Class 2　分類模型與泛化

課程地圖

分類的機率與損失
支持向量機與泛化

- Kaggle鐵達尼號生還預測競賽
- 台積電股票預測



鐵達尼號與股票，是用來檢驗理論的兩個案例。

三堂課的共同問題

課程地圖

- Class 1：已知函數與樣本，理解參數如何更新。
- Class 2：答案有不確定性，學會建模與公平評估。
- Class 3：輸入保留時間順序，再研究 RNN / LSTM。

課程	輸入	重點
Class 1	取樣x / 曲線y 由Loss更新權重	梯度下降 與反向傳播
Class 2	乘客特徵 / 股票各 項公開指標	分類與泛化
Class 3	連續多日的視窗	RNN / LSTM

模型、資料與評估方式要一起考慮。

第一段　複習：模型怎麼學

章節導覽

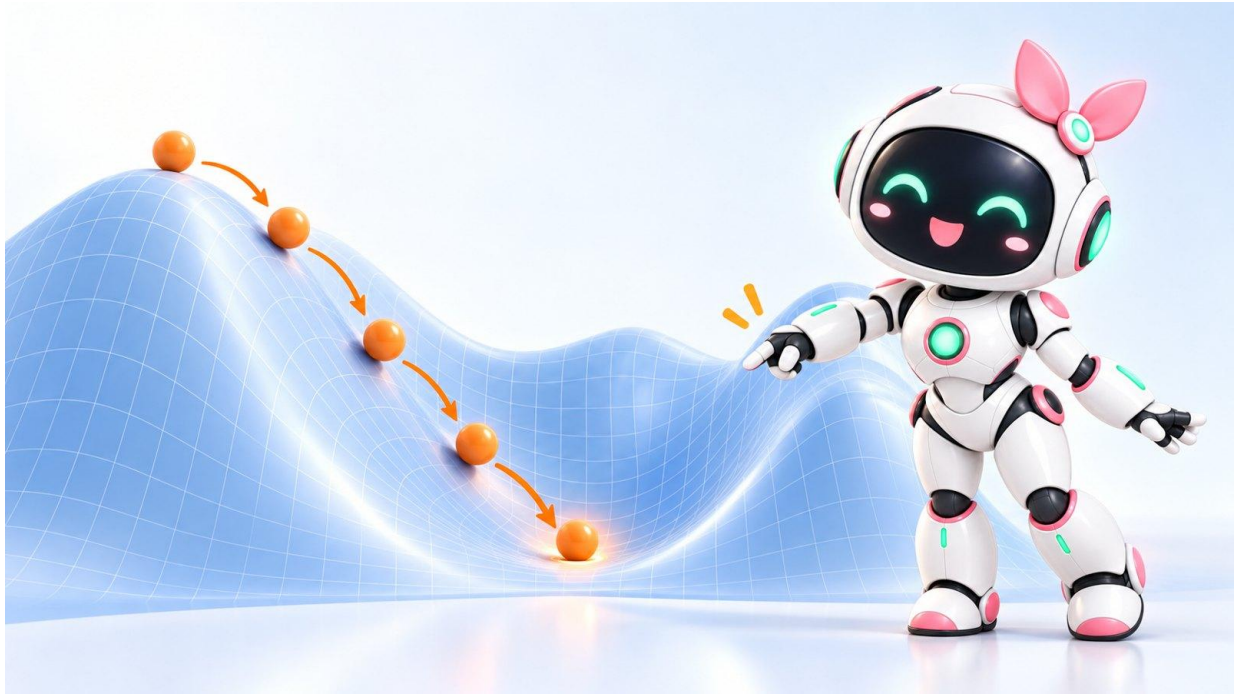
上一段

Class 1：用梯度下降，讓曲線逐步貼近資料點。

這一段要學習

複習如何合成一個正確（低損失）的函數？

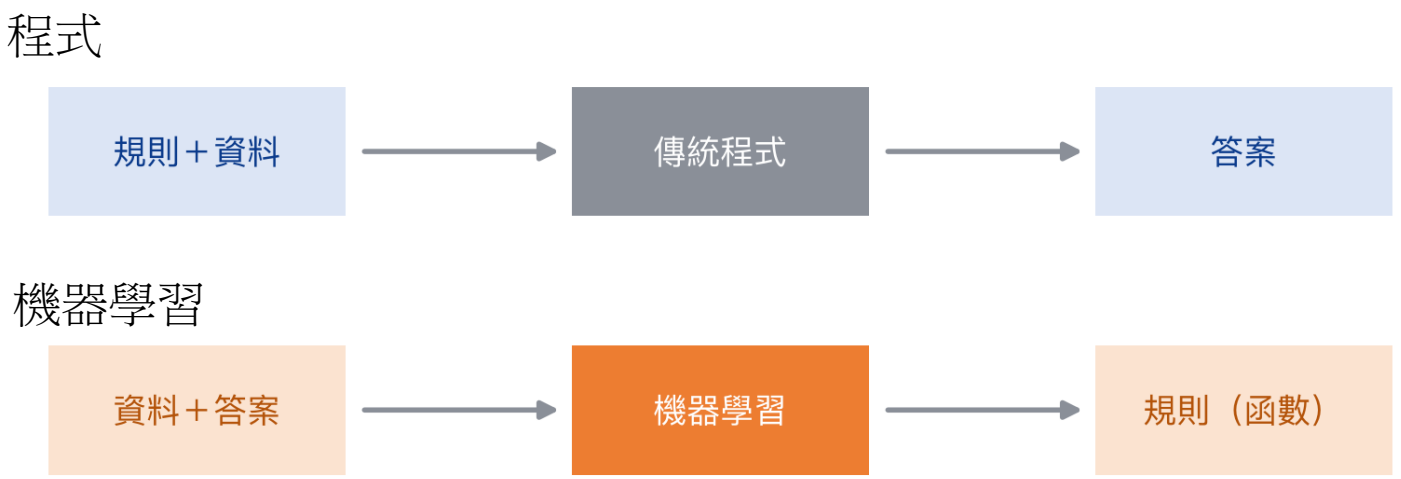
1. 參數與超參數：損失靠更新什麼降低
2. 前向計算圖與反向傳播
3. 梯度下降：更新前後比較



監督式學習的基本構成

$$\hat{y} = wx+b$$

- 每筆樣本包含特徵 x 與標籤 y。
- 模型用同一組參數 w, b 處理所有樣本
- (Class1)　訓練找出規律，預測時只能讀取 x，手動調整w, b
- (Class1) 相同的x 透過**梯度下降 * 學習率η** 透過**反向傳播**修改w,b得到更小的Loss, $(y-\hat{y})^2$



用已知答案學習，將函數用在尚無答案的新樣本。

降低損失要更新什麼？參數與超參數

參數 parameters　模型自己學

- 權重 w、偏差 b（神經網路裡的 W_1、b_1…）
- 由資料透過梯度下降自動學出
- 每更新一次，損失通常就降一點

本例　w = 1,　b = 0

超參數 hyperparameters　人事先設定

- η (學習率)、batch (批次)、epoch (輪次)、隱藏層寬度
- 梯度不會改它，由人決定
- 決定怎麼更新：步長、每步看幾筆、走幾輪

本例　η = 0.1,　batch = 1,　epoch = 1

損失下降，是因為更新了參數；超參數決定這個更新怎麼進行，之後用驗證集來挑。

學習目標：最小化平均損失

核心理論

$$L(\theta) = (1/N)\ \Sigma_i\ \ell(f\theta(x_i), y_i)$$

公式參考．看懂即可

- 單筆損失 ℓ 衡量一個預測錯多少。
- 平均損失 L 把全體訓練樣本的錯誤彙整。
- 不同任務會選不同損失，並非一律用 MSE。

樣本	預測−答案	平方誤差（MSE）
A	+1	1
B	−2	4
C	+1	1

MSE = 2

共同參數必須兼顧多筆資料，而不是逐筆背一組答案。

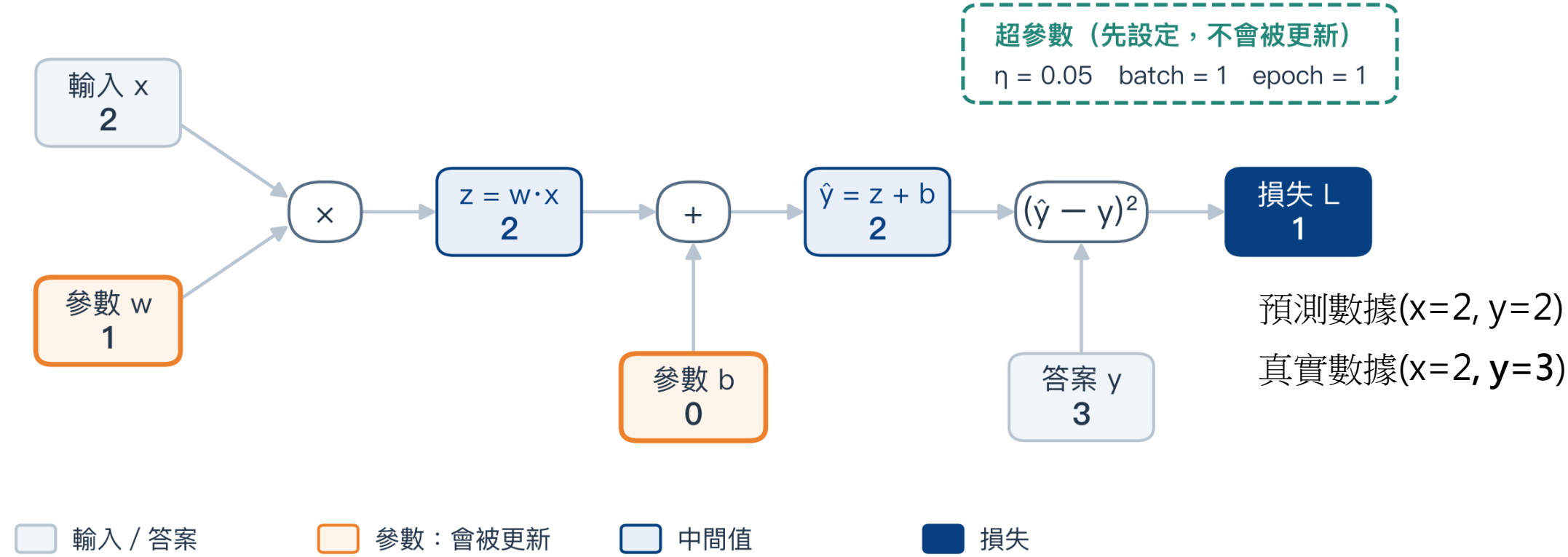
前向計算：從輸入一路算到損失

圖解

初始： y = 1 * x + 0

$z = w \cdot x$　　$\hat{y} = z + b$　　$L = (\hat{y} - y)^2$

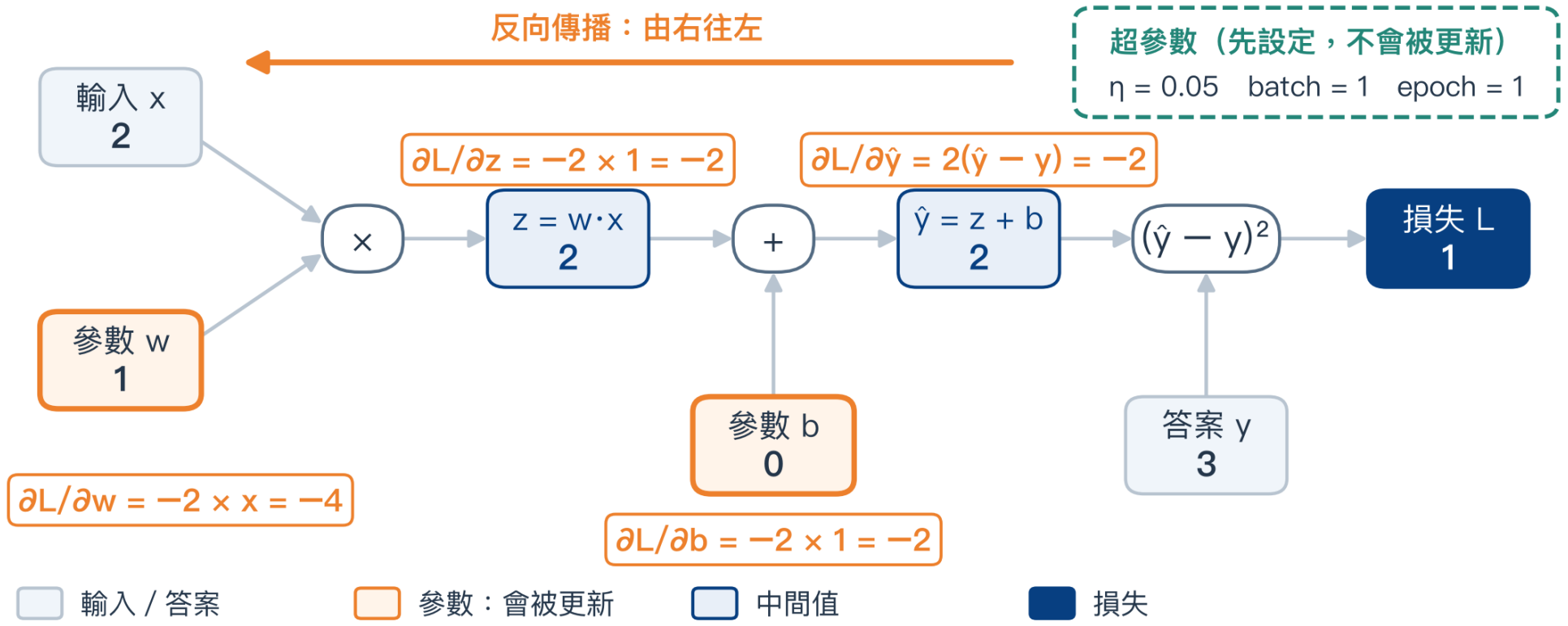
真實數據(x=2, y=3)



先標出超參數 η、batch、epoch；橘框的 w、b 是等一下要更新的參數。

反向傳播：沿同一張圖往回算梯度

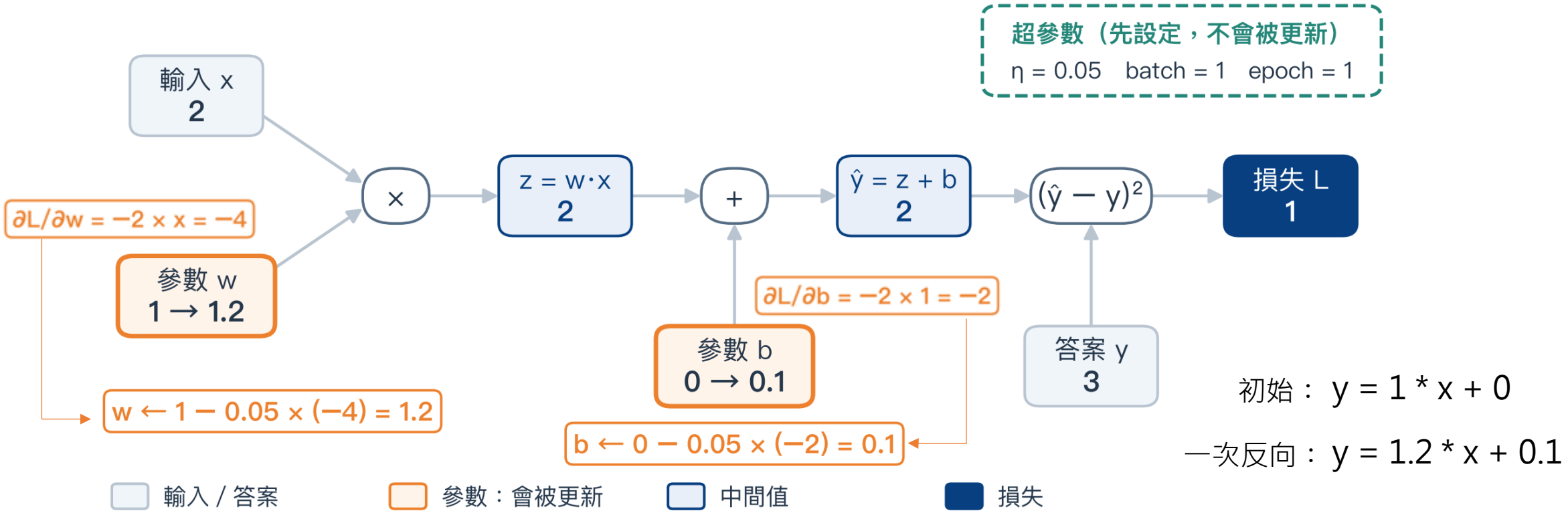
$$\partial L/\partial w = (\partial L/\partial \hat{y})(\partial \hat{y}/\partial z)(\partial z/\partial w) = (-2)(1)(2) = -4$$



每經過一條邊就乘上局部導數；b 只經過一個加法，所以 $\partial L/\partial b = (\partial L/\partial \hat{y})(\partial \hat{y}/\partial b) = -2$。

梯度下降：用梯度更新參數

$$\theta_{t+1} = \theta_t - \eta \cdot \partial L/\partial \theta \qquad \theta = (w, b)$$



初始： y = 1 * x + 0

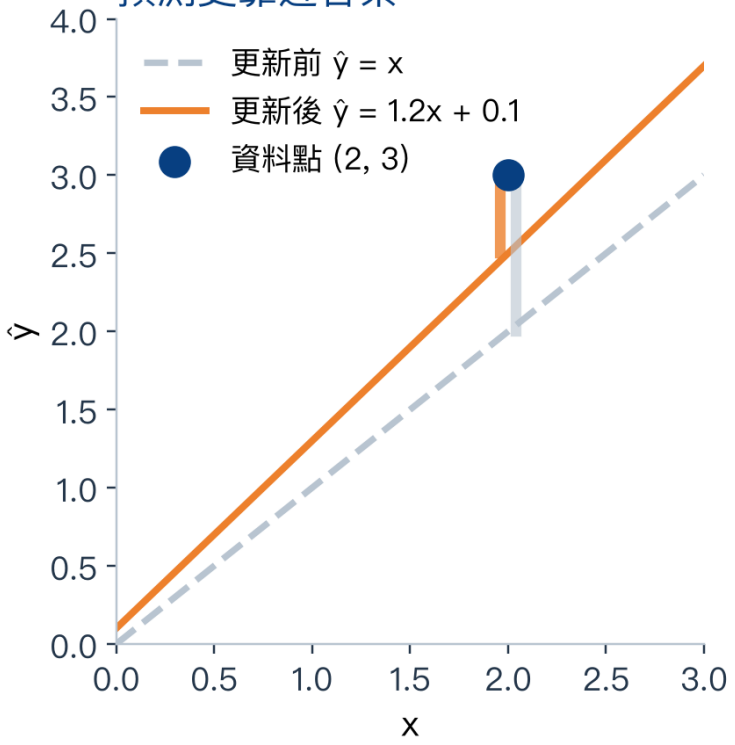
一次反向： y = 1.2 * x + 0.1

學習率 η 是超參數：它決定每一步走多遠，自己不會被梯度更新。

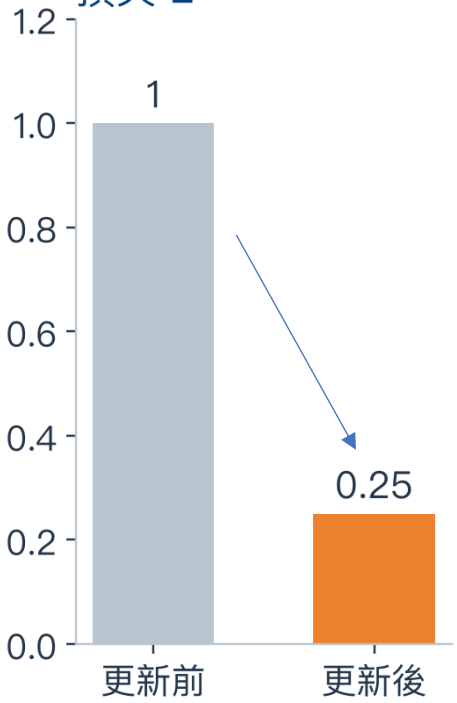
更新前後比較：損失從 1 降到 0.25

L: 1 → 0.25

預測更靠近答案



損失 L



數值對照

	更新前	更新後
w	1	1.2
b	0	0.1
ŷ	2	2.5
ŷ − y	−1	−0.5
L	1	0.25

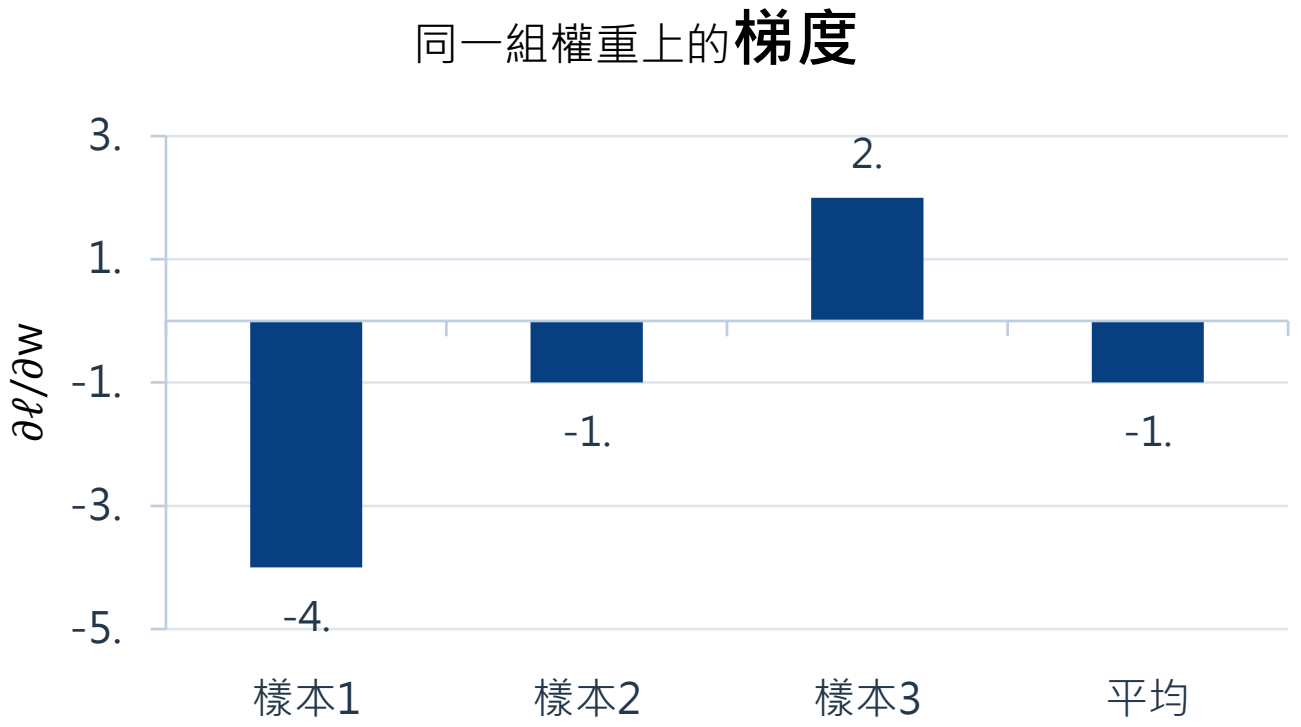
只更新一次，預測就從 2 靠近到 2.5，損失降為原來的四分之一；重複很多次就是訓練。

多筆資料如何共用一組權重

公式參考．看懂即可

$$\nabla L = (g_1 + g_2 + \dots + g_N) / N$$

- 不同樣本可能要求參數朝相反方向改。
- 先求每筆梯度，再用平均方向更新同一組參數。
- 個別樣本的損失可能升高，整體仍可能改善。



三筆梯度 −4、−1、+2，平均 −1；η=0.1 時 w 增加 0.1。

Epoch (輪次)與Batch (批次)

S (批) = ceil(N / B)　　T (總更新數) = E (輪次) × S (批)

樣本 sample	批次 batch	更新 step	回合 epoch
一位乘客	一次算梯度的一小群樣本	權重被修改一次	全部訓練資料都看過一次

320組資料

batch = 32



同樣 320 位乘客＝1 個 epoch；每一格是一批，算完就更新一次

320 ÷ 32 ＝ 10 批
1 個 epoch 更新 10 次

batch = 64



同樣 320 位乘客＝1 個 epoch；每一格是一批，算完就更新一次

320 ÷ 64 ＝ 5 批
1 個 epoch 更新 5 次

10 個 epoch



batch 32：10 × 10 ＝ 100 次
batch 64：10 × 5 ＝ 50 次

Colab A5 實際：訓練集 534 人、batch 32
534 ÷ 32 ＝ 16.7，最後一批只有 22 人也算一次 → 每個 epoch 更新 17 次；60 個 epoch 共 60 × 17 ＝ 1,020 次

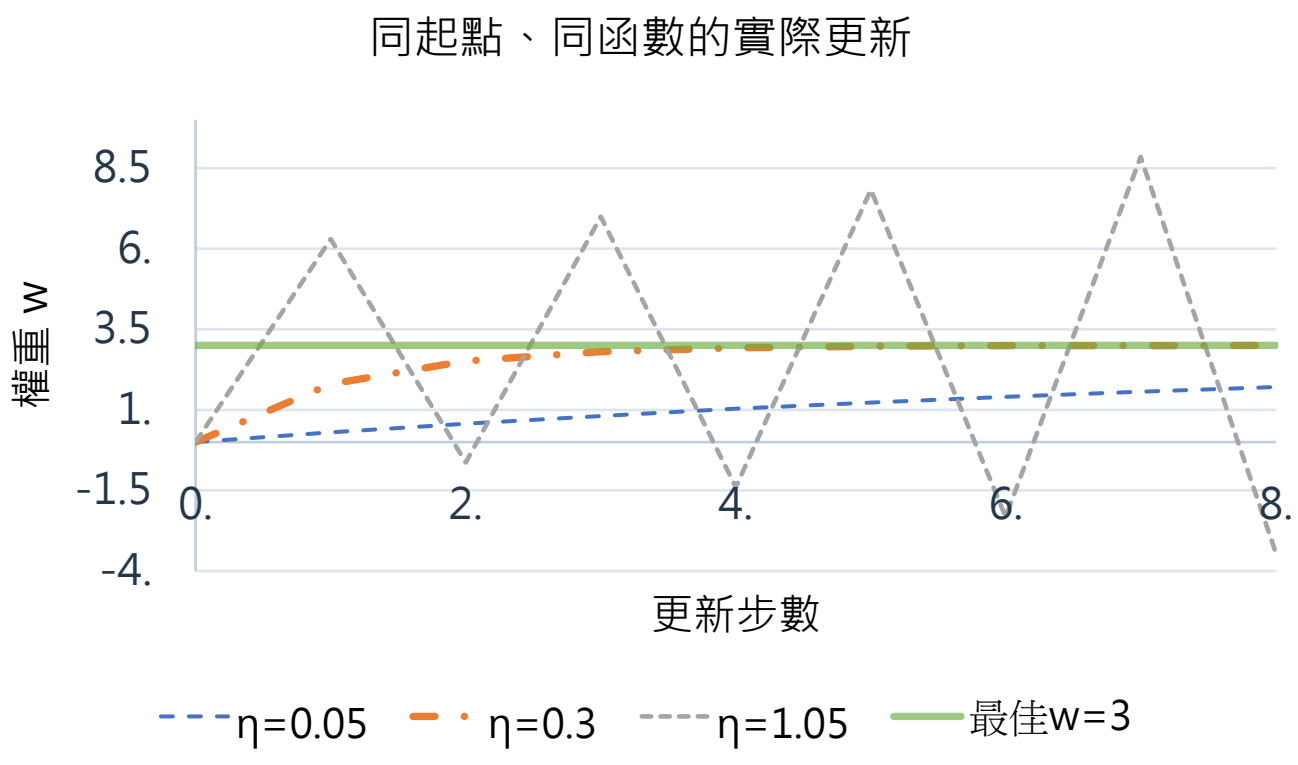
N 樣本數　B batch 大小　E epoch 數　S 每個 epoch 的更新次數　T 總更新次數

學習率的穩定範圍

$L = (w - 3)^2$　　$w_{t+1} = w_t - \eta \cdot \partial L/\partial w = w_t - 2\,\eta(w-3)$

公式參考．看懂即可

- 小學習率走得慢，有限步數內可能尚未收斂。
- 適中步長可快速靠近低點。
- 太大時可能來回擺盪，甚至離低點愈來愈遠。



圖為同一個二次函數的實際迭代，不是模型排行榜。

第二段　從數字預測到機率預測(把握度)

