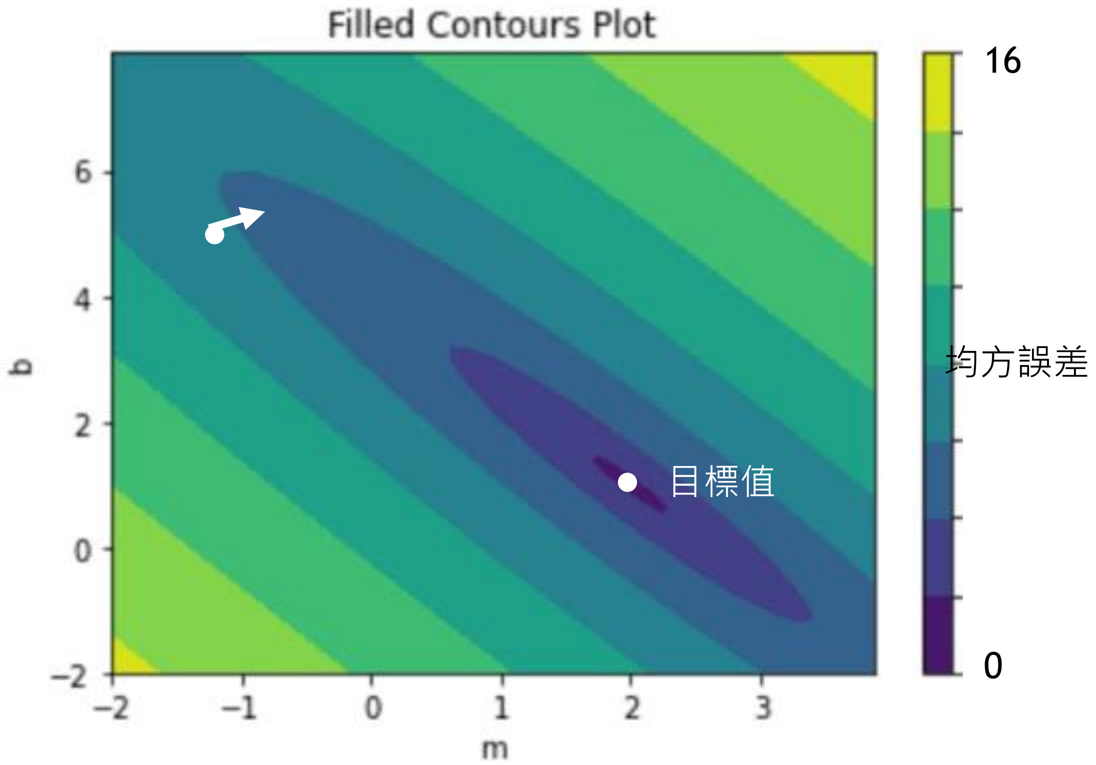
$m = 0$

$b = 4$


Filled Contours Plot
現在
目標
b
6
4
2
0
−2
−2
−1
0
1
2
3
m
16
MSE
0

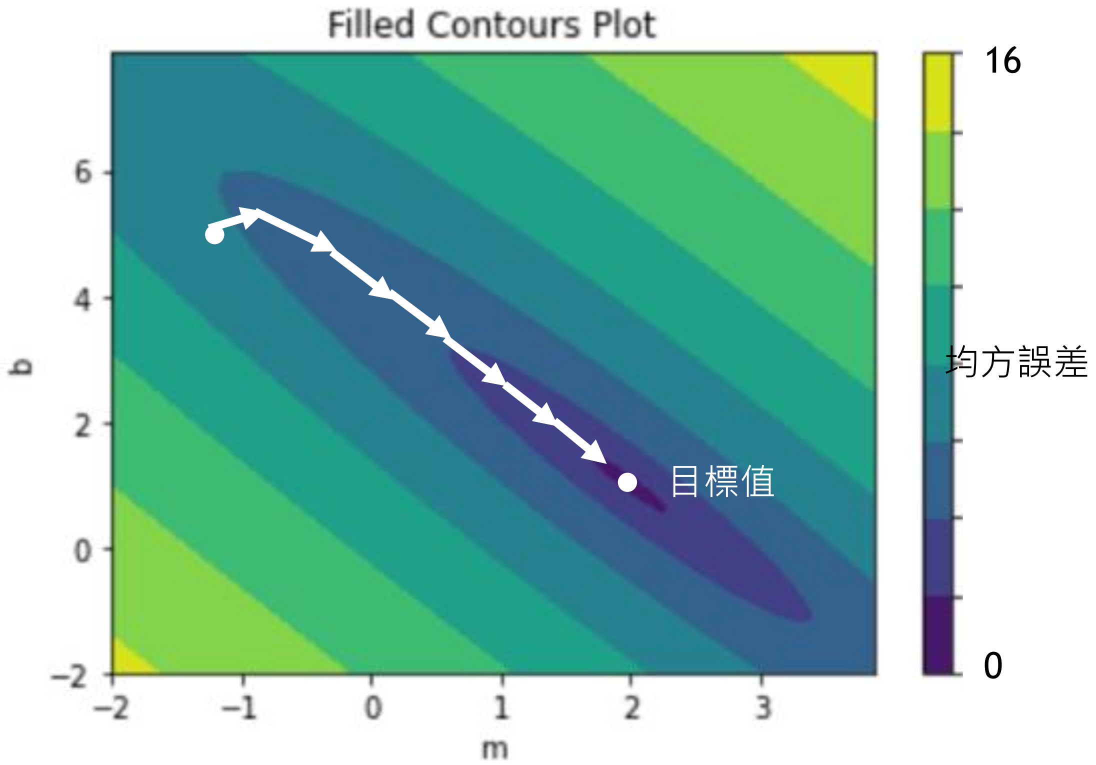
# 損失曲線

| | |
|---|---|
| 梯度 | 哪個方向的損失率降低最多 |
| λ：學習率 | 移動距離 |
| Epoch | 以完整資料集進行模型更新 |
| 批次 | 完整資料集的範例 |
| 步驟 | 更新權重參數 |

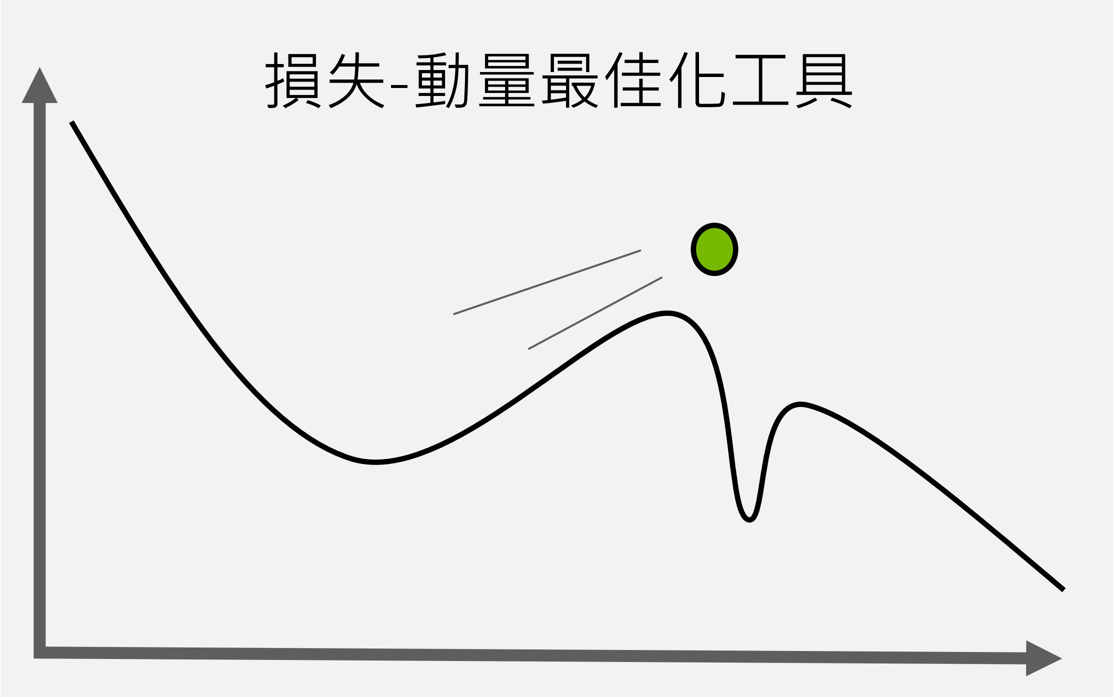


# 損失曲線

| | |
|---|---|
| 梯度 | 哪個方向的損失率降低最多 |
| λ：學習率 | 移動距離 |
| Epoch | 以完整資料集進行模型更新 |
| 批次 | 完整資料集的範例 |
| 步驟 | 更新權重參數 |



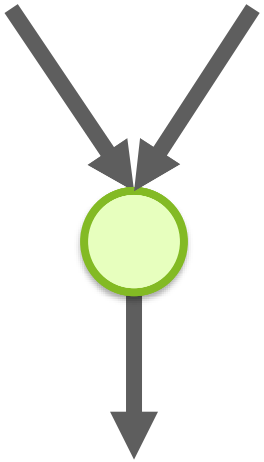
# 最佳化工具




- Adam
- Adagrad
- RMSprop
- SGD

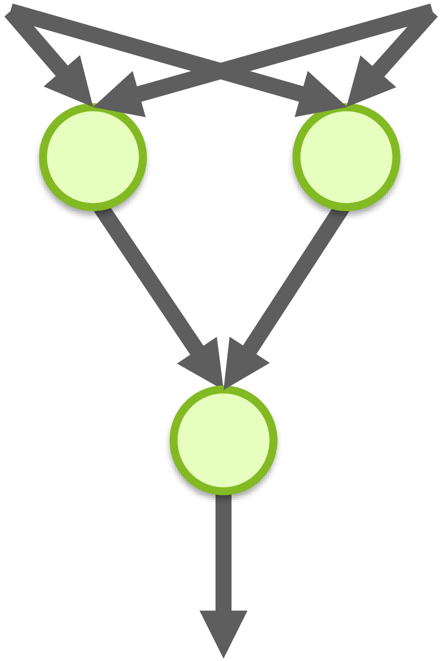
# 從神經元到網路

# 建立網路



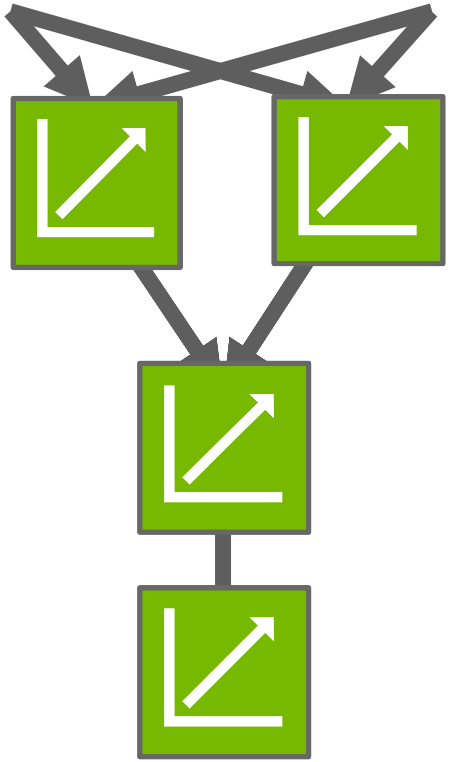
- 擴充為更多輸入值

# 建立網路



- 擴充為更多輸入值
- 可以連結神經元

# 建立網路



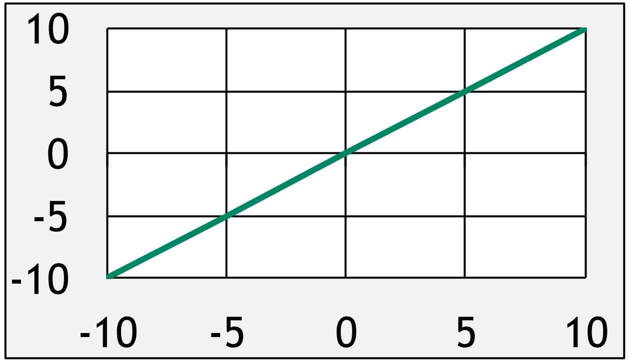
- 擴充為更多輸入值
- 可以連結神經元
- 如果所有迴歸都是線性的，則輸出也會是線性迴歸

