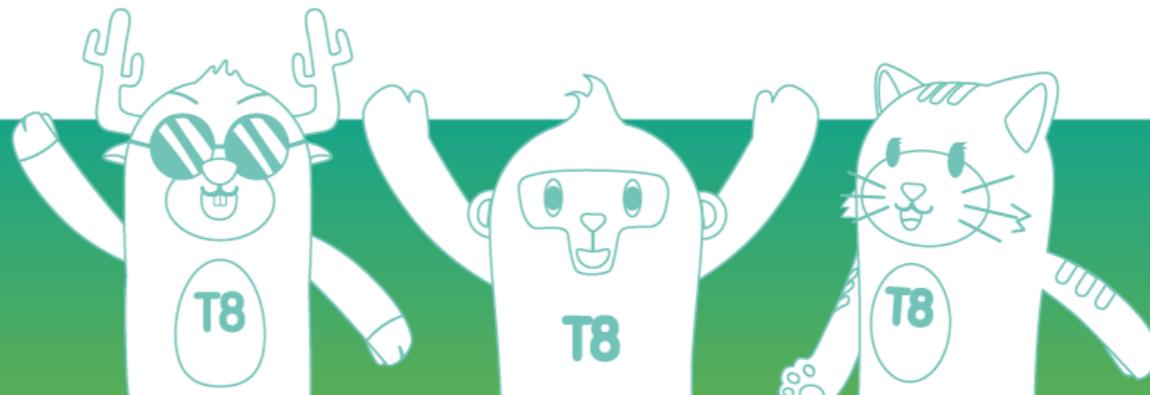


機器學習精要 Machine Learning

授課講師 陳少君 Paul
教材編寫 陳少君 Paul



緯育 *TibaMe*

即學 · 即戰 · 即就業
<https://www.tibame.com/>

課程規劃 與範疇

課程規劃 及大綱

學習目標：

- 了解本課程的目的
- 了解本課程的 12 個模組

課程目的

- 經由講解和範例，以及使用 Python 語言的實作，以由淺入深的方式讓大家逐步了解**機器學習的原理、步驟以及應用**。

學習步驟

- 課前對機器學習課綱相關知識的預習，課中專注聆聽講解、釋疑與實作，加上課後的復習與評量。同時也對Python套件和幾何概率微分隨時溫習上手，達到最佳學習效果。

課程內容

- 我們從機器學習入門最基本的原理及相關知識開始講解，逐漸帶入迴歸預測，資料分類以及分群的原理與技巧，佐以各單元完整的實作和評量。

學習環境一

- 使用 Google Colab，需有 Google Account 與 Google Drive。

學習環境二

- 也可以下載並安裝 Anaconda，包括 Python 3 和 Jupyter，在自己環境內編譯執行。
- 需考量自身電腦是否有足夠 GPU / TPU 運算能力。

學習環境三

- 有些課程範例會放在 Kaggle.com，Github.com 及其他學習網站上面，若想在 網站上直接體驗，建議建立相關帳號。

<u>• 課程規劃與範疇</u>		
<u>• 學習工具與準備</u>		
1.機器學習基礎介紹	5. 支持向量機SVM	9.經典安隆案預測模型
2.迴歸分析	6.樸素貝葉斯 Naïve Bayes	10. 階層式分群
3.決策樹模型	7. 類神經網路	11. 分裂式分群
4.羅吉斯迴歸模型	8. 集成學習	12. 密度式分群

Module 1. 機器學習 基礎介紹

機器學習導論

學習目標：

- 認識機器學習的四大分類

甚麼是機器學習

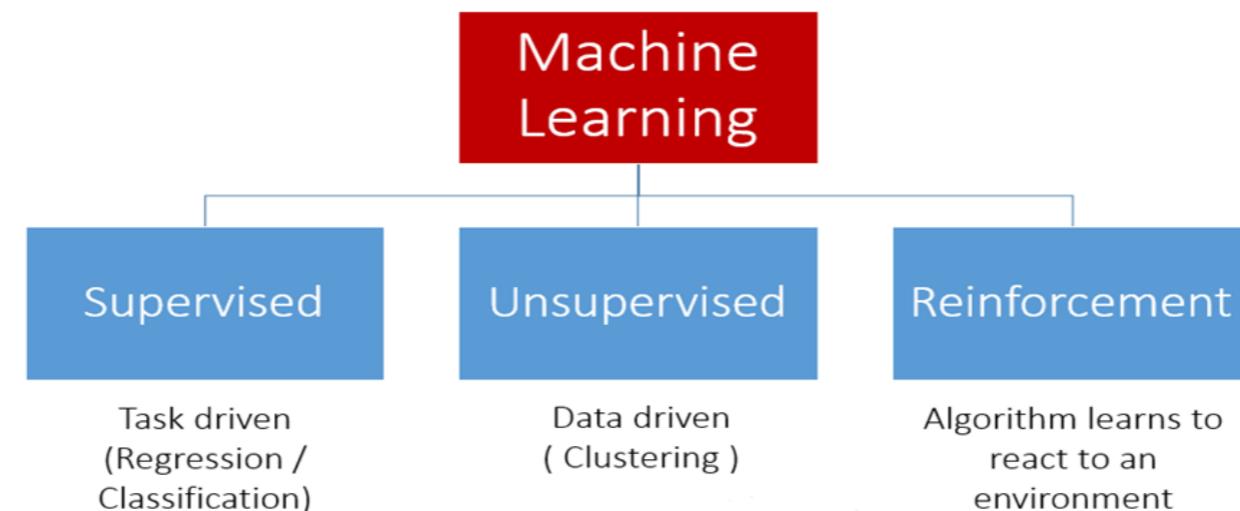
簡單地說，所謂機器學習就是機器根據資料，不靠程式師的程式設計，自行發展出演算法。

資料可能是標籤過的(監督式學習)，部分標籤過的(半監督式學習)或者未標籤(非監督式學習)的。

學習過程需要依據離目標的距離被懲罰或獎勵，可能是靜態或動態(強化式學習)。

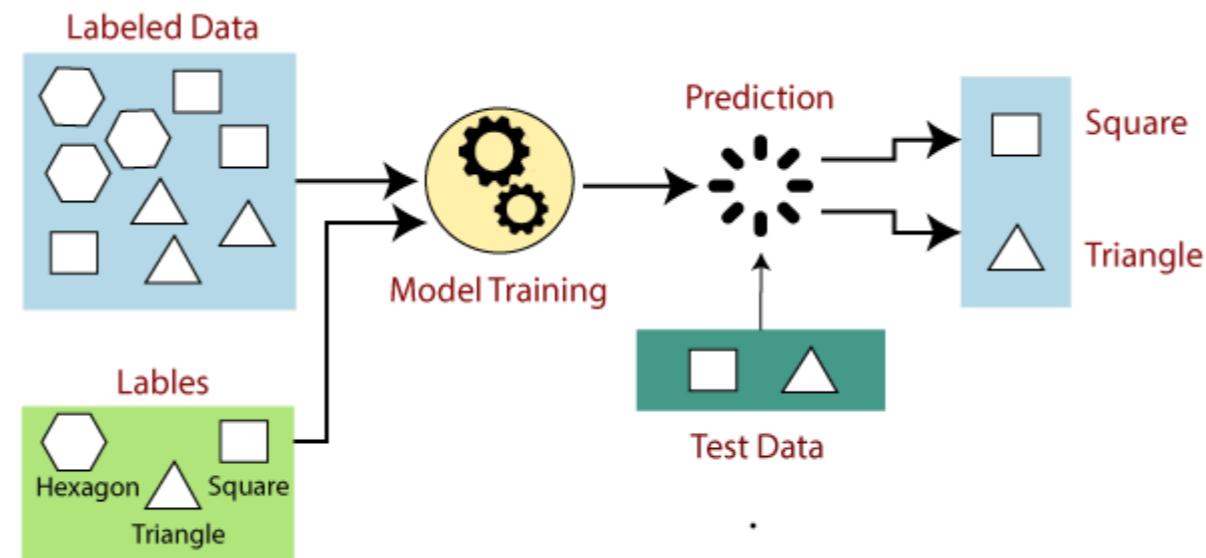
一般來說，我們可以根據資料是否註記(標籤)將學習分成：監督式學習和非監督式學習；而學習環境為動態而非靜態者稱為強化學習。

Types of Machine Learning



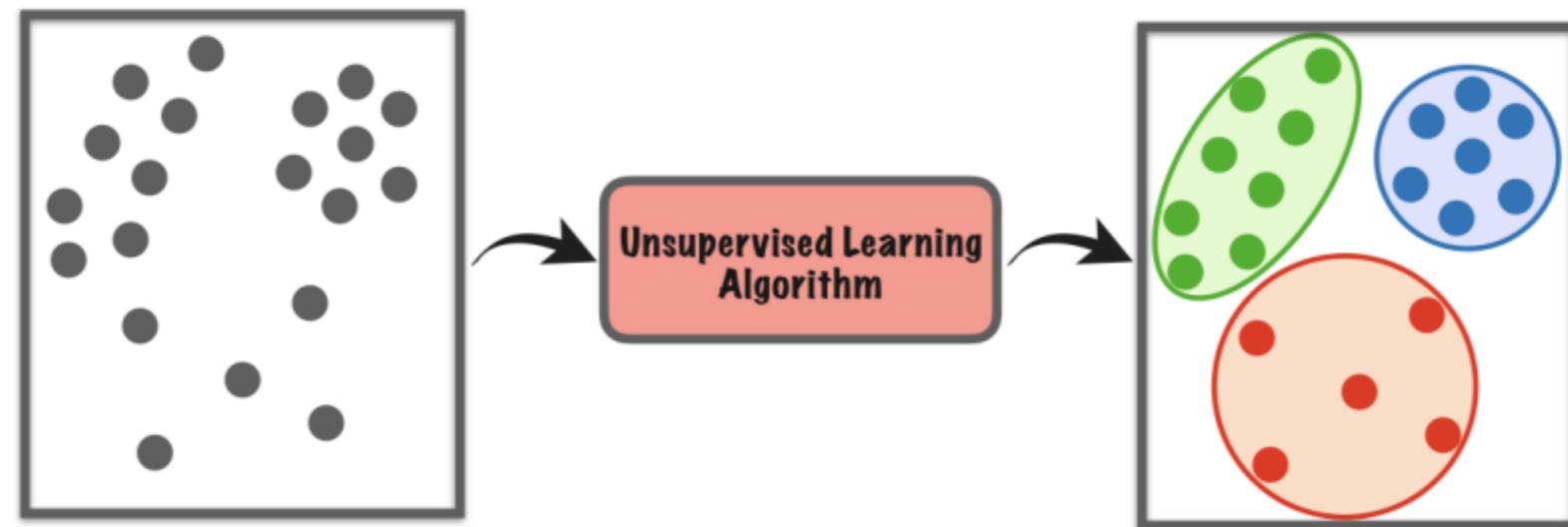
<https://mahiprashanth866.medium.com>

監督式學習表示給予機器資料(Data)以及每筆資料所對應的標籤(Label)這些標籤就用來訓練電腦辨別東西的答案。主要應用在分類(Classification)與迴歸(Regression)