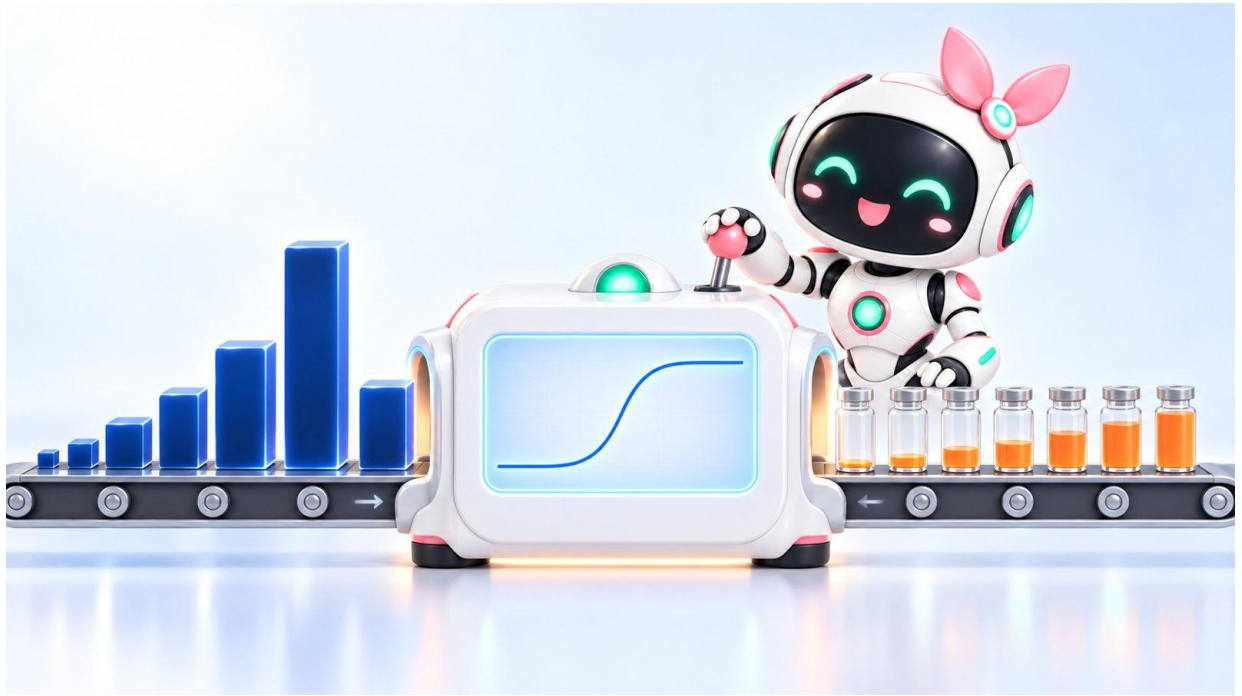
章節導覽

上一段

上週模型輸出一個數字，例如曲線的高度。

這一段要學習

如果答案只有「是 / 否」，模型該輸出什麼？ (非黑即白)

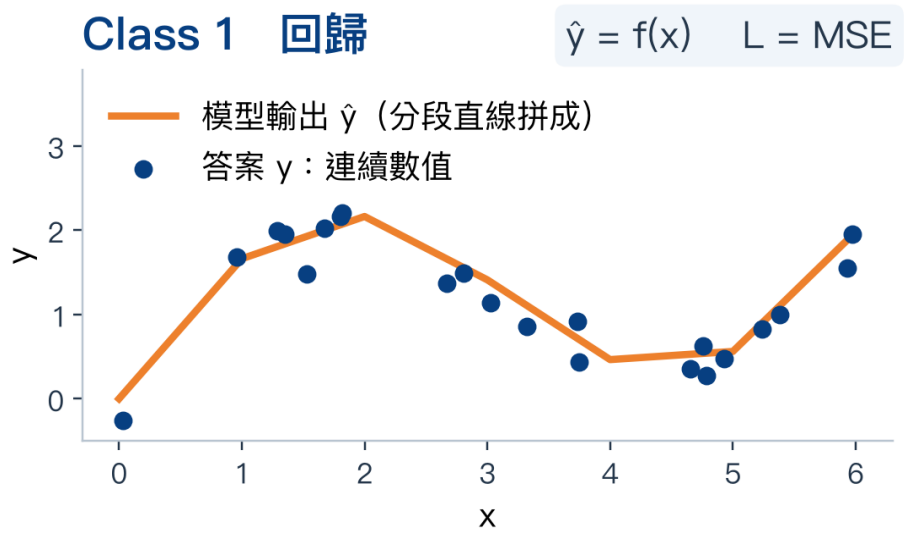


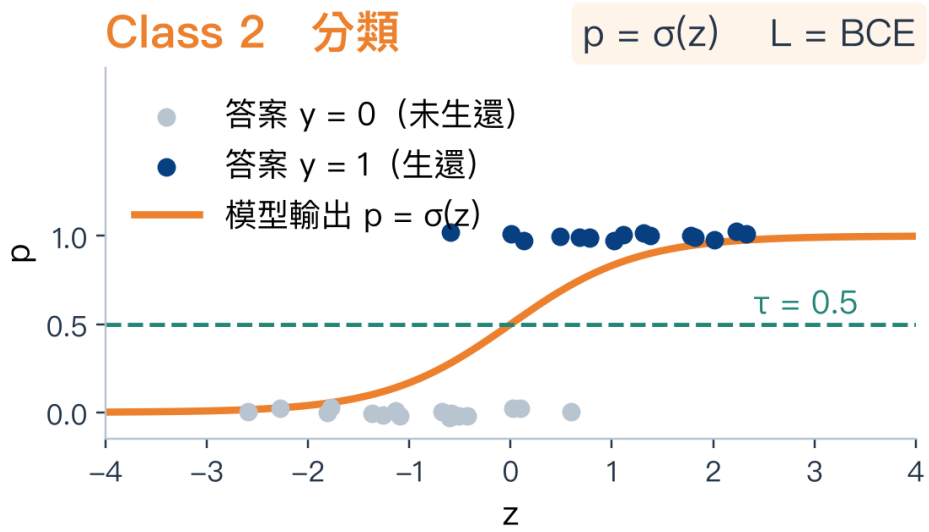
1. 加總證據，得到一個分數（可正可負）
2. 把分數換成機率/把握程度（0%～100%）
3. 決定是否合格（超過 50% 判「是」）

模型不直接回答「是 / 否」，而是先說「有幾成把握」。

Class 1 與 Class 2：從回歸到分類

對照





	Class 1　回歸 regression	Class 2　分類 classification
答案 y	連續數值（曲線高度）	類別 0 / 1（是否生還）
模型輸出	ŷ：任意實數	p = σ(z)：0～1 的機率，再用門檻判類別
損失	MSE：差多少	交叉熵：有把握地答錯，懲罰最大
評估	誤差大小	正確率、Precision、Recall

答案從「數值」變成「類別」：輸出改成機率、損失改成交叉熵；學習方法與class1一樣。

全景：當一回補習班老師來估算台大電機錄取機率

圖解全景

$$z = w_1x_1 + w_2x_2 + w_3x_3 + b \qquad p = \sigma(z)$$

① 假想申請者

數學 $x_1 = 95$ $w_1 = 0.08$

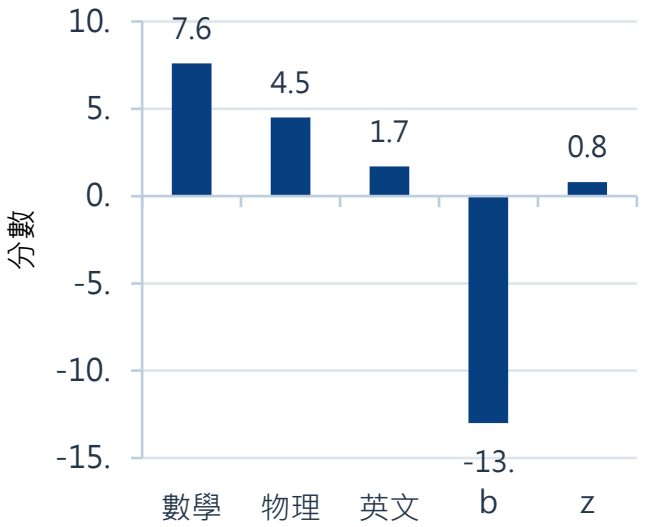
物理 $x_2 = 90$ $w_2 = 0.05$

英文 $x_3 = 85$ $w_3 = 0.02$

今年 b = –13 (比去年簡單)

（學生平均程度微調數字）

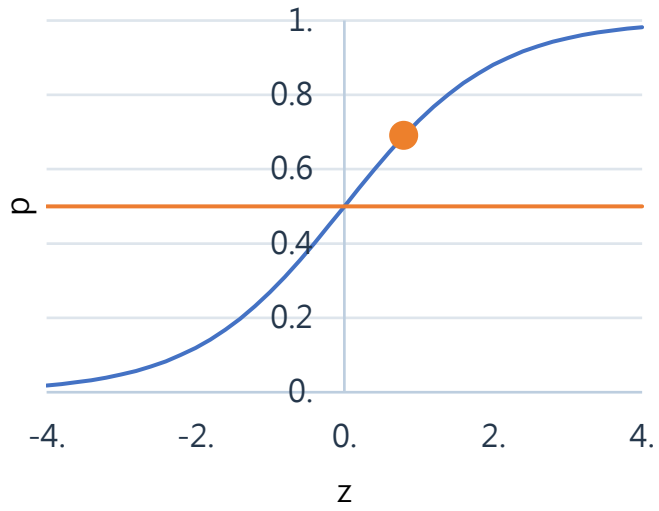
② 加權相加得到 z



$7.6 + 4.5 + 1.7 - 13$

$= 0.8$

③ Sigmoid 與門檻



$p = \sigma(0.8) \approx 69\%$

$\geq 50\% \rightarrow$ 預測錄取把握程度

權重 w 決定各科有多重要 (你目前並不知道)，b 反映當年整體的競爭；加總後再經 sigmoid 變成錄取機率。

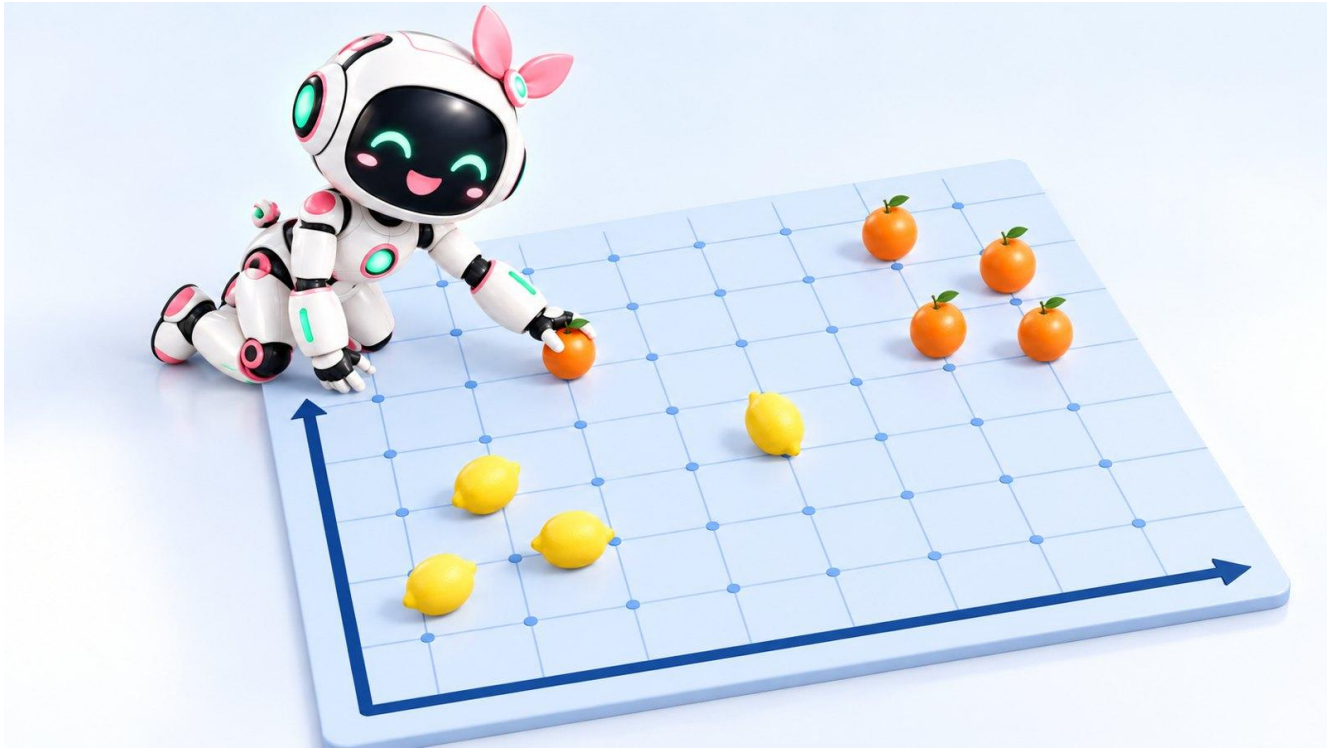
直覺圖解：分類就是替每個點找區域

生活中的例子

分揀水果時量重量、看顏色，就能把每顆水果放到地板方格上的一個位置。橘子與檸檬自然聚成兩群，中間少數幾顆最難判斷。

對應到模型

一顆水果	**一筆樣本**
重量、顏色	**特徵 x_1、x_2**
地板上的位置	**特徵空間中的點**
橘子或檸檬	**標籤 y**



分類模型要學的是「哪一塊區域屬於哪一類」；兩群交界處最容易判錯。

回歸與分類：答案不同

- 回歸輸出連續數值，例如明日乖離率。
- 分類輸出類別，可先估計分數或機率再判定。
- 類別用0、1編碼，不代表問題就變成一般數值回歸。

任務	標籤意義	輸出	
溫度預測	連續數值	例如26.4°C	回歸
乖離率預測	連續數值	例如+2.5%	
生還預測	二元事件	機率 / 0或1	分類

任務由標籤的意義決定，不能只看它是不是數字。

鐵達尼號生存率分析

1912 年 4 月 15 日，在她的首次航行中，被廣泛認為是“不可沉沒”的“泰坦尼克”號，因與冰山相撞而沉沒。不幸的是，船上沒有足夠的救生艇，導致 2224 名乘客和船員中，1502 人喪生

雖然倖存下來也與一些運氣有關，但似乎某些人群的生存幾率更高。

在本次挑戰中 (取891筆數據)，我們要求您構建一個預測模型，該模型能夠回答問題： “哪些類型的乘客更有可能存活？”

資料來源：
https://www.kaggle.com/c/titanic/data

kaggle

#	Team	Score	Entries	Last
2753	Gabriel Barros #3	0.78229	3	7h
2754	Michael Toth #2	0.78229	3	6h
2755	josephWithData	0.78229	4	3h
2756	llxiongfei	0.77990	10	2mo
2757	ryosei-akatsuka	0.77990	9	2mo
2758	Dean Whitbread	0.77990	4	2mo
2759	Teendangor	0.77990	5	2mo

2,756/10,000

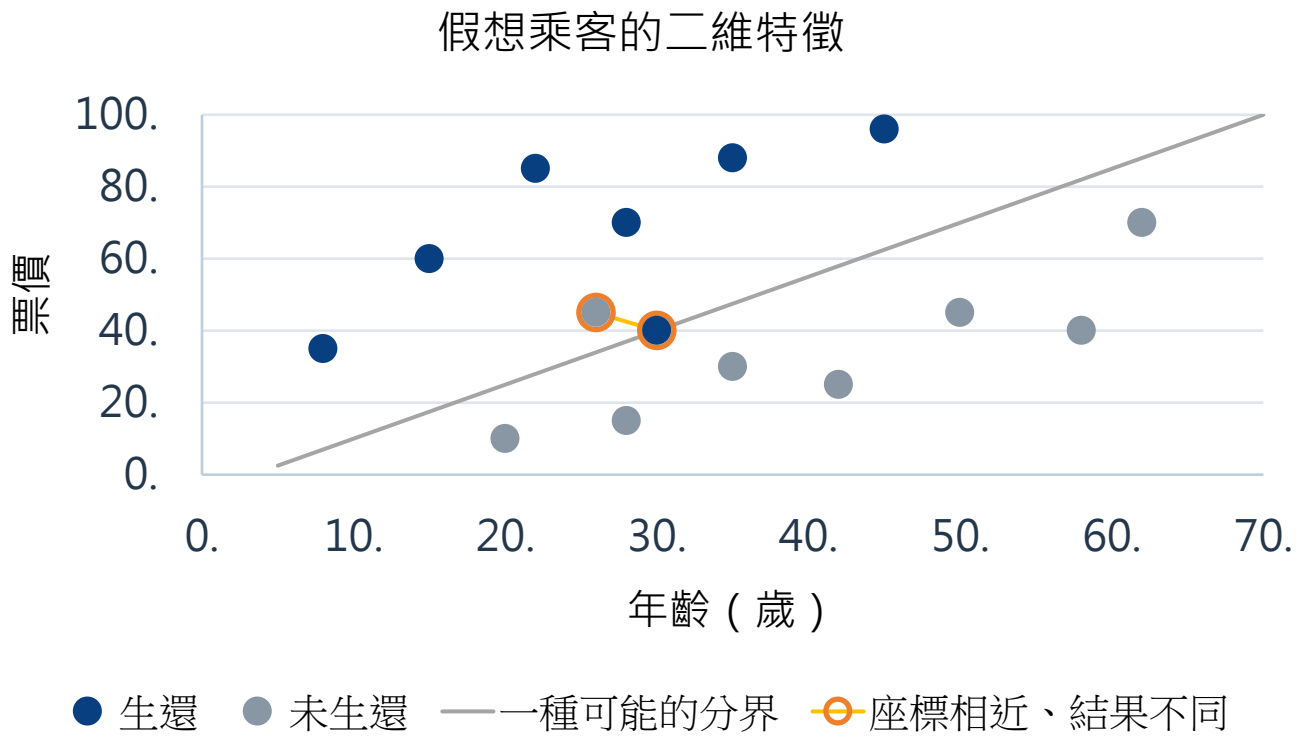
class2

試試看你能第幾名

任務由標籤的意義決定，不能只看它是不是數字。

特徵空間：把乘客想成一個數據集合體

- 選兩個特徵，就能將每位乘客畫在數據平面。
- 相似座標只代表特徵相近，標籤仍可能不同。
- 分類器在空間中找出分界規則。



藍 = 生還、灰 = 未生還；橘線是一種可能的分界，圈起的兩位乘客座標相近、結果卻不同。

直覺圖解：線性分數像一座天平

生活類比

生活中的例子

每項證據放上天平，份量由權重決定；支持生還的放右邊、不利的放左邊。天平上還有一塊事先放好的固定砝碼。

對應到模型

砝碼大小	**權重 w**
放上去的證據	**特徵 x**
事先放好的砝碼	**偏差 b**
指針傾斜程度	**分數 z**

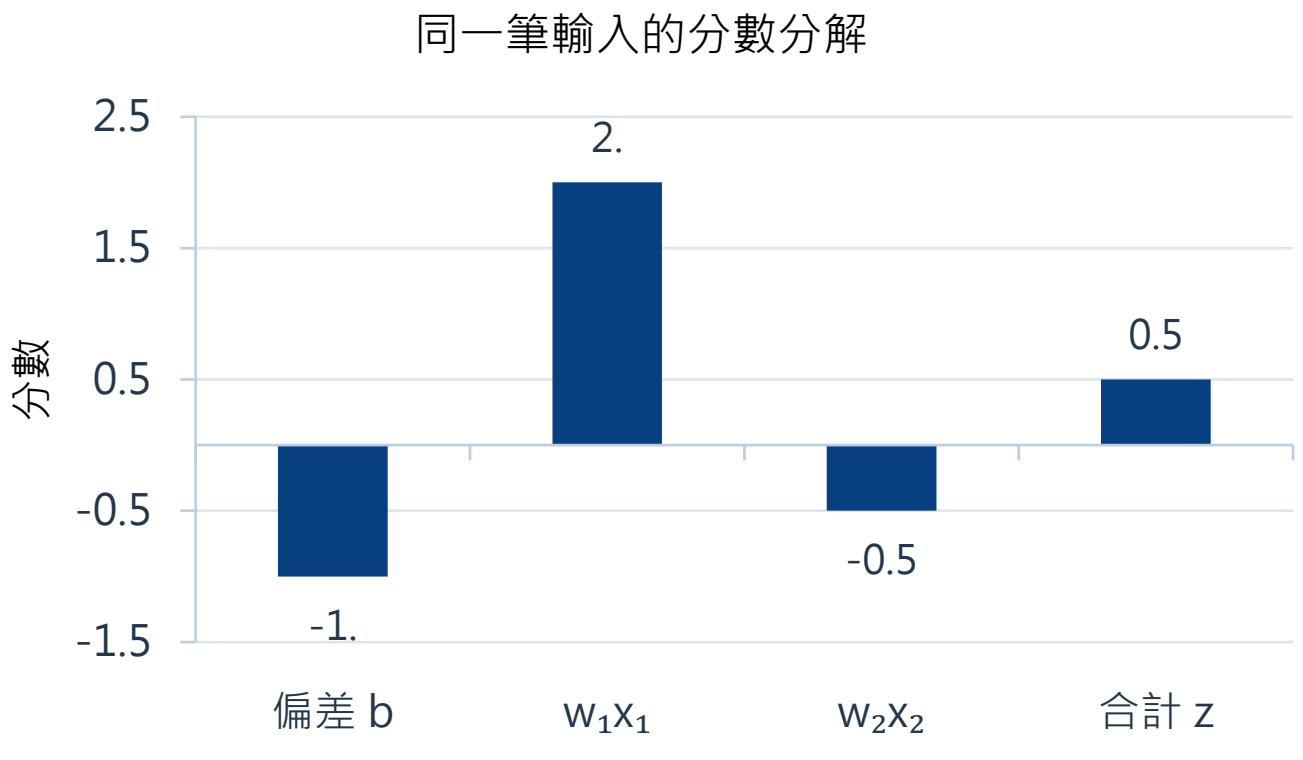


z 只說明往哪邊傾、傾多少；下一步再用 sigmoid 把傾斜程度換成機率。

線性分數：權重如何組合特徵

$$z = b + w_1x_1 + w_2x_2$$

- 每個特徵乘以自己的權重，再加上偏差。
- 權重正負決定其他條件固定時的影響方向。
- 權重大小受單位影響，不能直接當重要性排名。

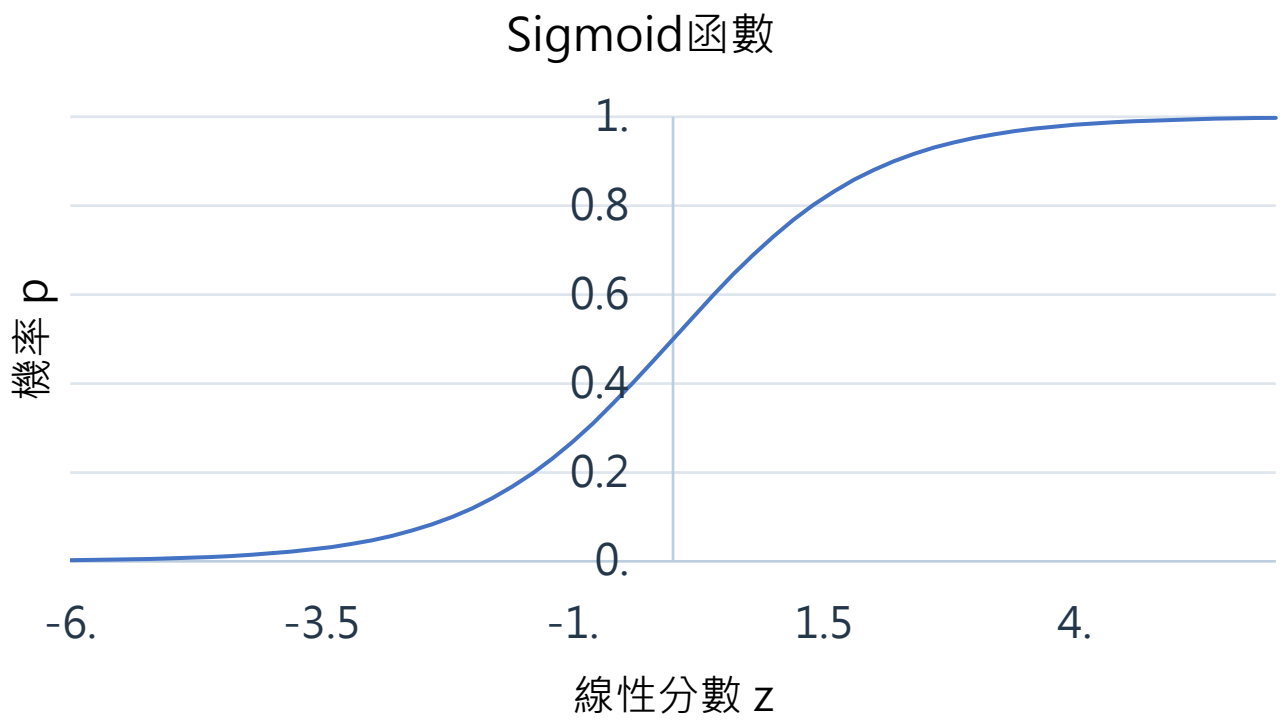


示例：b=−1、x=(1,0.5)、w=(2,−1)，得到 z=0.5。

Sigmoid：分數轉為事件機率

$$p = \sigma(z) = 1 / (1 + e^{-z})$$

- z=0 對應0.5，正分數高於0.5。
- 兩端逐漸飽和，有限z的結果介於0與1之間。
- 曲線限制範圍，但機率是否可信仍需驗證。



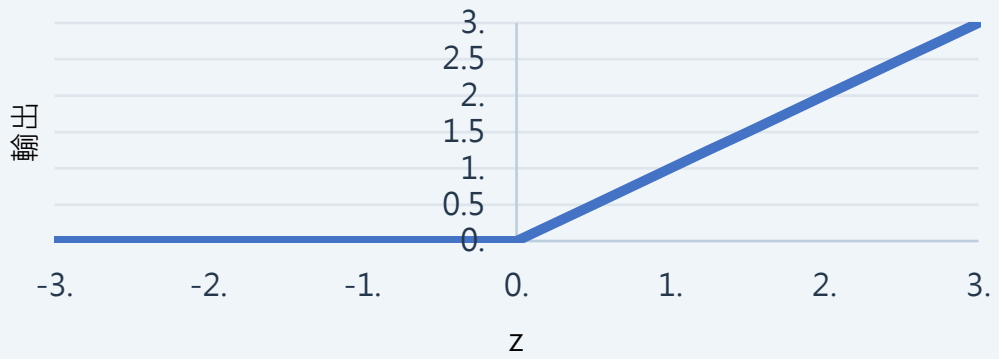
z=−2、0、2 分別約為0.119、0.500、0.881。

複習：ReLU 與 Sigmoid：同樣是激活函數，任務不同

左右對照

Class 1　ReLU　隱藏層

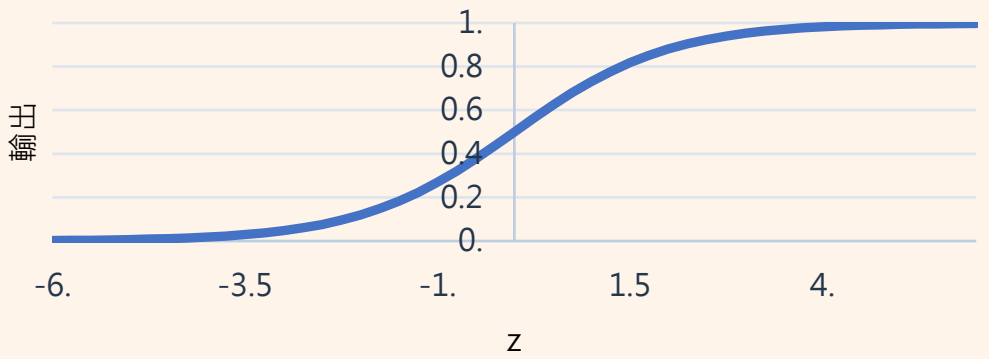
ReLU(z) = max(0, z)