# 激活函數

# 激活函數

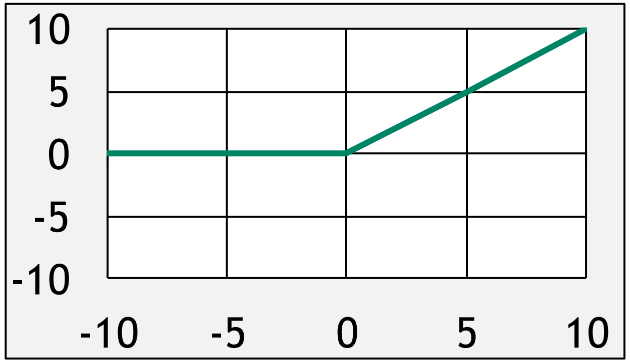
## 線性

```
# Multiply each input
# with a weight (w) and
# add intercept (b)
y_hat = wx+b
```



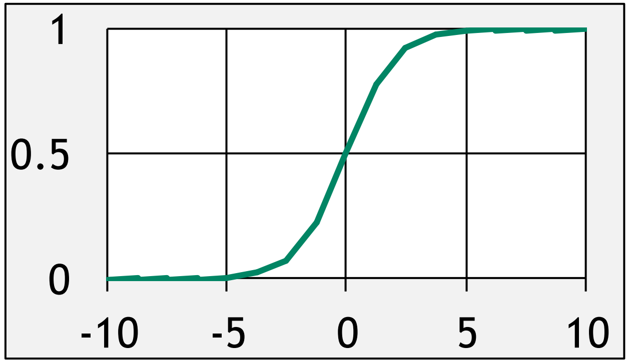

## ReLU

```
# Only return result
# if total is positive
linear = wx+b
y_hat = linear * (linear > 0)
```



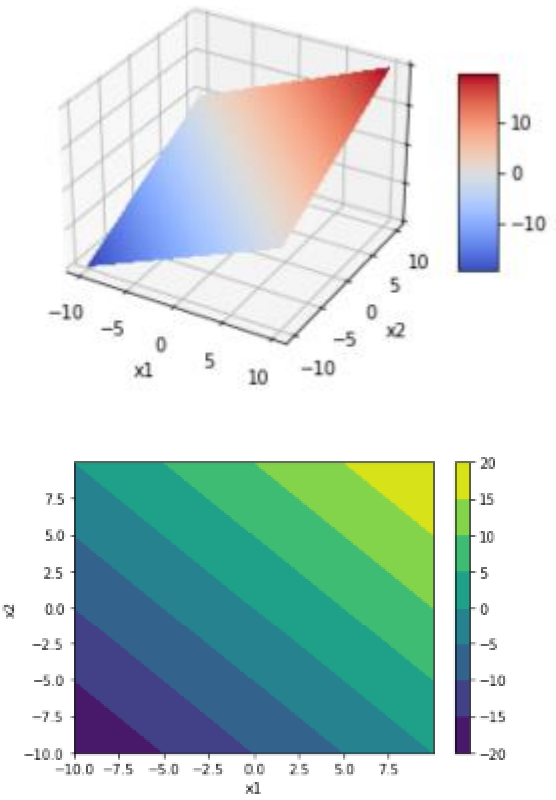

## Sigmoid

```
# Start with line
linear = wx + b
# Warp to - inf to 0
inf_to_zero = np.exp(-1 * linear)
# Squish to -1 to 1
y_hat = 1 / (1 + inf_to_zero)
```