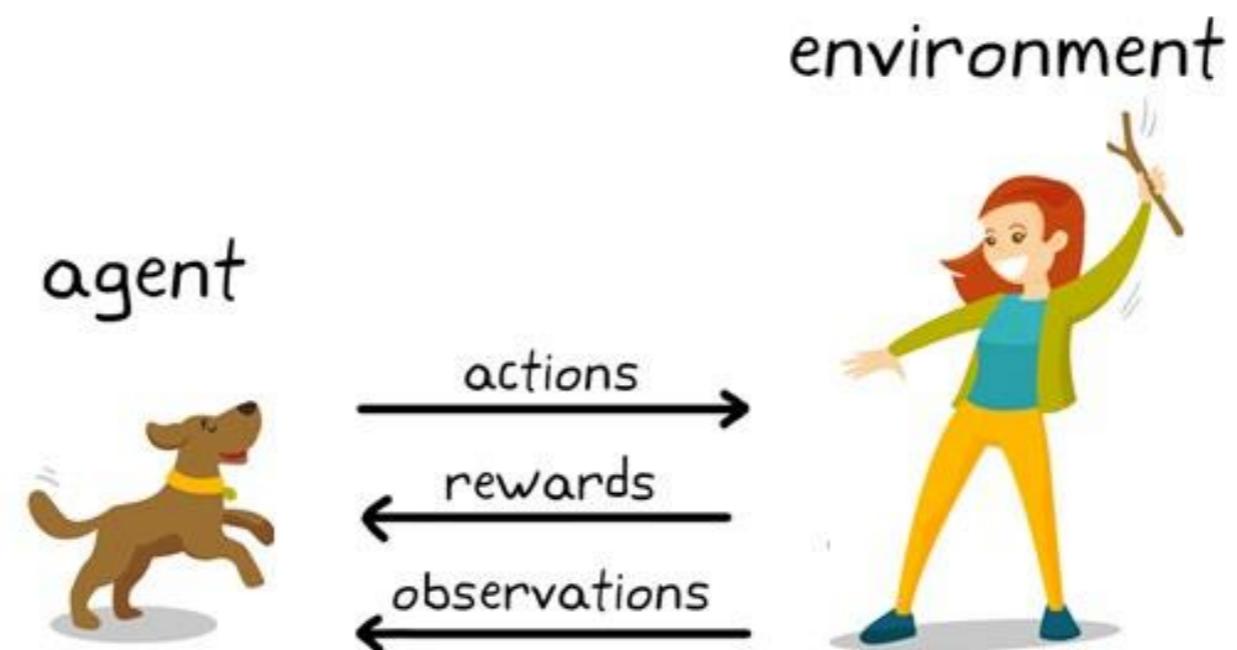


非監督式學習提供機器資料(Data)但沒有對應的標籤(Label)

分群(類聚)就是一種常見的非監督式學習，自動把相似度高的資料放在一起。



強化學習是經由智能體與環境的互動、給予不同數量的賞酬，以動態方式修正智能體的一連串策略的學習行為，直至任務完成。圍棋和自駕車是典型的例子



Pic from: KD Nuggets.com

本課程重點

監督式學習：

- 迴歸分析：
線性迴歸
羅吉斯迴歸
- 分類
樸素羅吉斯
決策樹
支持向量機器(SVM)

非監督式學習：

- 分群(類聚)
階層式分群
K-means 分群
DBSCAN 分群

Module 1. 機器學習 基礎介紹

建模流程
與框架

學習目標
Learning Objectives

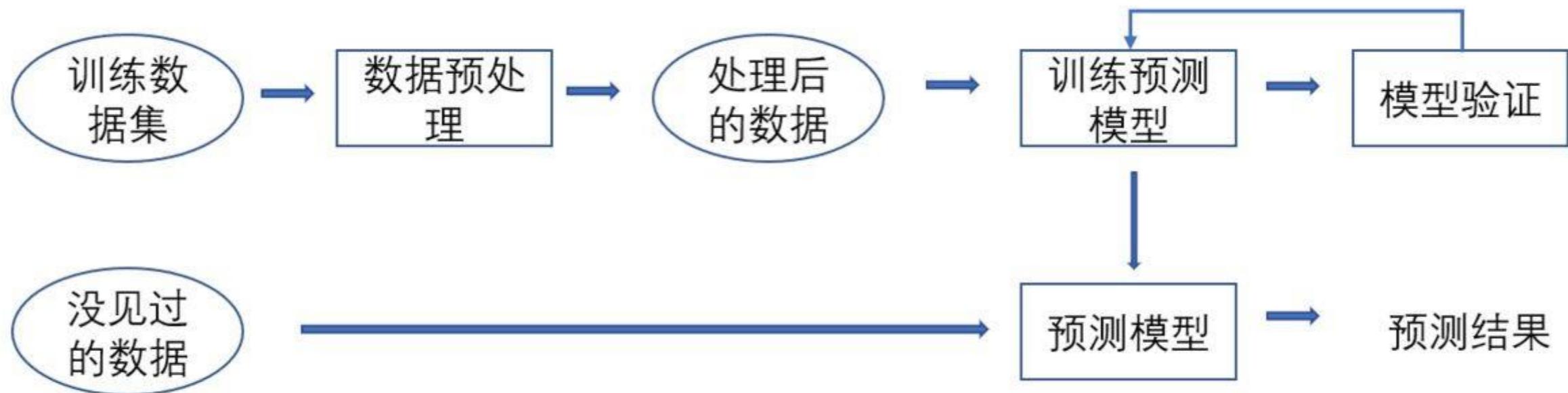
- 認識模型建立流程

機器學習模型訓練流程

蒐集

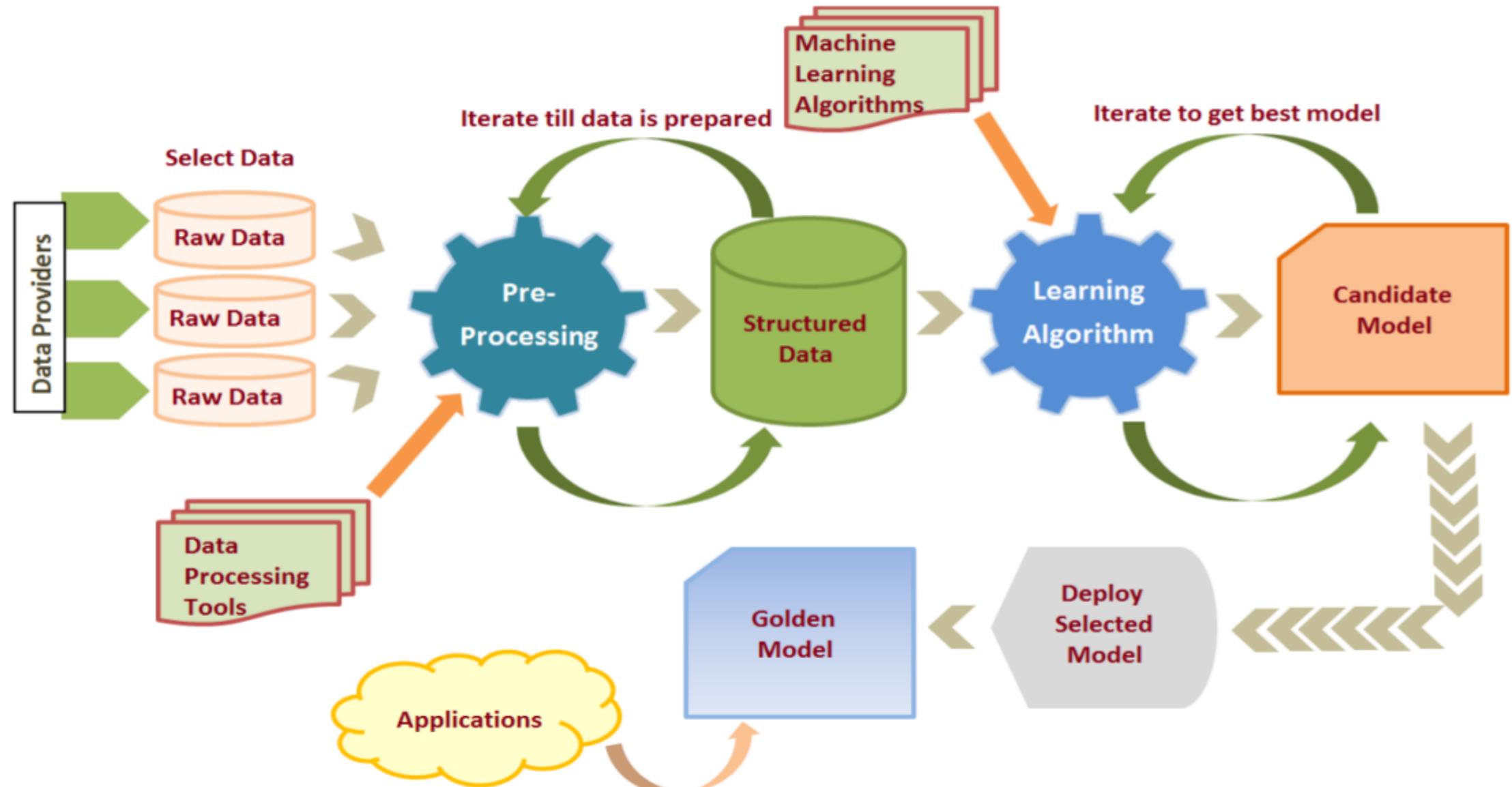
清洗(預處理)