- 負的變 0、正的照傳
- 多個 ReLU 相加，拼出任意折線（曲線拼圖）
- 輸出 0 ~ ∞，不能當機率

Class 2　Sigmoid　輸出層

$\sigma(z) = 1 / (1 + e^{-z})$



- 任何分數都壓進 0 ~ 1
- z = 0 時剛好 0.5，越大越接近 1
- 輸出可解讀為機率，再用門檻判類別

同一個網路常兩者並用：隱藏層用 ReLU 學出形狀，輸出層用 sigmoid 給出機率。

完整算一次分類預測

$x=(1, 0.5), w=(2,-1), b=-1$

- 先以特徵與權重算出z，再套用sigmoid。
- 得到p後，依門檻τ決定最後類別。
- 同一個p可以配合不同任務門檻。

$$z = b + w_1x_1 + w_2x_2$$

步驟	計算	結果
線性分數	$-1 + 2\times1 - 1\times.5$	$z = 0.5$
轉成機率	$1 / (1+e^{-0.5})$	$p \approx 0.622$
門檻τ=0.5	$0.622 \geq 0.5$	預測1
門檻τ=0.7	$0.622 < 0.7$	預測0

z=0.5，p≈0.622；τ=0.5 判1，τ=0.7 判0。

直覺圖解：決策邊界是一道圍籬

生活中的例子

牧場裡綿羊在左、山羊在右，一道直直的圍籬就能分開兩群。靠近圍籬的動物最難判斷，離得遠的一眼就知道。

對應到模型

圍籬	**決策邊界 z = 0**
貼著圍籬	**機率接近 0.5**
跑錯邊的動物	**分類錯誤**

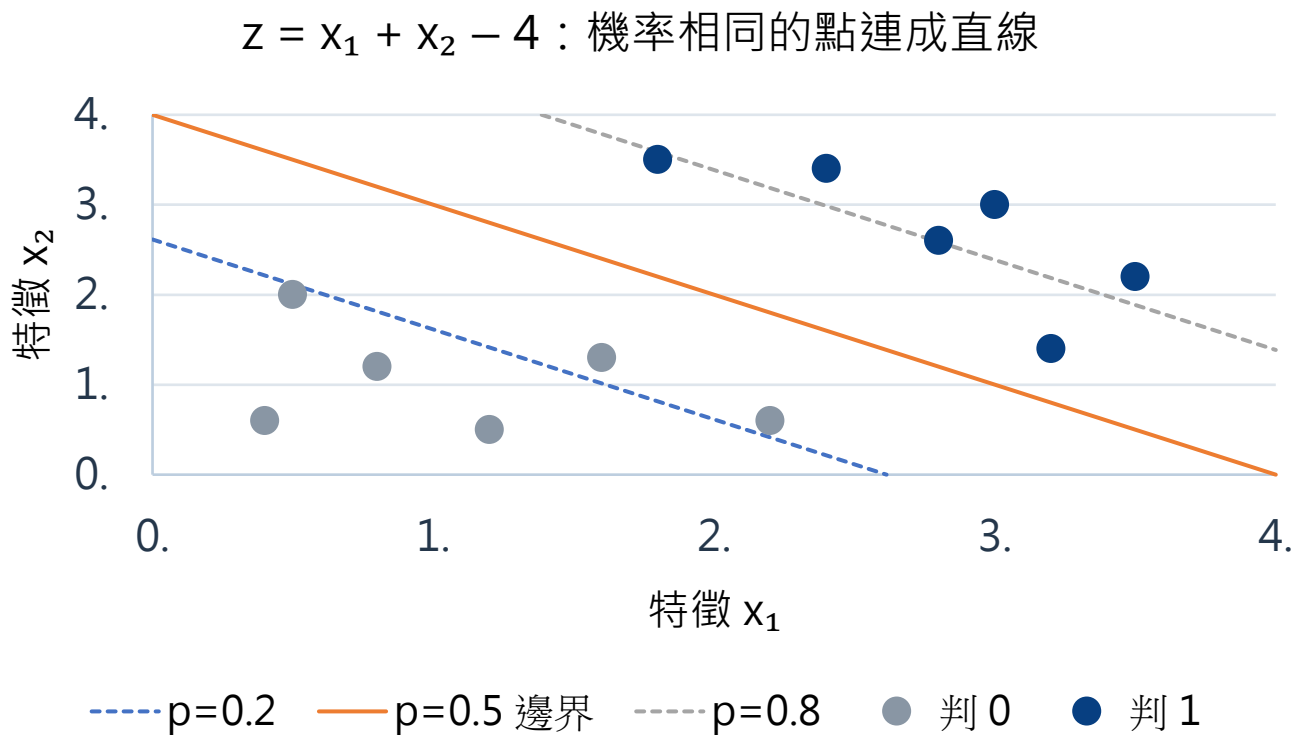


Logistic 的圍籬一定是直的；想要彎曲的圍籬，需要新特徵或隱藏層。

決策邊界：哪裡恰好判成一半

$$p = 0.5 \Leftrightarrow z = 0 \Leftrightarrow w_1x_1 + w_2x_2 + b = 0$$

- 平面上每一點都有分數 z，經 sigmoid 變成機率 p。
- 機率相同的點連成直線；p = 0.5 那條就是決策邊界。
- 離邊界越遠，機率越接近 0 或 1，模型越有把握。

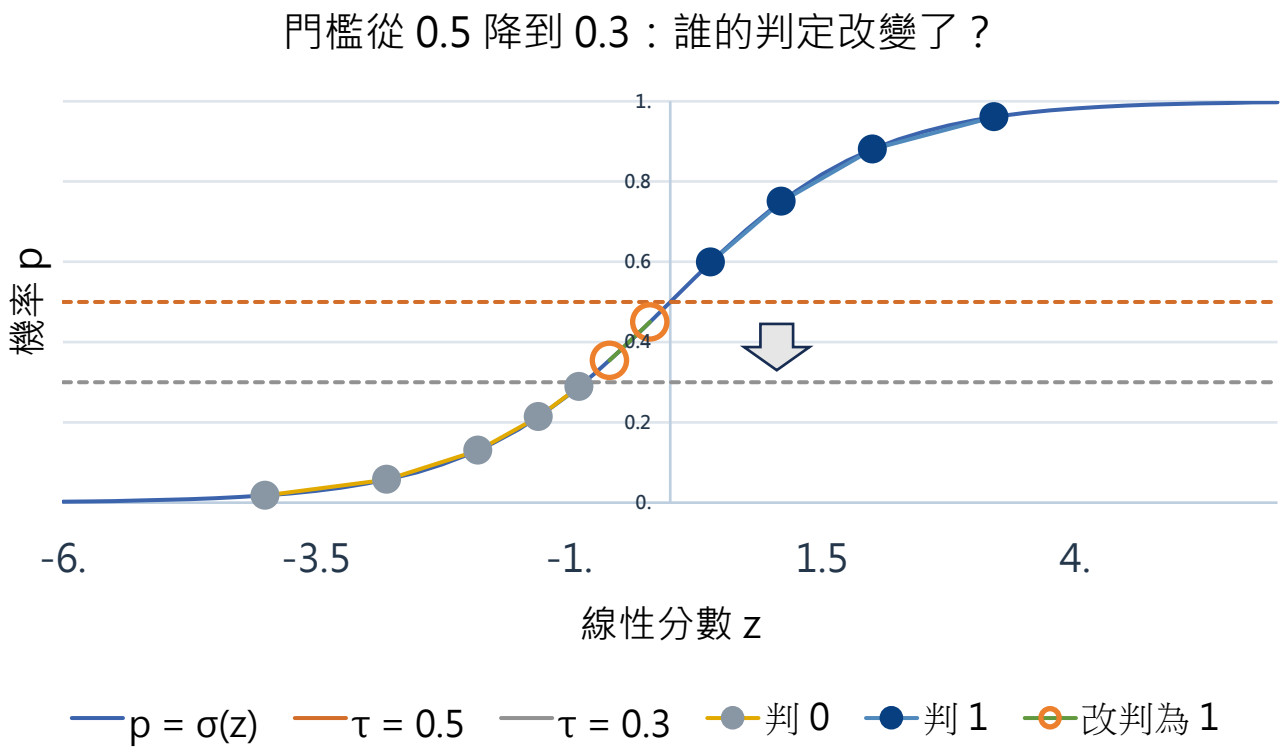


像地圖的等高線：邊界是 p = 0.5 那一條，兩側的 p = 0.2、0.8 與它平行。

門檻改變判定，模型機率保持原值

$\hat{y} = 1$ if $p \geq \tau$, otherwise $\hat{y} = 0$

- 門檻只是一條判定線：$p \geq \tau$ 就判為 1。
- τ 從 0.5 降到 0.3：兩線之間的人改判為 1；曲線沒動，不用重新訓練。
- 降門檻能找回漏掉的生還者，但誤報也可能變多（第六段再談）。

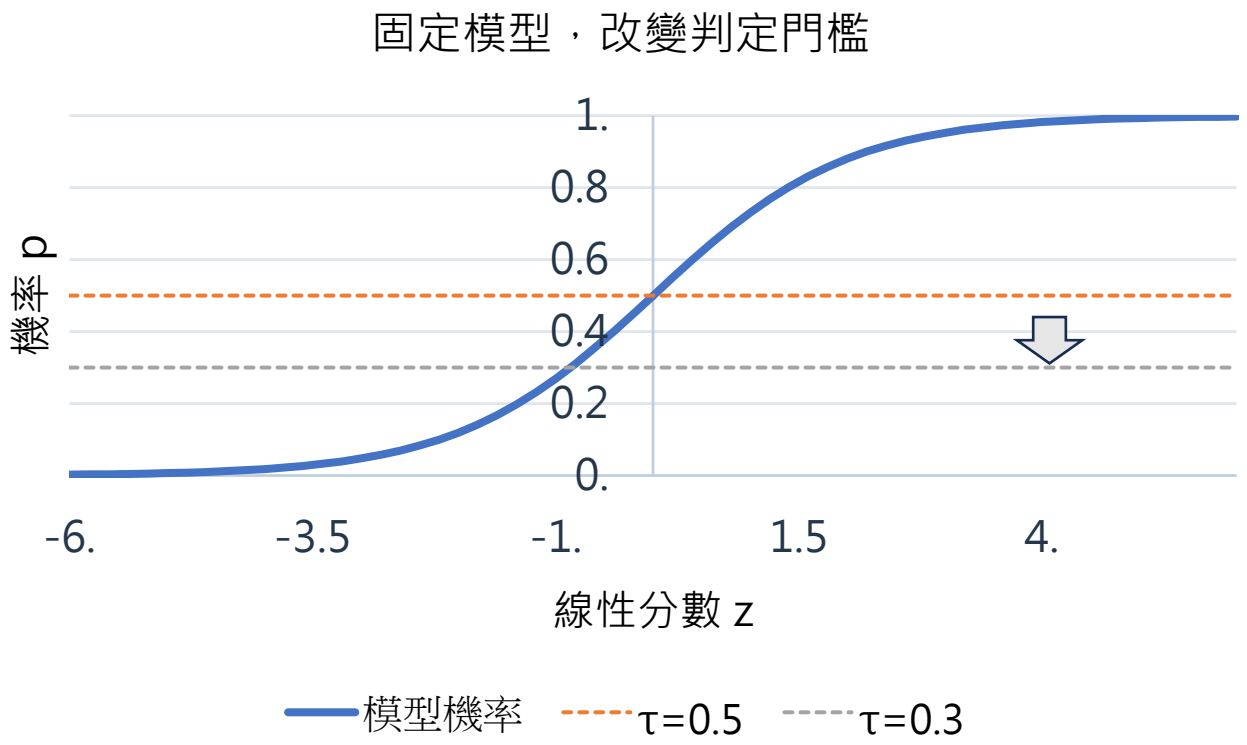


圈起的兩位乘客改判為 1，但他們的機率完全沒變。

想一想②　改門檻會改掉什麼

思考題

模型預測p=0.40，把門檻從0.5改成0.3。



A　類別 0 改 1

B　機率自動變成0.60

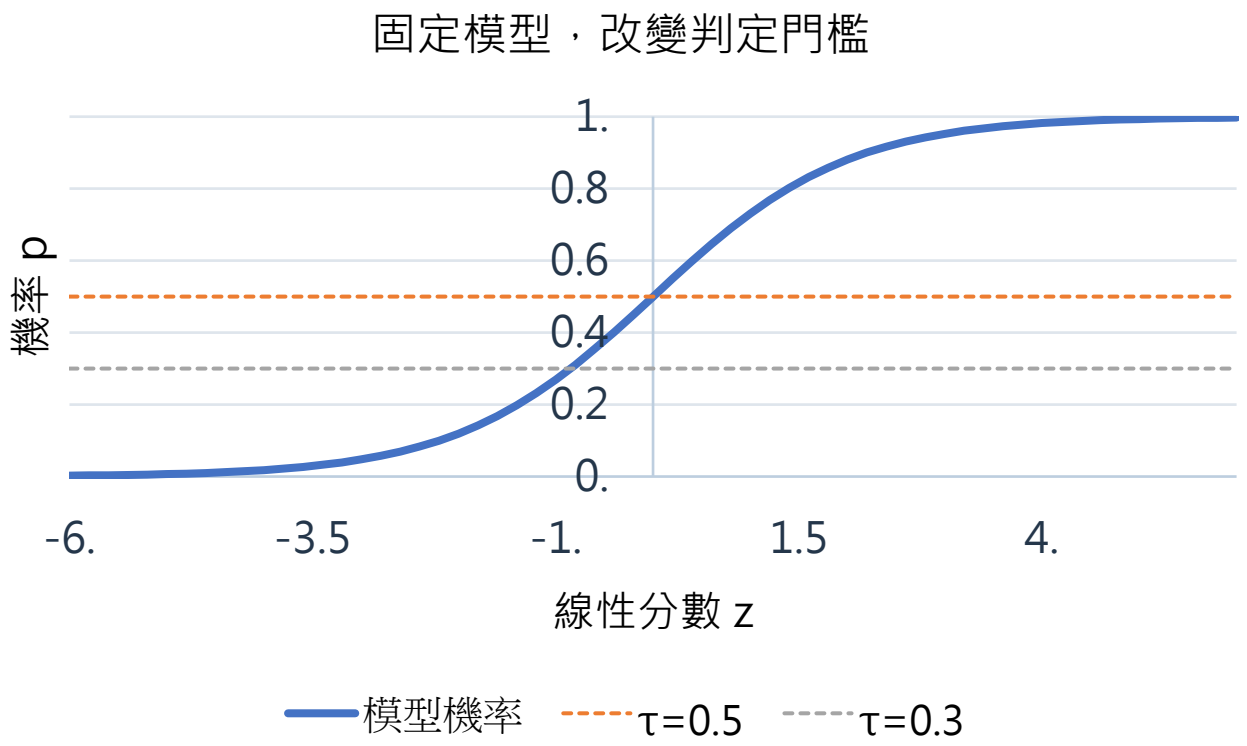
C　必須重新反向傳播

請先選擇，再用圖中的證據解釋。

想一想②　改門檻會改掉什麼：解答

思考題

模型預測p=0.40，把門檻從0.5改成0.3。



A　類別 0 改 1

B　機率自動變成0.60

C　必須重新反向傳播

答案：A　類別 0 改 1

0.40低於0.5但高於0.3。變的是判定規則，不是模型算出的機率。

第三段　損失與神經網路

上一段

模型已能輸出機率，也能依門檻做決定。

這一段要學習

把握度如何定義損失？神經網路又是怎麼學的？



1. 交叉熵：有把握地答錯，代價最大
2. 神經網路：隱藏層自己學新特徵
3. 訓練一回合：前向、反傳、最佳化器

押上全部籌碼卻押錯，損失最慘——這就是交叉熵的直覺。

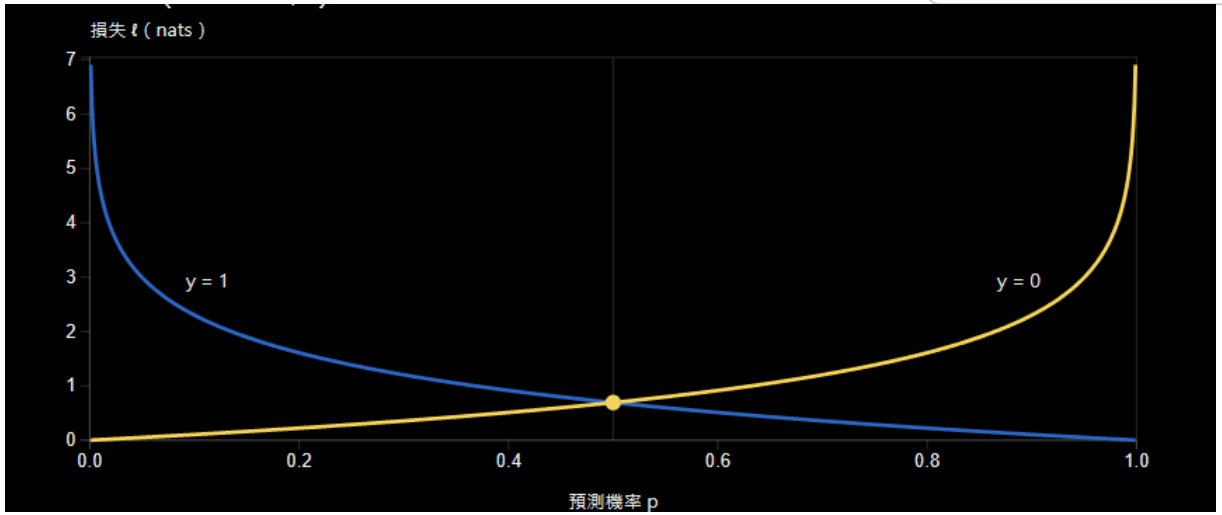
二元交叉熵：一個公式涵蓋兩類

核心理論

公式參考，看懂即可

$$L = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \ln p_i + (1-y_i)\ln(1-p_i)\right]$$

- y=1時，第二項消失，留下–ln(p)。
- y=0時，第一項消失，留下–ln(1–p)。
- 平均所有樣本，得到分類訓練的損失。



真實標籤	保留下的項	較佳方向
y = 1	–ln(p)	p靠近1
y = 0	–ln(1–p)	p靠近0

Binary Cross-Entropy（BCE），二元分類時亦稱Log loss。

附註：–ln(1) = 0,

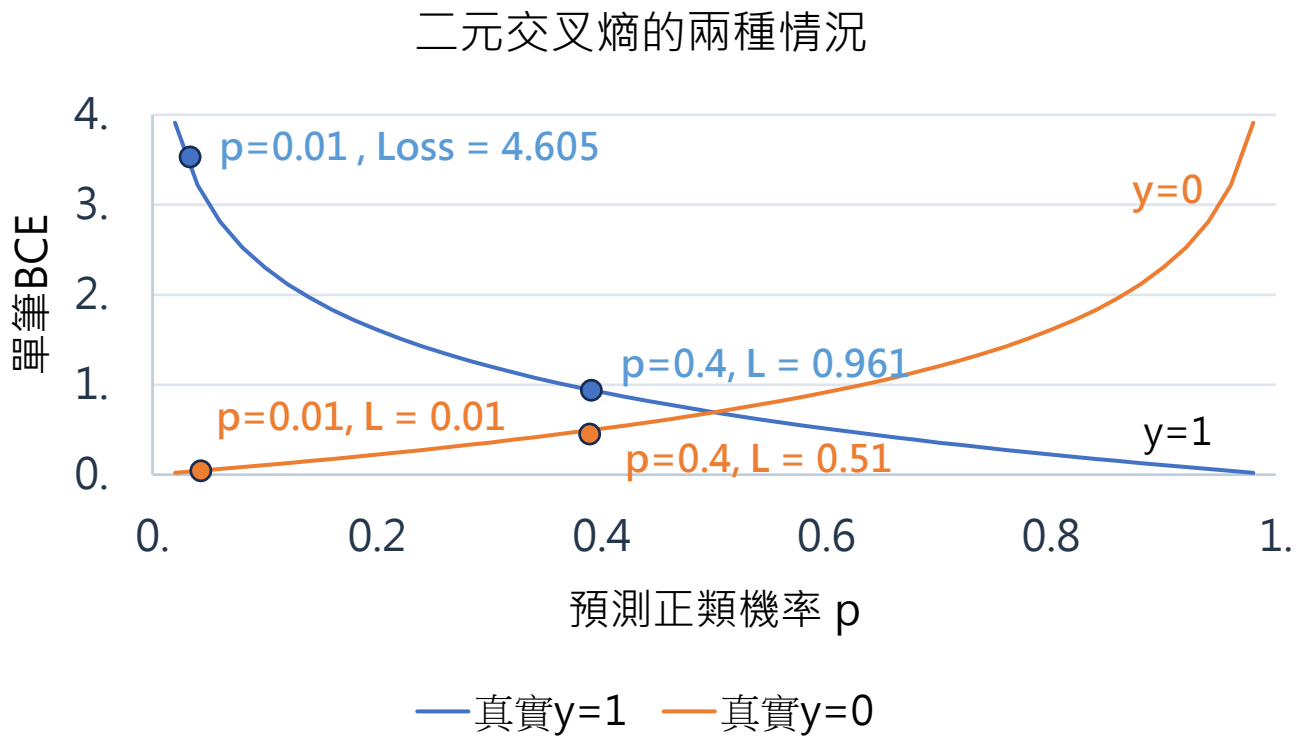
交叉熵 (BCE)曲線：自信地答錯代價很大

- 真實y=1時，p愈接近1，loss愈小

  y=1：p=0.4，$\ell \approx 0.916$；
  p=0.01，$\ell \approx 4.605$

- 真實y=0時，趨勢相反。

  y=0：p=0.4，$\ell \approx 0.51$；
  p=0.01，$\ell \approx 0.01$



Accuracy只數對錯，BCE還會區分機率估計的品質。

隱藏層 難題：神奇樓梯開關：不一樣燈才亮

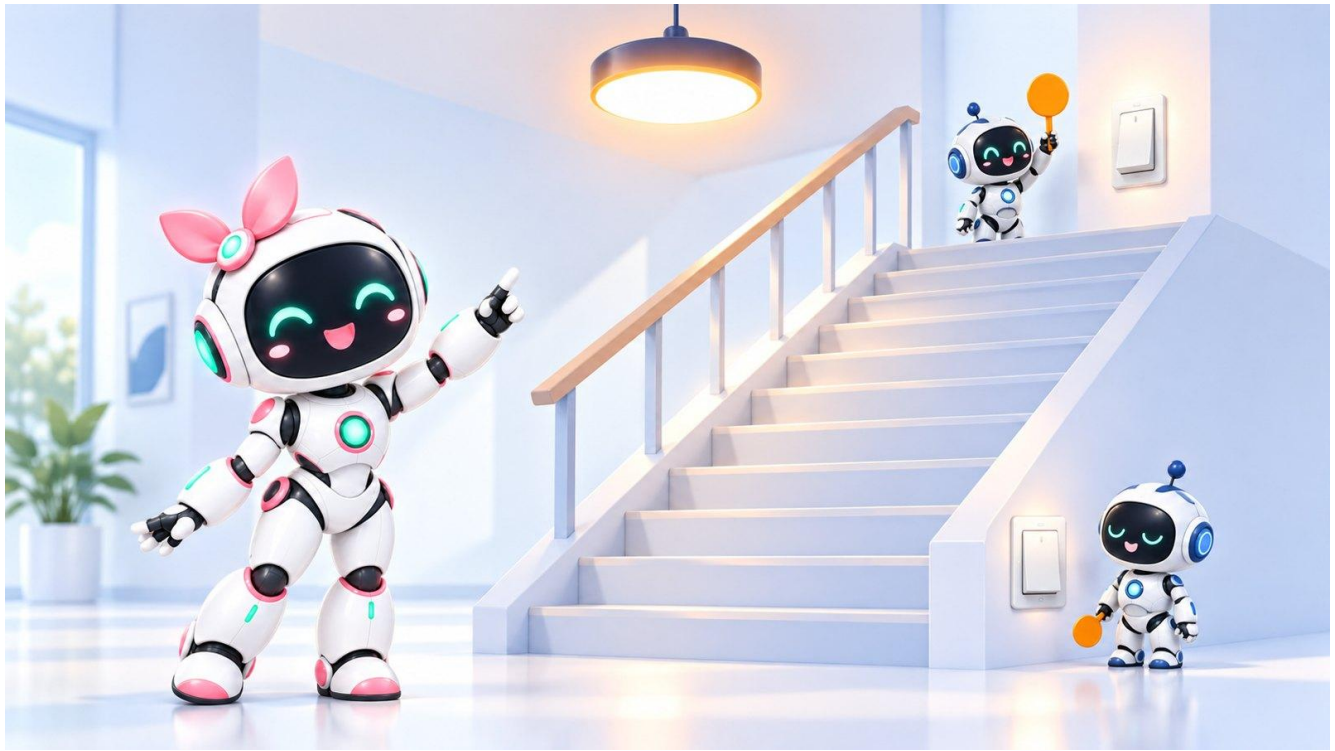
生活類比

生活中的例子

樓梯上下各一個開關，兩個「不一樣」時燈才亮。只用一個分數，兩個都開時分數最高，燈卻該是暗的。改請兩位小評審：A 管「上開下關」、B 管「上關下開」，有人舉牌燈就亮。

對應到模型

一個分數	**都開最高，卻該是暗**
小評審 A、B	**神經元 h_1、h_2**
非自己的情況	**ReLU：不舉牌（0）**
有人舉牌就亮	**輸出層 sigmoid**



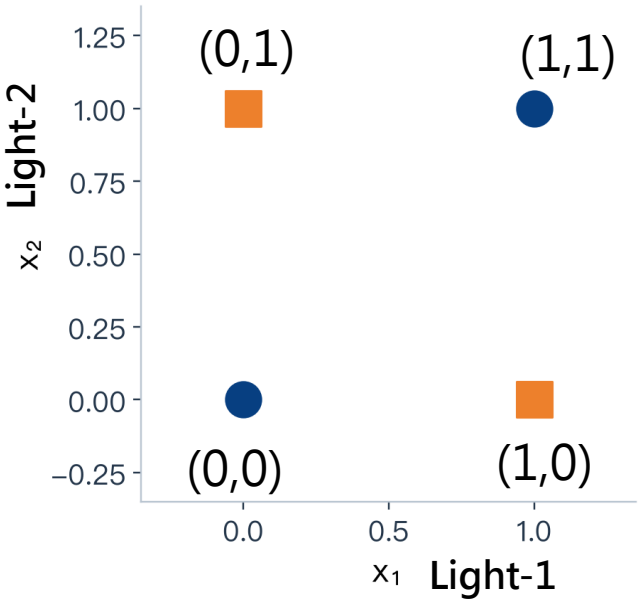
一個分數會被「兩個都開」騙；請小評審各管一種情況，再由總評審判斷，矛盾就解開了。

特徵交互作用：單一直線的限制

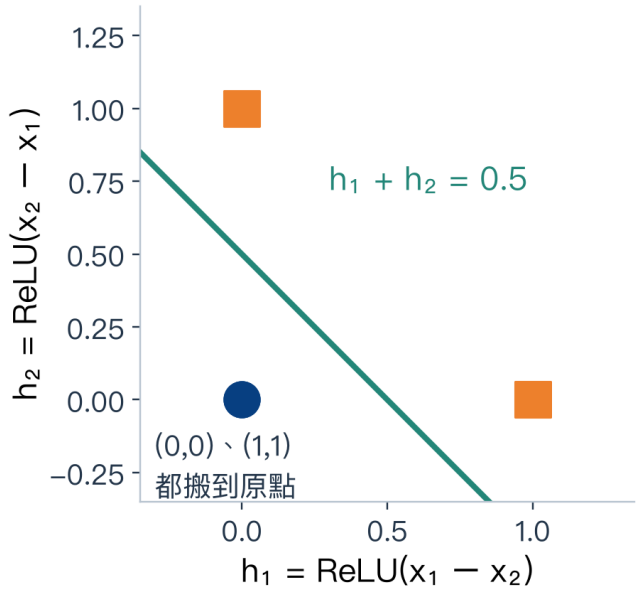
$y = 1$ if $x_1 \neq x_2$, otherwise $y = 0$