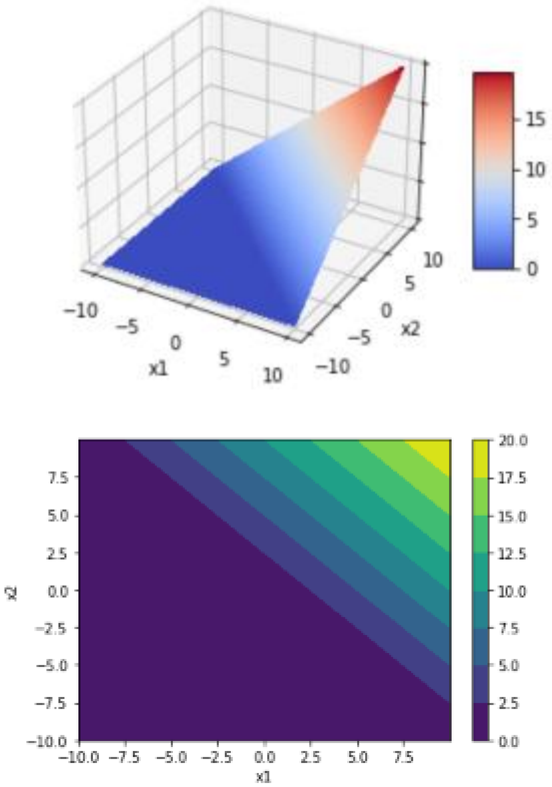


# 激活函數

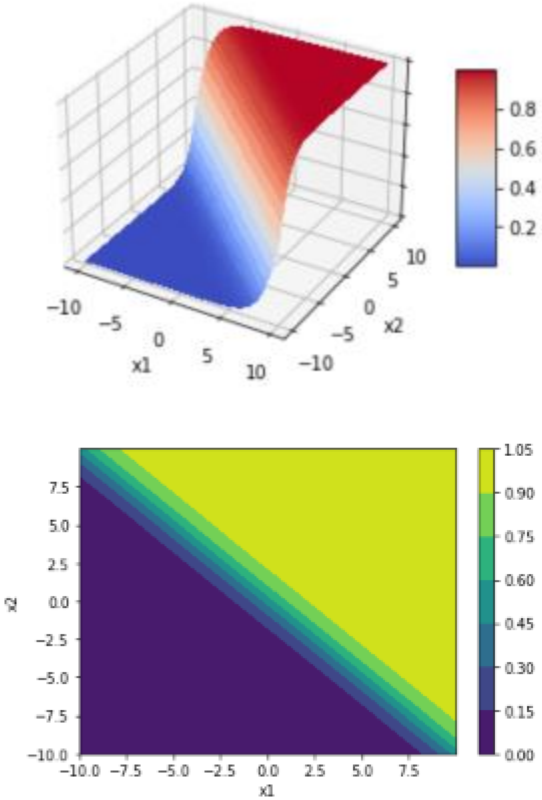
## Linear



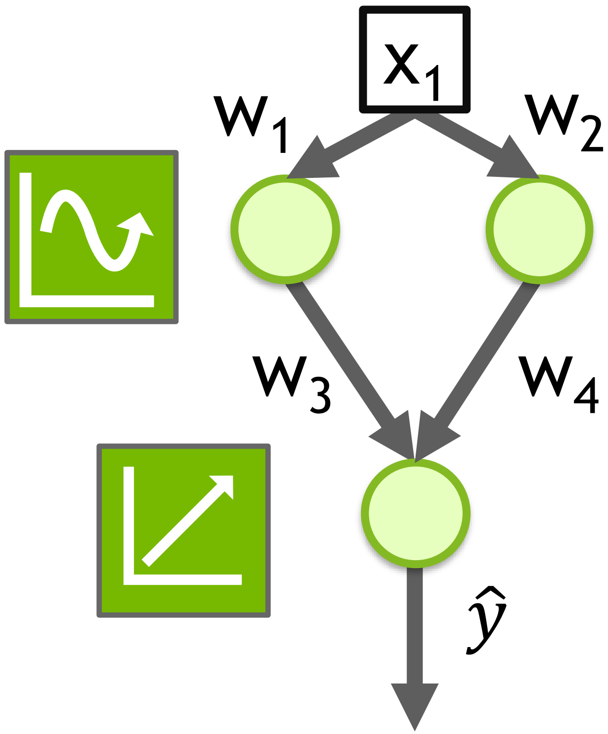
## ReLU

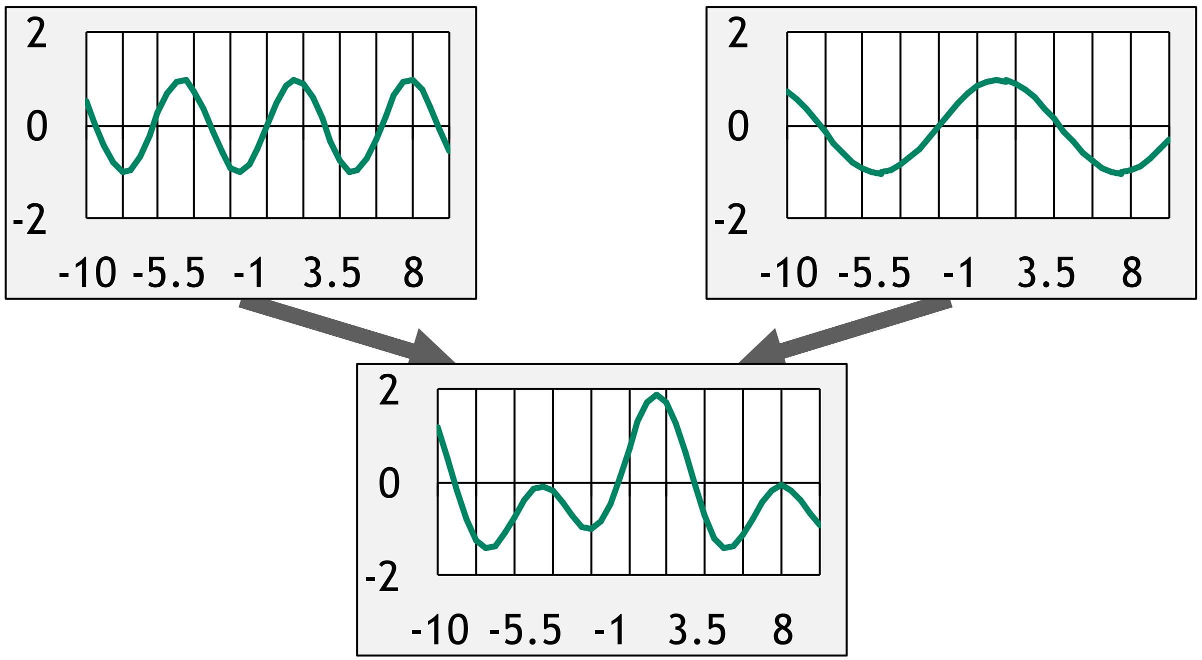


## Sigmoid



# 激活函數



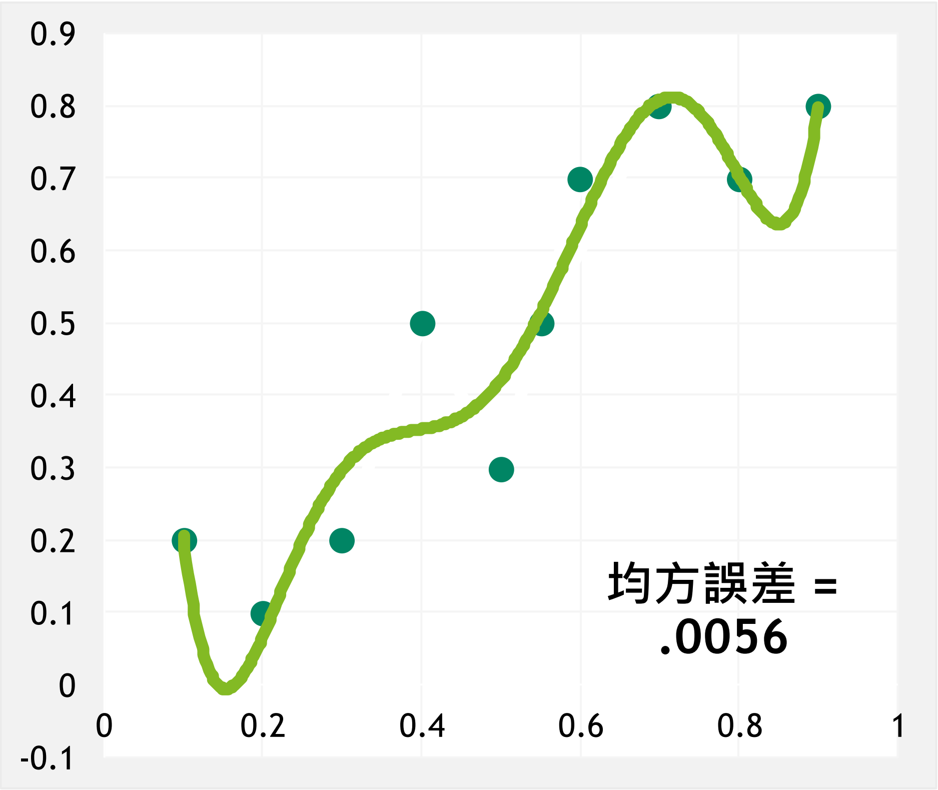
# 過度擬合

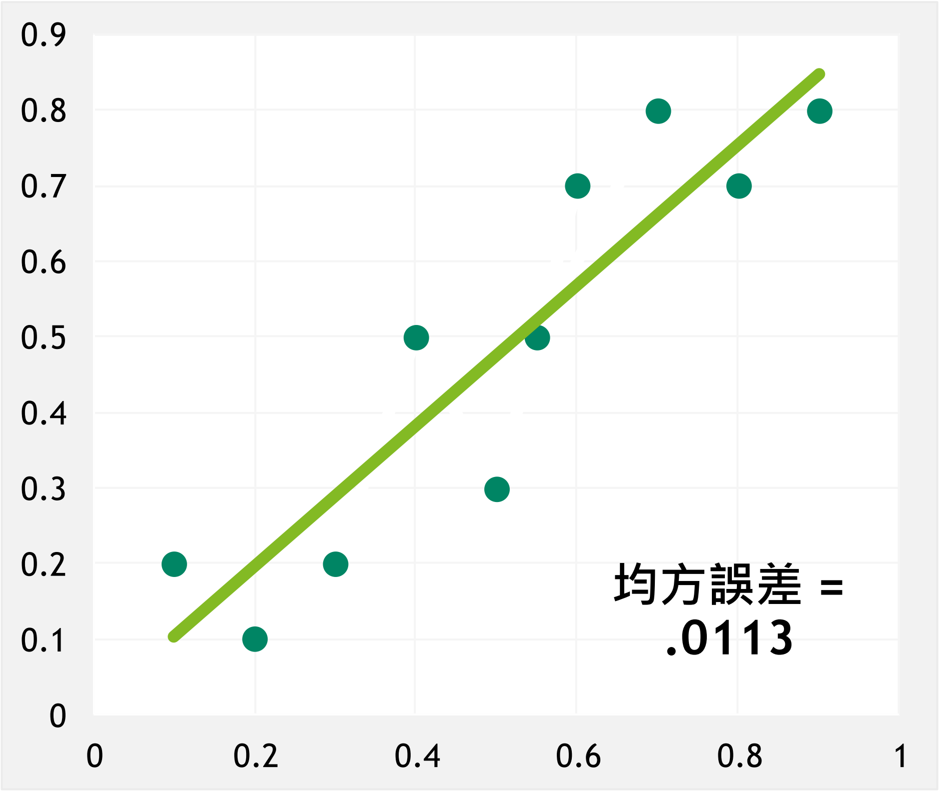
# 過度擬合

## 為何不建構一個超大型神經網路？

# 過度擬合

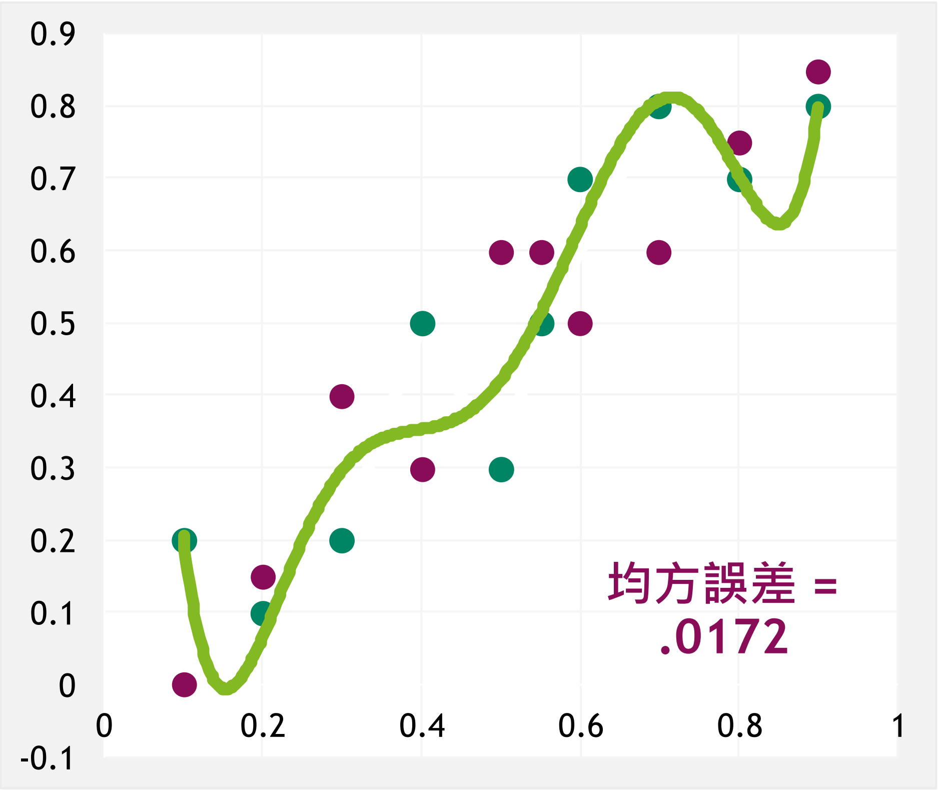
## 哪條趨勢線更理想？

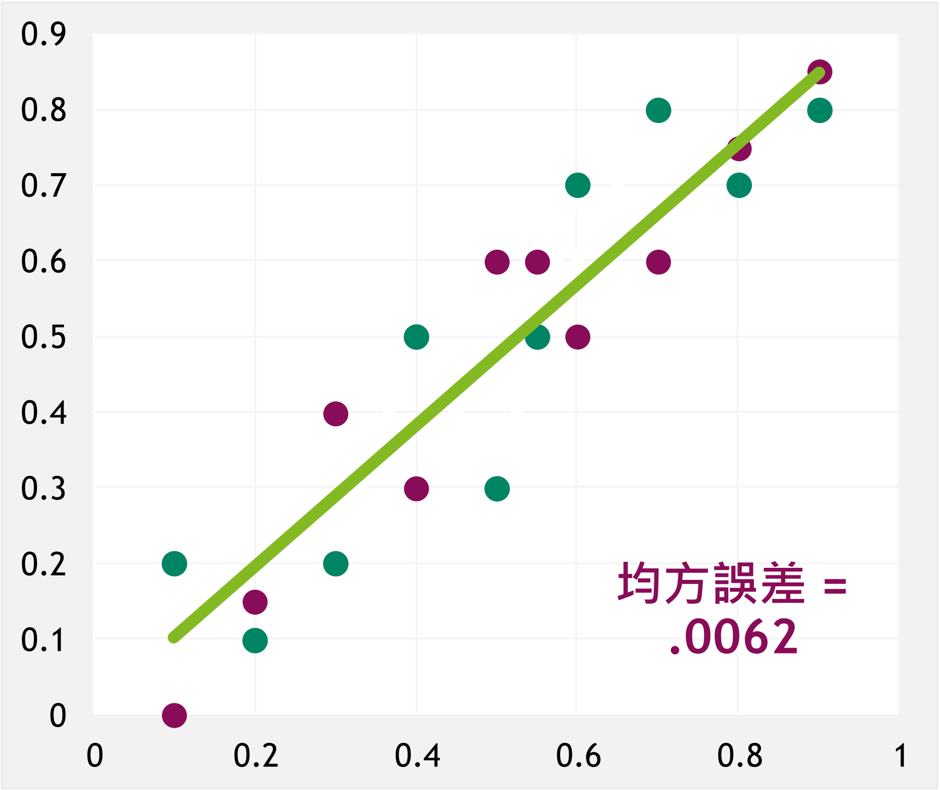


# 過度擬合

## 哪條趨勢線更理想？



# 訓練 VS 驗證資料

## 避免背誦

### 訓練資料

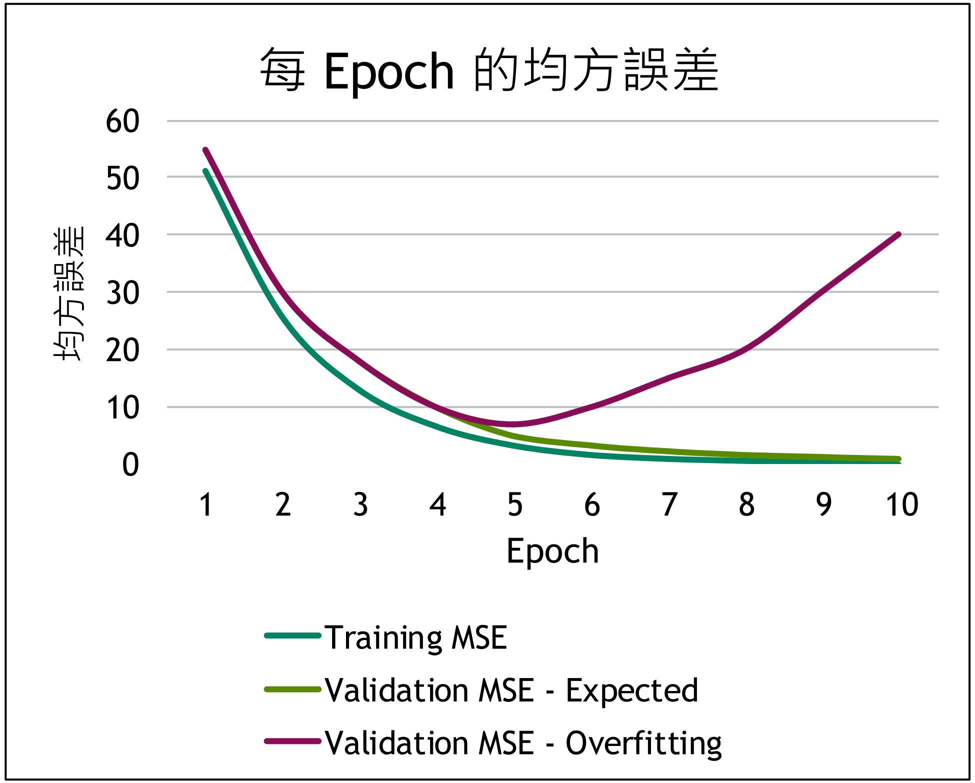
- 讓模型進行學習的核心資料集

### 驗證資料

- 用來測試模型是否真的瞭解規則的新資料 (可歸納)