特徵工程 建模



知乎 @赜赜

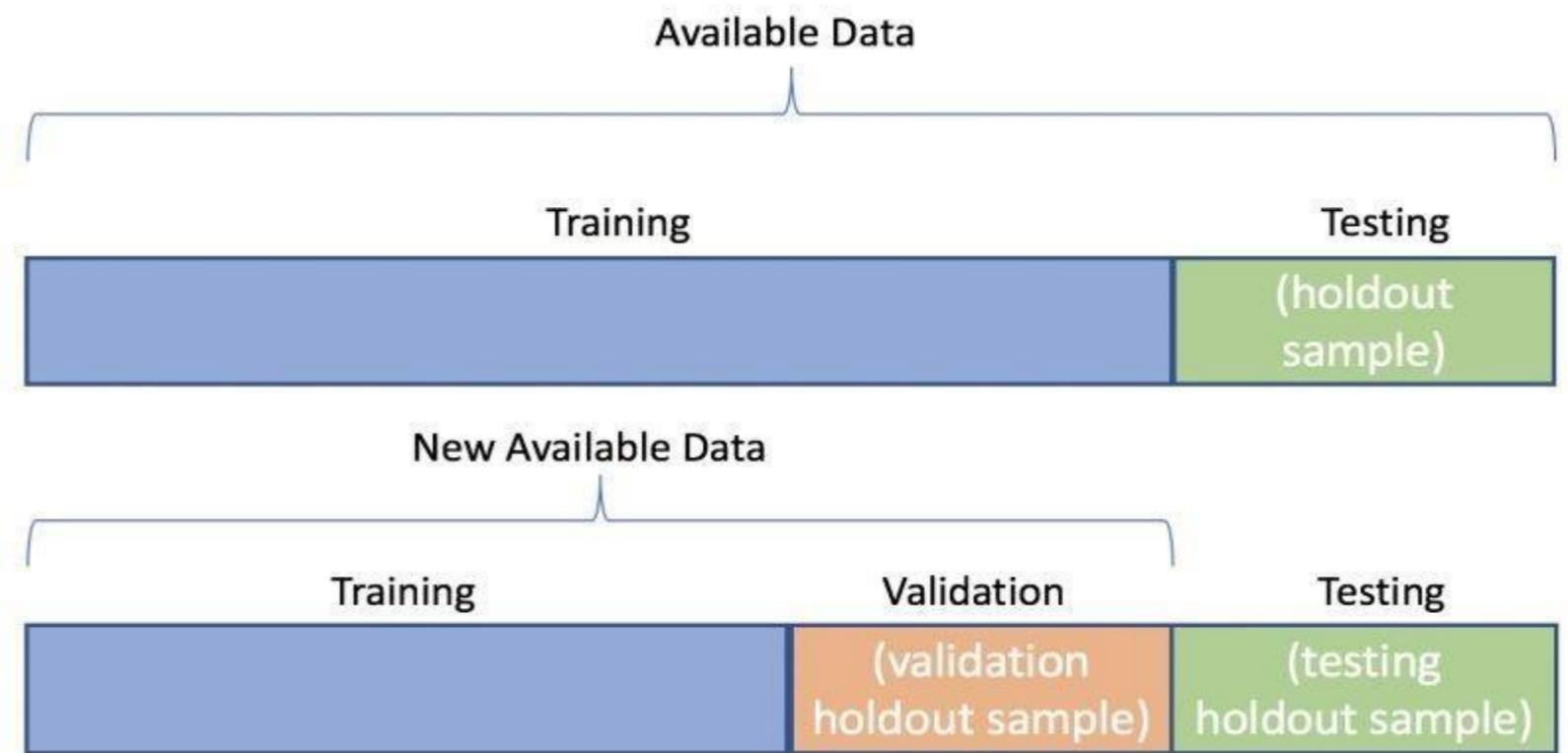
機器學習模型訓練流程



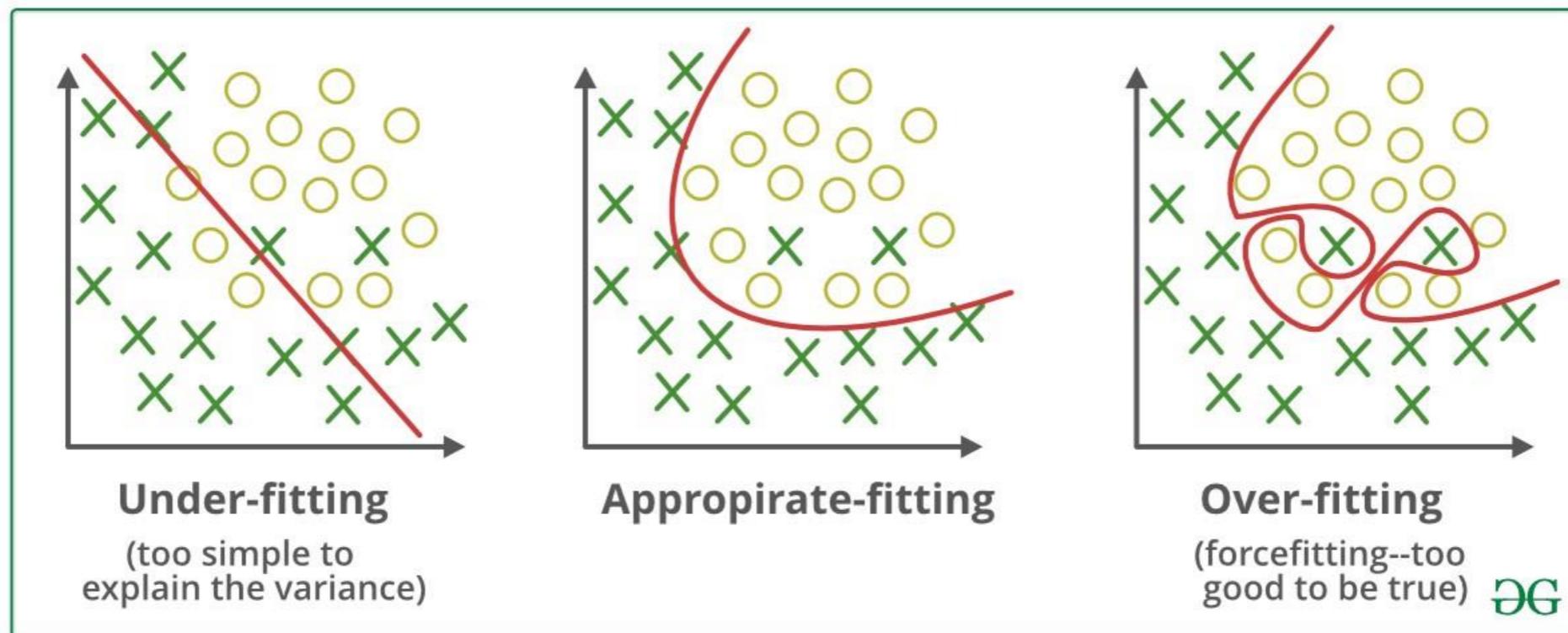
數據集分割

原數據訓練-測試 (8:2,
7:3, 2/3:1/3...)

新數據訓練-驗證 (8:2)
測試：保留樣本



驗證資料集的目的是避免在訓練時很準，使用時卻失準，也就是避免過適(overfitting)，以提升模型的泛化能力。



Module 1

機器學習

基礎介紹

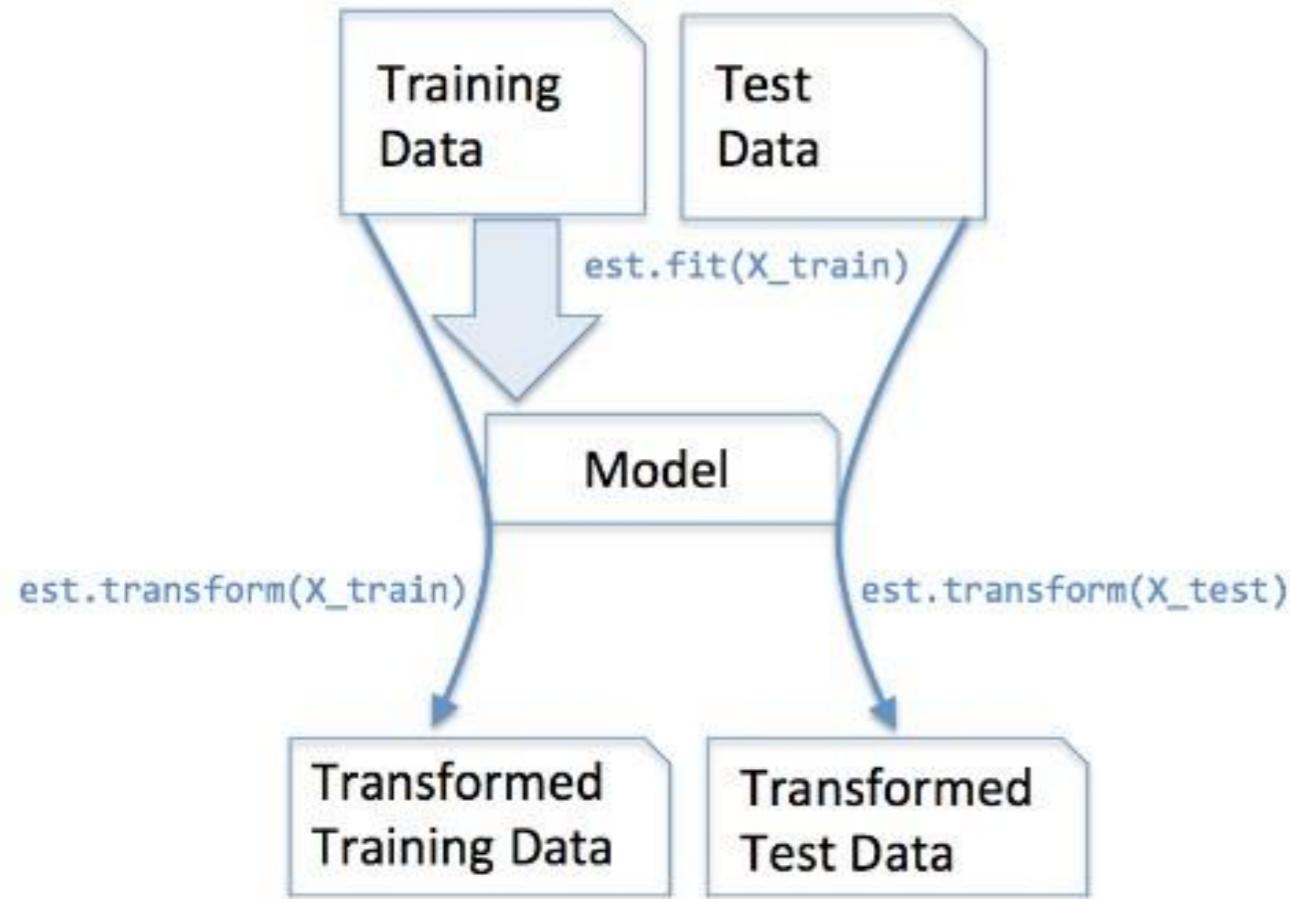
Scikit-learn 套件介紹

學習目標

- Scikit-learn API 介紹

Scikit-learn (曾叫做scikits.learn還叫做sklearn)

- 用於Python程式語言的自由軟體機器學習庫。
- 具有各種分類、回歸和聚類算法，包括支持向量機、隨機森林、梯度提升、k-平均聚類和DBSCAN，
- 協同於Python數值和科學庫NumPy和SciPy



Scikit對各種演算法都有一致的API，所以置換比較也很方便。主要方式如下：

- Fit: 擬合與訓練 (建模)
- Transform: 根據訓練模型做轉換
- Predict: 根據訓練模型做預測
- Fit_transform/Fit_predict : 同時做以上步驟

Module 2. 迴歸分析

迴歸模型簡介

學習目標

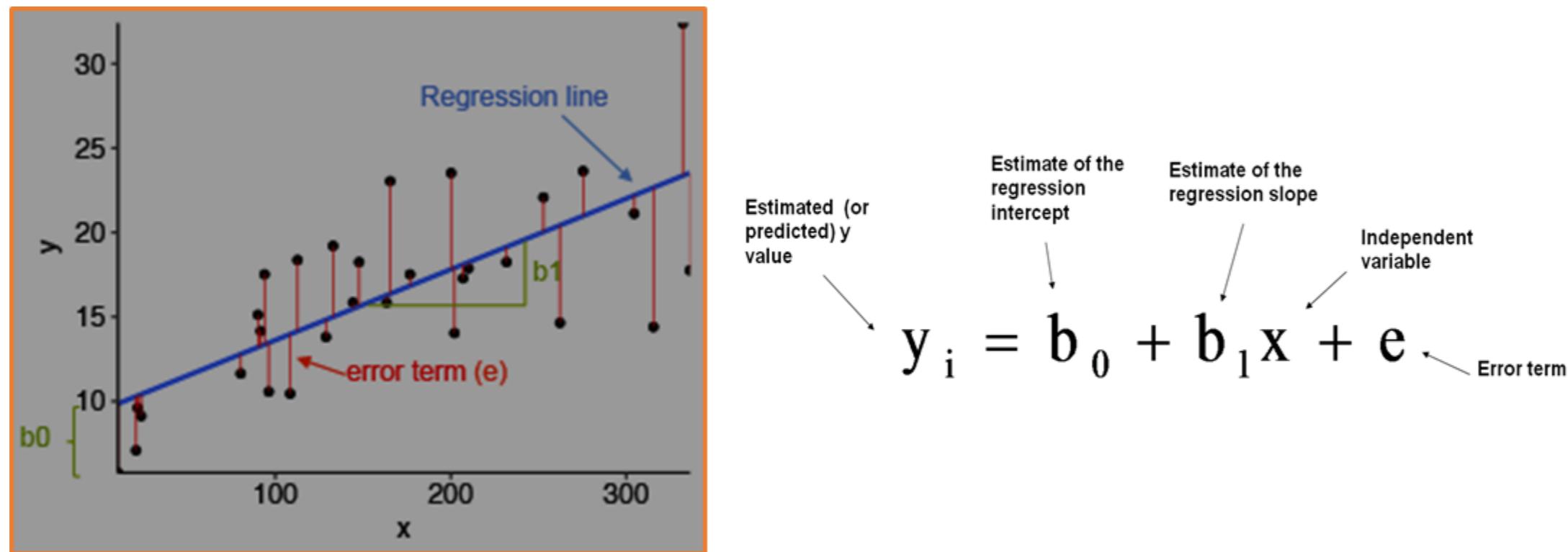
- 認識線性迴歸

如果以多元一次多項式表達，我們稱為線性回歸，格式如下：

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 \dots$$

簡單線性迴歸，則只有一個自變數，我們易於在二維空間的紙張上表達。

線性回歸是以一組獨立特徵去預測一個連續型的依變數，用來解決迴歸（預測）問題。目的就是要找到一條線性方程式，能夠最符合，我們的數據分布，最能夠代表我們的變數間的關係。



From: Datascience.foundation

線性迴歸分析主要的目的是預測，自變數是連續數，依變數也是連續數最適合，比如某地區的房價預測

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	...	grade	sqft_above	sqft_basement
0	7129300520	20141013T000000	221900.00	3	1.00	1180	5650	1.00	0	0	...	7	1180	0
1	6414100192	20141209T000000	538000.00	3	2.25	2570	7242	2.00	n	n	...	7	2170	400
2	5631500400	20150225T000000	180000.00	2	1.00	770	10000							
3	2487200875	20141209T000000	604000.00	4	3.00	1960	5000							
4	1954400510	20150218T000000	510000.00	3	2.00	1680	8080							



\$82000



\$55500



???