- XOR中，相同輸入值組合為0，不同組合為1。
- 四個角點無法用單一直線完全分開。
- 增加合適非線性特徵或隱藏層，才能改變邊界能力。

原始特徵：一條直線分不開



隱藏層座標：一條直線就分開

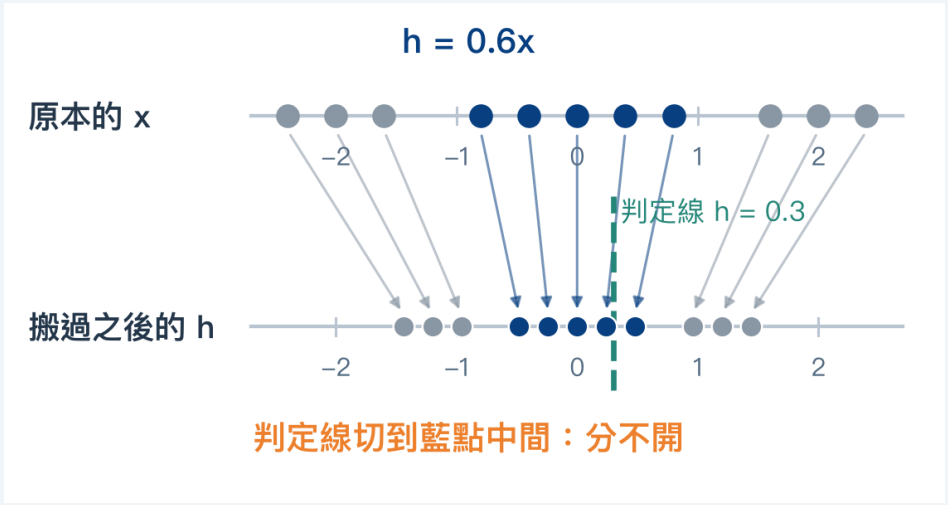


兩個 ReLU 神經元把四個角搬到新座標，一條直線就分得開。

為什麼隱藏層要加 ReLU：把資料搬開

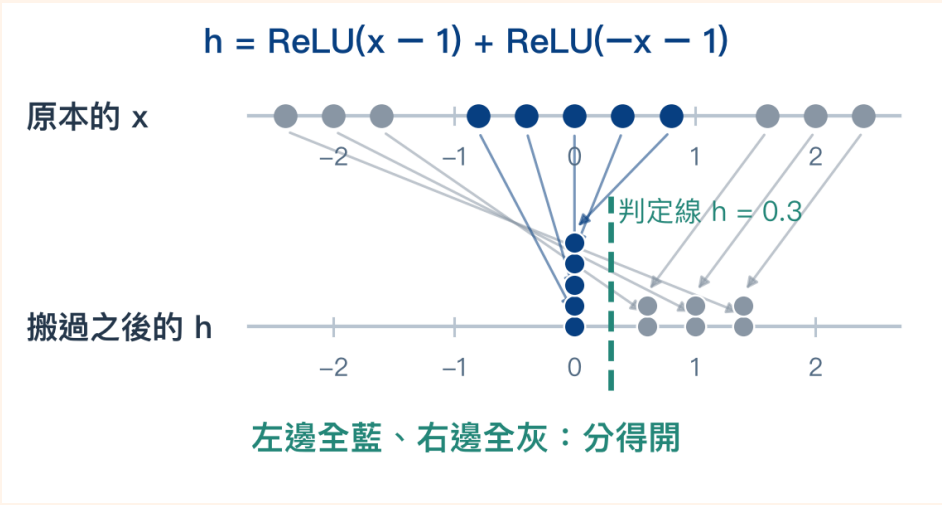
左右對照

沒有 ReLU　只能切一刀



- 上排：同一排原始資料，中間藍、兩端灰
- 箭頭平行：只縮放，順序還是灰藍灰
- 判定線 h = 0.3 切到藍點中間，分不開

加上 ReLU　把資料搬開



- 中間的藍點全部搬到 0（沒感覺不舉牌）
- 兩端的灰點都搬到右邊（0.6 ~ 1.4）
- 同一條判定線 h = 0.3 就分開

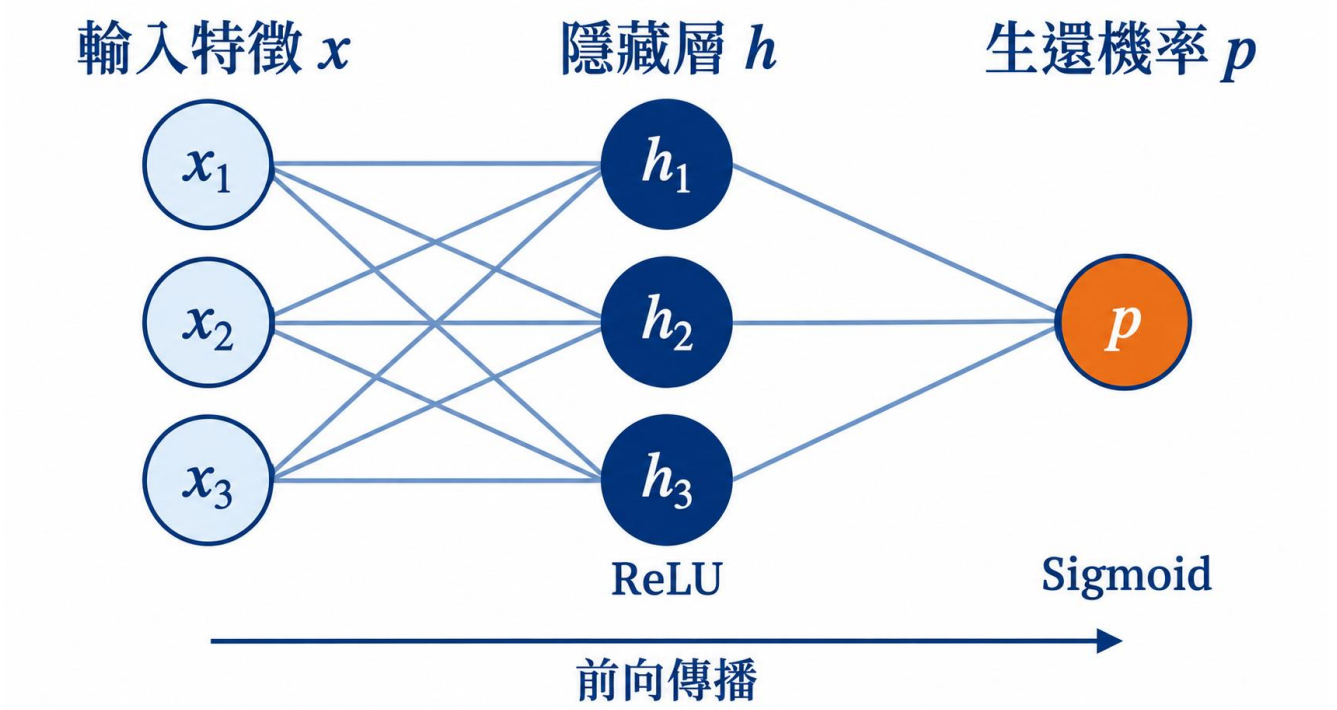
ReLU 把中間的藍點搬到 0、兩端的灰點搬到右邊：原本一刀切不開，搬完一條線就分開。

神經網路：先學新特徵，再分類

$h=\mathrm{ReLU}(W_1x+b_1)$　　$p=\sigma(w_2^{T}h+b_2)$

公式參考．看懂即可

- 隱藏層將原始輸入轉換成新的表示h
- 非線性激活讓多層模型能表達更複雜的關係。
- 最後一個sigmoid仍可輸出生還機率。



圖例：3個輸入、1個隱藏層（3個神經元）、1個輸出，與上方公式一致。

訓練一回合：神經網路怎麼學

核心理論

1

取一小批

從訓練集取 batch 位乘客

2

前向傳播

逐層計算：加權、ReLU、sigmoid，得到每人的 p

3

反向傳播

算交叉熵，用鏈式法則求出每個權重的梯度

4

更新權重

最佳化器沿負梯度，依序調整全部權重

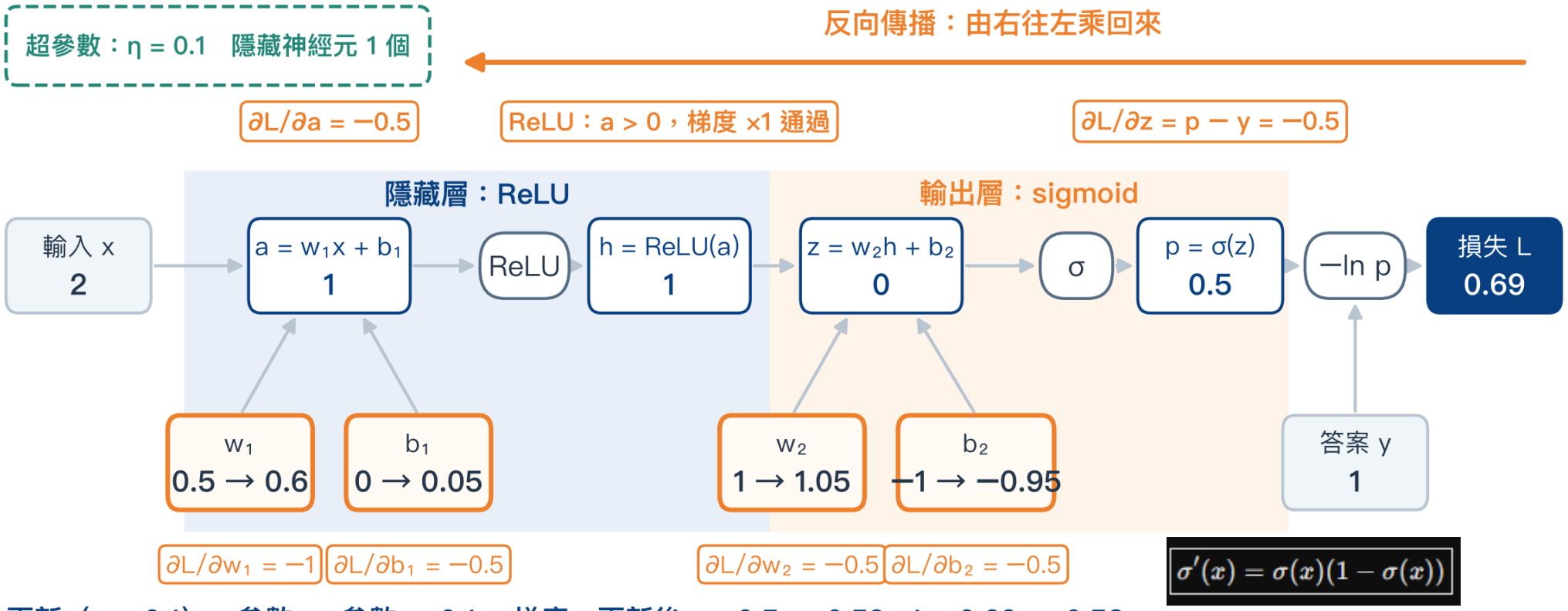
和 Class 1 的連結

步驟 3 就是上週的鏈式法則，只是損失換成交叉熵。

全部資料輪過一次 = 1 個 epoch；Colab A5 控制台預設 20 個 epoch，會畫出訓練 / 驗證曲線。

神經網路的計算圖：和 Class 1 同一套流程

圖解



更新（η = 0.1）：參數 ← 參數 − 0.1 × 梯度；更新後 p：0.5 → 0.59，L：0.69 → 0.53

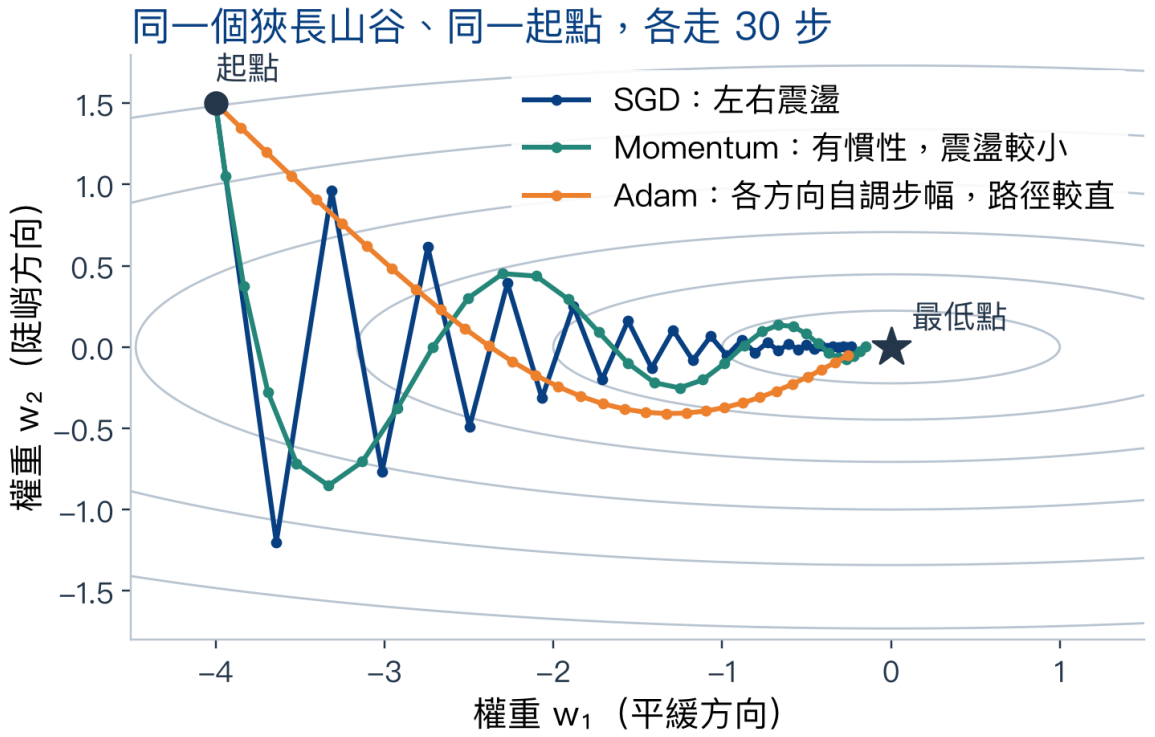
前向 → 反向 → 更新：和 Class 1 同一套流程，只多了 ReLU 隱藏層與 sigmoid 輸出。

最佳化器：SGD、Momentum、Adam

公式參考．看懂即可

$$v \leftarrow \beta v + g \qquad w \leftarrow w - \eta \cdot v$$

- SGD：每步只看當下的坡度；山谷狹長時會左右震盪。
- Momentum：像滾下坡的球，保留前幾步的方向，震盪變小。
- Adam：每個權重依過去梯度大小自調步幅；Colab 可切換。



本例三者都接近谷底，差別在路徑；Adam 對學習率較不敏感，所以常當預設。

第四段　資料準備

章節導覽

上一段

模型怎麼學已經清楚；接下來要把真實的乘客表格交給它。

這一段要學習

真實表格有缺值、有文字，還要分出訓練與測試，怎麼整理才公平？



1. 訓練 / 驗證 / 測試：各有不同權限
2. 缺值、One-hot 與標準化
3. 只向訓練集學規則，避免資料洩漏

整理完這一段，直接進入實作①。

訓練、驗證、測試各有不同權限

- 訓練集用來學權重與前處理統計量
- 驗證集選模型、特徵、門檻和訓練設定
- 測試集保留到定案後，估計未見資料表現



練習本 = 訓練；模擬考 = 驗證；封印的期末考 = 測試，定案後才拆封。

補缺值用的平均，也只能從訓練集算

核心理論

正確：只看訓練集 ✓ 公平

- 訓練集年齡：20、30、40
- 平均 = 30（只用這三筆算）
- 測試集有缺值，就補 30，不重算

本例　平均 = 30

錯誤：全部一起算 ✗ 偷看

- 把測試集的 80 歲也算進來
- 平均變成 42.5
- 模型等於先偷看了測試集

本例　平均 = 42.5

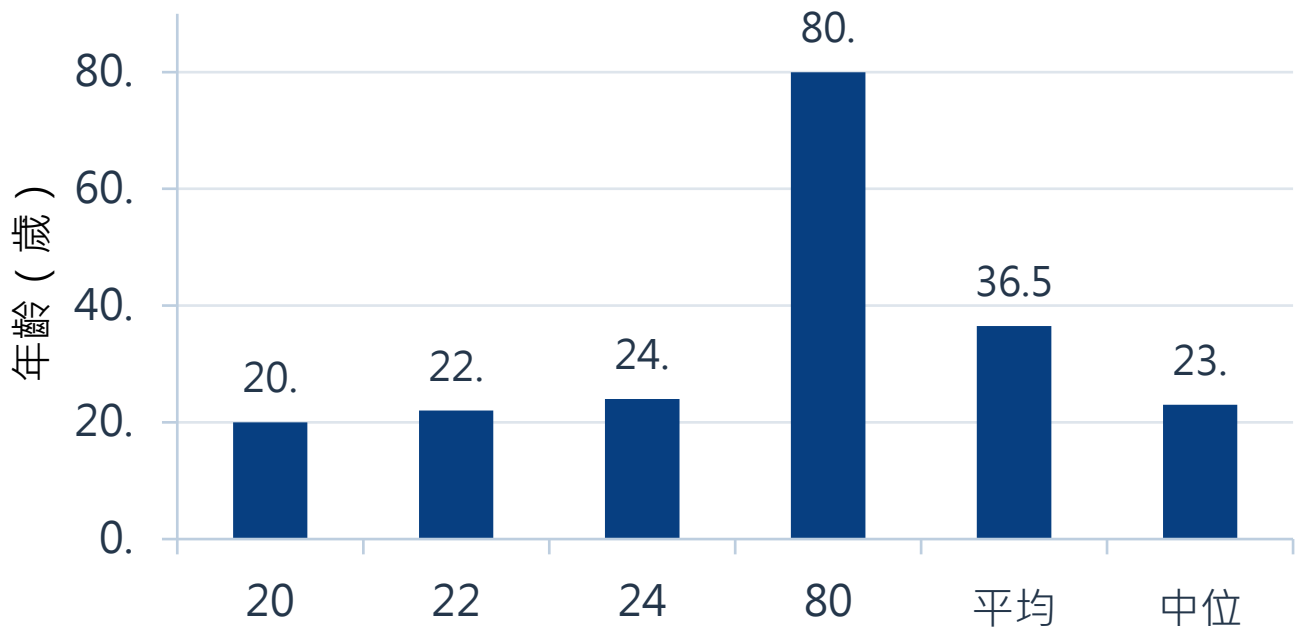
平均、標準差只從訓練集算，測試集照用；就像考前只能看練習題，不能先看期末考卷。

缺值補齊：平均數與中位數

ages = {20, 22, 24, 80}　　mean = 36.5　　median = 23

- 缺值表示未知，填0會額外假設數值真的等於0。
- 平均數容易受到極端值影響，中位數較穩健。
- 補值只是近似，會改變分布與變異程度。

極端值對補值統計量的影響



若只是缺少一個年齡，不能因此斷定乘客是嬰兒。

One-hot：類別不能隨便編成距離

- S、C、Q若寫成0、1、2，會暗示順序與等距。
- One-hot為每個類別建立0/1欄，避免這種暗示。
- 預測時需保持欄位順序，也要處理未見類別。

港口	S欄	C欄	Q欄
S	1	0	0
C	0	1	0
Q	0	0	1

類別編碼增加欄數，不等於增加新的樣本。

One-hot：文字類別怎麼變成數字

錯誤：編成 0、1、2

✗ 假裝有大小

- 登船港口 S=0、C=1、Q=2
- 模型以為 Q「比 C 大」，還是 C 的兩倍
- 乘上同一個權重：Q 的影響被綁成 C 的兩倍

本例　Q − S = 2 ？

正確：One-hot

✓ 各自一欄

- 每個港口各開一欄，是就填 1、不是填 0
- C 港上船的乘客 → S=0、C=1、Q=0
- 每一欄有自己的權重，互不牽制

本例　C 港 → [0, 1, 0]

類別沒有大小和距離，所以不能直接編成 0、1、2；one-hot 讓每個類別各有一欄、各有一個權重。

標準化：讓尺度更容易比較

$$z = (x - \mu_{train}) / \sigma_{train}$$

- 年齡與票價若量級不同，距離與梯度會受單位支配。
- 扣除訓練平均，再除以訓練標準差
- 標準化不會自動讓資料變成常態分布。

特徵	原始值	訓練μ / σ	標準化
年齡	50歲	30 / 10	2
票價	100元	50 / 25	2

同一位乘客的兩項特徵，都高於各自訓練平均2個標準差。

Pipeline：固定前處理與模型的順序

- 數值欄先補值，再標準化。
- 類別欄先補值，再one-hot，最後合併矩陣。
- 驗證與未來輸入都使用同一套已學規則。

欄位群	步驟1	步驟2
數值欄	中位數補值	標準化
類別欄	眾數補值	One-hot
合併後	固定欄位順序	交給模型

正確順序可降低漏做前處理與資料洩漏的風險。

Python 小教室① 讀進來、列出來：先看清楚這張表

上週用 NumPy 陣列，只能用位置取值；今天的資料是表格，每一欄都有名字，用 pandas 處理。

語法	結果	為什麼
`df = pd.read_csv("titanic.csv")`	讀成一張表 DataFrame	一般讀檔寫法；Colab 已讀好，名字叫 raw
`df.shape`	(891, 15)	891 位乘客、15 欄；和 NumPy 的 shape 一樣
`df.head(3)`	前 3 列	先看資料長什麼樣子
`df.dtypes`	每欄是數字或文字	sex、embarked 是文字，之後要轉成數字
`df["age"].describe()`	筆數 714、平均 29.7	891 – 714：年齡有缺值

Colab A0

```
from sklearn.model_selection import train_test_split
df = raw.copy()           # 練習用副本，不動到原始資料
print(df.shape)
display(df.head(3))
print(df["age"].describe().round(1))
```

NumPy 用位置取欄 H[:, 1]；pandas 用名字取欄 df["age"]。先列出來看，再決定怎麼處理。

Python 小教室② 提取：選欄、選列、條件篩選

和上週的 H[列, 欄] 一樣：先說要哪些列，再說要哪些欄。

語法	結果	為什麼
`df["age"]`	一欄（22, 38, 26, …）	用欄名取一欄
`df[["age", "fare"]]`	只有兩欄的表	雙中括號 = 欄名清單
`df.iloc[0:2, 0:4]`	前 2 列、前 4 欄	用位置取，和 H[0:2, 0:4] 一樣
`df["age"] >= 50`	True / False 一整欄	每位乘客各判斷一次
`df.loc[條件, 欄位]`	符合條件的列	loc[列, 欄]：列用條件、欄用名字
`(條件1) & (條件2)`	50 歲以上男性：52 位	每個條件加括號；「且」用 &

Colab A0

```
selected = df.loc[(df["sex"] == "male") & (df["age"] >= 50),
                  ["age", "sex", "pclass", "survived"]]
print(len(selected))
```

iloc 用位置、loc 用名字與條件；逗號前選列、逗號後選欄。

Python 小教室③　清洗：拿掉不能用的，檢查怪的

清洗不是把資料弄漂亮，而是確認每一欄在「預測當下」真的能用、而且意義正確。

語法	結果	為什麼
`df.drop(columns=["alive"])`	(891, 14)	alive 是答案的另一種寫法，會洩漏
`df["embarked"].value_counts()`	S 644、C 168、Q 77	檢查類別有沒有拼錯或怪值
`df.duplicated().sum()`	107	沒有姓名欄，可能是不同乘客，不能直接刪
`pd.to_numeric(s.str.replace(",", ""))`	"1,050.00" → 1050.0	帶逗號的文字數字要先轉成數字（股票 B1）

Colab A0

```
df = df.drop(columns=["alive"])
print(df["embarked"].value_counts())
print("重複列：", df.duplicated().sum())
```

每刪一欄、每改一筆，都要說得出理由；看到重複也要先想是不是真的重複。

Python 小教室④　補數據：先數缺值，再決定補或刪

先數有幾格缺，再決定：缺很少可以刪；缺很多就補，而且補的數字只用訓練集算。

語法	結果	為什麼
`df.isna().sum()`	age 177、embarked 2	每欄各缺幾格
`df.dropna(subset=["embarked"])`	(889, 15)	只缺 2 筆，刪掉影響很小
`train["age"].median()`	29.0	補值的數字只用訓練集算
`train["age"].fillna(29.0)`	缺值補成 29.0	驗證、測試也照用 29.0

Colab A0

```
print(df.isna().sum()[["age", "embarked"]])
train, rest = train_test_split(df, test_size=.4, stratify=df.survived, random_state=SEED)   # 先切分
age_median = train["age"].median();  print(age_median)
train_age = train["age"].fillna(age_median)
```

缺 2 格可以刪，缺 177 格要補；補值用的中位數只能從訓練集算。

Python 小教室⑤　表格內運算：新增欄位、分組統計、排序

表格可以一整欄一起算，不用寫迴圈；也能分組比較，先看出資料裡的規律。

語法	結果	為什麼
`df["family"] = df["sibsp"] + df["parch"]`	新欄：同行家人數	整欄相加，和 NumPy 逐元素運算一樣
`df.groupby("sex")["survived"].mean()`	女 0.742、男 0.189	分組算平均 = 各組生還率
`df.groupby("pclass")["survived"].mean()`	1 等 0.63、3 等 0.24	艙等越高，生還率越高
`df.sort_values("fare", ascending=False)`	最高票價 512.33	由大到小排序，找極端值

Colab A0

```
df["family"] = df["sibsp"] + df["parch"]
print(df.groupby("sex")["survived"].mean().round(3))
display(df.sort_values("fare", ascending=False).head(3))
```

整欄運算造新特徵，groupby 看各組差異；先看懂資料，再交給模型。

Python 小教室⑥　切分，最後變成模型要的矩陣 X

處理好的表格要先切成訓練、驗證、測試，再轉成全是數字的矩陣 (N, d)，回到 Class 1。

語法	結果	為什麼
`train_test_split(df, test_size=0.4, stratify=...)`	訓練 60%、其餘 40%	其餘再切一半：驗證、測試
`len(train), len(valid), len(test)`	534、178、179	三份各自的筆數
`t_preprocess(FEATURES).fit_transform(train[FEATURES])`	補值 + 標準化 + one-hot	規則只從訓練集學（fit）
`X_train.shape`	(534, 12)	7 個特徵經 one-hot 變成 12 欄：(N, d)

Colab A0

```
FEATURES = ["age", "fare", "sibsp", "parch", "pclass", "sex", "embarked"]
train, rest = train_test_split(df, test_size=.4, stratify=df.survived, random_state=SEED)
valid, test = train_test_split(rest, test_size=.5, stratify=rest.survived, random_state=SEED)
X_train = t_preprocess(FEATURES).fit_transform(train[FEATURES]);  print(len(train), len(valid), len(test), X_train.shape)
```

表格 → 切分 → 補值與編碼 → 矩陣 X：之後的模型都從這個 X 開始。

Python 小教室⑦　scikit-learn：fit、predict、predict_proba

核心理論

- fit(X,y)用訓練資料學模型參數。
- predict(X)輸出預測，predict_proba(X)提供支援模型的機率。
- 同樣的介面不代表所有模型使用同一個最佳化方法。

方法	輸入	工作
fit	X_train, y_train	學模型參數
predict	X_new	類別或數值
predict_proba	X_new	支援模型的機率

所有模型都是同三個動作：fit 學、predict 判類別、predict_proba 給機率。

鐵達尼號實作：先建立可解釋的基準

實作連結

- A3先以固定特徵訓練logistic模型。
- 逐筆對照機率、預測類別與真實標籤。
- 再改一項特徵，記錄驗證結果與原因。

實驗項目	先固定	再比較
起點	資料切分	多數類 vs logistic
特徵實驗	模型與樣本	移除一個特徵
門檻實驗	訓練好的模型	0.3 / 0.5 / 0.7

先有基準，才知道更複雜的方法增加了什麼。

實作① Titanic A1–A3：先建立基準

▶ 動手做・打開 Colab

實作時間

建議 17 分鐘　Colab 步驟

- ☐ A1　看資料：shape、缺值、選欄位
- ☐ A2　先切分，再只用訓練集補值
- ☐ A3　多數類基準 vs Logistic（0 層網路）



完成標準

能說出：驗證 accuracy 比「永遠猜未生還」高多少？

建議 17 分鐘；卡住先舉手，不要跳過切分直接看測試集。

休息回來：四個問題熱身

休息回來

1 分數 z 和機率 p 差在哪裡？

提示：z 沒有上下限；sigmoid 把它壓進 0 ~ 1

2 為什麼有把握地答錯，交叉熵特別大？

提示：−ln p 沒有上限，梯度 p−y 也夠大

3 為什麼補值要先切分、只用訓練集？

提示：否則驗證、測試的資訊會漏進訓練

4 驗證集和測試集各做什麼決定？

提示：驗證：選設定；測試：定案後只看一次

每題請一位同學用一句話回答，答不出來就回到對應頁。

第五段　泛化

章節導覽

上一段

實作①：第一個 logistic 模型已經跑出訓練與驗證成績。

這一段要學習

訓練分數很高，代表模型真的學會了嗎？

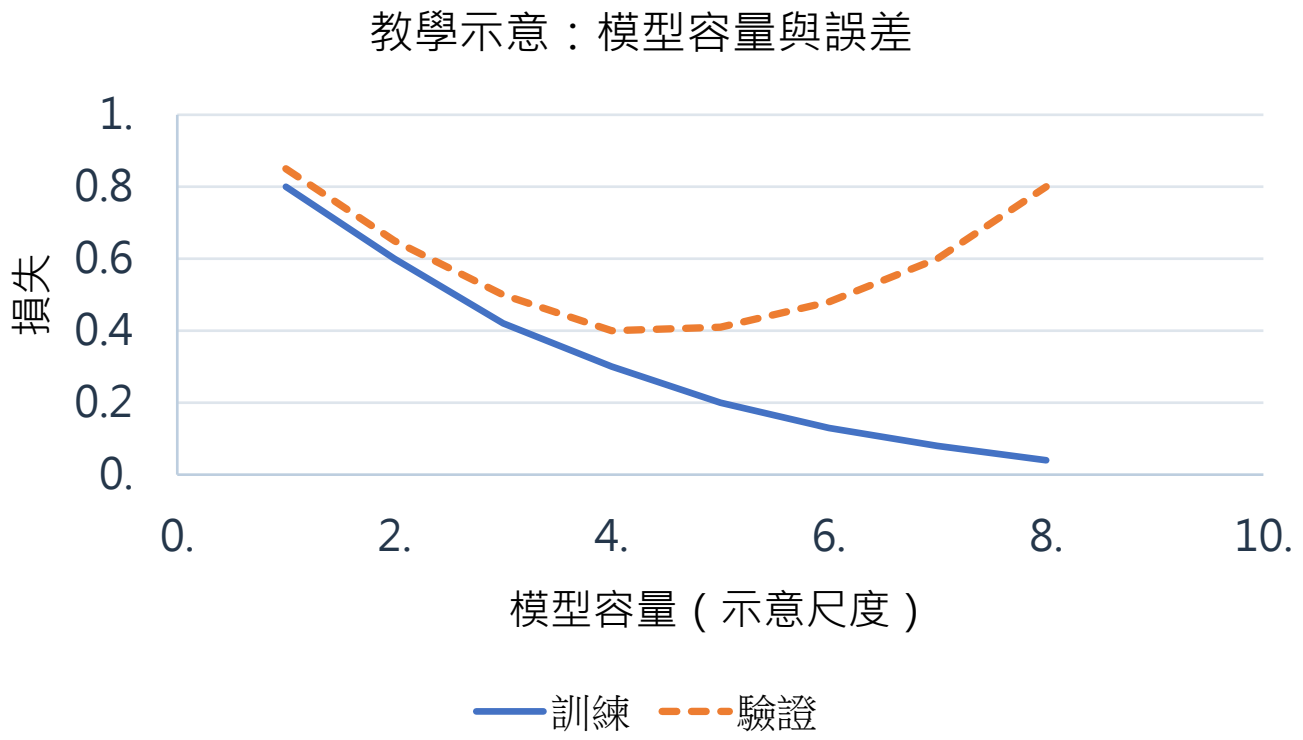
1. 模型容量與過擬合（Demo 4）
2. L2、Dropout 與早停：別讓模型背答案
3. 接著實作②：自己訓練神經網路看曲線