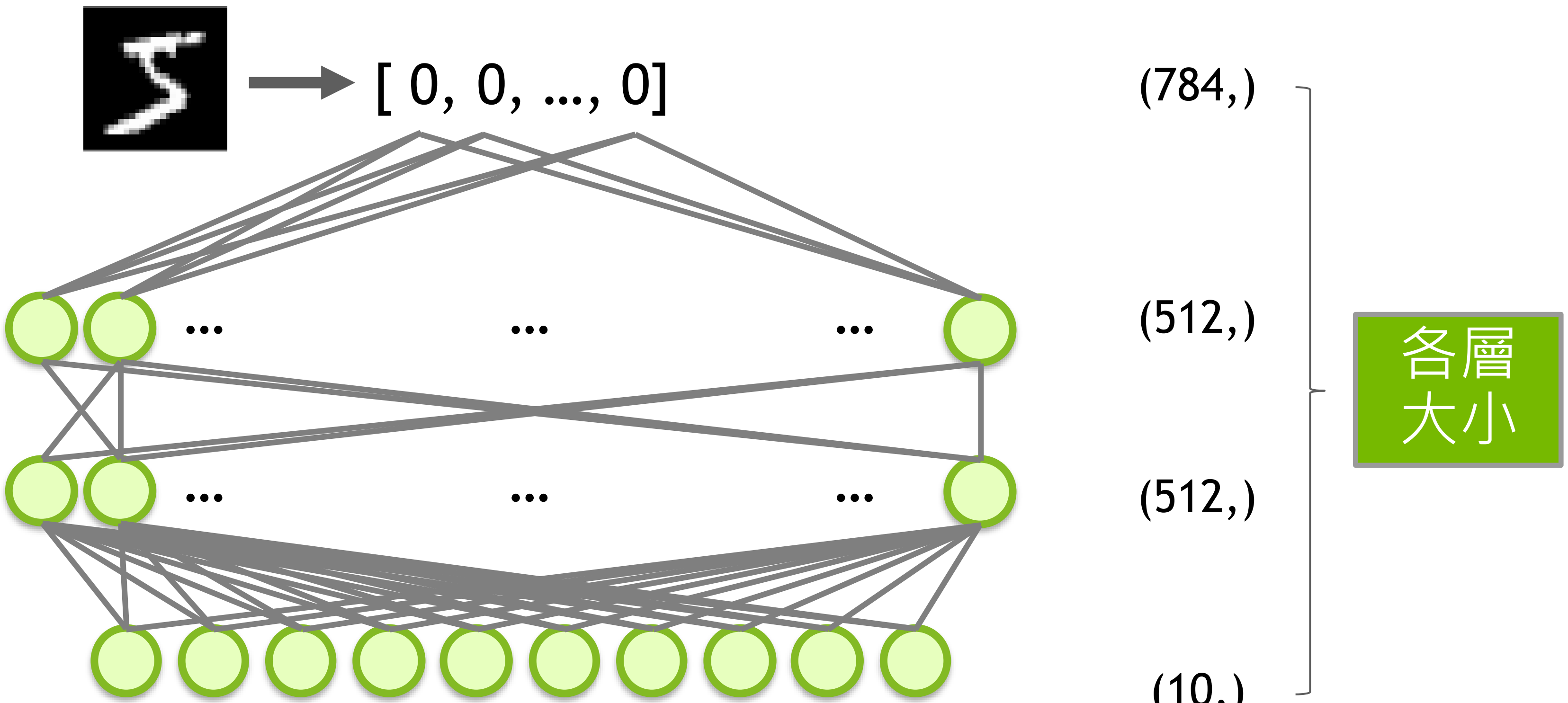
### 過度擬合

- 模型在訓練資料上效果很好，在驗證資料上卻效果不佳 (背誦的證據)
- 理想情況下，兩個資料集的精確度和損失率應該趨於接近

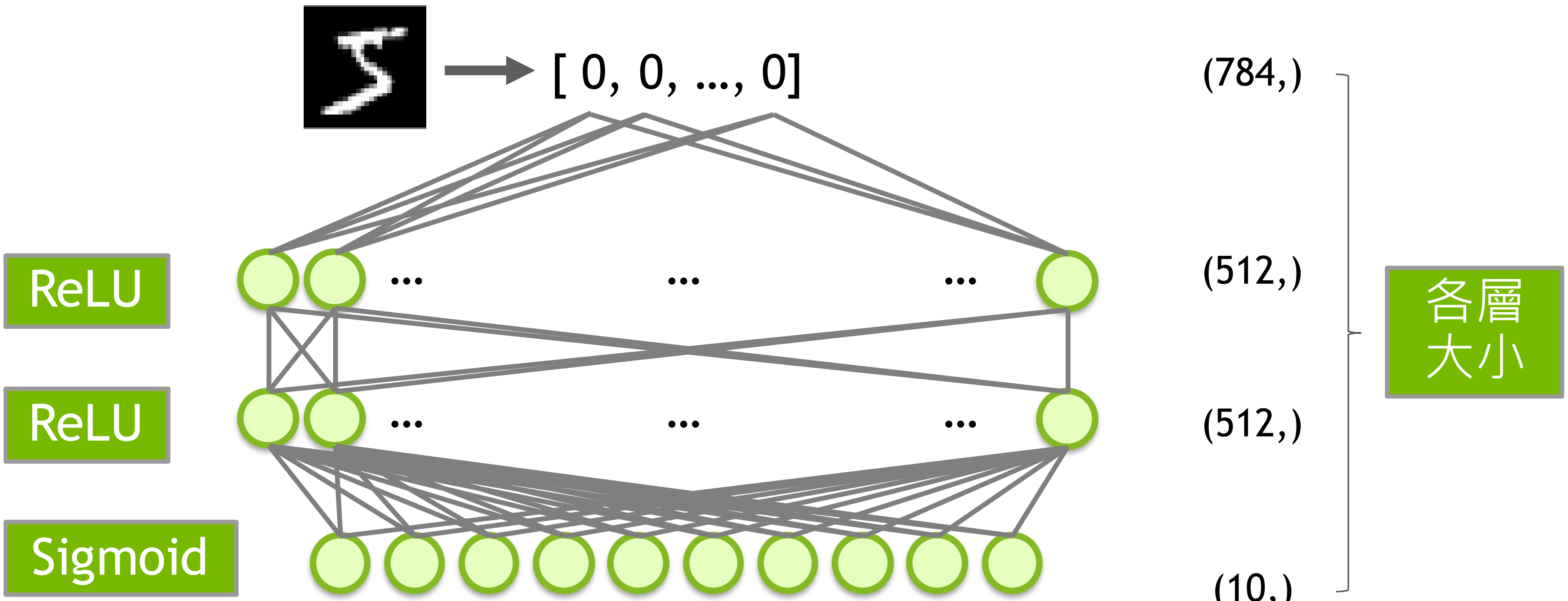


# 從迴歸到分類

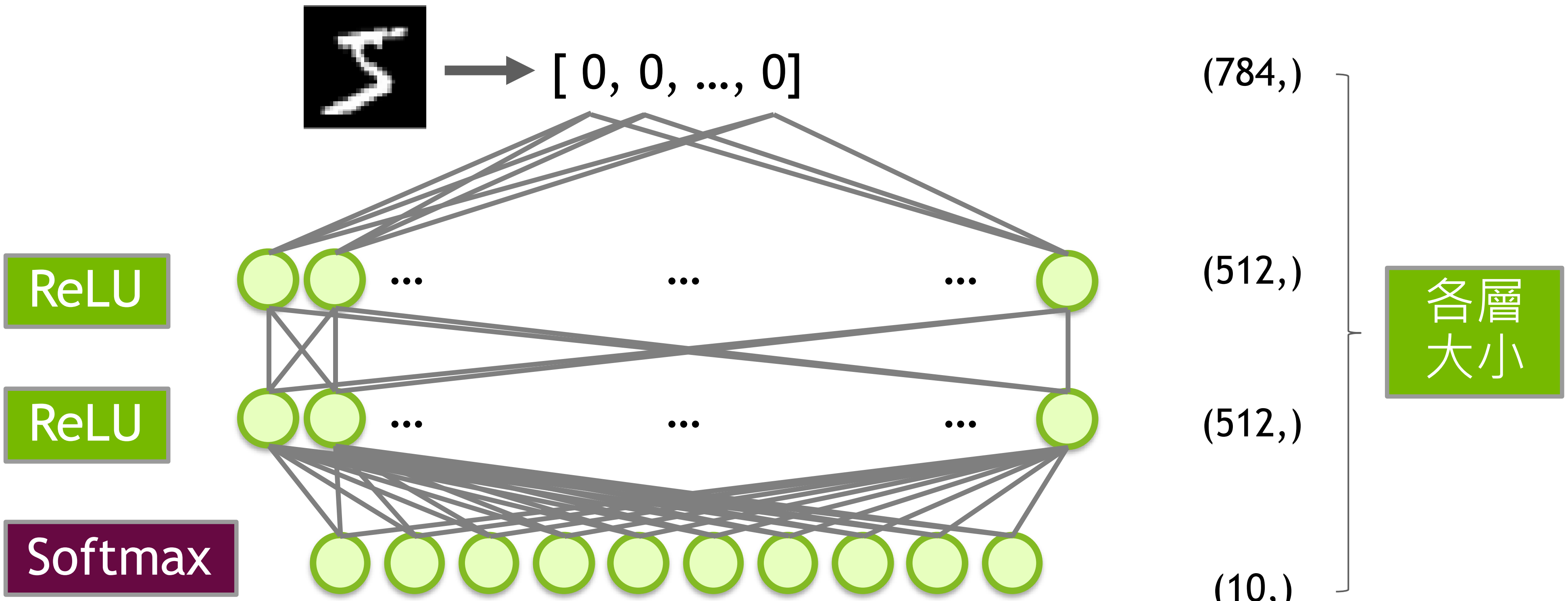
# MNIST 模型



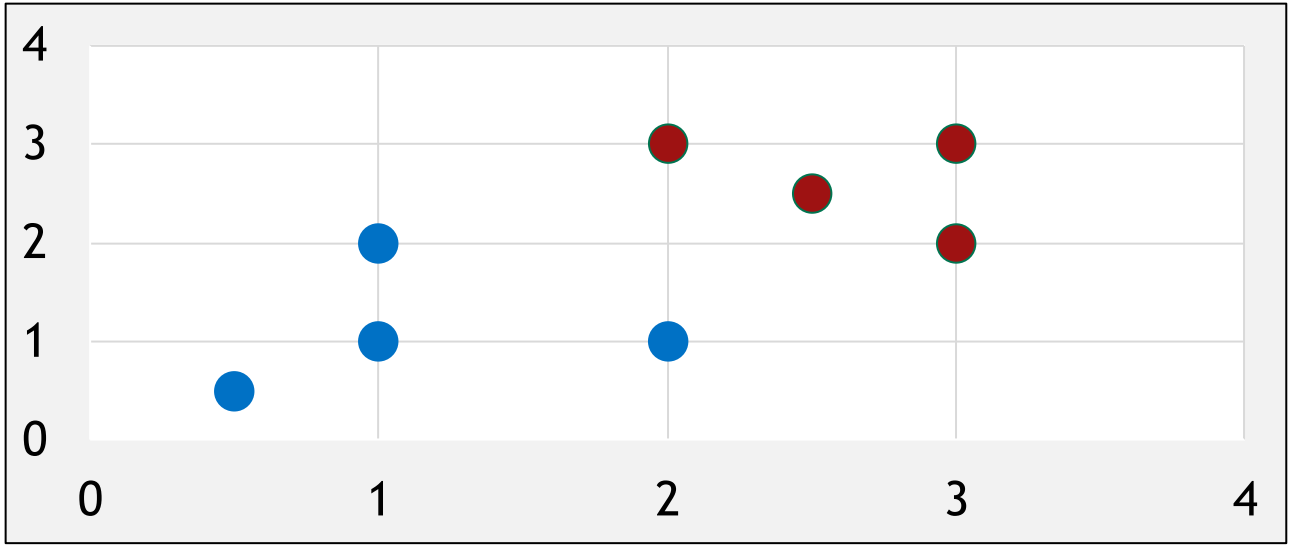
# MNIST 模型



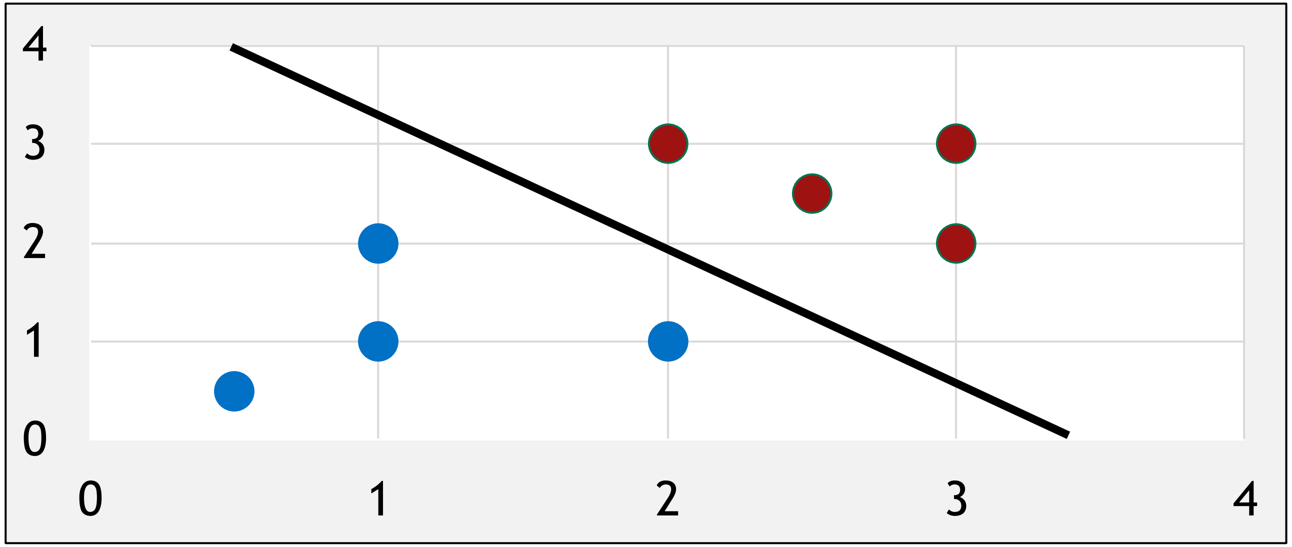