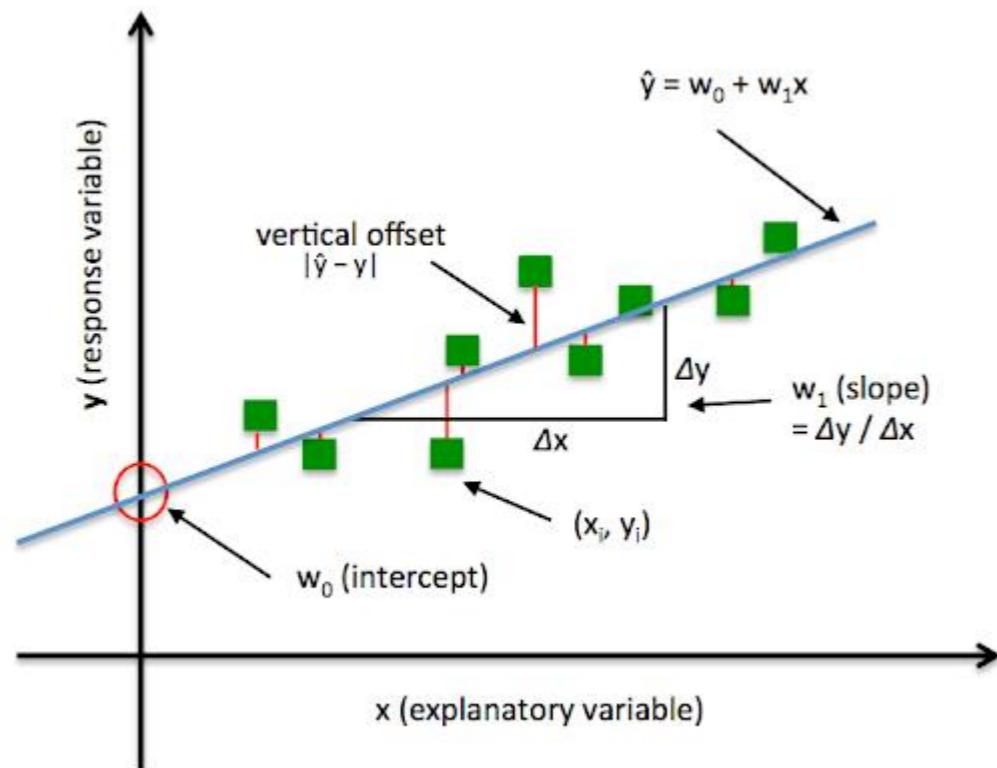
Module 2. 迴歸分析

迴歸模型評估

學習目標：

- 認識MSE
- 認識R2 Score

要比較線性迴歸模型之間的好壞，需要客觀的判準。我們可定義一目標函數，往往是計算誤差的大小或是損失，據以判斷，比方：



$$MAE = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}|$$

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

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

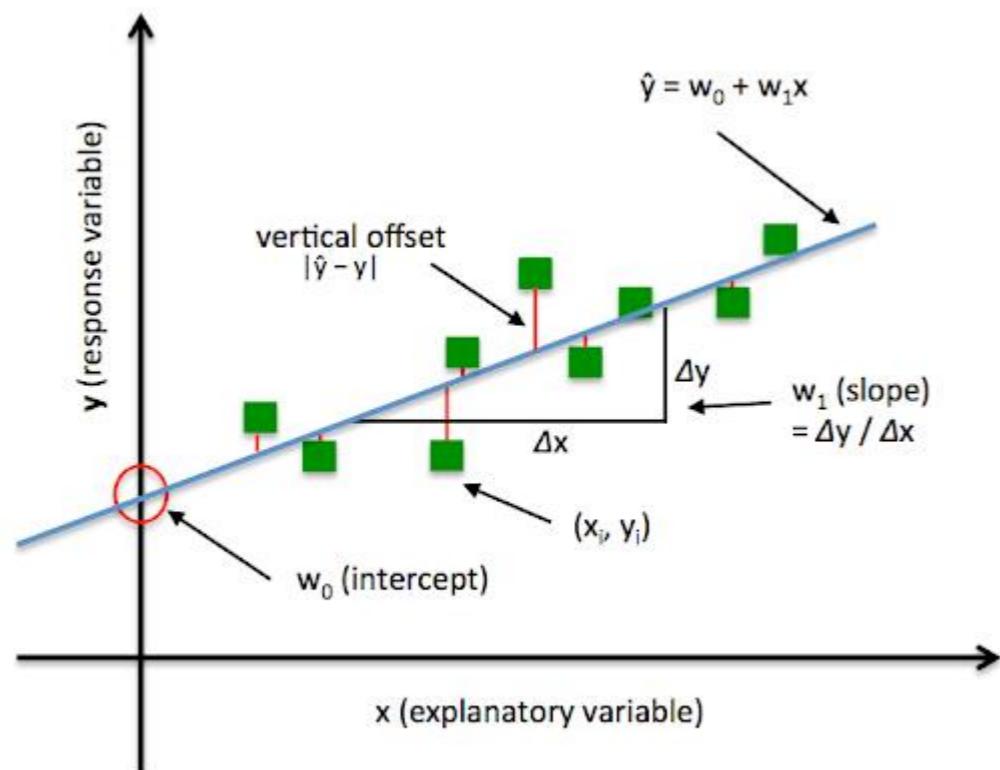
$$R^2 = 1 - \frac{\sum(y_i - \hat{y})^2}{\sum(y_i - \bar{y})^2}$$

Where,

\hat{y} – predicted value of y
 \bar{y} – mean value of y

以均方誤差 MSE(Mean Square Error)來評估

- 計算每個資料點與預估值的距離平方和再除以樣本數
- 數字愈接近0愈好。

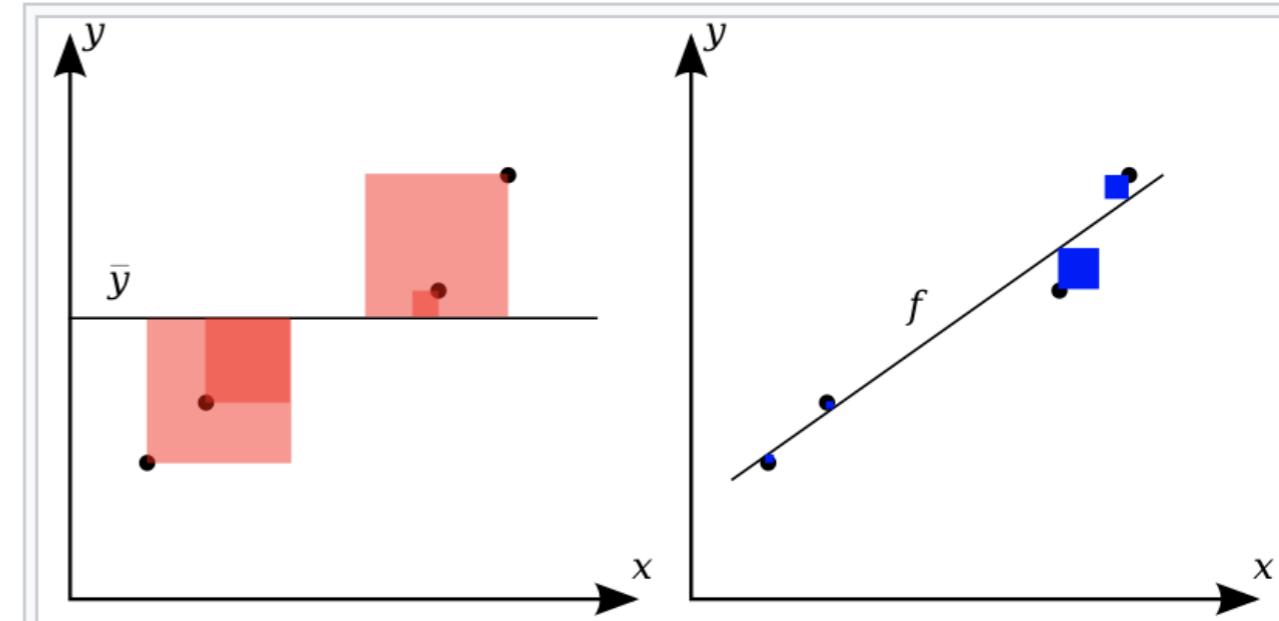


$$MSE = \frac{1}{n} \sum \underbrace{\left(y - \hat{y} \right)^2}_{\text{The square of the difference between actual and predicted}}$$

The square of the difference
between actual and
predicted

以R squared(coefficient of determination)來評估：

- 可由自變量解釋因變量的變異部分所占的比例，以此來判斷統計模型的解釋力。
- 解釋力趨近於1最好。



$$R^2 = 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}}$$

The better the linear regression (on the right) fits the data in comparison to the simple average (on the left graph), the closer the value of R^2 is to 1. The areas of the blue squares represent the squared residuals with respect to the linear regression. The areas of the red squares represent the squared residuals with respect to the average value.

Pic from: medium.com

Module 2. 迴歸分析

實作 – 房屋價格 預測

學習目標：

- 認識sklearn 線性迴歸模型API
- 實作線性迴歸預測

`sklearn.linear_model.LinearRegression` : 線性迴歸

- `coef_` : 可以看迴歸模型的參數值(係數)