背熟考古題，不等於會解新題目。

模型容量與資料量

- 更寬、更深或更彎曲的模型，可表達更多函數。
- 容量增加可能減少訓練誤差，也可能記住雜訊。
- 需要驗證資料判斷額外複雜度是否值得。



示意曲線：訓練誤差下降，不保證新資料誤差跟著下降。

Demo 4　親手製造過擬合

互動 Demo

①複習 ②機率 ③損失·NN ④資料 **⑤泛化** ⑥評估 ⑦SVM ⑧股票

現場 5 分鐘　操作步驟

- ☐ 複雜度從 1 慢慢拉到 15
- ☐ 每一格先讓學生猜：測試誤差升還降？
- ☐ 找出測試誤差最低的複雜度

觀察重點

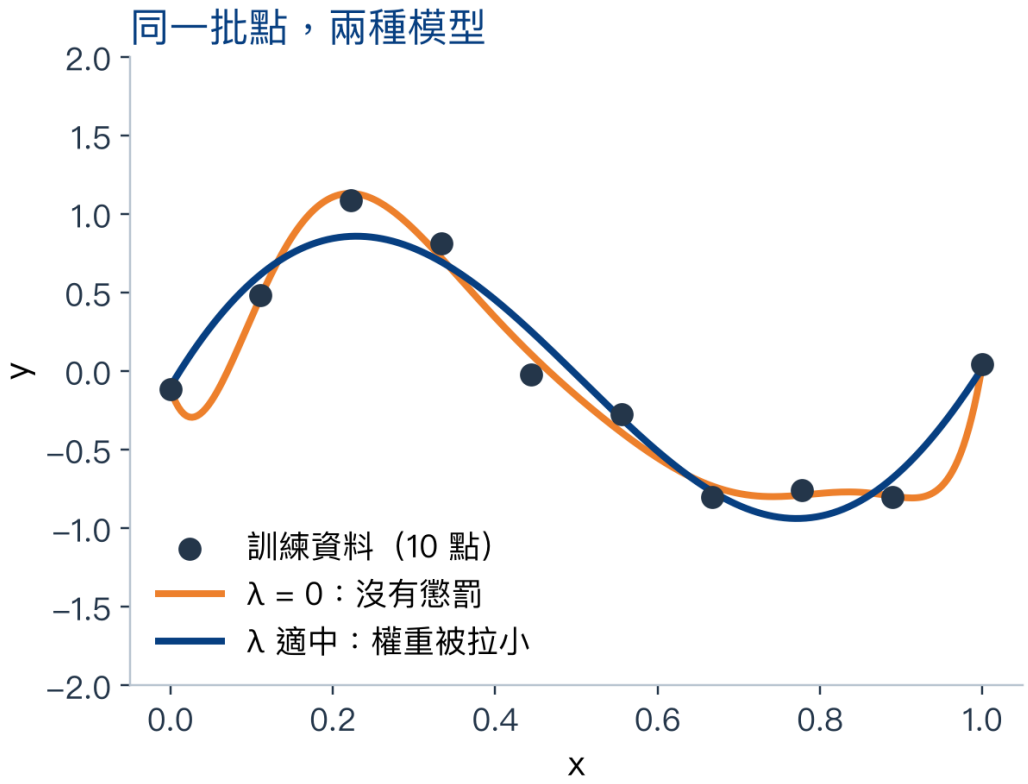
訓練誤差一路下降，測試誤差先降後升：這就是過擬合。

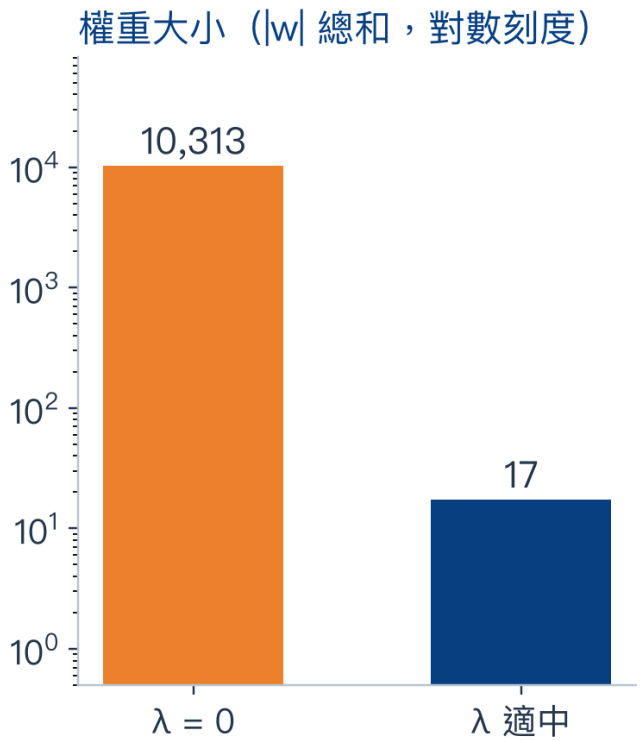


檔案：CLASS2/互動demo/demo4_過擬合.html（離線可開）

L2 正則化：權重太大就罰分，曲線才不會亂彎 (修改走勢)

$$J = L + \lambda \cdot \Sigma w^2$$



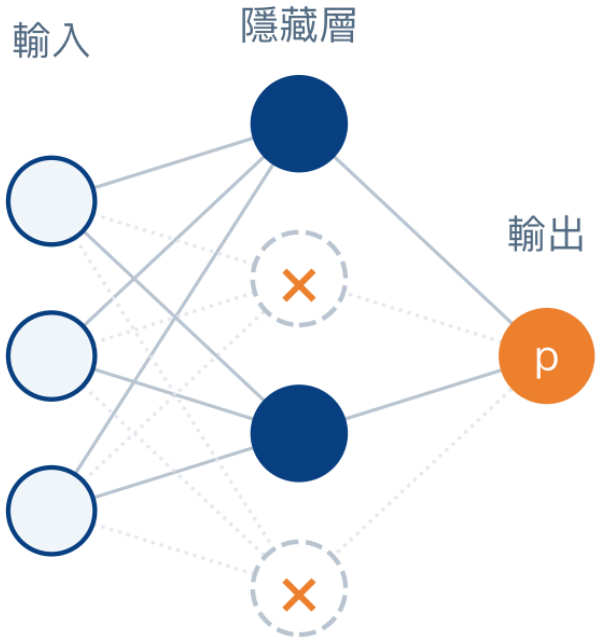


權重越大，曲線越能繞彎去貼每一個點；L2 對大權重罰款，w越大梯度變化越大 ，模型只好用小權重→畫出平滑的線。

λ是超參數：太小等於沒罰，太大會連該學的都學不到

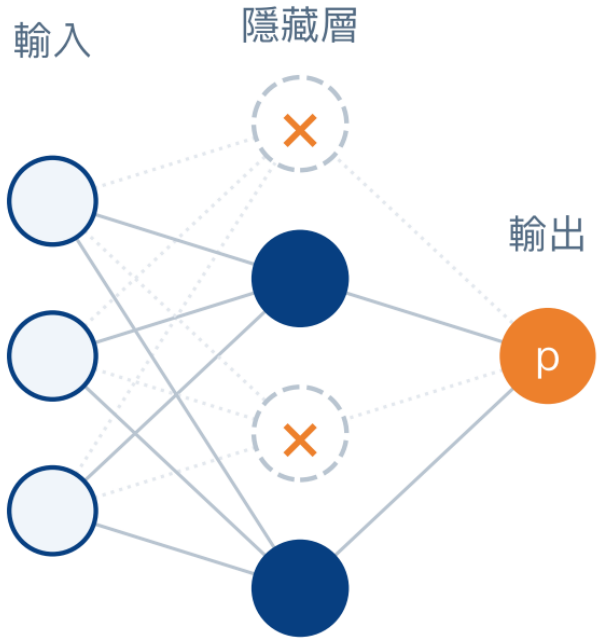
Dropout：訓練時隨機關掉隱藏層的神經元

訓練第 1 批



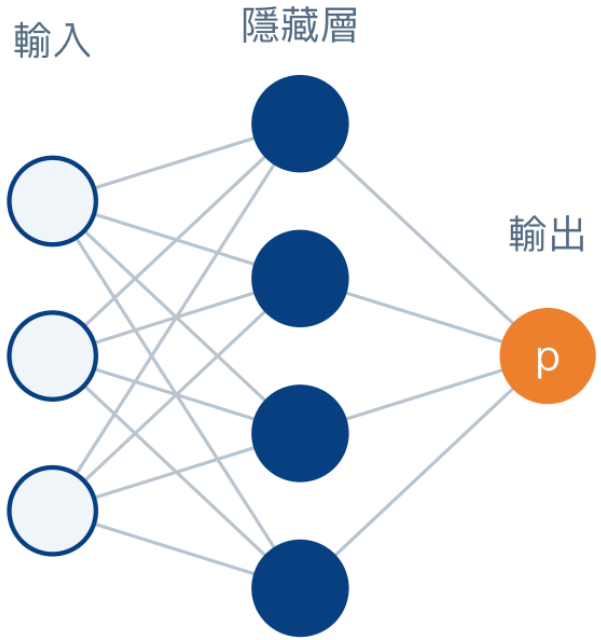
隨機關掉 2 個隱藏神經元
(輸出暫時設為 0)

訓練第 2 批



隨機關掉 2 個隱藏神經元
(輸出暫時設為 0)

預測時

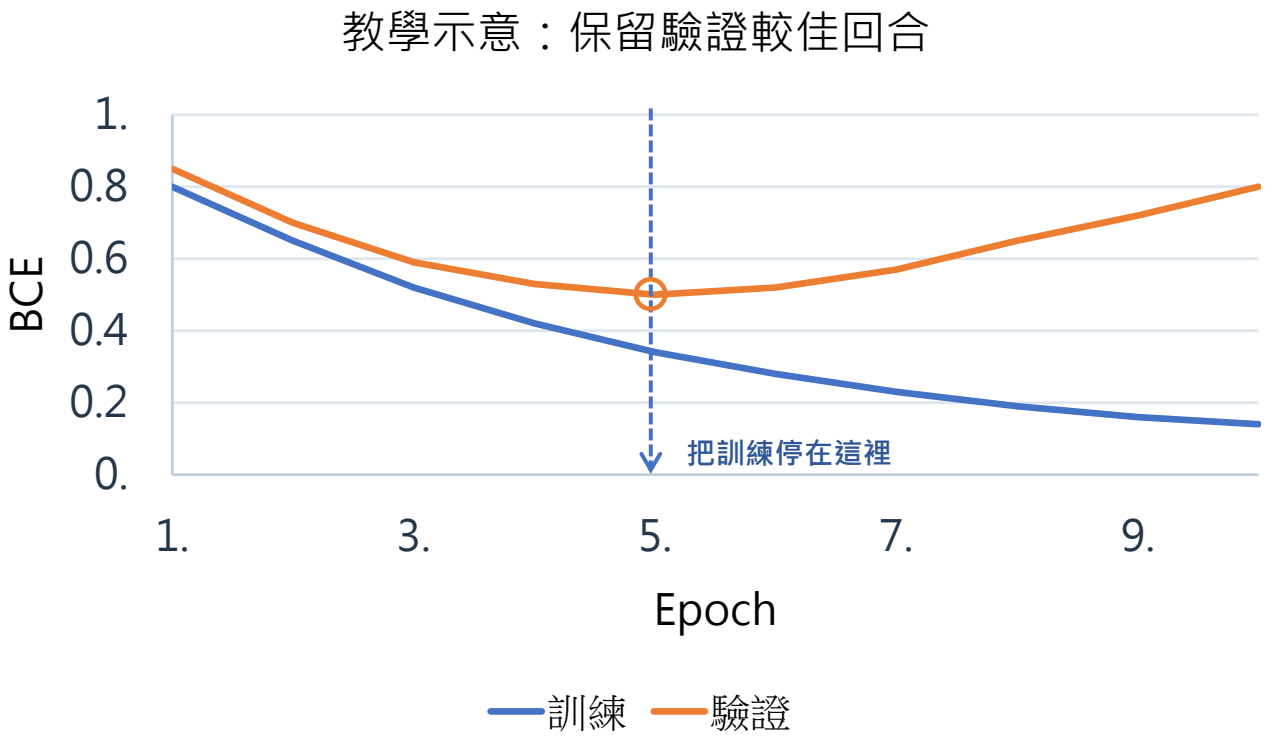


全部神經元都上場

每一批訓練都隨機關掉一些隱藏神經元，其他神經元就得學會補位；預測時全部上場，不再依賴少數幾個。

早停 Early stopping：以驗證表現選停止點

- 訓練loss通常持續下降，驗證loss可能先降後升。
- 保留驗證最佳時的權重，避免只拿最後一輪。
- 需容許短暫波動，不能看到一次上升就下結論。



圖為教學示意；Colab 控制台勾選 EARLY_STOP 就會早停並退回最佳權重。

實作② Titanic A5：訓練第一個神經網路

▶ 動手做・打開 Colab

實作時間

①複習 | ②機率 | ③損失·NN | ④資料 | ⑤泛化 | ⑥評估 | ⑦SVM | ⑧股票

建議 14 分鐘　Colab 步驟

- ☐ A5-1　填空：手算一步訓練（N1–N6）
- ☐ A5-2　控制台：調旋鈕，看曲線與目標看板
- ☐ A5-3 / A5-4　挑戰任務，把損失壓到目標線下



完成標準

能說出：你拿到哪面牌？哪一個旋鈕讓損失降最多？

時間15 分鐘

第六段　評估：答對 80% 夠用嗎

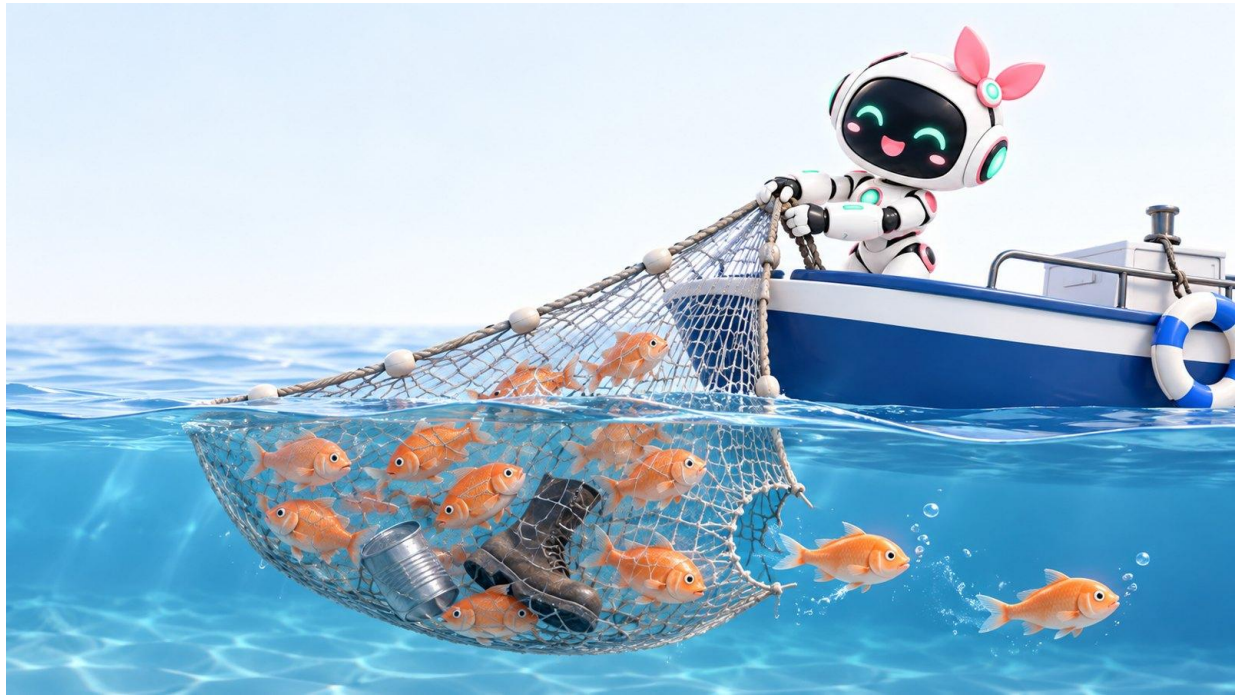
上一段

實作②：logistic 與神經網路都有了驗證成績。

這一段要回答

Accuracy 78% 夠好嗎？漏掉生還者與誤判的人，代價一樣嗎？

1. 類別不平衡與多數類基準
2. 混淆矩陣、Precision、Recall、F1
3. 門檻取捨與機率校準

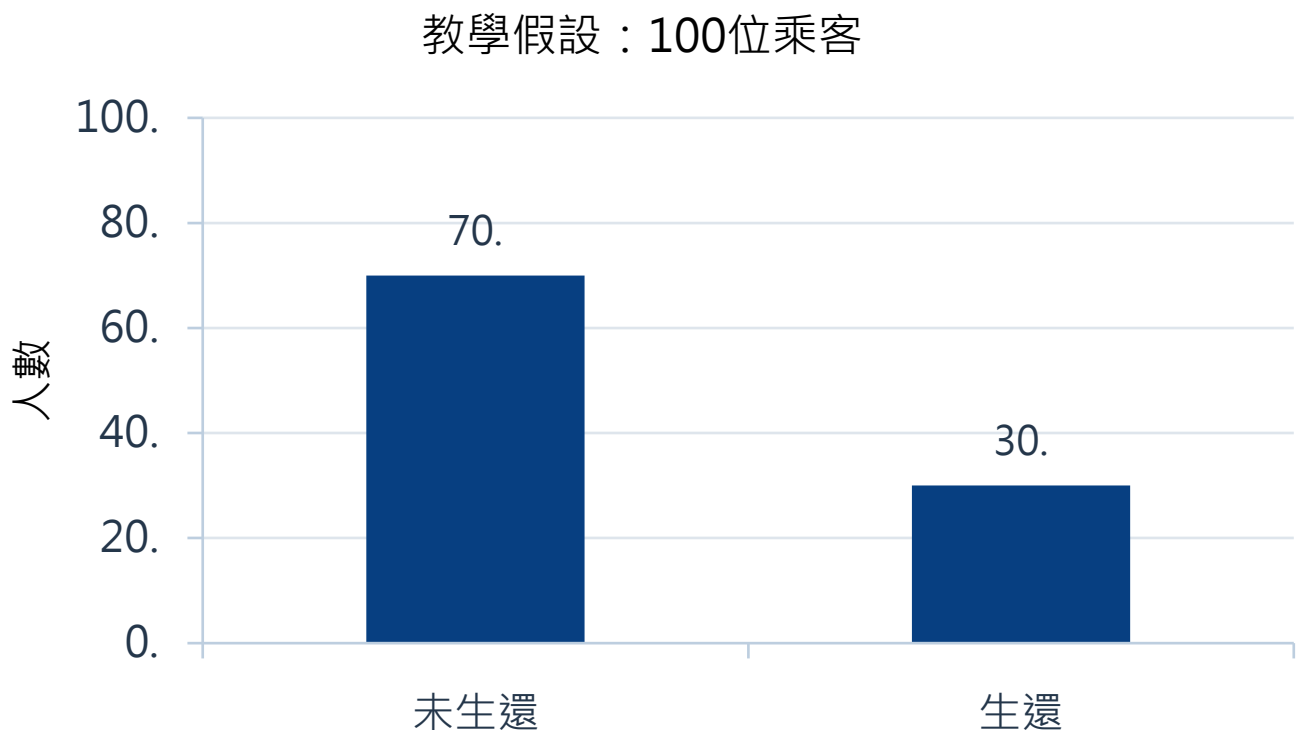


撒網捕魚：撈到多少目標魚？撈上來的又有多少是雜物？

類別不平衡：高Accuracy (準確率)也可能沒有用

Accuracy = correct / N

- Accuracy = 猜對的人數/全部人數（N）。
- 100 人中 70 人未生還：全部猜未生還就有 70%。
- 這樣的模型一個生還者都沒找到，所以要看錯在哪裡。



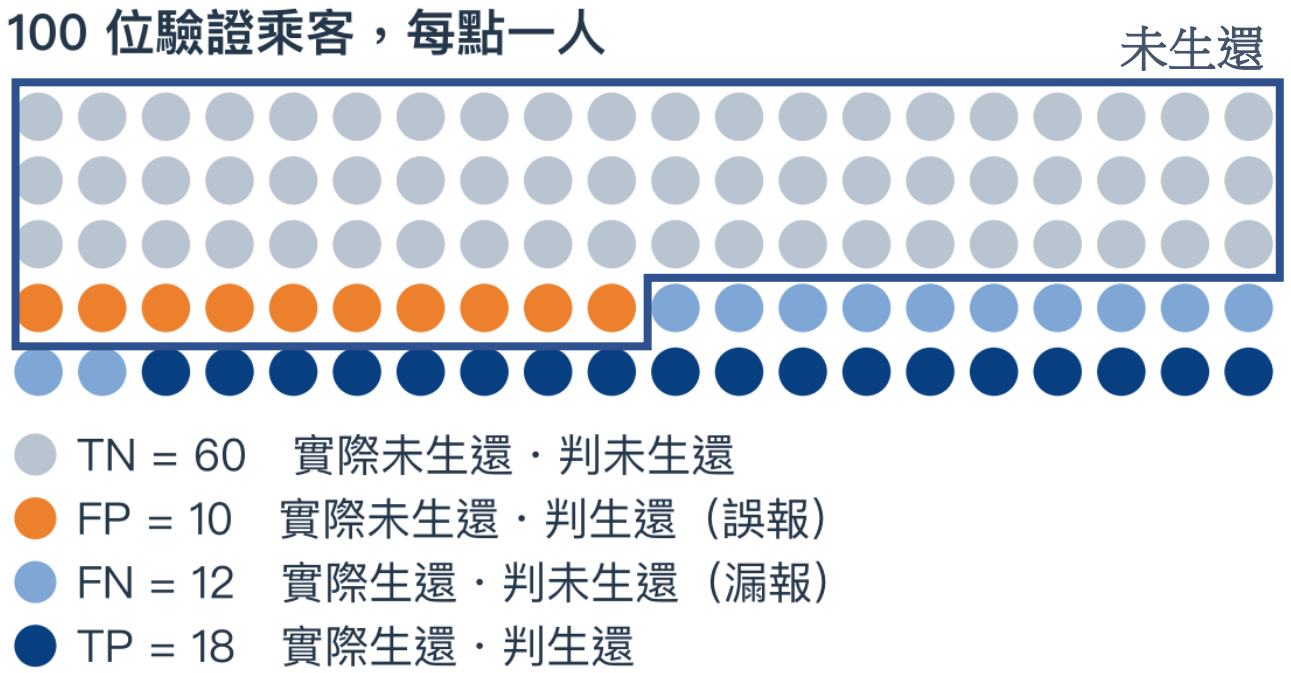
模型100人全猜未生還，猜中70人，準確率70/100=70%，但生還者Recall=0/30 (實際生還)=0

混淆矩陣：兩種正確與兩種錯誤

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN) = (18 + 60) / 100 = 78\%$$

T/F：猜對或猜錯　P/N：模型判生還或未生還

符號	意思	本例
TP	實際生還、判生還（抓對）	18
TN	實際未生還、判未生還	60
FP	實際未生還、判生還（誤報）	10
FN	實際生還、判未生還（漏報）	12

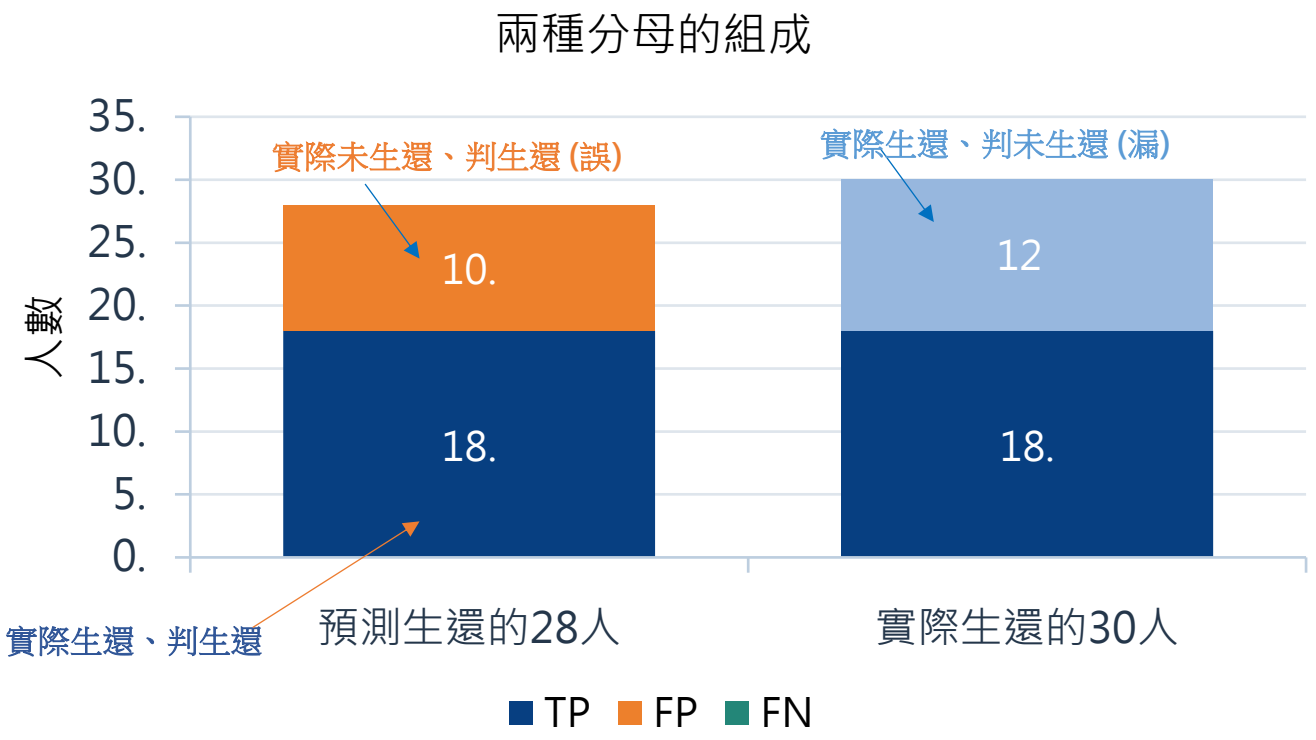


100 位驗證乘客：TN 60、FP 10、FN 12、TP 18；列 = 實際、欄 = 預測。

Precision (精確率)與Recall (召回率)的分母不同

Precision=TP/(TP+FP)　Recall=TP/(TP+FN)

- Precision：模型喊生還的人，有多少真的生還？
- Recall：真正生還的人，有多少被模型找出？
- 同樣的TP，搭配不同分母會回答不同問題。



例：Precision=18/28≈64.3%，Recall=18/30=60%。

Accuracy = (TP + TN) / (TP + TN + FP + FN)