# MNIST 模型



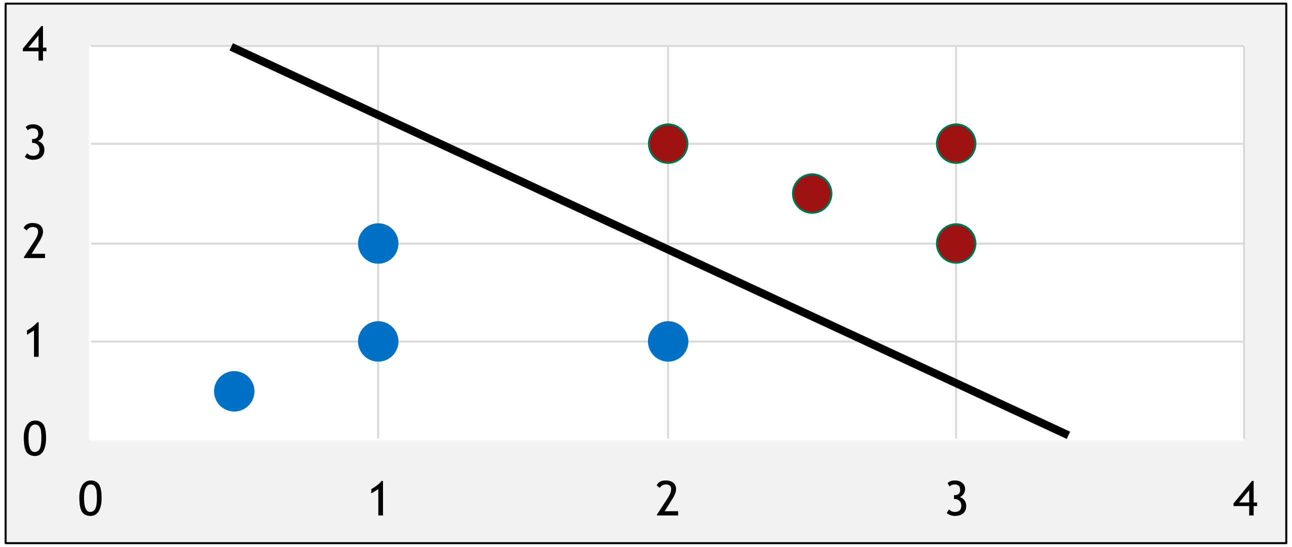
# 使用均方根誤差預測機率？



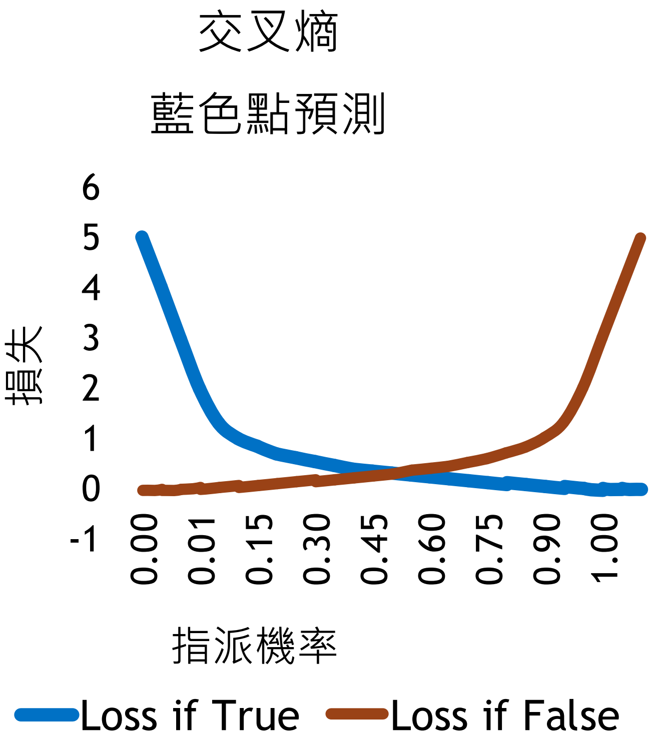
# 使用均方根誤差預測機率？



# 交叉熵

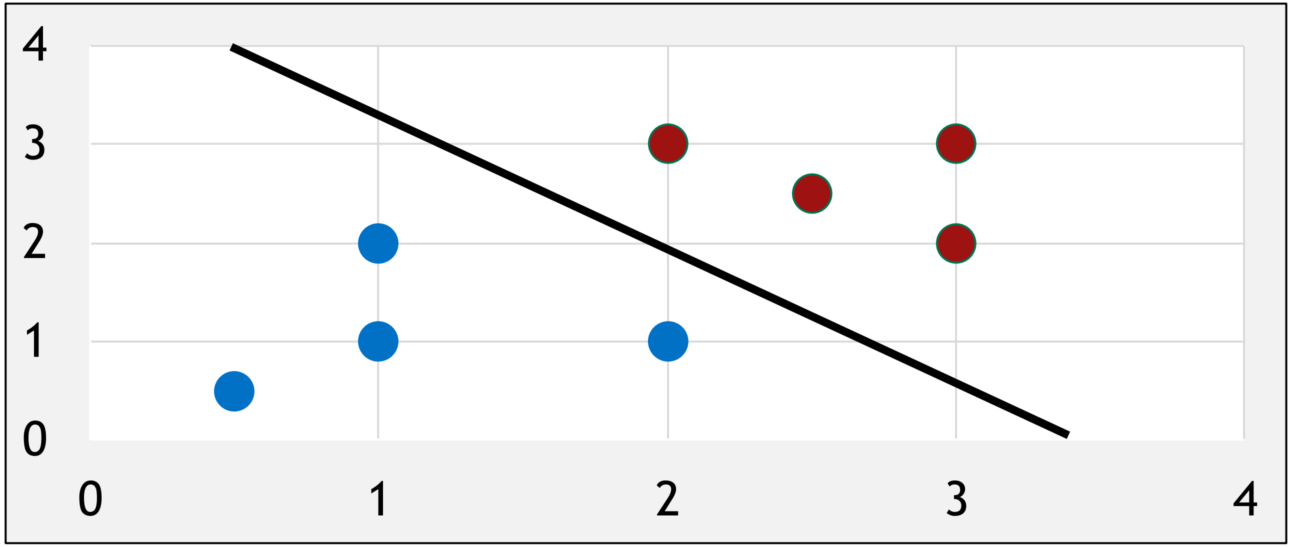


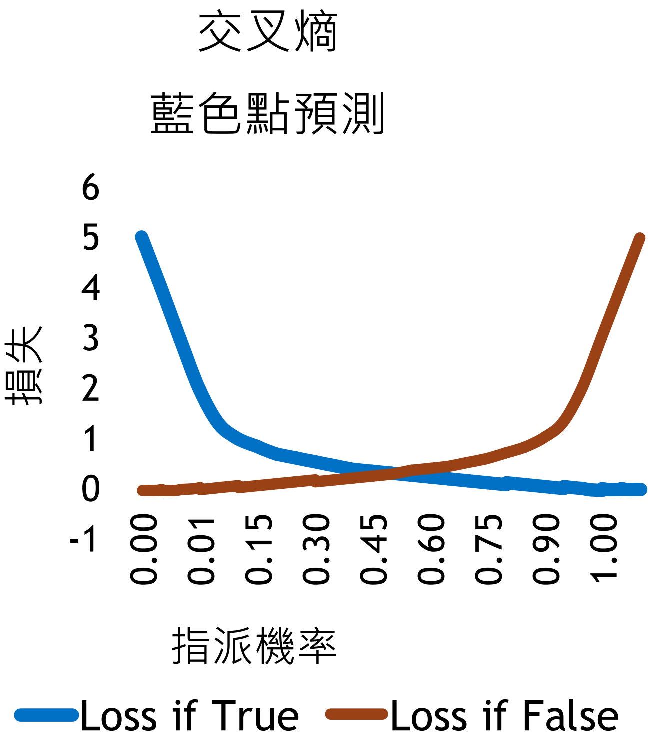
4
3
2
1
0
0
1
2
3
4

交叉熵
藍色點預測
損失
6
5
4
3
2
1
0
-1
0.00
0.01
0.15
0.30
0.45
0.60
0.75
0.90
1.00
指派機率
Loss if True
Loss if False