`sklearn.metrics.mean_squared_error` : MSE

- `squared` : 是否要開平方 => RMSE

`sklearn.metrics.r2_score` : R-Square

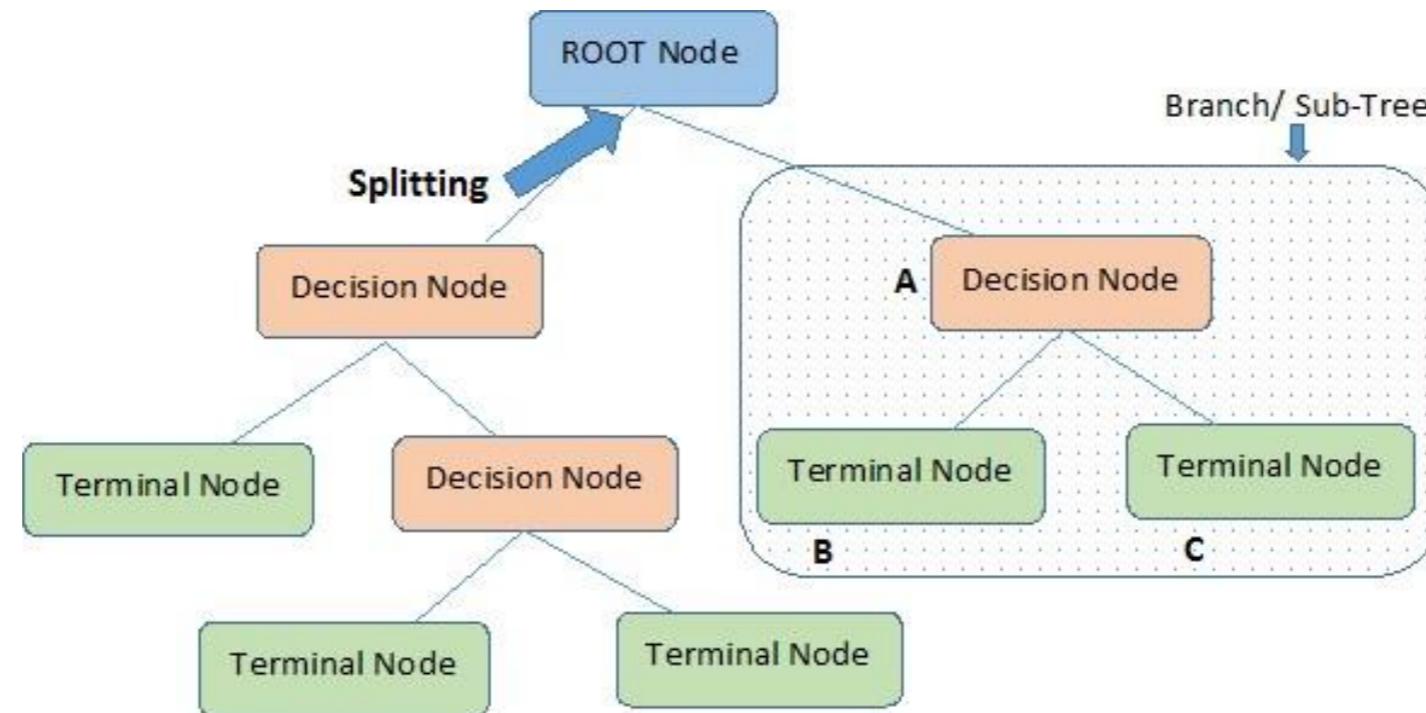
Module 3. 決策樹模型

決策樹

學習目標：

- 認識決策樹的運作原理

決策樹模型會用 features 切分資料，透過一系列的是非問題，幫助我們將資料進行切分。

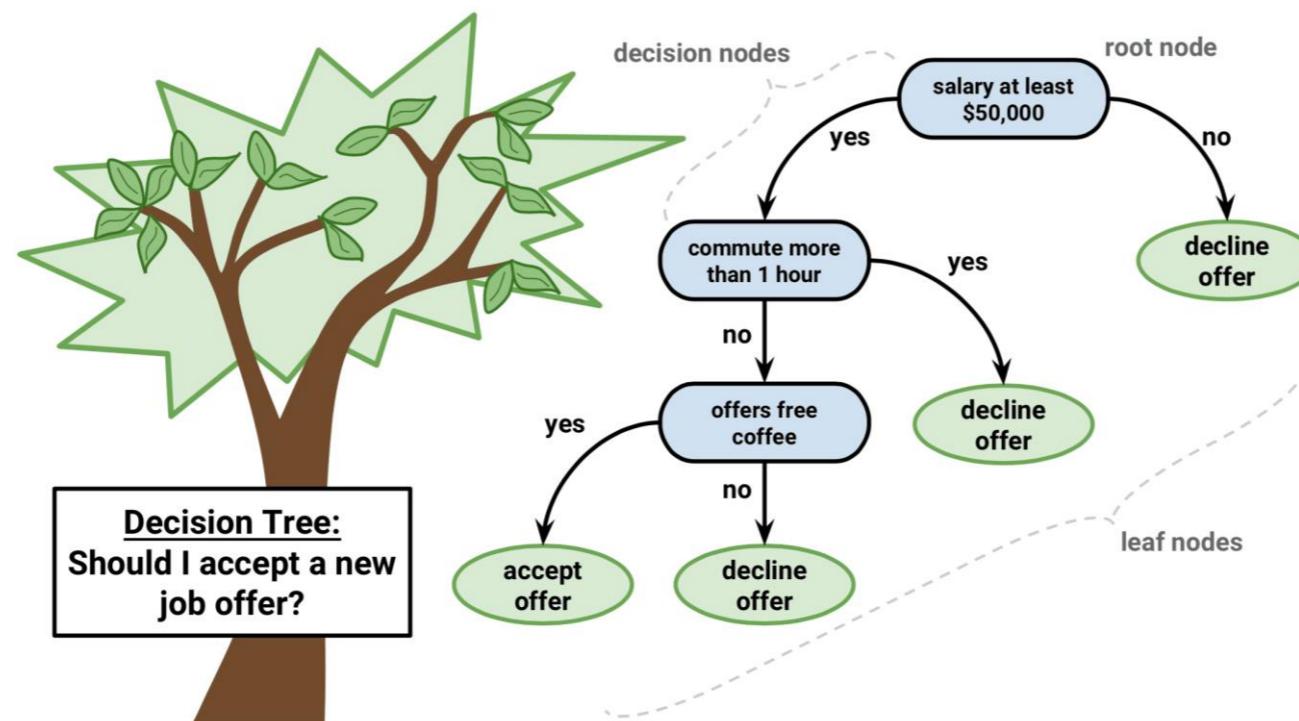


Note:- A is parent node of B and C.

<https://medium.com/towards-artificial-intelligence/decision-trees-explained-with-a-practical-example-fe47872d3b53>

決策樹時一個樹狀結構的決策模型，

- 由根部不斷分裂(split)，
- 每次分裂都以某特徵的數值作為切分的判準(Yes/No)，
- 而終於達到停止切分的標的。



pic from: datacamp.com

- 切分的目的：切分後在同一組內相似度高，組間相異度高
- 如何定義相似度：熵Entropy (亂度)，亂度愈小，相似度愈高
- 資訊量的度量類似亂度，以 $I(x) = -\log_2 p(x)$ 表達， $p(x)$ 為或然率
- 小明每次考試都90幾100，小華則每次都在60邊緣，以預測考試及格來說，誰的資訊量大？

訊息熵Entropy :

$$H(x) = \mathbb{E}(I(x)) = - \sum_j p_j \log_2 p_j$$

可以使用Entropy衡量資料相似程度（亂度）

Class 1	0
Class 2	6

$$p(\text{class 1}) = \frac{0}{6}, \quad p(\text{class 2}) = \frac{6}{6}$$

$$\text{Entropy} = -0 * \log(0) - 1 * \log(1) = 0$$

Class 1	1
Class 2	5

$$p(\text{class 1}) = \frac{1}{6}, \quad p(\text{class 2}) = \frac{5}{6}$$

$$\text{Entropy} = -\frac{1}{6} * \log\left(\frac{1}{6}\right) - \frac{5}{6} * \log\left(\frac{5}{6}\right) = 0.65$$

Class 1	2
Class 2	4

$$p(\text{class 1}) = \frac{2}{6}, \quad p(\text{class 2}) = \frac{4}{6}$$

$$\text{Entropy} = -\frac{2}{6} * \log\left(\frac{2}{6}\right) - \frac{4}{6} * \log\left(\frac{4}{6}\right) = 0.91$$

On line Entropy Calculator: [Shannon Entropy Calculator | Information Theory \(omnicalculator.com\)](https://www.omnicalculator.com/information-theory/shannon-entropy-calculator)

訊息增益 (Information Gain):

$$\text{Information Gain} = \text{Entropy}(\text{before splitting}) - E(\text{Entropy}(\text{After Splitting}))$$

Before Splitting:

Class 1	6
Class 2	6

$$\text{Entropy}(\text{before splitting}) = 1.000$$

After Splitting:

A

Class 1	4
Class 2	3

Class 1	2
Class 2	3

$$\text{Entropy} = 0.985$$

$$\begin{aligned}E(\text{Entropy}(\text{after splitting})) &= 7/12 * 0.985 + 5/12 * 0.971 \\&= 0.979\end{aligned}$$

B

Class 1	1
Class 2	4

Class 1	5
Class 2	2

$$\text{Entropy} = 0.722$$

$$\begin{aligned}E(\text{Entropy}(\text{after splitting})) &= 5/12 * 0.722 + 7/12 * 0.863 \\&= 0.804\end{aligned}$$

決策樹

before splitting

Class 1	6
Class 2	6

Entropy(before splitting) = 1.000

after splitting

Information Gain on A way

$$\begin{aligned} &= \text{Entropy}(\text{before splitting}) - E[\text{Entropy}(\text{after splitting})] \\ &= 1.000 - 0.979 = 0.021 \end{aligned}$$

Information Gain on B way

$$\begin{aligned} &= \text{Entropy}(\text{before splitting}) - E[\text{Entropy}(\text{after splitting})] \\ &= 1.000 - 0.804 = 0.196 \end{aligned}$$

因為 $0.196 > 0.021$ ，由以上得知：B 方法較優

Module 3. 決策樹模型

決策樹- ID3

學習目標：

- 認識ID3演算法



Iterative Dichotomiser 3 (ID3)

使用訊息增益量作為選擇特徵切割點的指標

$$\text{Entropy}(\text{before split}) = -5/14 *$$

$$\log(5/14) - 9/14 * \log(9/14) = 0.94$$

$$E(S) = \sum_{i=1}^c -p_i \log_2 p_i$$

Play Golf	
Yes	No
9	5

Entropy(PlayGolf) = Entropy (5,9)
 $= \text{Entropy} (0.36, 0.64)$
 $= - (0.36 \log_2 0.36) - (0.64 \log_2 0.64)$
 $= 0.94$

計算以Outlook切分的Entropy值

$$E(T, X) = \sum_{c \in X} P(c)E(c)$$

		Play Golf		
		Yes	No	
Outlook	Sunny	3	2	5
	Overcast	4	0	4
	Rainy	2	3	5
				14



$$\begin{aligned} E(\text{PlayGolf}, \text{Outlook}) &= P(\text{Sunny}) * E(3,2) + P(\text{Overcast}) * E(4,0) + P(\text{Rainy}) * E(2,3) \\ &= (5/14) * 0.971 + (4/14) * 0.0 + (5/14) * 0.971 \\ &= 0.693 \end{aligned}$$

		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			

		Play Golf	
		Yes	No
Temp.	Hot	2	2
	Mild	4	2
	Cool	3	1
Gain = 0.029			

		Play Golf	
		Yes	No
Humidity	High	3	4
	Normal	6	1
Gain = 0.152			

		Play Golf	
		Yes	No
Windy	False	6	2
	True	3	3
Gain = 0.048			

在接下來的節點按照相同的方式分類

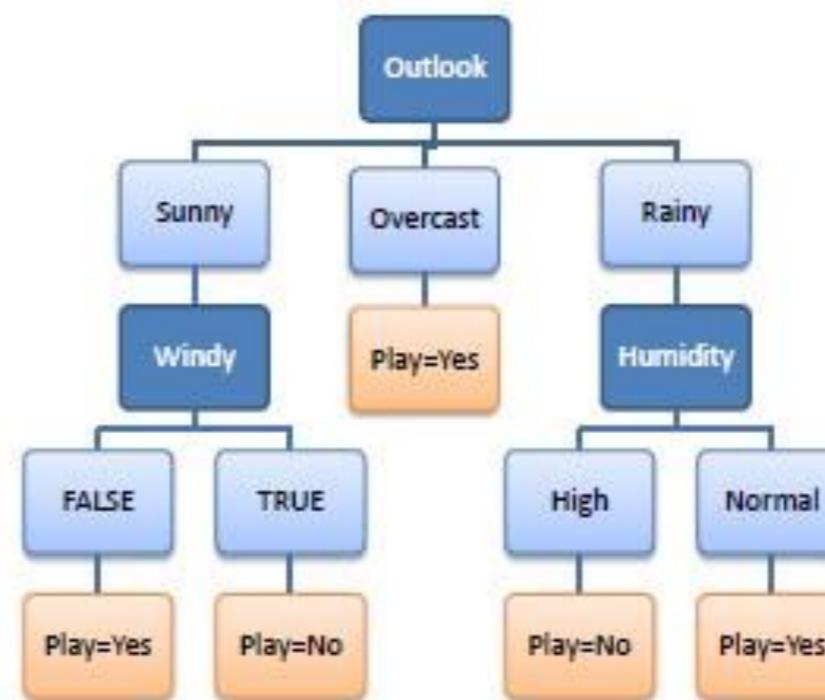
R₁: IF (Outlook=Sunny) AND
(Windy=FALSE) THEN Play=Yes