Demo 5　一條門檻線，四種結果

互動 Demo

現場 5 分鐘　操作步驟

- [ ] 門檻由 0.5 拉到 0.2，再拉到 0.8
- [ ] 每次唸出 TP、FN、FP、TN 的變化
- [ ] 橘框點 = 目前犯的錯：誤報還是漏報？

觀察重點

門檻左移時 Recall 上升、Precision 可能下降；模型權重完全沒變。



檔案：CLASS2/互動demo/demo5_評估指標.html（離線可開）

F1：兼顧Precision與Recall

$$F1 = 2PR/(P+R) = 2TP/(2TP+FP+FN)$$

- F1是兩者的調和平均，偏向受較低的一項限制。
- 它忽略TN，不是所有任務的最佳總分。
- 若兩種錯誤成本不同，還需另外設計評估。

指標	代入本例	結果
Accuracy	(60+18)/100	78%
Precision	18/(18+10)	64.3%
Recall	18/(18+12)	60%
F1	36/(36+10+12)	62.1%

同一例子：36/(36+10+12)=36/58≈62.1%。

直覺圖解：機率校準就像天氣預報

生活中的例子

氣象主播連續十天都說「降雨機率 70%」，結果其中七天下雨，這位主播就是校準良好。若十天都下雨或只下兩天，70% 這個數字就不可信。

對應到模型

每天的預報	**模型輸出 p**
十天放一組	**把 p 相近的樣本分組**
實際下雨天數	**該組實際正類比例**
兩者接近	**校準良好**



0.8 不是「這個人一定生還」，而是「一群像他的人，約八成生還」。

個人情境與壽險的延伸

應用思考｜課後閱讀（放映略過）

- A6輸入假想年齡、艙等、票價等現有欄位。
- 改一項特徵，觀察模型機率如何變動。
- 壽險也會估計族群風險，但需不同資料與時間定義。

面向	Titanic課堂	壽險延伸
事件	一次事故生還	指定期間死亡
資料	歷史乘客欄位	另需適當風險資料
輸出	模型估計機率	族群風險估計
限制	非個人命運	另需定價與法規考量

Titanic模型不能直接用來估計現代壽命或替個人定價。

實作③　門檻、最後測試、如果你在船上

▶ 動手做・打開 Colab

實作時間

建議 12 分鐘　Colab 步驟

- ☐ A4　門檻 0.5 → 0.3 → 0.7，看 TP、FP、FN
- ☐ A7　選定模型與門檻，開啟測試集
- ☐ A6　填入你的資料；A6-2 加入自己的特徵



完成標準

能說出：降低門檻時哪種錯誤變多？模型說你幾%，那代表什麼？

建議 12 分鐘；卡住先舉手，不要跳過切分直接看測試集。

鐵達尼號預測完成：從資料到最後測試

- 資料：切分、補值、one-hot → 矩陣 X (534, 12)
- 模型：先贏過基準，再比 Logistic 與神經網路
- 定案後才開測試集，而且只看一次

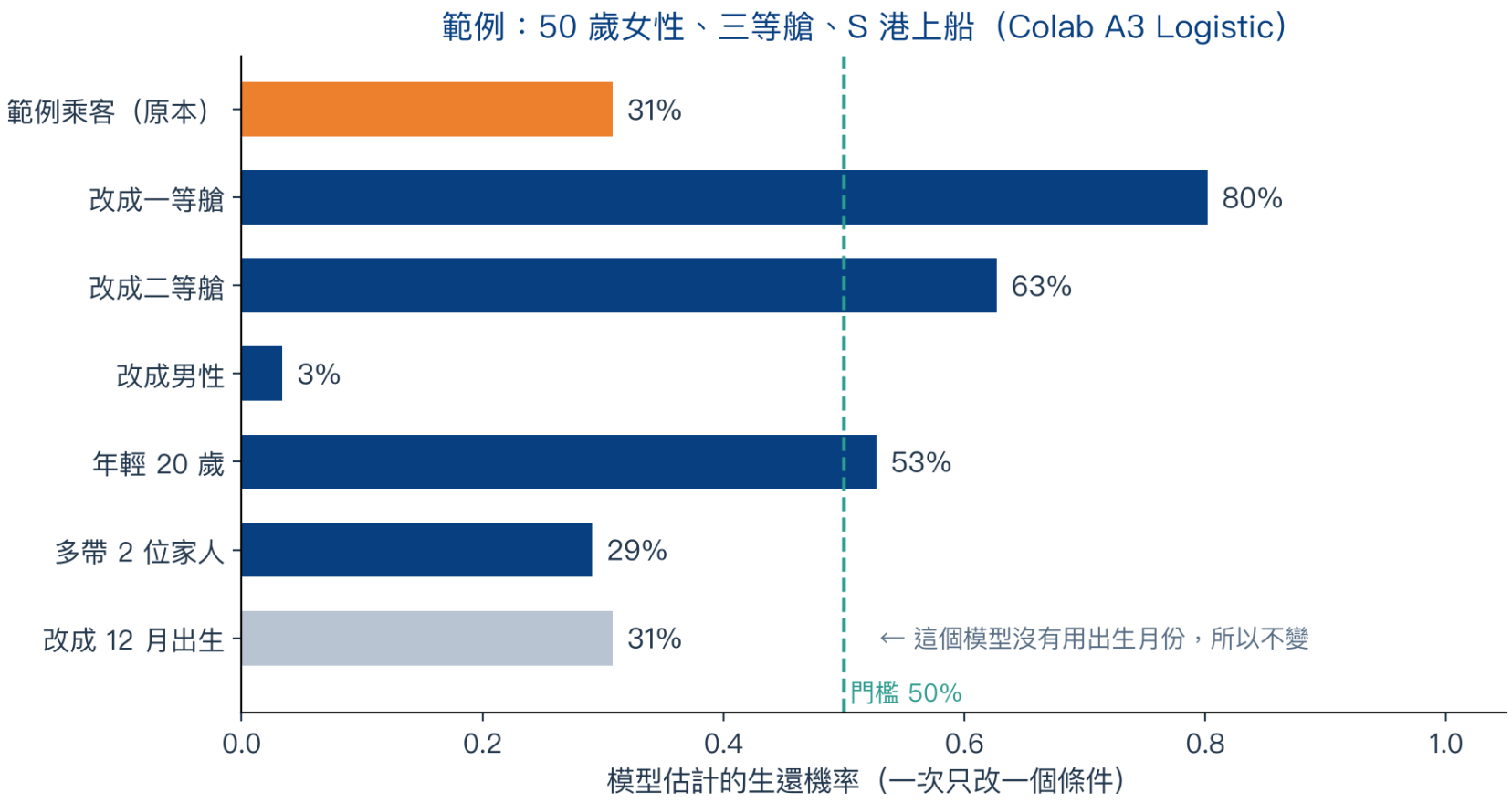
步驟	驗證 accuracy
多數類基準	0.618
Logistic	0.815
神經網路（16）	0.843
神經網路（64）	0.837
最後測試（Logistic）	0.805

測試 0.805 接近驗證 0.815：沒有對驗證集過度調整；神經網路只多一點點。

如果你在船上：填入你的資料，模型怎麼說？

實作延伸

①複習 ②機率 ③損失·NN ④資料 ⑤泛化 **⑥評估** ⑦SVM ⑧股票

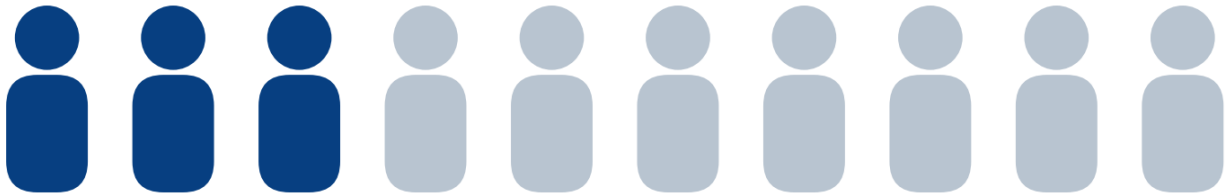


一次只改一個條件，就看得出模型最在意什麼：性別與艙等影響最大，家人多寡影響很小。

想一想⑥　生存率只有 30% 代表什麼

思考題

模型說一位乘客的生存率只有 30%，哪個解讀最合理？



10 位和你條件相似的 1912 年乘客：約 3 位生還、7 位未生還

A　這個人十次裡會死七次

B　像他這樣的一群人，約三成生還

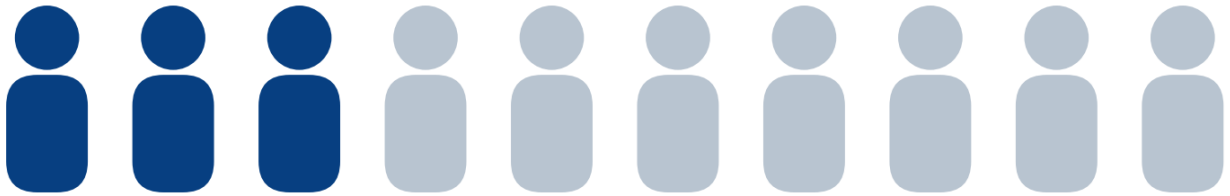
C　低於 50%，所以他一定不會生還

請先選擇，再用圖中的證據解釋。

想一想⑥　生存率只有 30% 代表什麼：解答

思考題

模型說一位乘客的生存率只有 30%，哪個解讀最合理？



10 位和你條件相似的 1912 年乘客：約 3 位生還、7 位未生還

A　這個人十次裡會死七次

B　像他這樣的一群人，約三成生還

C　低於 50%，所以他一定不會生還

答案：B　像他這樣的一群人，約三成生還

30% 描述的是「一群條件相似的人」的比例；門檻只是把機率切成判定，不會改變機率本身。

第七段　SVM：最寬的那條馬路

章節導覽

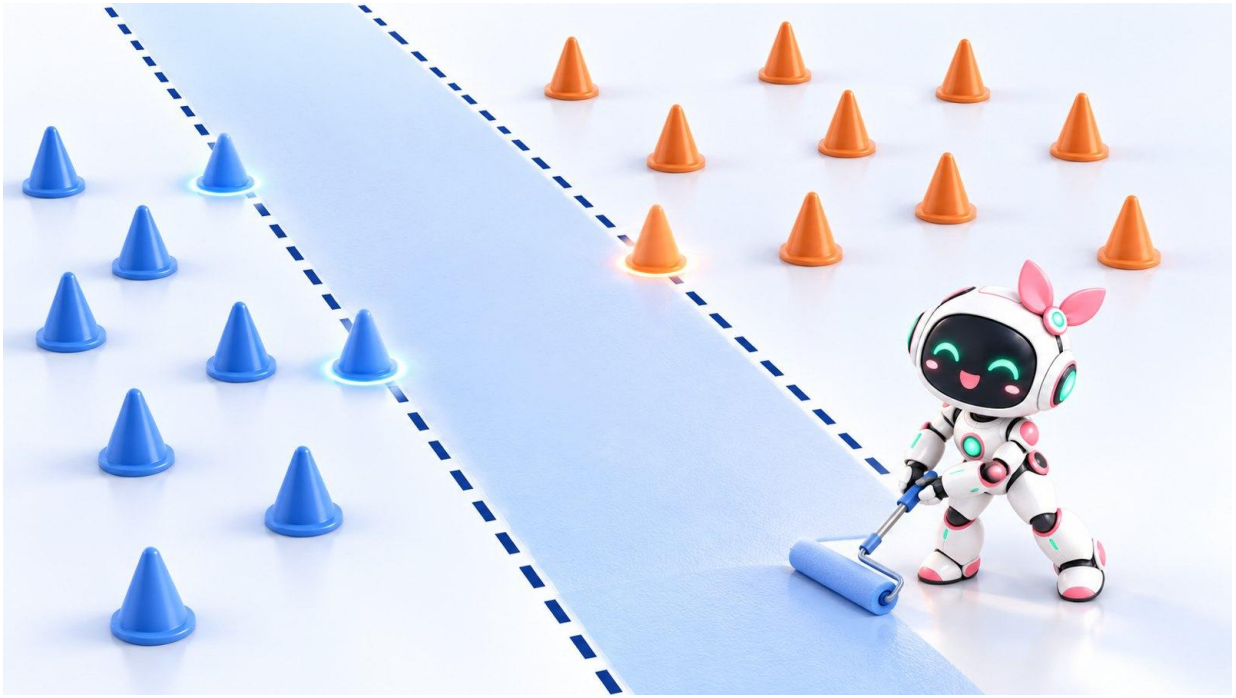
上一段

鐵達尼號已用 logistic 與神經網路完成預測。

這一段要學習

如果不追求機率，只追求「離兩類都最遠」的分界線呢？

1. 例子①：哪條線最好 → 超平面與間隔
2. 例子②：有例外 → C
3. 例子③：直線分不開 → 核函數與 RBF



在兩群交通錐之間，鋪出最寬的一條馬路。

SVM 的重點：和 Logistic、神經網路差在哪

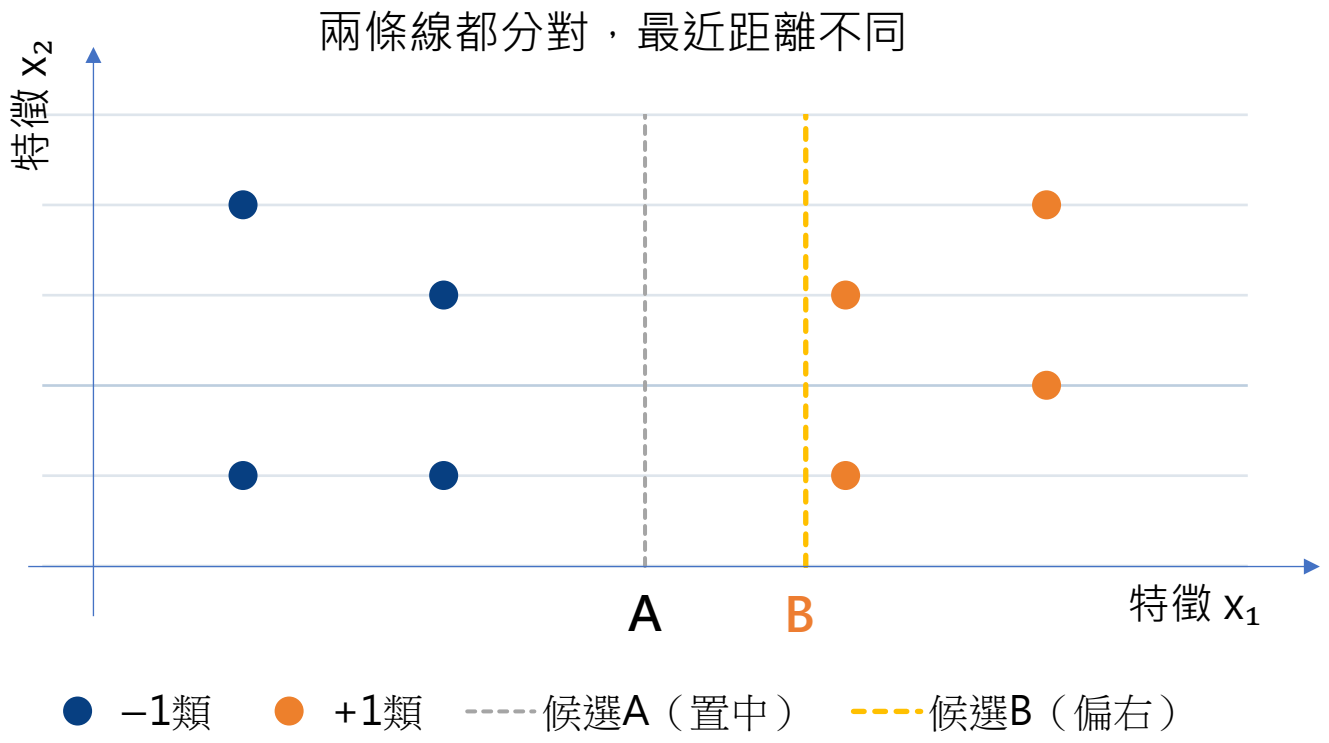
- Logistic、神經網路：追求機率準，每個樣本都影響邊界。
- SVM：追求分界線離兩類最遠，只看最靠近的點。
- 三個例子帶出三個工具：間隔、C、核函數。

ITEM	Logistic	神經網路	SVM
追求什麼	機率準	機率準	離兩類最遠
輸出	機率 p	機率 p	在哪一邊
邊界由誰決定	全部樣本	全部樣本	最近的少數點
彎曲邊界	--	隱藏層	核函數

SVM 不問「機率多少」，只問「這條線夠不夠安全」；邊界由少數支持向量決定。

例子①：兩條線都分對，哪條比較好？

- 兩條線都把藍、橘分對，訓練正確率一樣。
- 偏右的線離藍點很近，新資料稍微偏一點就錯。
- SVM 的答案：選離兩類都最遠的那條（間隔最大）。



先請學生選：A 置中、B 偏右，哪條比較放心？選 A 的理由就是「間隔」。

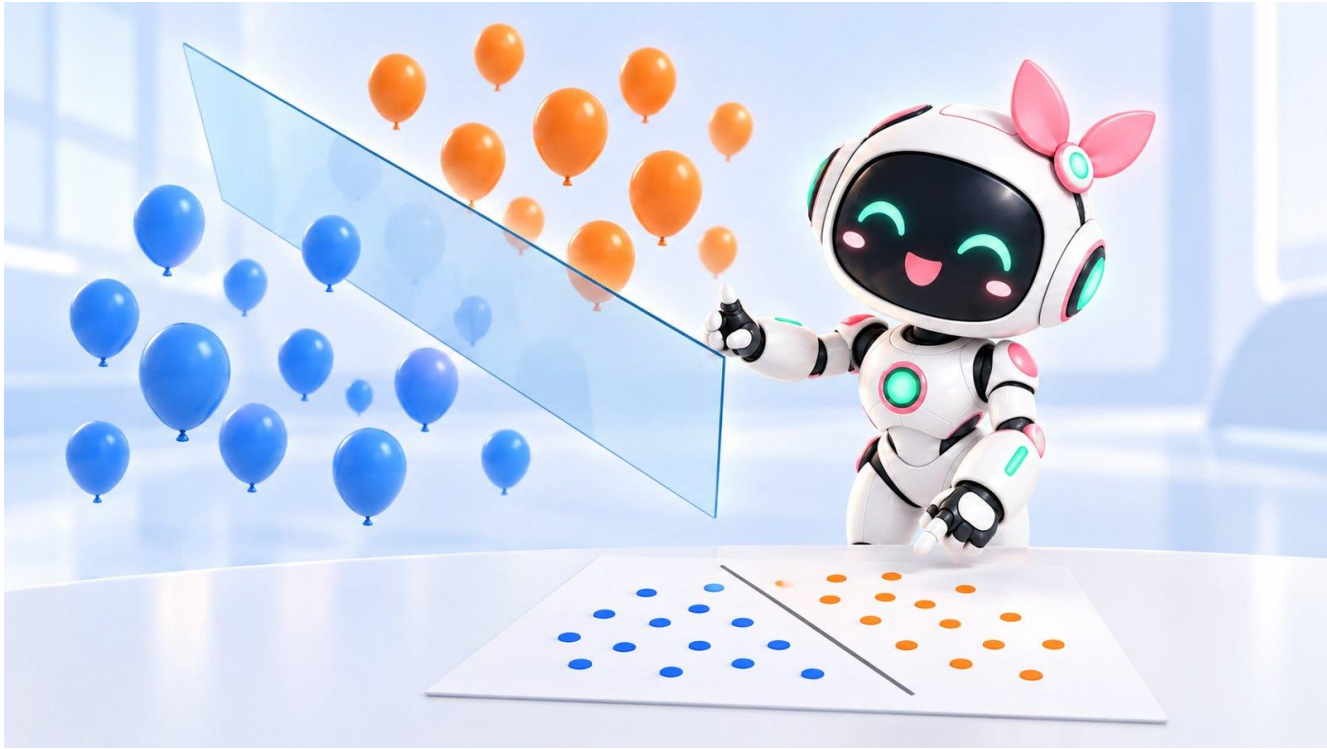
直覺圖解：超平面 = 更高維度的一刀切

生活中的例子

兩個特徵時，一條直線就能分開紙上的點；三個特徵時，點浮在空中，要用一片平板玻璃來分。特徵再多，仍是「一刀平切」，只是畫不出來。

對應到模型

2 個特徵	**直線**
3 個特徵	**平面**
d 個特徵	**d−1 維超平面**
垂直刀面的方向	**權重向量 w**



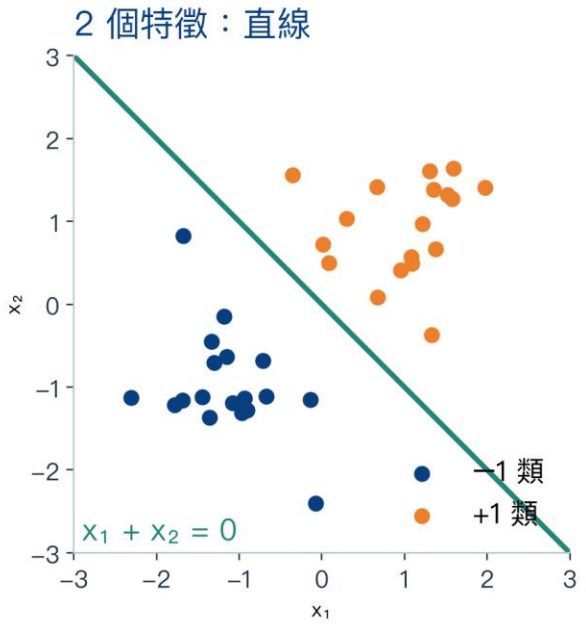
邊界方程式永遠是 w·x + b = 0，差別只在 x 有幾個分量。

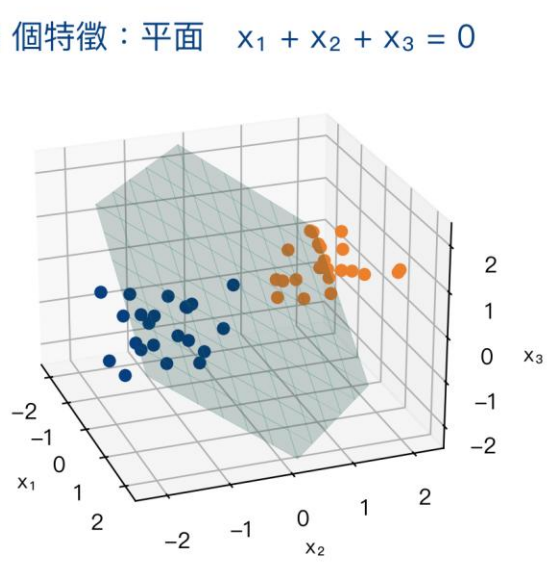
超平面與符號：SVM如何判類別

$\hat{y} = \text{sign}(w^T x + b)$

公式參考．看懂即可

- 分界線（超平面）：w·x + b = 0。
- 算出分數 s = w·x + b：正的判 +1、負的判 −1。
- SVM 慣用 −1 / +1 當標籤，方便寫「在哪一邊」。



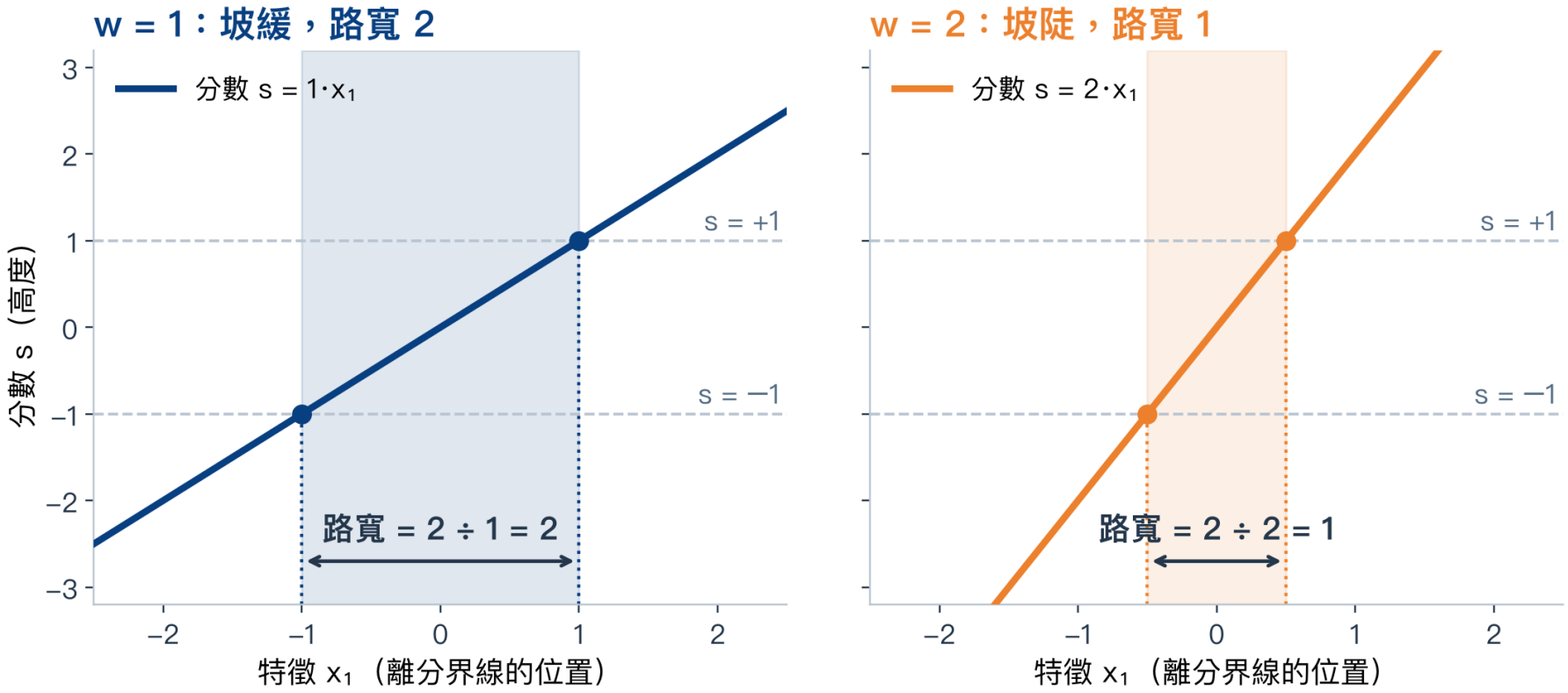


共同形式：w·x + b = 0　（d 個特徵時是 d−1 維的「超平面」）

直線、平面、超平面是同一件事：w·x + b = 0，只是特徵數不同。

為什麼路寬 = 2 / ||w|| ：把分數想成坡度

width = 2 / ||w||



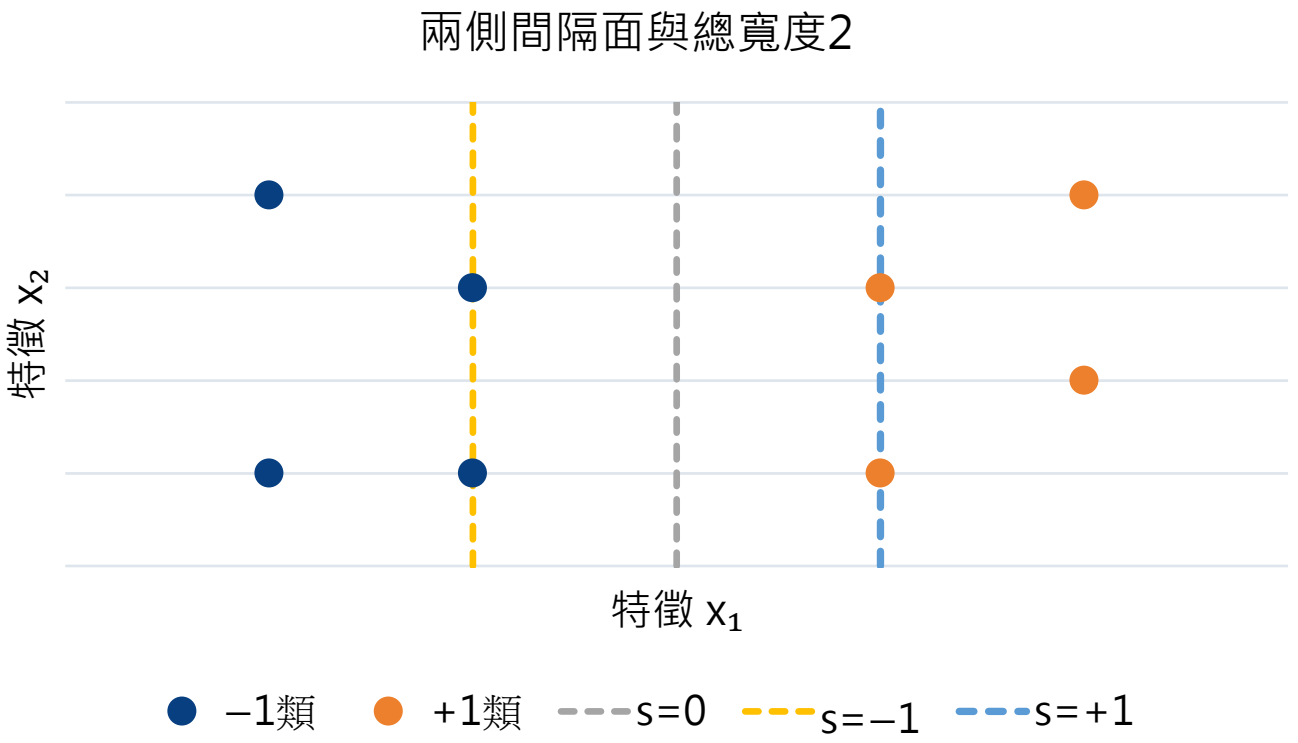
分數從 −1 爬到 +1 要升高 2；坡度 ||w|| 越大，走越短就爬完，路越窄。所以要路寬，就要 ||w|| 小。