# 交叉熵



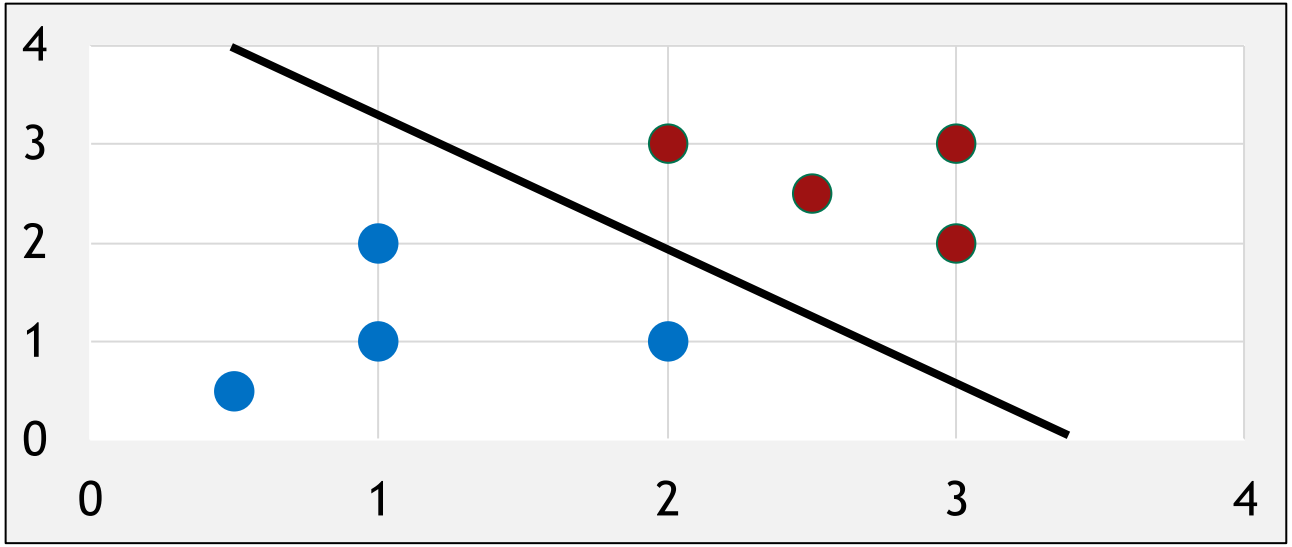




$$Loss = -(t(x) \cdot \log(p(x) + \left(1 - t(x)\right) \cdot \log(1 - p(x)))$$

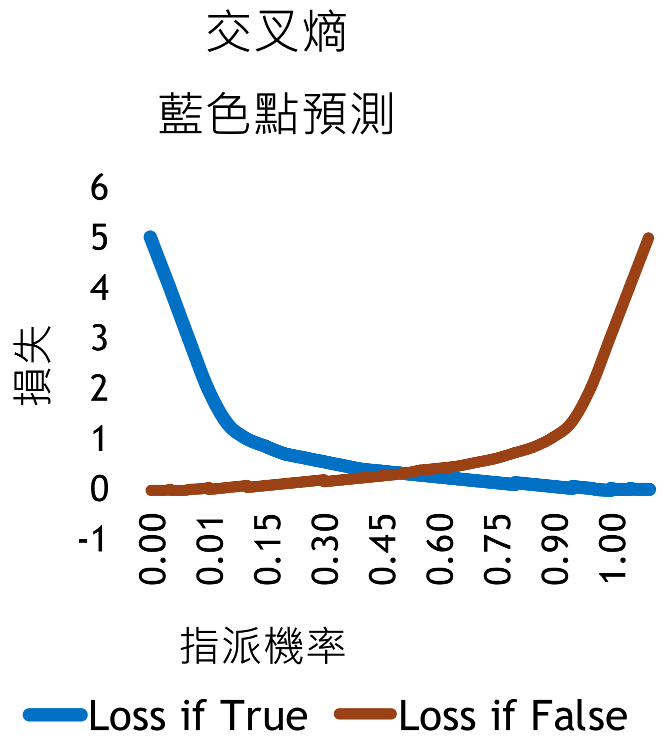
$$t(x) = target\ (\ 1\ if\ True, 0\ if\ False)$$

$$p(x) = probability\ prediction\ of\ point\ x$$

# 交叉熵



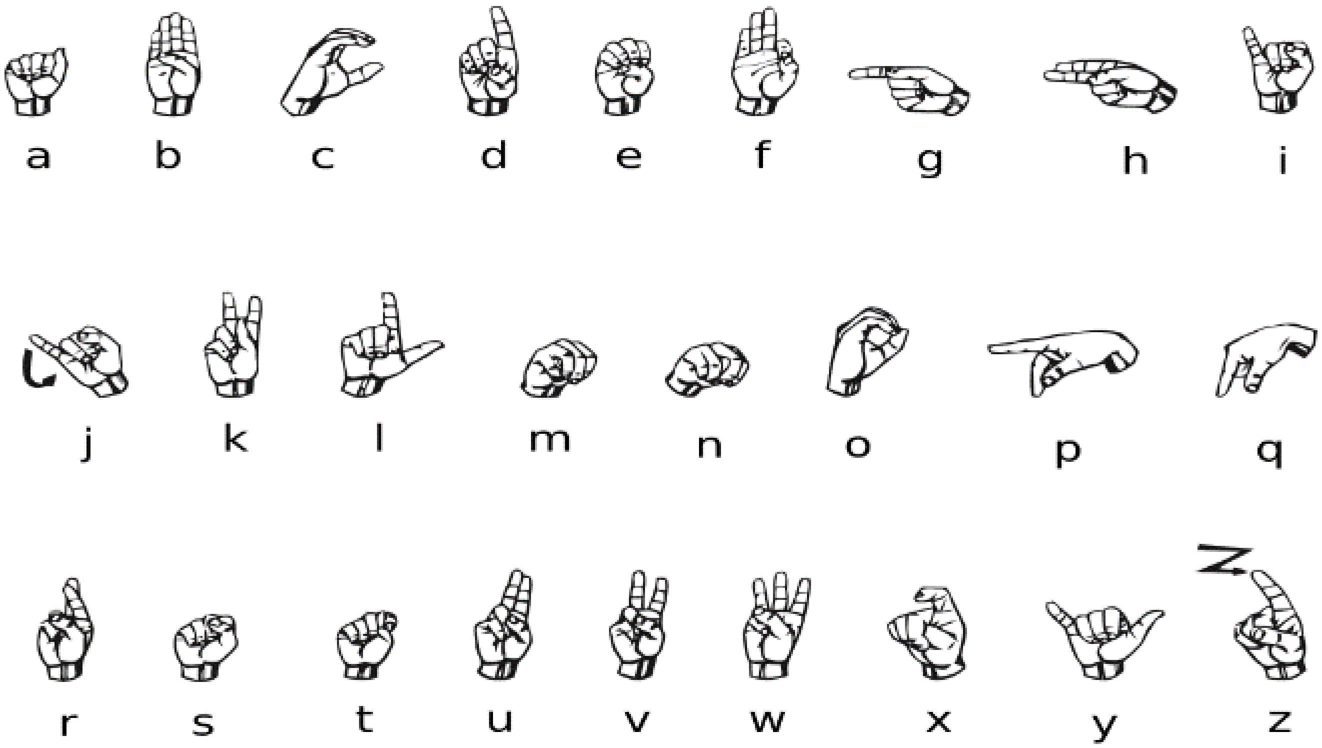
```
def cross_entropy(y_hat, y_actual):
    """Infinite error for misplaced confidence."""
    loss = log(y_hat) if y_actual else log(1-y_hat)
    return -1*loss
```



# 彙總整理

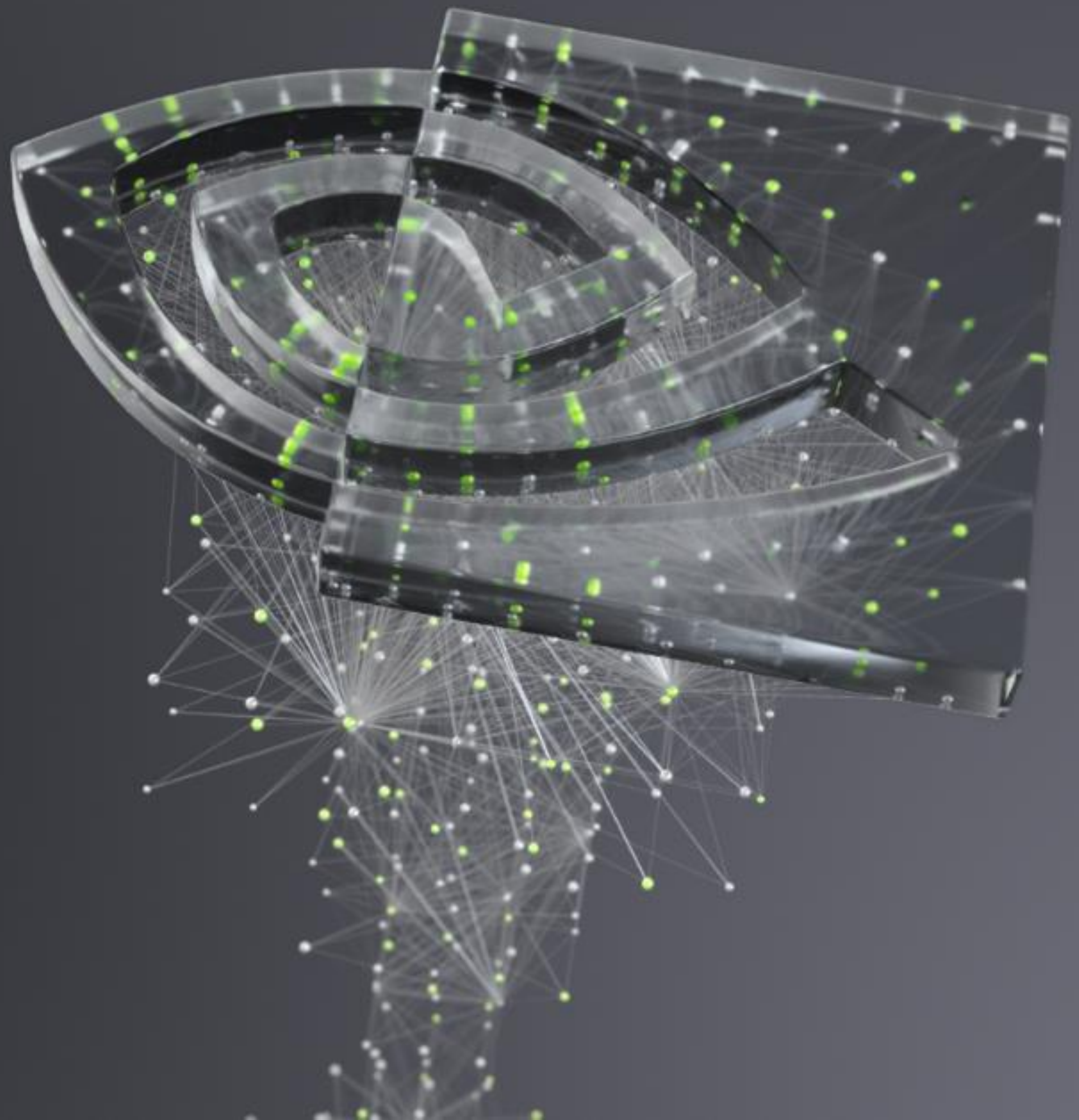
# 下一個練習

## 美國手語字母表



我們開始吧！

DEEP
LEARNING
INSTITUTE



# 附錄：梯度下降

協助電腦在微積分上作弊

# 從錯誤中學習

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y-\hat{y})^2 = \frac{1}{n}\sum_{i=1}^{n}(y-(mx+b))^2$$