R₂: IF (Outlook=Sunny) AND
(Windy=TRUE) THEN Play=No

R₃: IF (Outlook=Overcast) THEN
Play=Yes

R₄: IF (Outlook=Rainy) AND
(Humidity=High) THEN Play=No

R₅: IF (Outlook=Rain) AND
(Humidity=Normal) THEN
Play=Yes



決策樹優點：

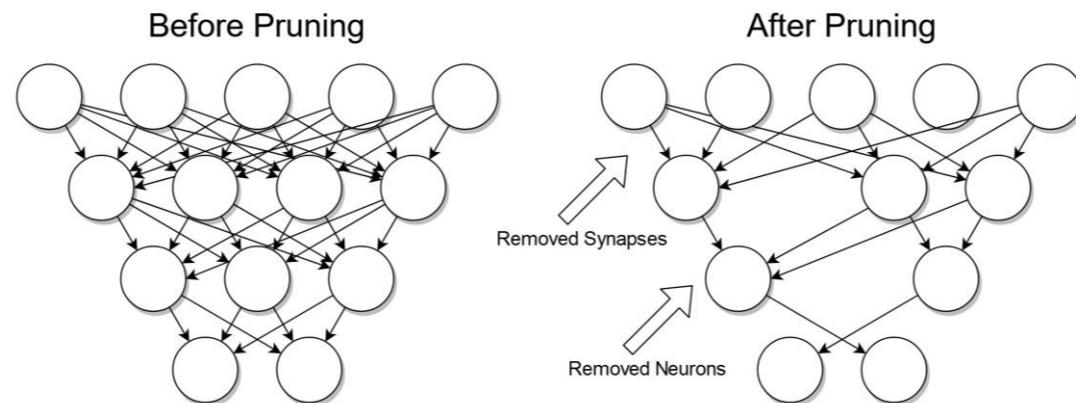
- 可視覺化每個切分決定的過程
- 具有高解釋性

決策樹缺點：

- 非常容易過適(Overfitting)，除非對決策樹進行限制（樹深度、葉子上至少要有多少樣本等）

在API中可調整的Hyper Parameters，也可說是決策樹枝修剪，以避免過適

- Max_Depth：深度限制
- Min_Samples_Split：至少要多少樣本數才切分
- Min_Samples_Leaf：終端節點上至少要有多少樣本



pic from:wikipedia.com

Module 3. 決策樹模型

隨機森林模型和分 類模型評估方式

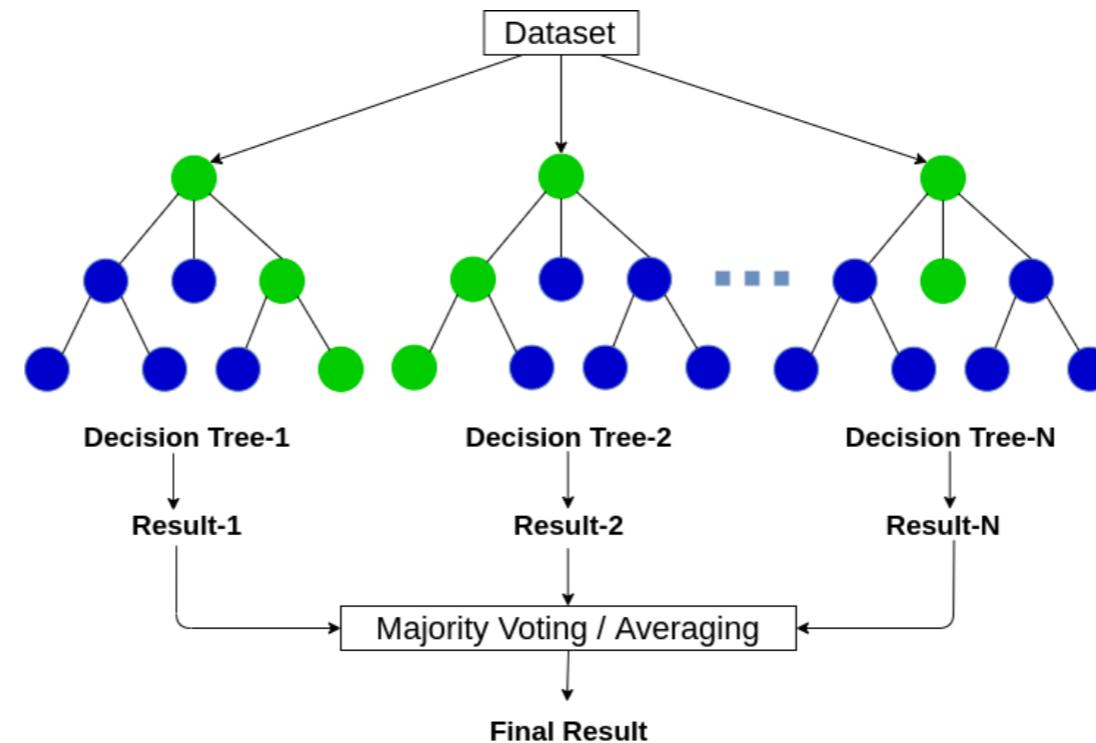
學習目標：

- 認識隨機森林
- 認識混淆矩陣

隨機森林模型

隨機森林的概念，以決策樹為基底，以集體協作的方式開發出的模型。

隨機森林每一棵決策樹在生成過程中，都是隨機使用部分資料與特徵，以隨機方式訓練而成。



pic from: analyticsvidhya.com

- n_estimators: 基礎決策樹的數量
- m_depth: 每棵樹的最深限制
- min_samples_split: 每棵樹分裂前至少樣本數
- min_samples_leaf: 每棵樹葉節點至少樣本數

混淆矩陣(Confusion Matrix)

混淆矩陣 (confusion matrix)，又稱為可能性表格或是錯誤矩陣。

用來評價算法或者分類器的結果分析表。

每一列代表預測值
每一行代表實際值

		實際 YES	實際 NO	
預測 YES	實際 YES	TP	FP	$\frac{TP}{TP+FP}$ Precision
	實際 NO	FN	TN	$\frac{FP}{TP+FP}$
預測 NO	實際 YES	TP	FP	$\frac{FN}{FN+TN}$
	實際 NO	FN	TN	$\frac{TN}{FN+TN}$
		$\frac{TP}{TP+FN}$ Recall	$\frac{FN}{TP+FN}$	$\frac{FP}{FP+TN}$
		$\frac{TP}{TP+FN}$	$\frac{FP}{FP+TN}$	$\frac{TN}{FP+TN}$

Accuracy = $(TP+TN) / \text{Tot. N}$

Precision = $TP / (TP+FP)$

Recall = $TP / (TP+FN)$

F1 Score = $\frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}}$

F Measure

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}}$$

pic from: ithome01.com

Module 3. 決策樹模型

實作 - 使用決策樹
與隨機森林建立分
類模型

學習目標：

- 認識隨機森林sklearn API

- `sklearn.tree.DecisionTreeClassifier(...)`：決策樹分類器
- `sklearn.ensemble.RandomForestClassifier(...)`：隨機森林分類器
- `sklearn.metrics.accuracy_score(Y, y_pred)`：計算Accuracy分數
參數：`Normalize(正規化):`
 - True : 求得比例
 - False : 求得數目

Module 4. 羅吉斯迴歸 模型

Log loss 介紹

學習目標：

- 認識cross entropy

甚麼是Log loss?

就是如右圖的函數，X軸基本上是機率 $p(x)$ ，y 軸 則是 $-\log(p(x))$ ，也就是將機率 $p(x)$ 先取對數(底一般為2)，再取負號。

為何用Log loss當機率損失目標函數？

$$\text{Log}(0) = -\infty, \text{Log}(1) = 0$$

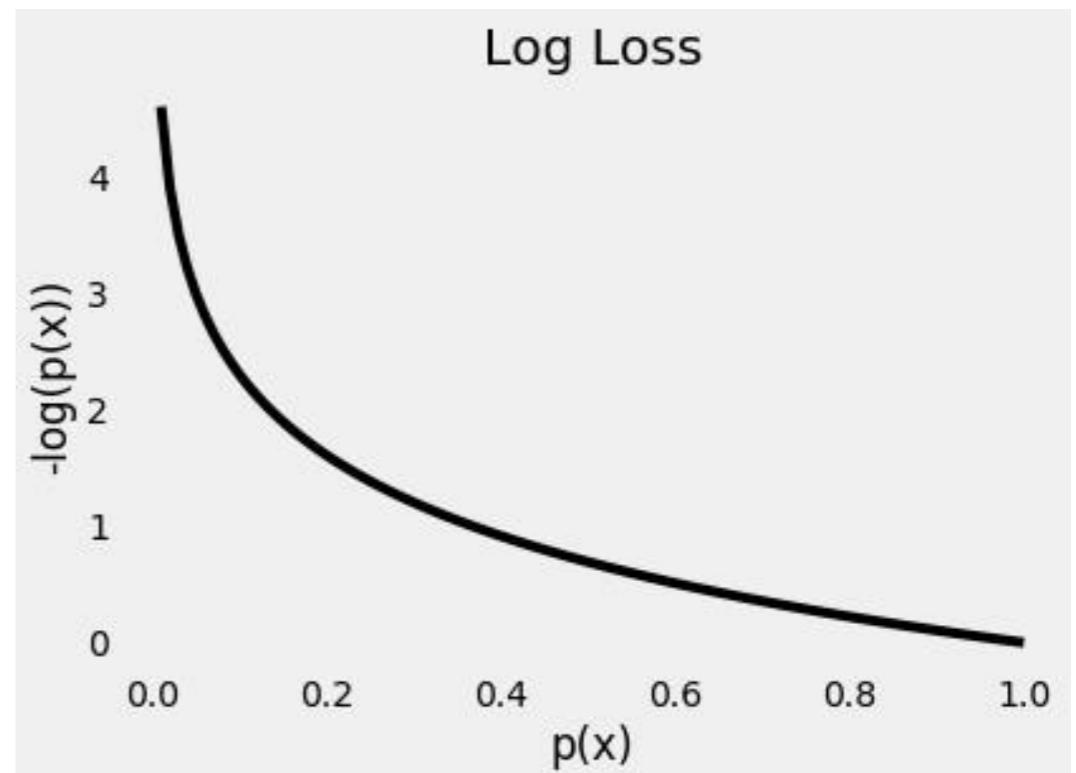
所以

$$-\text{Log}(0) = \infty, -\text{Log}(1) = 0$$

也就是機率為0時損失最大，1時最小，適合當機率損失

目標函數

另外在函數做乘法運算時Log可變成加法，十分方便



pic from: towarddatascience.com

分類經常是二元分類(True/False, Yes/No)，在二元分類時，我們可把Loss Function調整為 (此公式又稱為Binary Cross-Entropy Function)

$$\frac{-1}{m} \sum_{i=1}^m y_i \log(P(x_i)) + (1-y_i) \log(1-P(x_i))$$

Y_i 是第*i*項樣本的標籤(label)