數學：把「路要最寬」寫成公式

$$\min \tfrac{1}{2}||w||^2 \quad \text{subject to} \quad y_i(w \cdot x_i + b) \geq 1$$

公式拆解：路有多寬

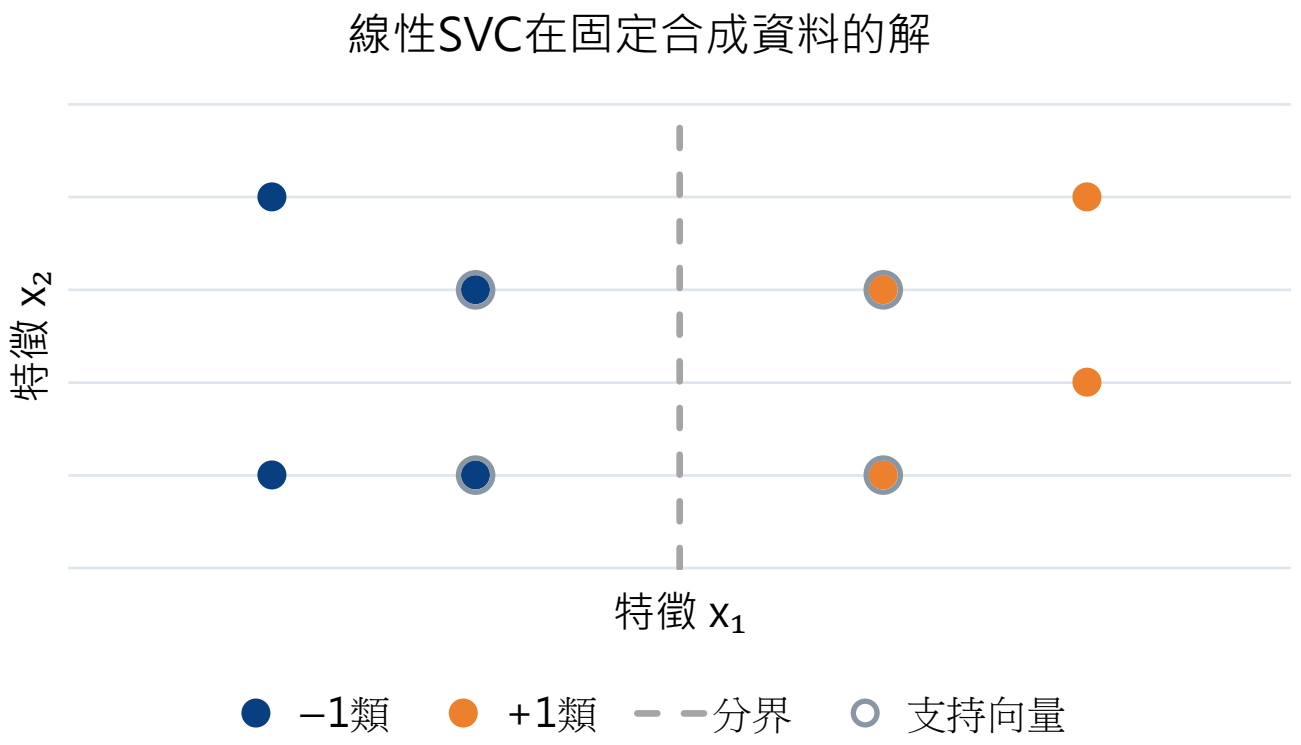
式子	意思	圖上
$w \cdot x + b = 0$	路的中線（分界）	$s = 0$
$w \cdot x + b = \pm 1$	兩側路邊	$s = \pm 1$
$y_i(w \cdot x_i + b) \geq 1$	每個點都在路外	不准壓線
$\min \tfrac{1}{2}\|\|w\|\|^2$	$\|\|w\|\|$ 越小，路寬 $2/\|\|w\|\|$ 越大	路越寬越好



路寬 = 2 / ||w||：讓 ||w|| 越小，路就越寬；條件是每個點都要在路外。

支持向量：最靠近邊界的關鍵樣本

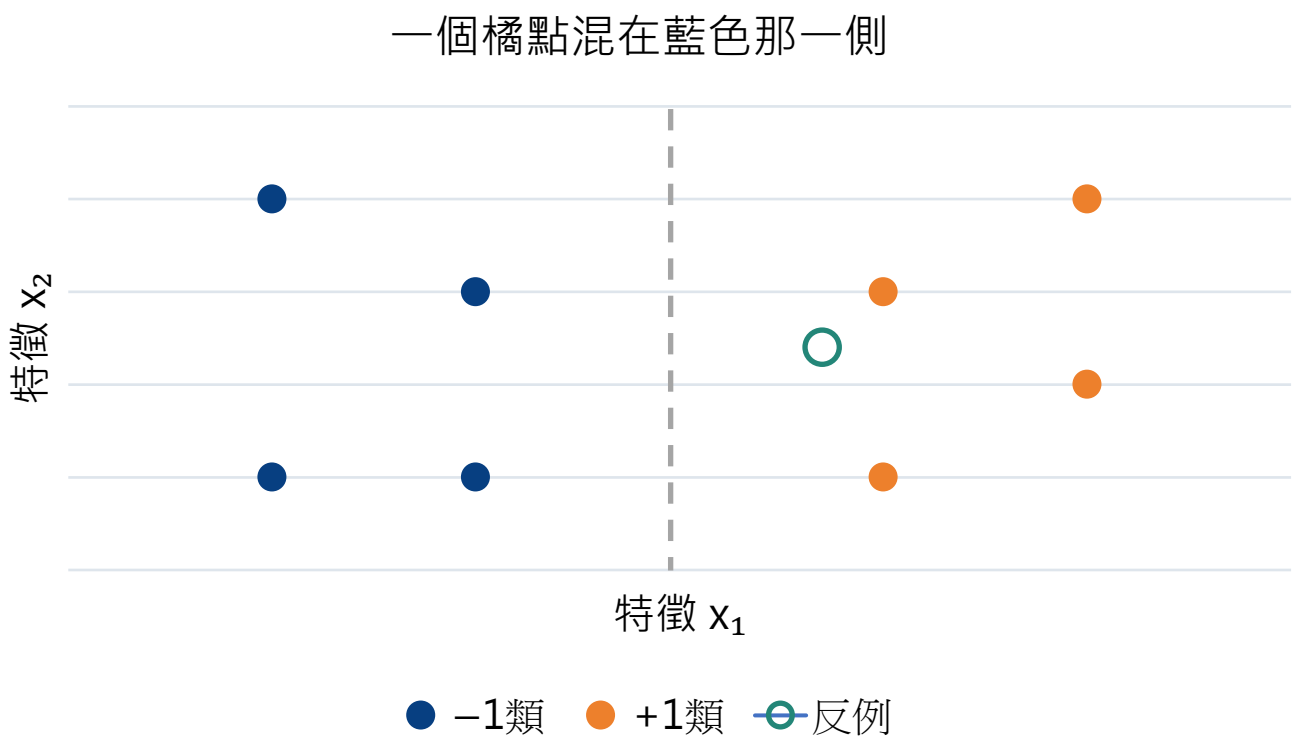
- 貼著路邊的點叫支持向量（圈起來的點）。
- 路的位置只由這幾個點決定。
- 移動遠處的點，路完全不變。



圈出的支持向量是演算法的幾何角色，不是「最重要的人」。

例子②：有一個點放錯邊，還要硬分全對嗎？

- 一個橘點混在藍色那一側。
- 硬要全部分對：線要歪過去、路變得很窄。
- 允許少數例外：路比較寬、比較穩。



先請學生想：為了一個可能是雜訊的點，值得把路擠窄嗎？這個取捨就是 C。

直覺圖解：C 是裁判有多嚴格

生活中的例子

有一個橘色交通錐誤放在藍色區。嚴格的裁判要求道路避開所有交通錐，路又窄又彎；寬容的裁判允許少數例外，換來又寬又直的道路。

對應到模型

嚴格裁判	**大 C：違規罰很重**
寬容裁判	**小 C：容許少數違規**
道路寬度	**間隔 margin**
誤放的交通錐	**雜訊或例外樣本**



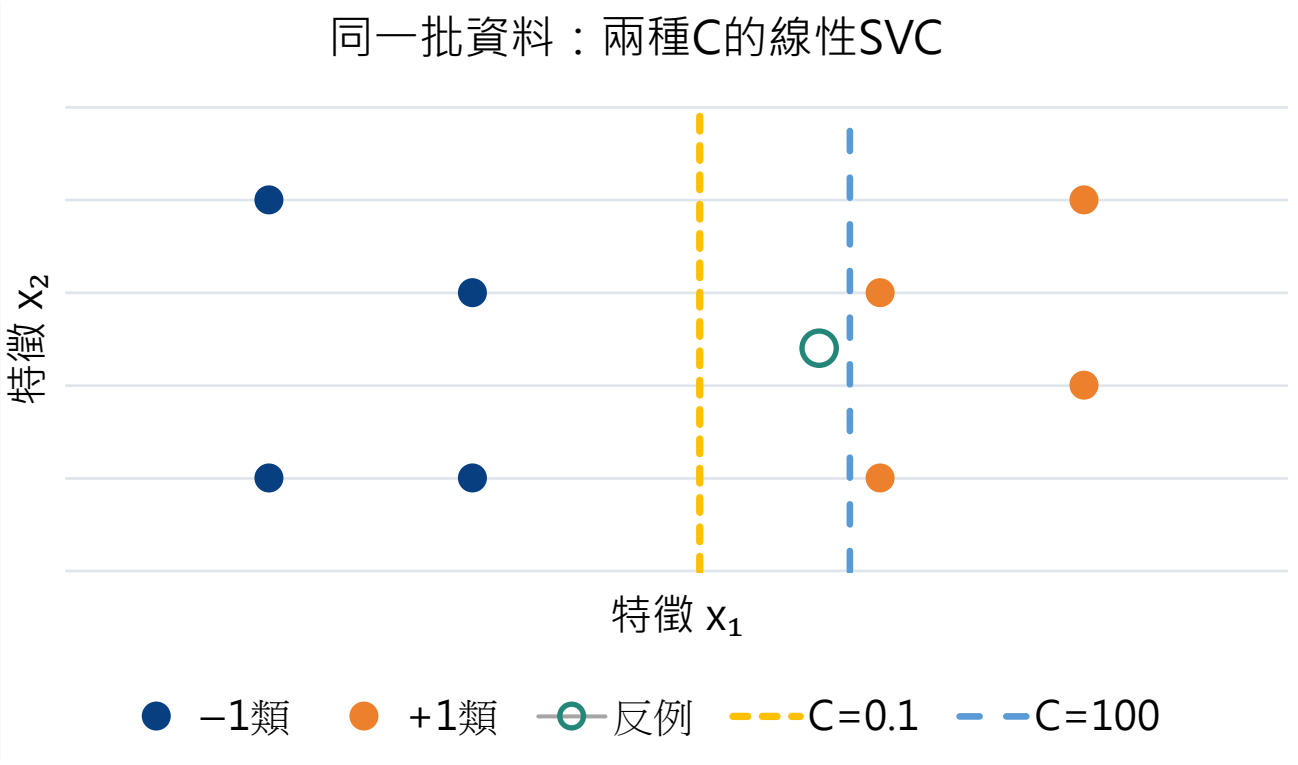
C 太大易被例外牽著走；彎路是非線性核下的情形。C 要用驗證集挑。

數學：C 在公式裡做什麼

$$\min \tfrac{1}{2}||w||^2 + C \cdot \Sigma\, \xi_i$$

公式拆解：路寬 vs 越界罰款

符號	意思	備註
$\frac{1}{2}\|\|w\|\|^2$	路要寬	和 L2 同一招
ξ_i	第 i 點越界多少	在路外 = 0
C	越界罰多重（超參數）	大→路窄； 小→路寬

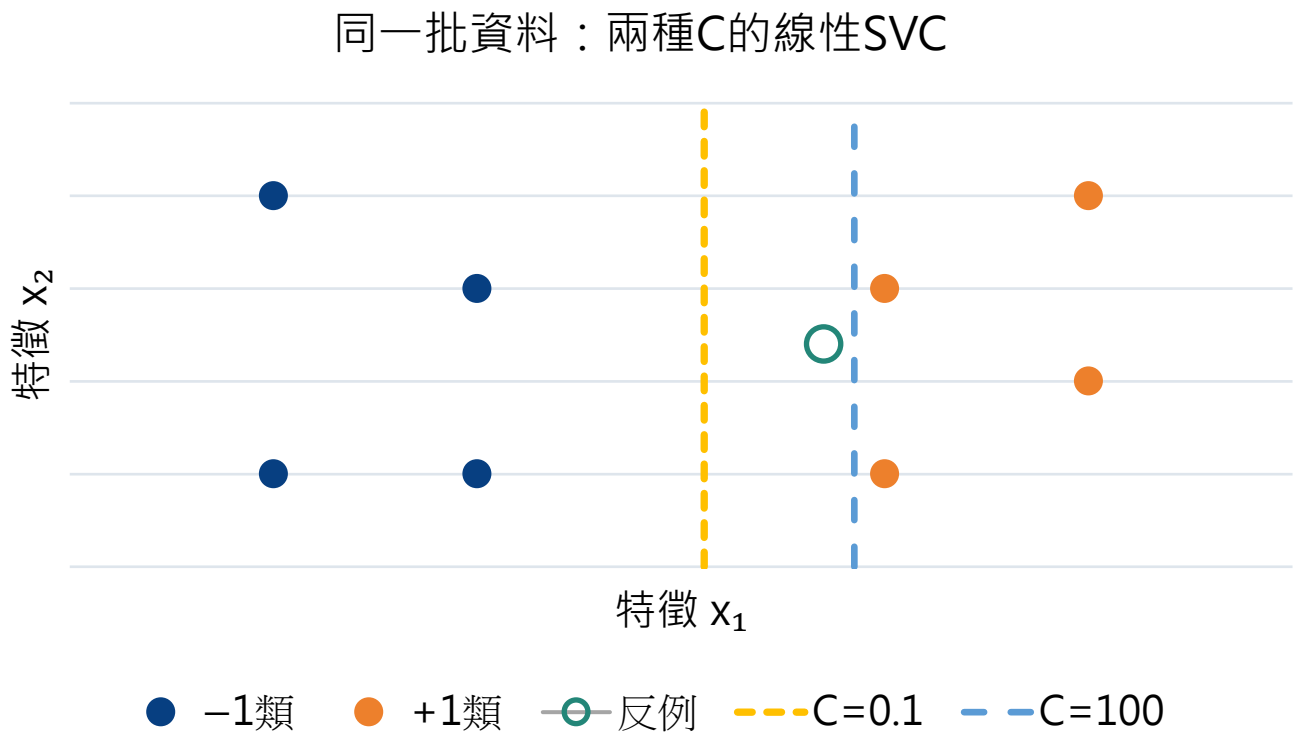


前半要路寬，後半罰越界；C 決定兩邊聽誰的：C 大就為了少越界把路縮窄。

想一想⑤　提高C一定更好嗎

思考題

把SVC的C調大後，訓練正確率上升，代表什麼？



A　未來正確率一定上升

B　對訓練違規更重視，仍需驗證

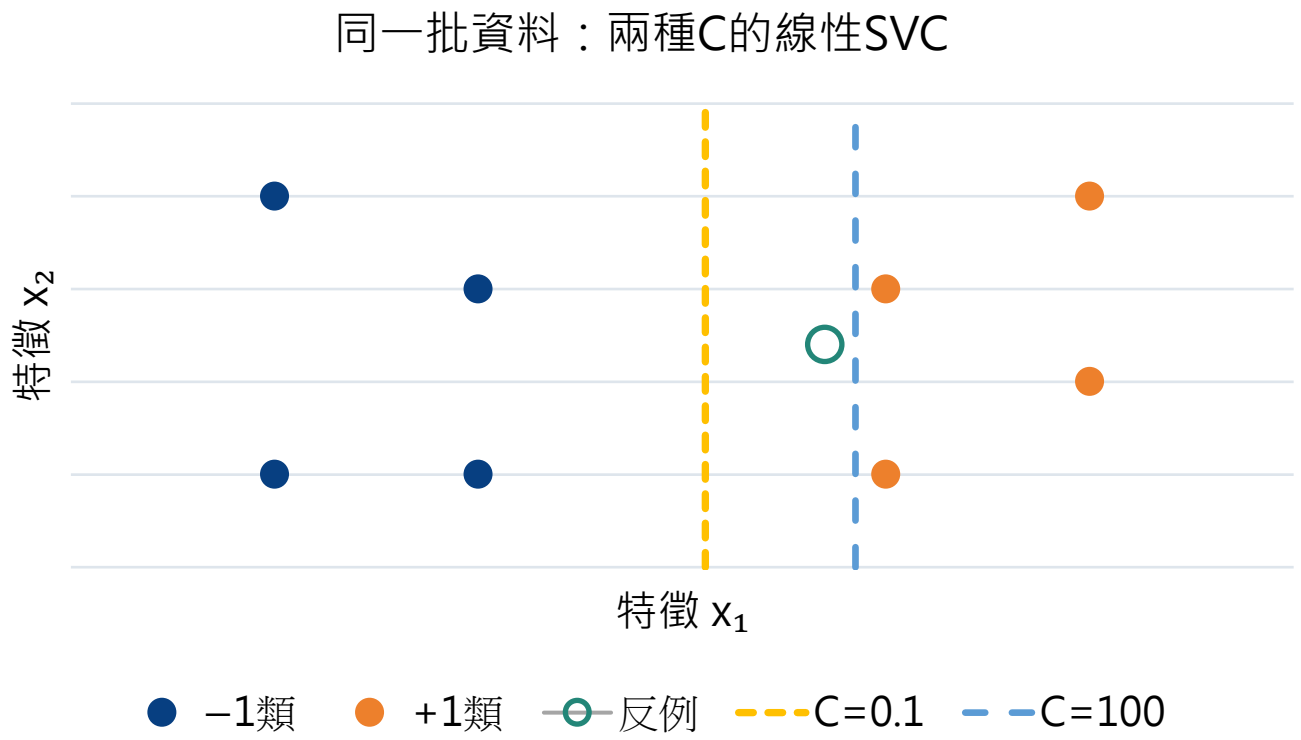
C　學習率變大了

請先選擇，再用圖中的證據解釋。

想一想⑤　提高C一定更好嗎：解答

思考題

把SVC的C調大後，訓練正確率上升，代表什麼？



A　未來正確率一定上升

B　對訓練違規更重視，仍需驗證

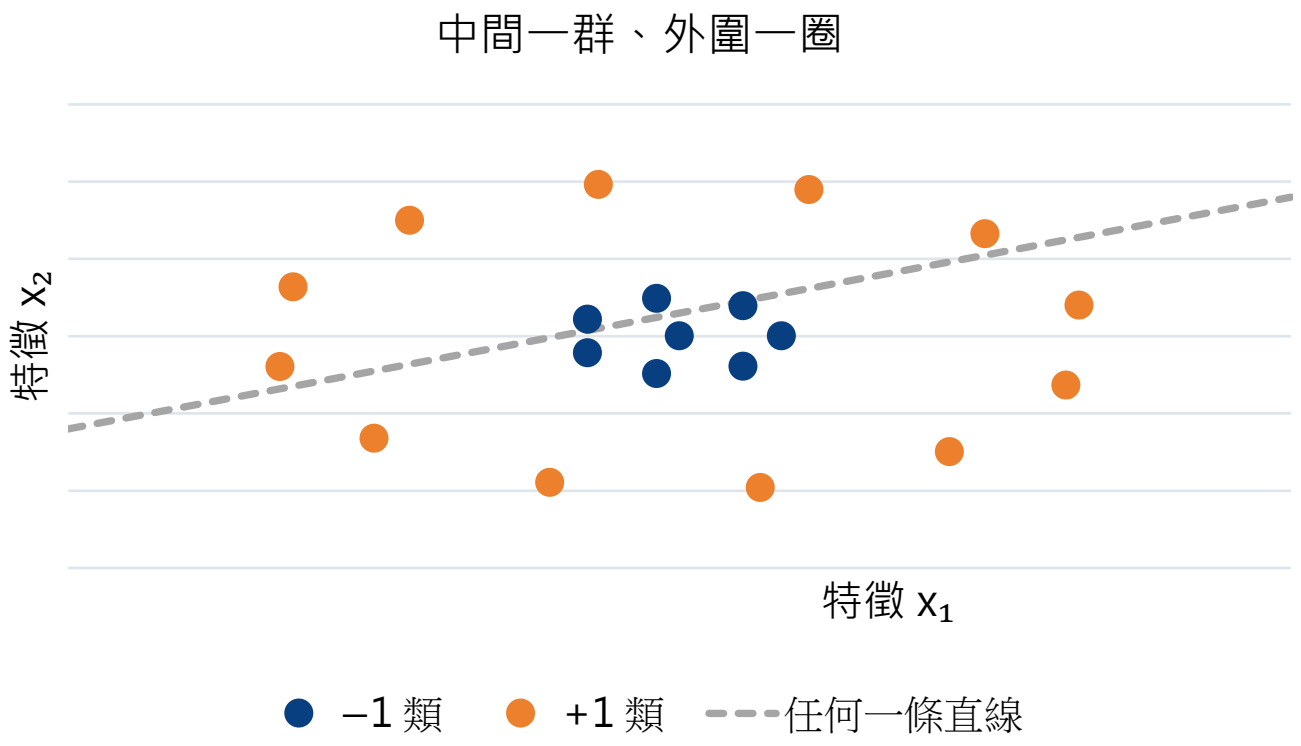
C　學習率變大了

答案：B　對訓練違規更重視，仍需驗證

C改變最佳化取捨。訓練改善只提供一部分證據，還要看相同驗證資料上的效果。

例子③：中間一群、外圍一圈，直線分不開

- 中間一群藍、外圍一圈橘。
- 不論直線怎麼畫，都會切到另一群。
- 要換個看法：用「離中心多遠」來分。



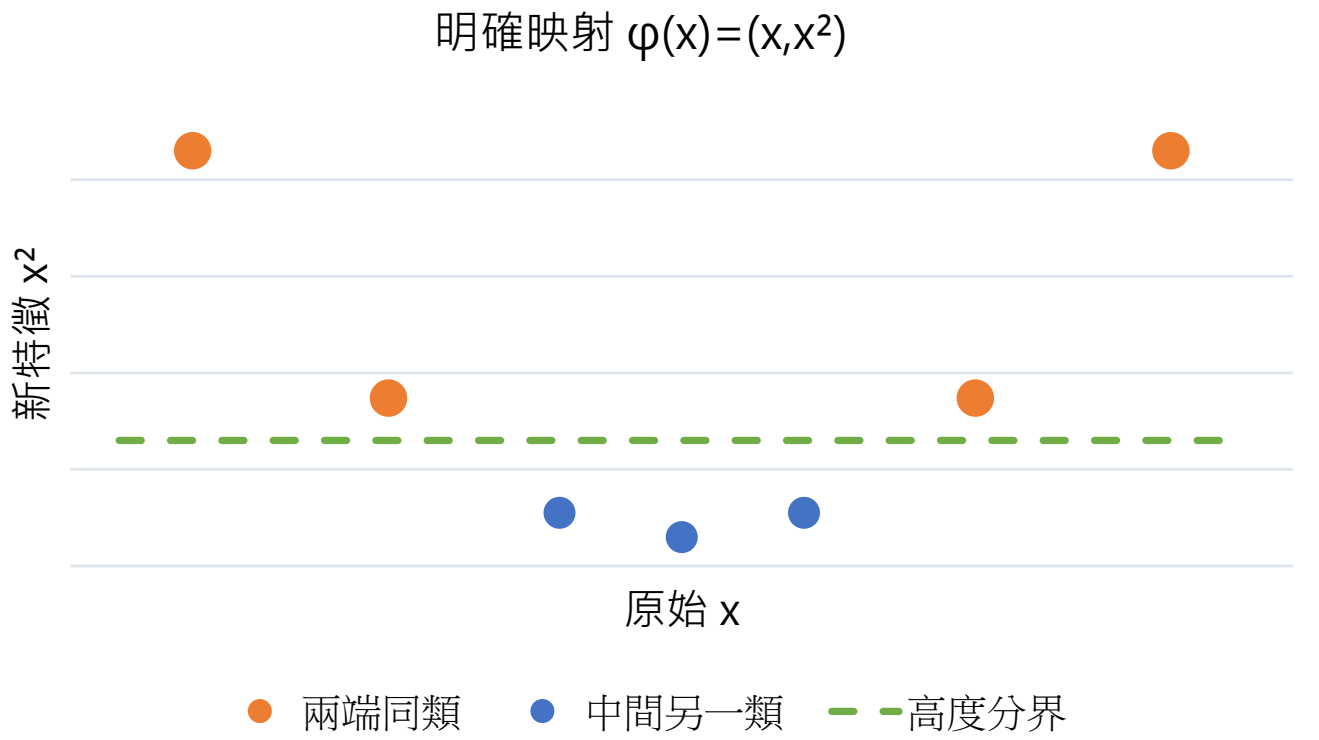
先請學生試著畫一條線分開：做不到。這時需要換一種特徵表示，也就是核方法。

核方法：加一個新特徵，直線就分得開

公式參考．看懂即可

$\varphi(x) = (x, x^2)$

- 原本只有 x：中間一類、兩端一類，一刀切不開。
- 加一個新特徵 x^2：兩端被抬高、中間留在低處。
- 在新座標 (x, x^2) 上，一條水平線就分開。



圖將一維樣本映到(x,x^2)，在高度處即可分開兩類。

直覺圖解：核方法 = 把桌布中央頂起來

生活類比

生活中的例子

藍色彈珠在桌子中央，橘色彈珠圍成一圈，桌面上畫不出一條直線分開。把桌布中央往上頂，藍色升高、橘色留在低處，一片水平玻璃就能分開。

對應到模型

桌面	**原本的特徵空間**
頂起的高度	**新特徵，如 $e^{-(x_1^2+x_2^2)}$**
水平玻璃	**新空間的超平面**
從上往下看	**原空間的圓形邊界**



高維空間中的直邊界，回到原空間看就是彎曲的；核函數讓 SVM 不必真的算出新座標。

直覺圖解：RBF 核 = 每個關鍵樣本是一盞燈

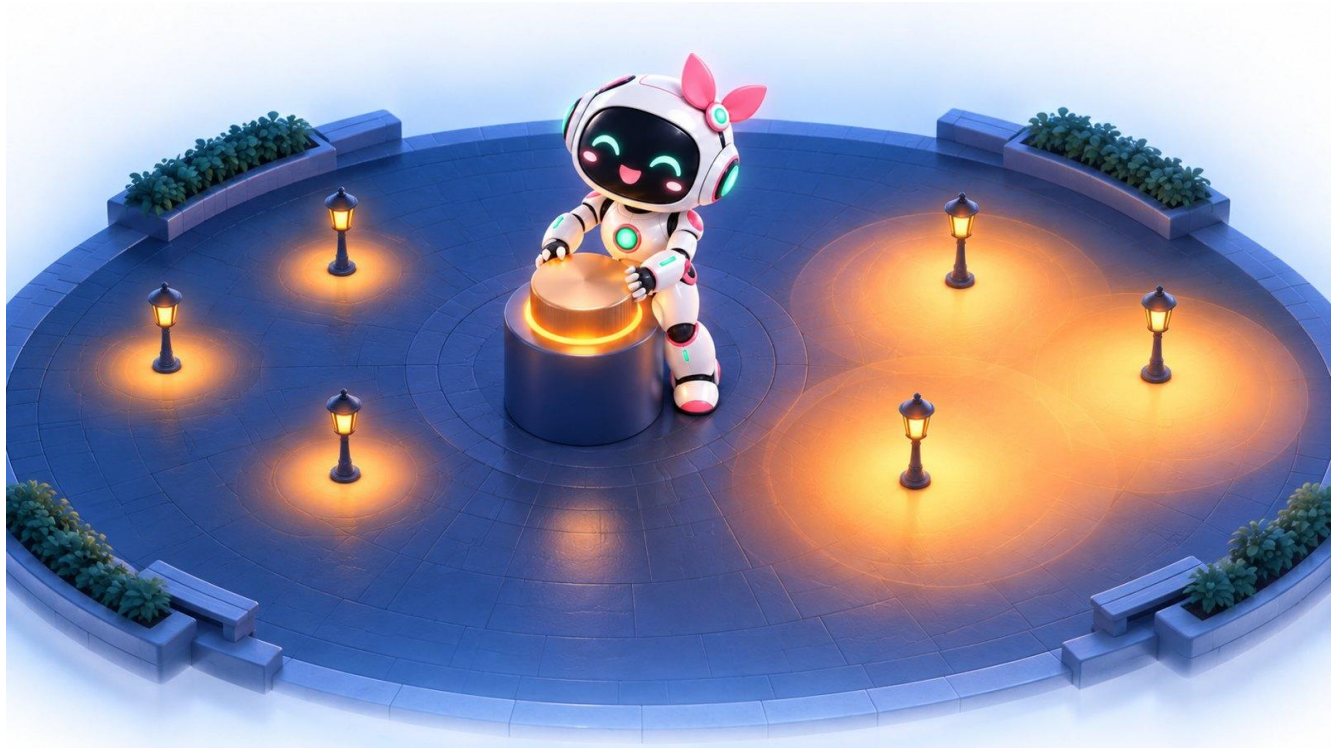
生活類比

生活中的例子

每個支持向量像一盞路燈，照亮附近區域。γ 大時光圈很小，只照到身邊；γ 小時光圈很大，彼此重疊成平滑的一片。

對應到模型

路燈	**支持向量**
光圈大小	**影響範圍，約 $1/(\gamma)^{0.5}$**
被照到的亮度	**相似度 K(x,z)**
光圈太小	**邊界破碎、易過擬合**