$$MSE = \frac{1}{2}((3 - (m(1)+b))^2 + (5 - (m(2)+b))^2)$$

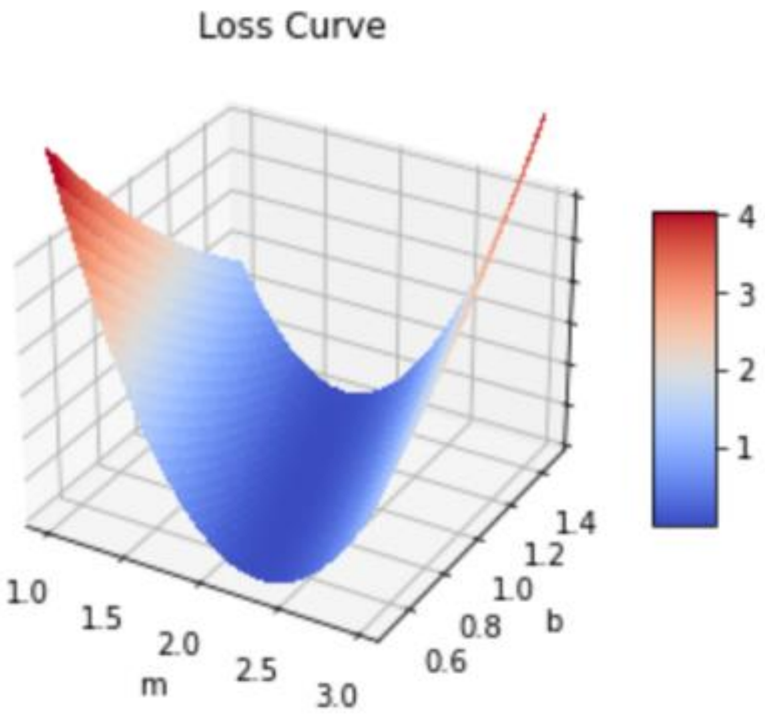
$$\frac{\partial MSE}{\partial m} = 9m + 5b - 23 \qquad \frac{\partial MSE}{\partial b} = 5m + 3b - 13$$

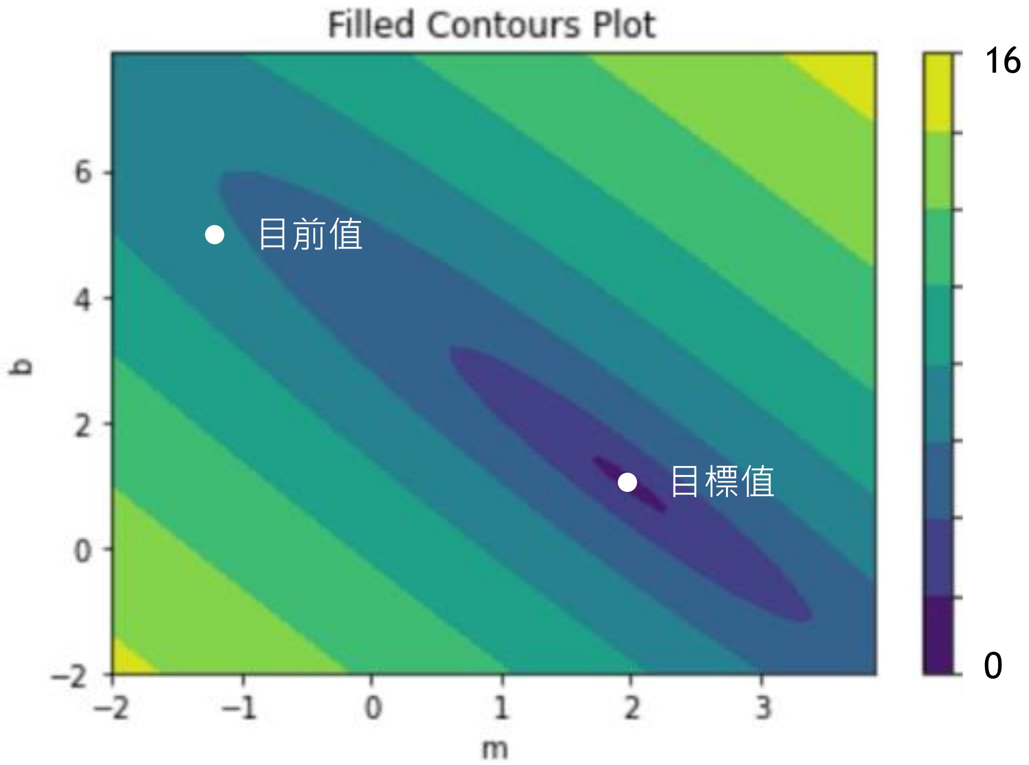
$$\frac{\partial MSE}{\partial m} = -7 \qquad \frac{\partial MSE}{\partial b} = -3$$

$m = -1$

$\mathrm{b} = 5$

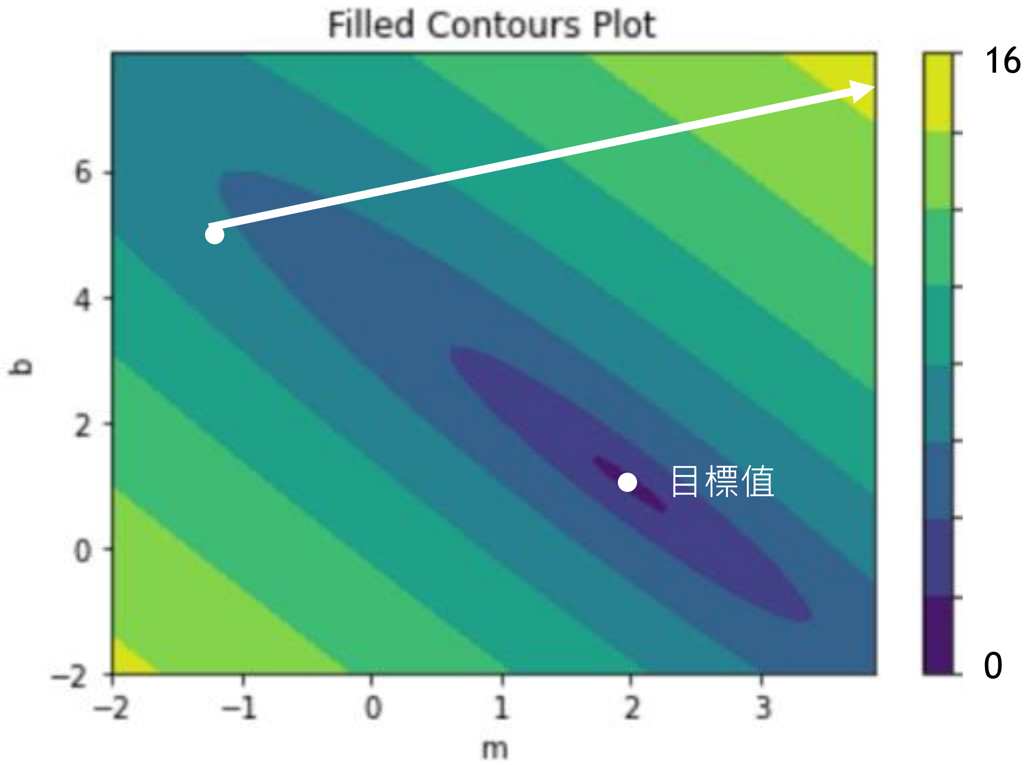
# 損失曲線


Loss Curve
1.0
1.5
2.0
2.5
3.0
m
0.6
0.8
1.0
1.2
1.4
b
1
2
3
4


Filled Contours Plot
目前值
目標值
b
m
6
4
2
0
−2
−1
1
3
16

# 損失曲線

$$\frac{\partial MSE}{\partial m} = -7 \qquad \frac{\partial MSE}{\partial b} = -3$$

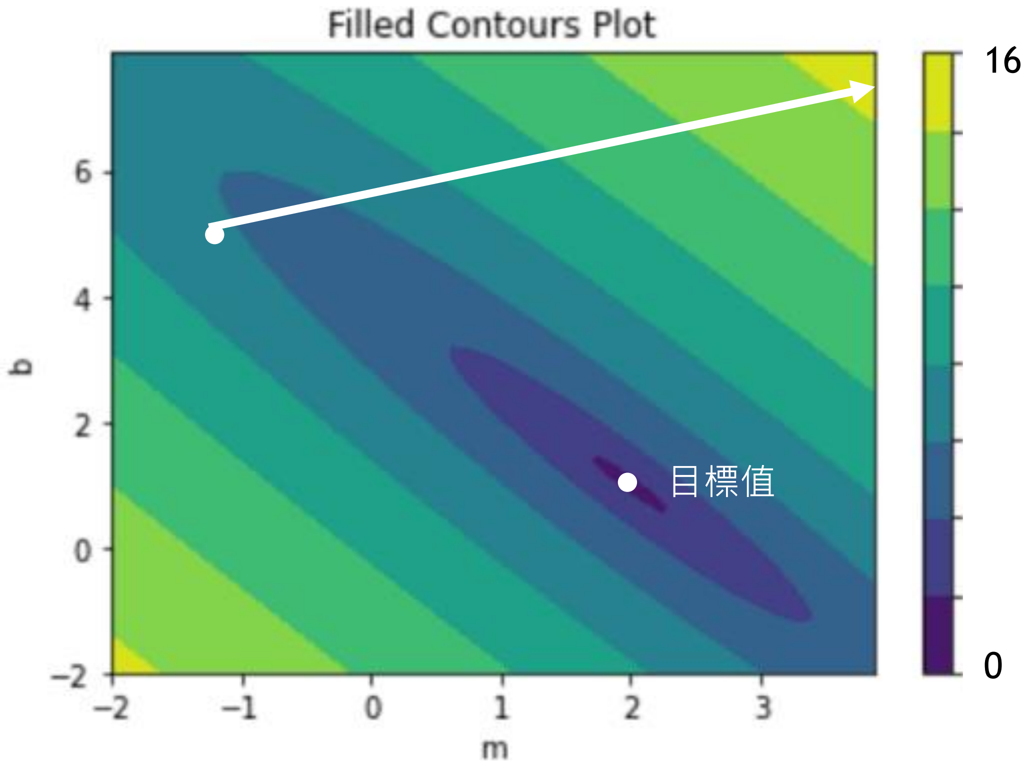


# 損失曲線

$$\frac{\partial MSE}{\partial m} = -7 \qquad \frac{\partial MSE}{\partial b} = -3$$