如果 $y_i = 1$ ，該項值會是 $\log(P(x_i))$

如果 $y_i = 0$ ，該項值會是 $\log(1-P(x_i))$

Module 4. 羅吉斯迴歸 模型

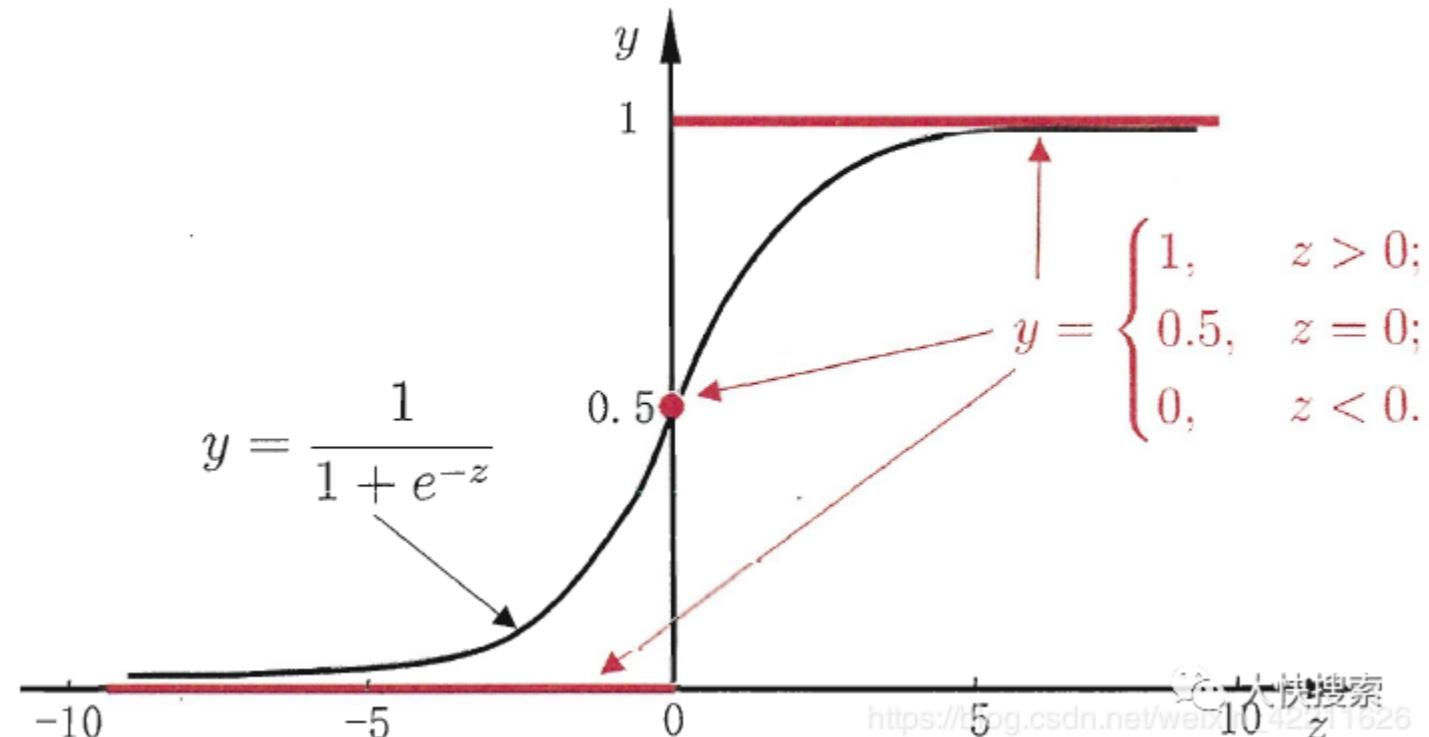
學習目標：

- 認識Logistic Regression

羅吉斯迴歸介紹

非線性激活函數: Sigmoid Function

Sigmoid是一個有界，可微分的實函數，每一點之導數皆為正，且只有一個拐點。因輸出值介於0與1之間，適合做機率的邏輯分類(0,1)。



pic from: sohu.com

羅吉斯迴歸損失計算

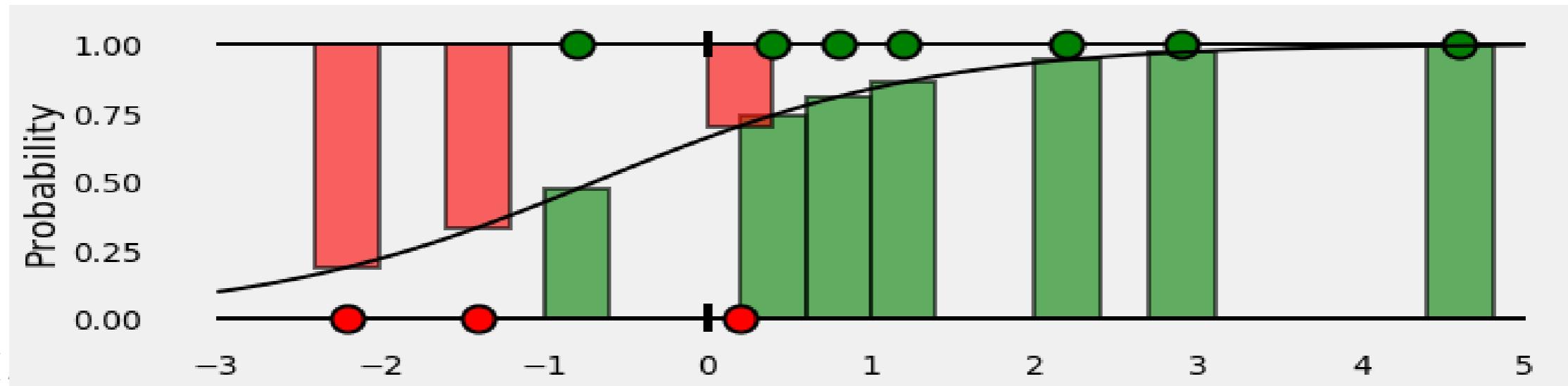
Sigmoid 模型
$$h\theta(X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$

損失函數：

$$Cost(h\theta(x), y) = \begin{cases} -\log(h\theta(x)) & \text{if } y = 1 \\ -\log(1 - h\theta(x)) & \text{if } y = 0 \end{cases}$$

也可表達為
$$J(\theta) = -\frac{1}{m} \sum [y^{(i)} \log(h\theta(x(i))) + (1 - y^{(i)}) \log(1 - h\theta(x(i)))]$$

 機器學習的參數是： β_0, β_1



Module 4. 羅吉斯迴歸 模型

-

實作 – 使用羅吉斯
迴歸建立分類模型

學習目標：

- 認識何為 cross entropy

sklearn.linear_model.LogisticRegression() : logistic regression

常用參數:

Penalty : “L1” , “L2” 。使用L1或L2的正則化參數

C : 正則化的強度，數字越小，模型越簡單

常用屬性:

coef_ : 形狀 (n_classes, n_features) , 決定函數的特徵值

Intercept_ : 決定函數的截距.