γ 決定「多近才算相似」；它和 C 一樣要用驗證集挑選。

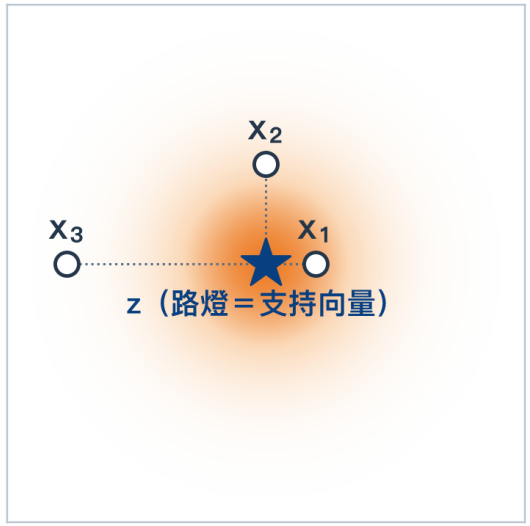
數學：RBF 核 = 用距離算相似度

$$K(x, z) = \exp(-\gamma \cdot ||x - z||^2)$$

公式拆解：x、z 各是什麼

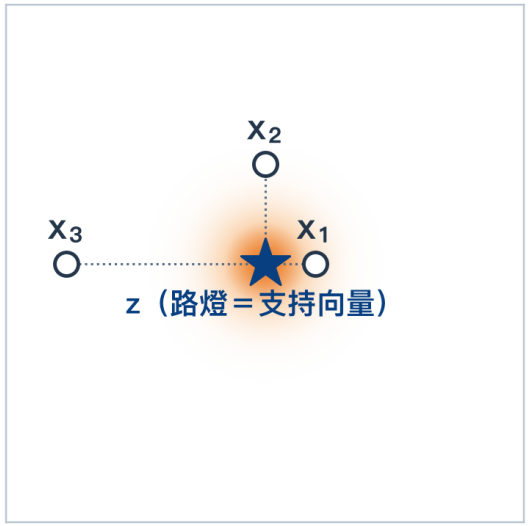
符號	意思
x	要判斷的新點（新乘客）
z	一個支持向量（路燈）
$\|\|x - z\|\|$	兩點的距離 d
γ	光圈大小（超參數）
K(x, z)	有多像：0～1

γ = 1：光圈大



x_1：距離 d = 0.5 → K = 0.78
x_2：距離 d = 1 → K = 0.37
x_3：距離 d = 2 → K = 0.02

γ = 4：光圈小



x_1：距離 d = 0.5 → K = 0.37
x_2：距離 d = 1 → K = 0.02
x_3：距離 d = 2 → K = 0.00

新點 x 離路燈 z 越近越亮（K 越接近 1）；γ 越大光圈越小，只有很近的點才算像。

SVM 小結：三個例子、三個工具

段落小結

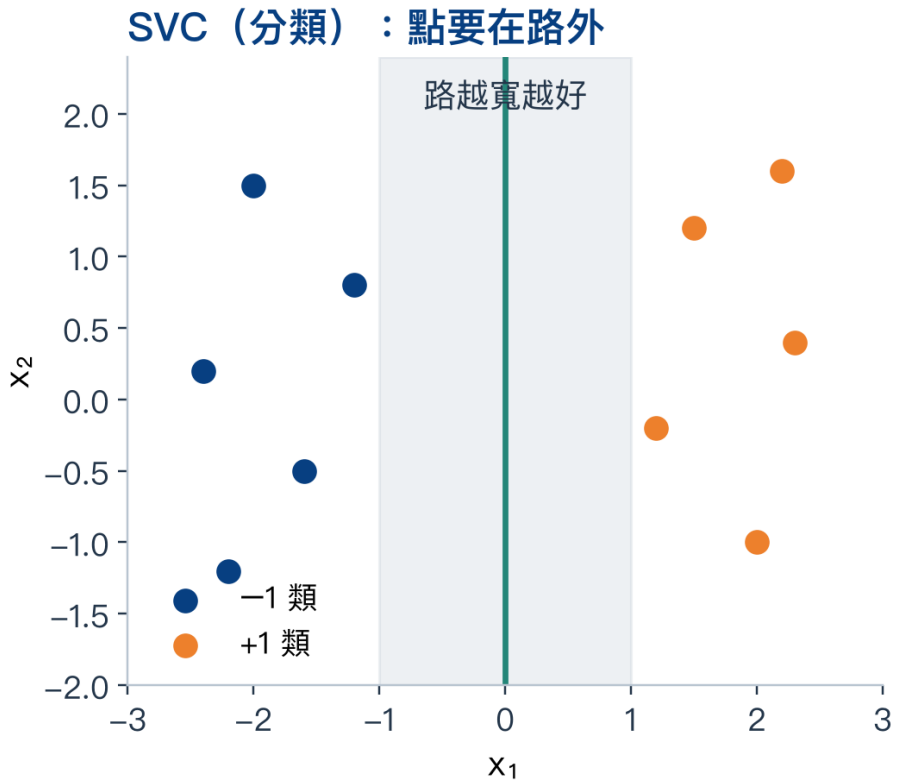
- 三個困難各對應一個工具：間隔、C、核函數。
- C 和 γ 都是超參數，要用驗證集挑
- Titanic：SVC 驗證 0.831，和 Logistic 0.815、神經網路約 0.84 差不多。

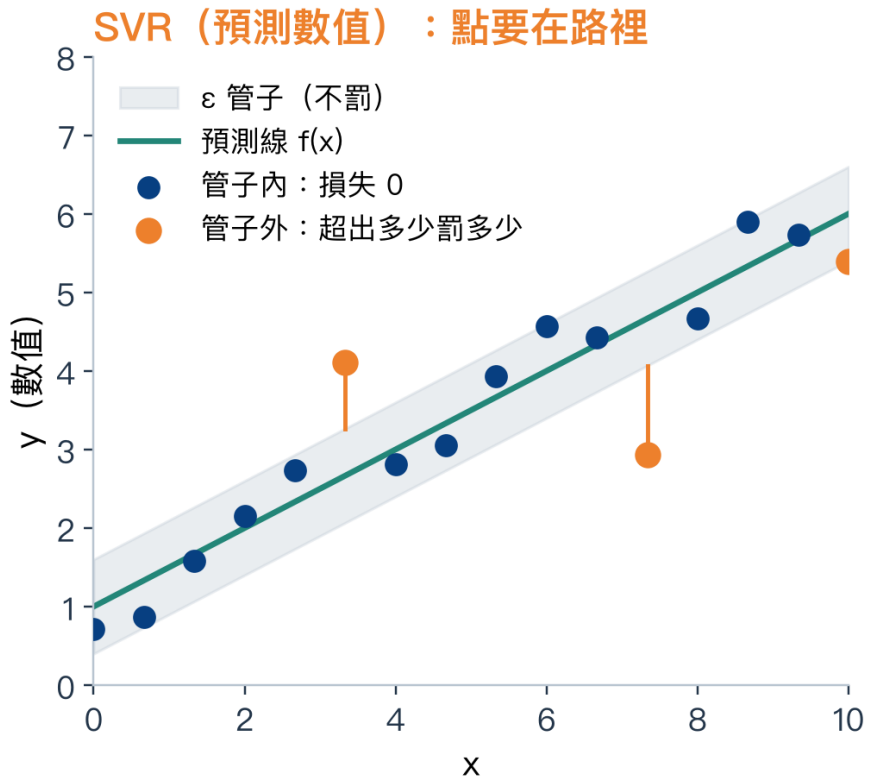
例子	遇到的困難	SVM 的工具
① 兩條線都分對	選哪條？	間隔最大
② 有點放錯邊	硬分全對路太窄	C：允許越界
③ 中間一群外圍一圈	直線分不開	核函數 RBF (γ)

SVM 用「最寬的路」分類；下一頁把路反過來用，就能預測數值。

SVR：把「路」反過來用，預測數值

$$\ell = \max(0, |y - f(x)| - \varepsilon)$$





SVC：點要在路外；SVR：點要在管子裡，只罰管子外的誤差。下一段預測一週後的乖離率，就是數值預測。

第八段　股票預測：看一週後的趨勢

上一段

SVR 讓 SVM 也能預測數值；換一個題目：台積電未來一週的趨勢。

這一段要回答

今天收盤後，能預測「7 天後的 15 日均線」會在今天收盤價的上面還是下面嗎？



1. 定義答案、對齊時間、只用過去訓練
2. 基準 vs 模型：特徵分組 A ~ D
3. 換你試：改天數、加入自己的特徵

明天的日曆還封著——任何偷看都是資料洩漏。

股票預測：先定義答案是什麼

核心理論

$$y_t = (Close_t / MA15_{t+7} - 1) \times 100$$

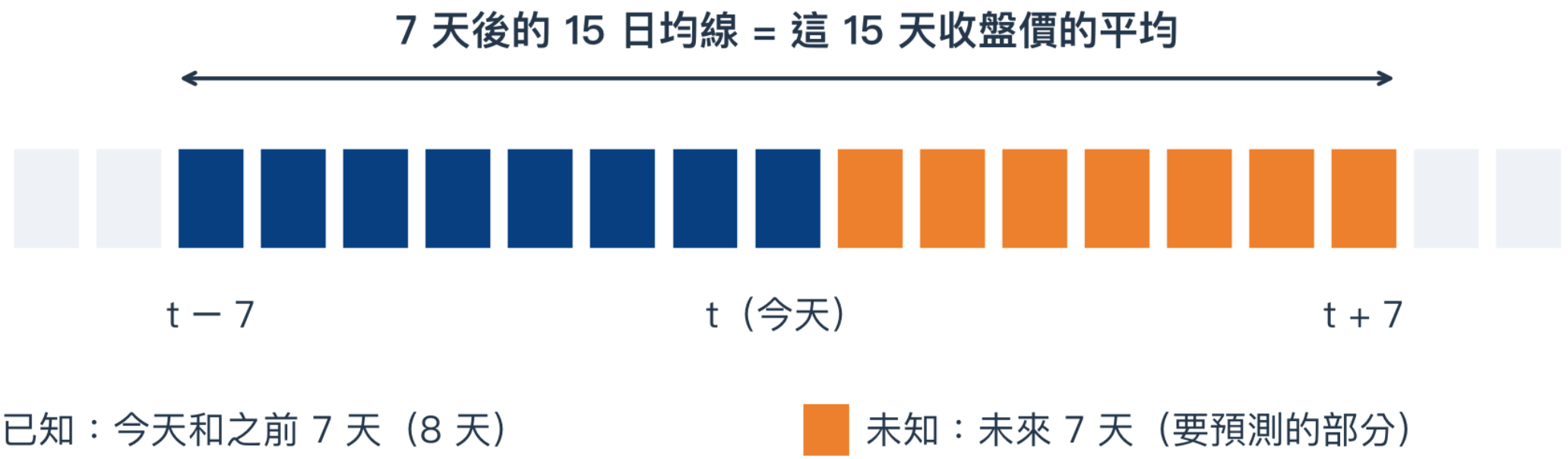
- 答案：今天收盤價，相對「7 天後的 15 日均線」高或低幾 %。
- $y < 0$：一週後的平均價比今天高（趨勢向上）。
- 今天收盤後只能用今天以前的資料來預測。

時點	可用的資料	答案
今天收盤後	今天以前的價量、均線	還不知道
7 天後收盤後	多了未來 7 天價格	可算出 7 天後的 MA15
建訓練資料時	保留當天的輸入	把 7 天後的答案對齊回來

不是猜明天漲跌，而是猜未來一週的平均價會落在今天收盤價的上面還是下面。

為什麼看「7 天後的 15 日均線」？

圖解



15 天裡有 8 天今天已經知道，只需猜未來 7 天；均線也把每天的雜訊平均掉，所以比猜明天穩定。

乖離率：將價格改成相對位置

BIAS15 = (Close / MA15 – 1) × 100

- 乖離率：收盤價離均線多遠，用 % 表示。
- 用比例而不是價差，100 元和 1000 元的股票才能比。
- 今天的 15 日乖離率，就是等一下的「基準」。

例子	Close	MA15	BIAS15
A	105	100	+5%
B	1050	1000	+5%
C	95	100	–5%

105/100–1=5%；1050/1000–1也等於5%。

時間對齊：shift(–7) 把一週後的答案拉回今天

target = (Close / MA15.shift(–7) – 1) × 100

- shift(–7) 把 7 天後的 MA15 往上搬到今天這一列。
- 今天的特徵和一週後的答案放在同一列，才能訓練。
- 最後 7 天還不知道答案，不能拿來訓練。

日期	Close	7 天後的 MA15	target
第 1 天	100	102	–1.96%
第 2 天	103	101	+1.98%
最後 7 天	…	還不知道	未知

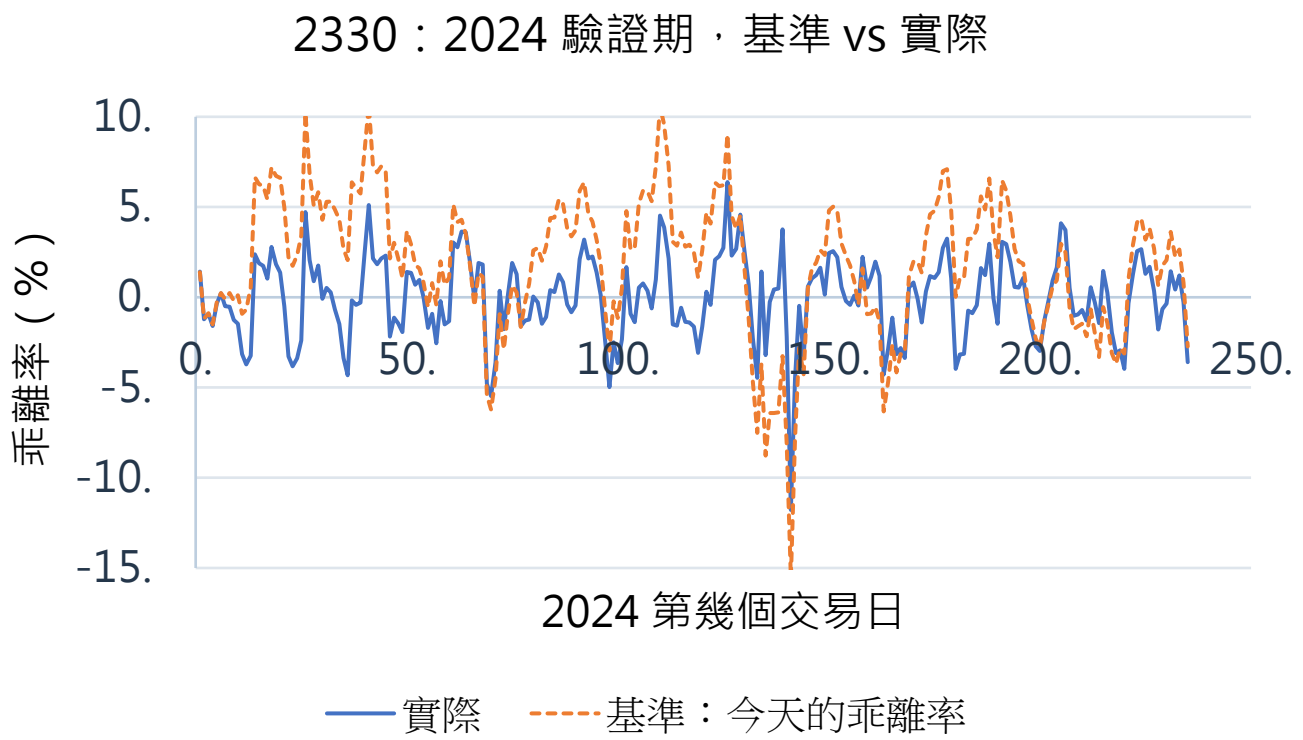
shift(–7) 只用來建立答案；把未來的價格放進今天的特徵，就是偷看。

基準：直接用今天的乖離率當預測

核心理論

$$\hat{y}_t = BIAS15_t$$

- 假設一週後的均線和今天一樣。
- 橘線總是慢半拍：趨勢轉彎時就猜錯。
- 模型要贏過這條線，才算真的學到東西。



真實 2330 驗證資料：基準 MAE 2.77 個百分點，方向正確率 68%。

特徵分組：A、B、C、D 各放了什麼

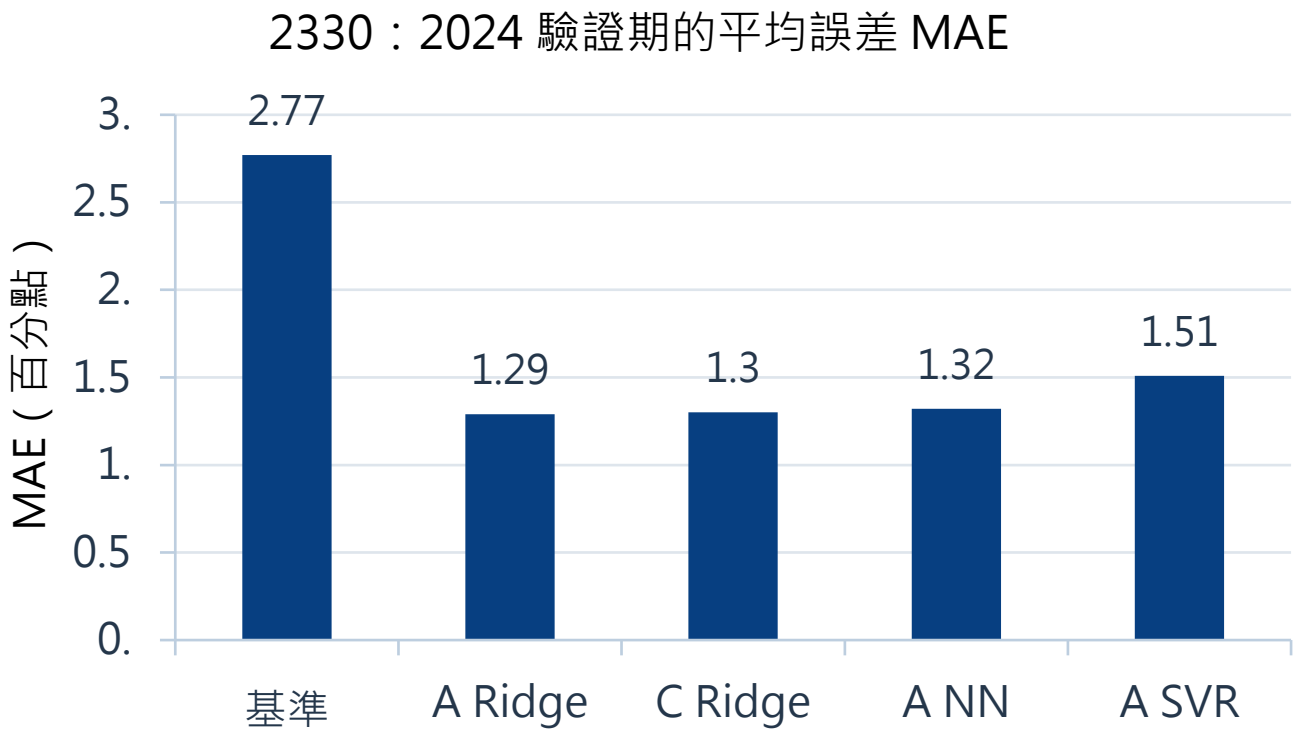
- 每組都含今天的 15 日乖離率，再逐組加特徵。
- 一次只加一組，才知道是哪些特徵有用。
- 「A 組 Ridge」= 用 A 組特徵、Ridge 模型。

組別	放了哪些特徵（都含今天的 15 日乖離率）
A	乖離率、過去一日報酬
B	A + 開高低收比率、成交量
C	B + 5 / 10 / 20 / 60 日均線比率
D	C + KD、RSI 技術指標
E（自己做）	B4-4 自己寫 my_features

後面寫的「A 組 Ridge」「C 組 Ridge」，就是特徵組別 + 模型名稱。

實際結果：模型誤差約為基準的一半

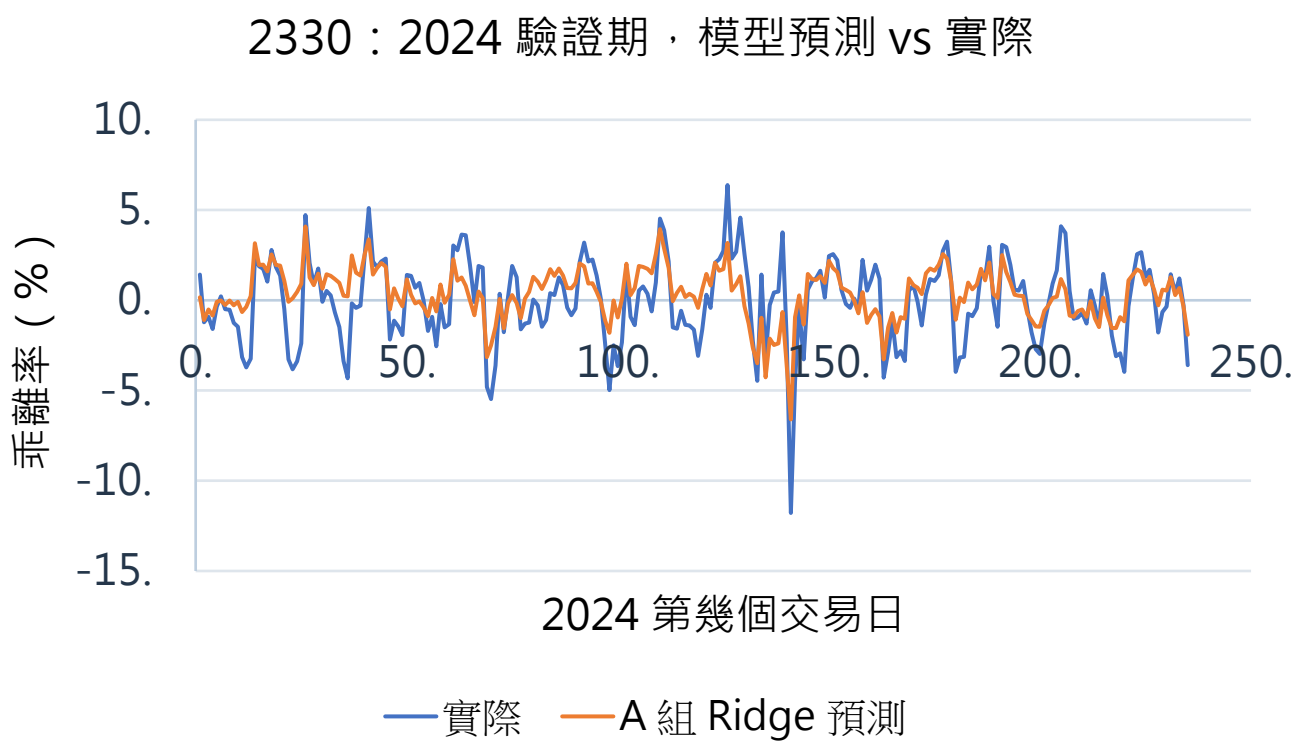
- 基準 2.77 → A 組 Ridge 1.29 個百分點。
- 加更多特徵（C 組）、換 NN 或 SVR，都沒有更好。
- 最後測試 2025：基準 2.64 → 1.21，方向正確率 70%。



簡單的 A 組 Ridge 就贏過基準；更複雜的模型不一定更準。

預測曲線 vs 實際：趨勢跟得上

- 灰線：實際答案；藍線：A 組 Ridge 的預測
- 上下轉折大多跟得上，只是幅度比較保守
- 和前面基準的橘線相比，不再慢半拍



預測與實際的相關係數 0.73；是否還有優化空間

換你試①：改均線天數與預測天數

- 在 Colab B2 改 MA_DAYS 與 HORIZON，重跑一次。
- 已知天數越多（均線長、預測近）越容易準。
- 5 日均線 3 天後：幾乎猜不準（方向 56%）。

均線天數	預測幾天後	基準 MAE	模型 MAE	方向正確率
15 日	7 天	2.77	1.29	69%
10 日	3 天	1.43	0.69	85%
10 日	5 天	2.33	1.22	66%
20 日	5 天	1.77	0.84	88%
20 日	10 天	3.43	1.62	68%
5 日	3 天	1.92	1.17	56%
30 日	15 天	4.69	1.91	71%

準不準和「有多少天已經知道」有關；要找的是有用、又不是太容易的題目。

換你試②：加入你自己的特徵

- 特徵可以來自股價以外：外資、比特幣、天氣、新聞。
- 只能用「今天收盤前已知」的資訊；收盤後才公布的要往後移一天。
- Colab B4-4：自己寫特徵，自動檢查有沒有偷看未來。

點子	可能的來源	注意
KD、RSI、MACD	股價自己算	D 組已有 KD、RSI
外資買賣超	TWSE 三大法人	收盤後公布 → 移一天
比特幣、美股、費半	國際行情	時差：用前一晚收盤
天氣、新聞、川普發文	氣象署、新聞、社群	先量化成數字

先猜、再驗證：新特徵要在同一批樣本上贏過原本的組別，才算有用。

股票實作：把理論變成一次比較

實作連結

- B1–B3 填空 S1–S9：清洗、MA15、答案、基準、切分。
- B4 控制台：Ridge / SVR / NN、特徵組 A–D。
- B4-4 自己做特徵；B7 最後測試與交接。

步驟	做什麼	課堂
B1–B3	填空 S1–S9：清洗、MA15、答案	老師示範
B2	設定題目、看基準線	老師示範
B4	控制台與挑戰任務	回家
B4-4	自己做特徵，衝獎牌	回家（選做）

多模型調參可選做，先完成公平比較與解釋。

實作④　股票 B1–B3：老師示範，B4 回家做

▶ 老師示範・Colab

實作時間

建議 6 分鐘　Colab 步驟

- ☐ B1–B3　填空 S1–S9：清洗到基準
- ☐ B2　設定題目（15 日、7 天），看基準線
- ☐ B4 / B4-4　回家做挑戰

完成標準

能說出：誤差比和方向正確率各是多少？有沒有贏過基準？



建議 6 分鐘；卡住先舉手，不要跳過切分直接看測試集。

時間視窗：一列特徵變成一段歷史

Class2：X=(N,d)　Class3：X=(N,T,d)

- Class2每個樣本是一日的特徵向量。
- Class3每個樣本包含過去T天，保留前後順序。
- 目標與切分規則先維持一致，才看出序列模型的作用。



窗口只往過去取資料；答案日（橘色）不能出現在輸入。

本堂課的四個核心理解

課程收束

①複習 ②機率 ③損失·NN ④資料 ⑤泛化 ⑥評估 ⑦SVM **⑧股票**

- 模型把特徵轉成分數、機率或數值，選擇取決於任務。
- 損失函數與最佳化方法決定模型如何學。
- 泛化需要隔離資料、公平比較與謹慎解讀機率。
- 特徵更多、模型更大、勝率更高，都需要額外證據。

主題	閱讀資源
分類與泛化	Google ML Crash Course
間隔與核方法	Stanford CS229 §6
模型與資料流程	scikit-learn User Guide
時序視窗與LSTM	TensorFlow時間序列教學

完成一次比較後，能說明「改了什麼、為什麼、證據是什麼」。