$$\mathrm{m} := \mathrm{m} - \lambda \frac{\partial MSE}{\partial m}$$

$$\mathrm{b} := b - \lambda \frac{\partial MSE}{\partial b}$$



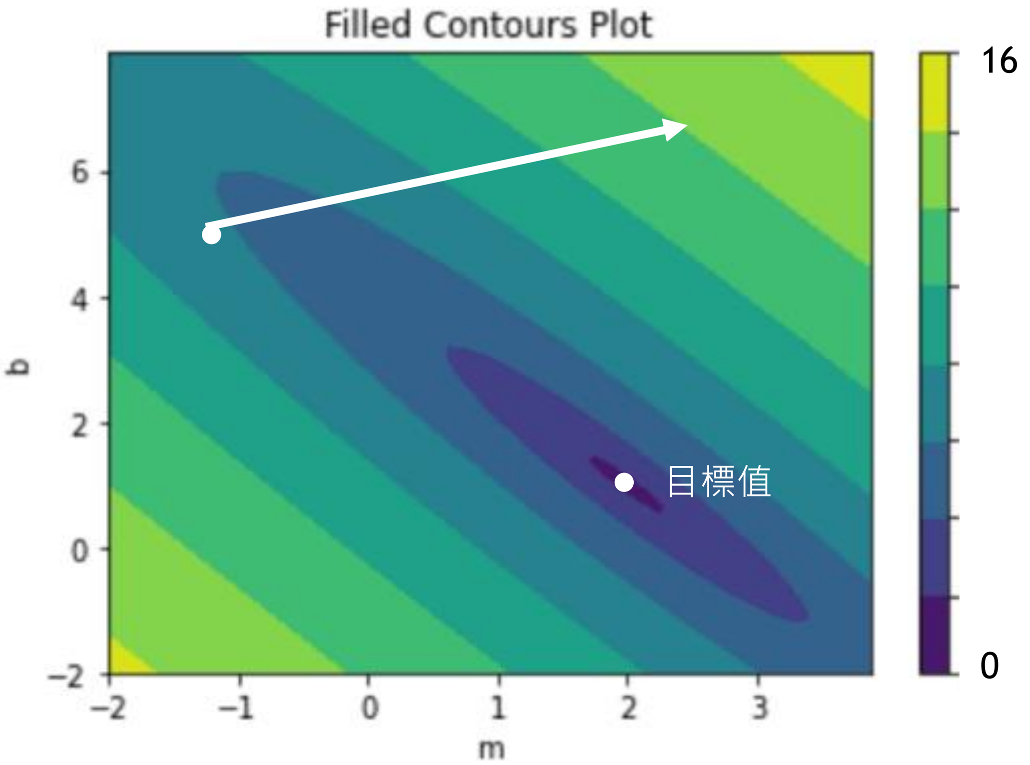
# 損失曲線

$$\frac{\partial MSE}{\partial m} = -7 \qquad \frac{\partial MSE}{\partial b} = -3$$

$$\mathrm{m} := \mathrm{m} - \lambda \frac{\partial MSE}{\partial m}$$

$$\lambda = .5$$

$$\mathrm{b} := b - \lambda \frac{\partial MSE}{\partial b}$$



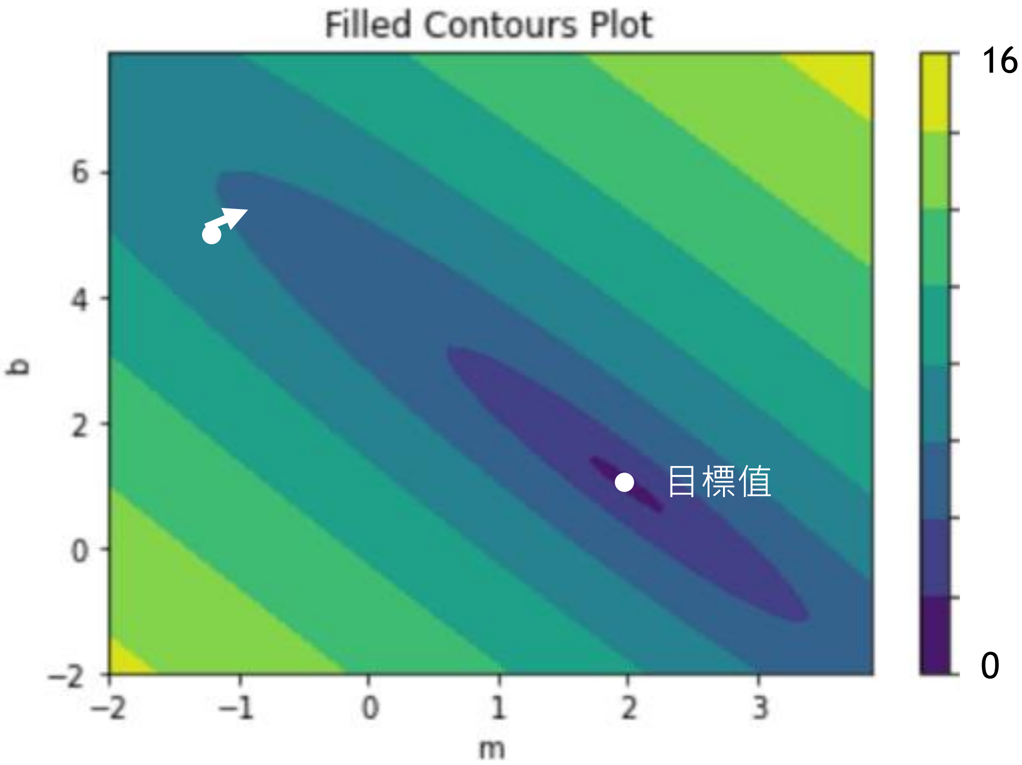
# 損失曲線

$$\frac{\partial MSE}{\partial m} = -7 \qquad \frac{\partial MSE}{\partial b} = -3$$

$$\mathrm{m} := \mathrm{m} - \lambda \frac{\partial MSE}{\partial m}$$

$$\lambda = .005$$

$$\mathrm{b} := b - \lambda \frac{\partial MSE}{\partial b}$$

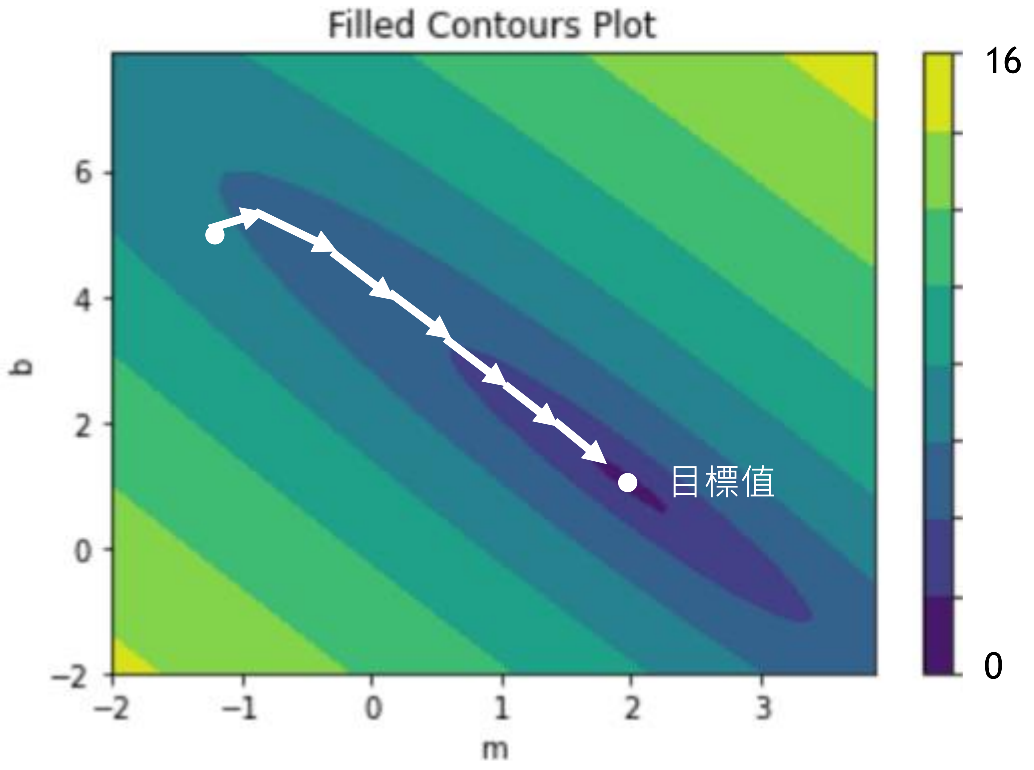


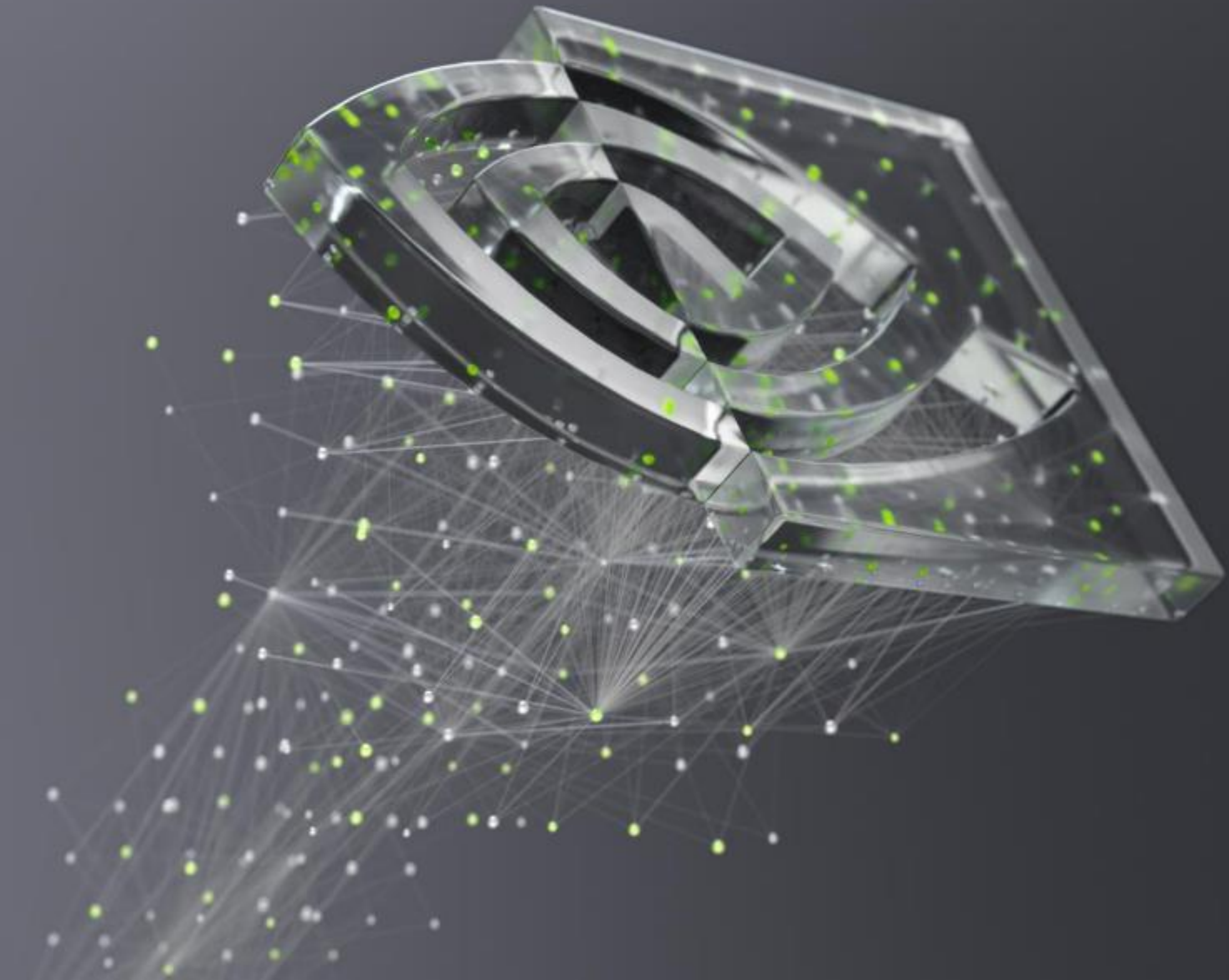
# 損失曲線

$$\lambda = .1$$

$$m := -1 \; - 7\,\lambda = -1.7$$

$$b \; := 5 \; - 3\,\lambda = 5.3$$





NVIDIA

DEEP
LEARNING
INSTITUTE