正則化是用來限制學習來的參數的數值範圍，數值範圍越大，模型就越複雜，越容易過適。

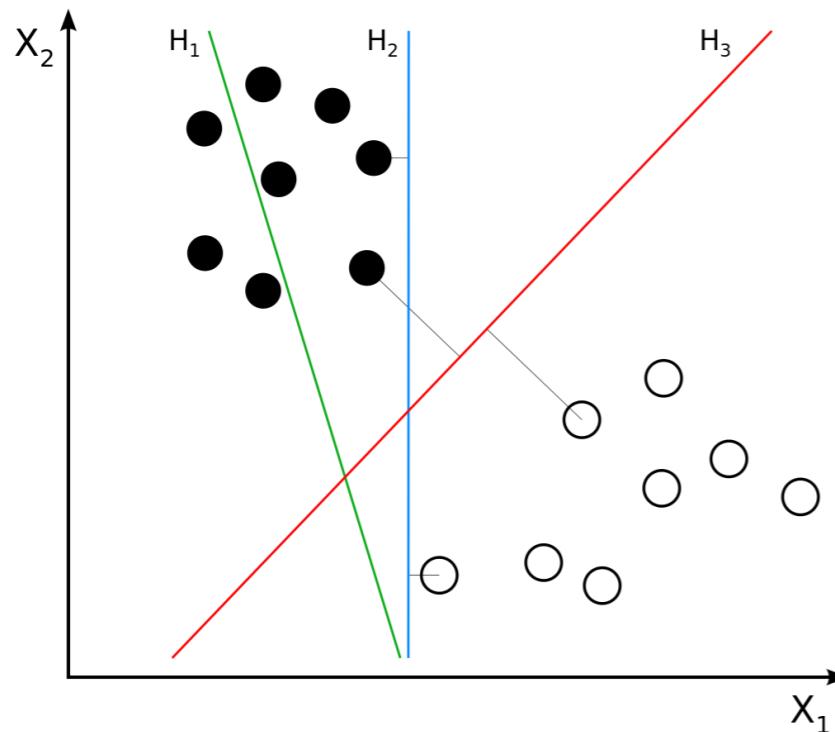
Module 5. 支持向量機 SVM

SVM 介紹

學習目標：

- SVM原裡介紹

簡單地說，支持向量機(SVM)在高維空間中構造超平面，其可以用於分類、回歸或其他任務。分類邊界距離最近的訓練資料點越遠越好，因為這樣可以縮小分類器的泛化誤差。



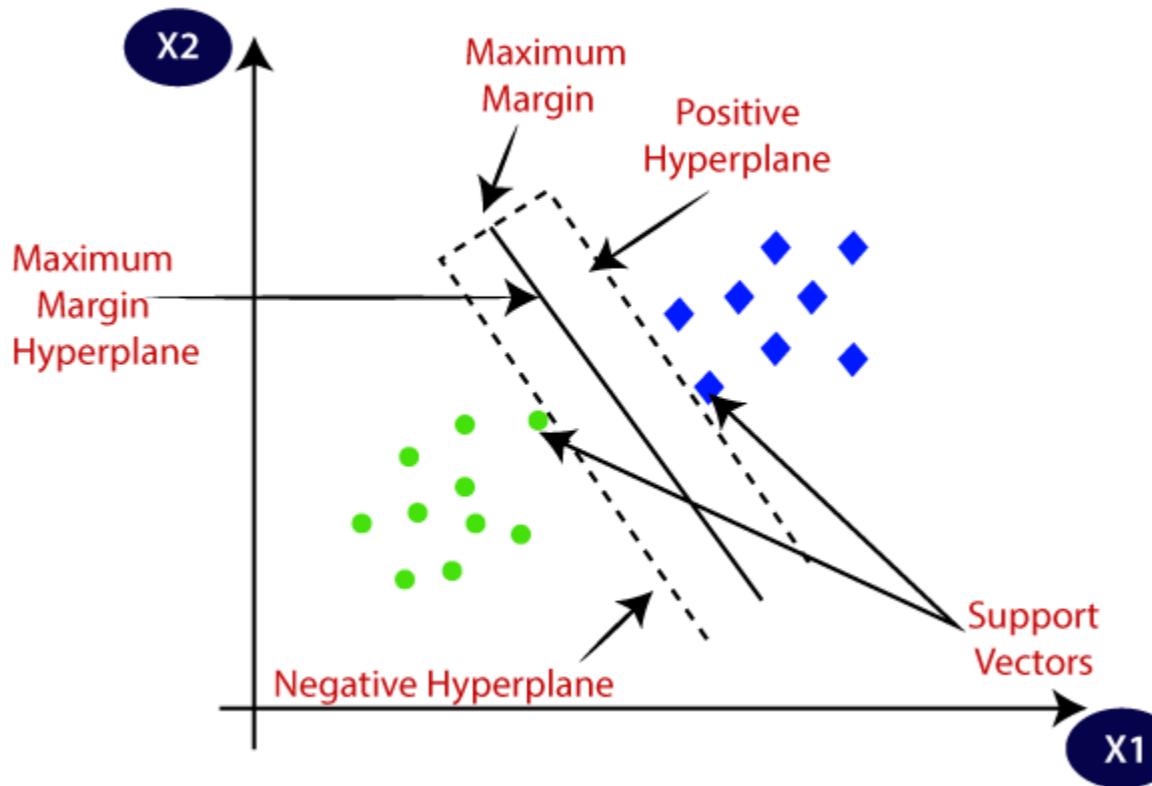
H_1 不能把類別分開。

H_2 可以，但只有很小的間隔。

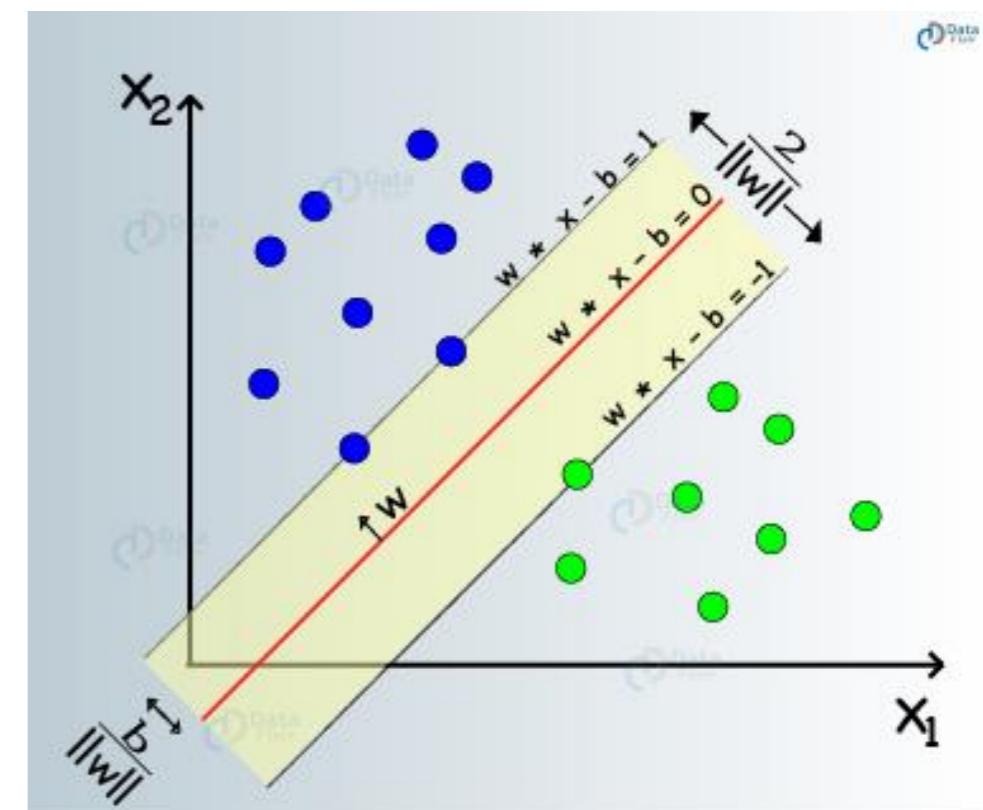
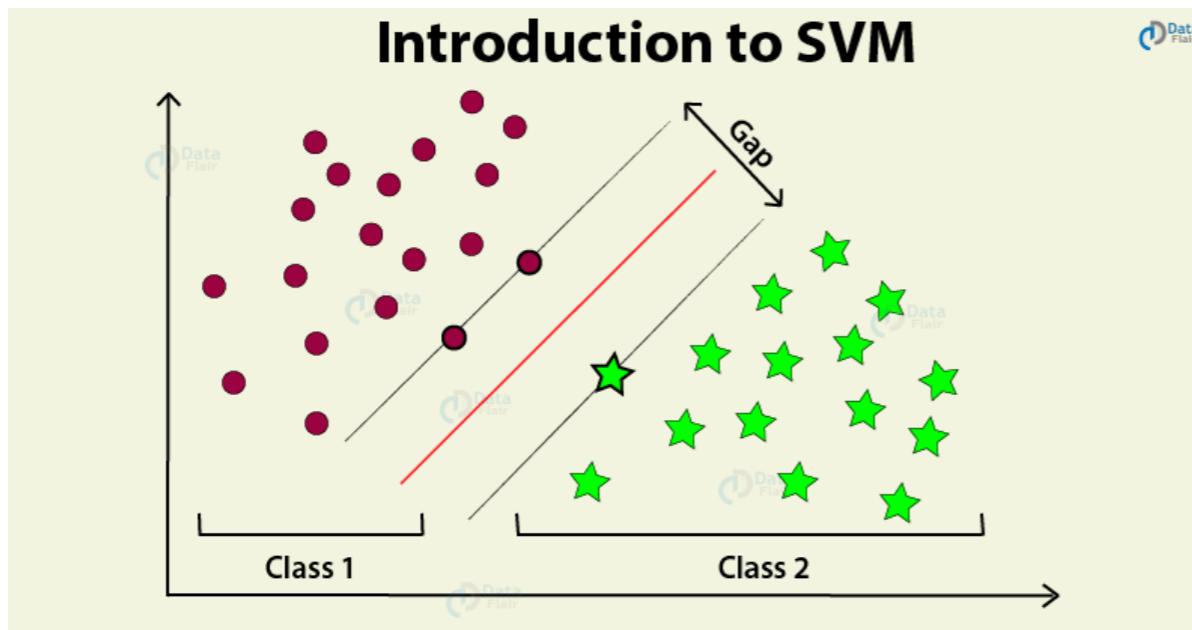
H_3 以最大間隔將它們分開。

SVM 介紹：何謂支持向量

支持向量指的是兩個類別(比方藍和綠)資料中最靠近彼此(或者最突出)的樣本。



1. 定義Class Label 為 1 and -1
2. 定義兩個Class 的支持超平面(Support Hyperplane)
3. 在兩支持超平面中間作為分類器(Classifier)
4. 定義目標函數為 $\text{margin} = 2 / \|w\|$
5. 限制兩邊距離至少為1
6. 結合目標函數與限制式求解 w



優點

- 高效高穩定度，不易被異常值影響
- 核函數解決非線性問題
- 適用於各種資料集
- 高維度小樣本時效果佳
- Margin清晰時很有效

缺點

- 黑箱方法，不提供機率預測
- 易於Overfitting
- 不適於大型數據或雜訊多的數據
- 訓練速度慢
- 只適合1次區隔2Classes

Module 5. 支持向量 機SVM

SVM Kernel Function

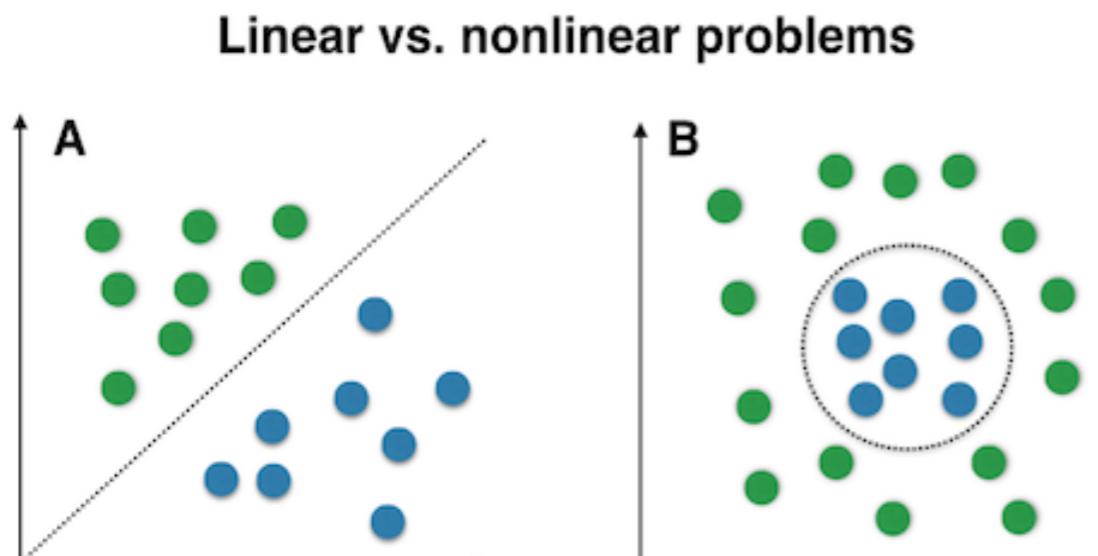
學習目標：

- 了解如何處理非線性
可分資料

SVM Kernel Function

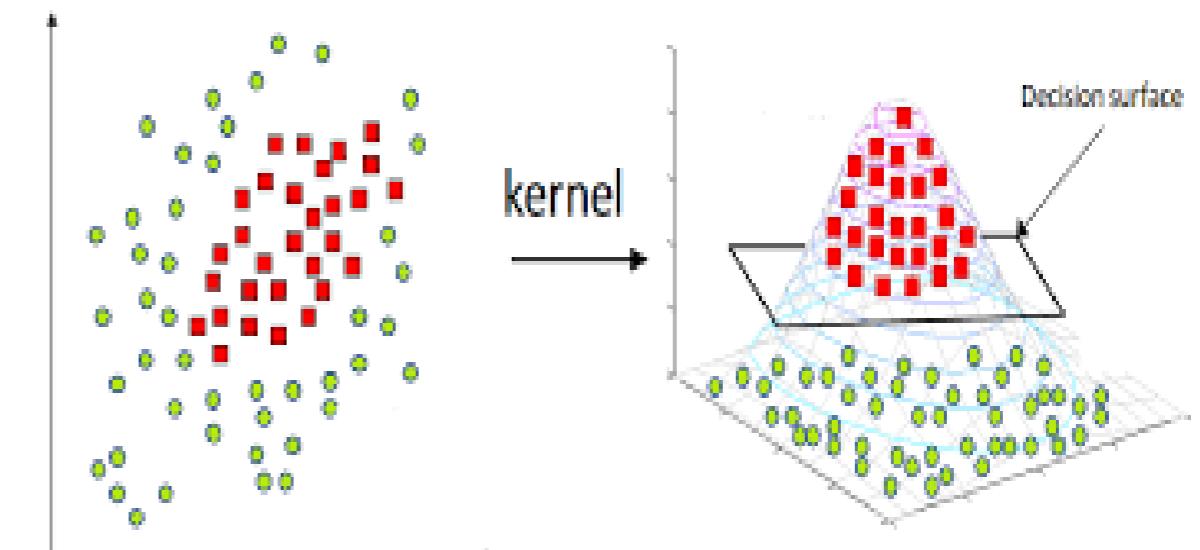
SVM 核函式可以映射至高維的方式解決非線性的問題

非線性問題



Pic from: stackoverflow.com

映射至高維



Pic from: medium.com

常見的Kernel Functions

Kernel	Equation
Linear	$K(x, y) = x \cdot y$
Sigmoid	$K(x, y) = \tanh(ax \cdot y + b)$
Polynomial	$K(x, y) = (1 + x \cdot y)^d$
KMOD	$K(x, y) = a \left[\exp\left(\frac{\gamma}{ x-y ^2 + \sigma^2}\right) - 1 \right]$
RBF	$K(x, y) = \exp(-a x - y ^2)$
Exponential RBF	$K(x, y) = \exp(-a x - y)$

pic from: esearchgate.net

Module 5. 支持向量 機SVM

實作 - 使用SVM
建立分類模型

學習目標：

- 認識 Scikit learn SVM API

sklearn.svm.SVC : SVM

- C: 正則化強度，數字愈小，模型愈簡單
- Kernel: 映射至高維度的方式 ('linear' , 'poly' , 'sigmoid' , 'precomputed' , or callable, default = 'rbf')
- degree: 'poly' kernel function 維度
- gamma: ('scale' , 'auto') for 'sigmoid' , 'poly' , 'rbf'

Module 6. 樸素貝葉斯 Naïve Bayes

貝葉斯定理
(Bayes Throrem)

學習目標：

- 認識條件機率
- 認識貝氏定理

當事件B發生時，事件A發生的機率如下

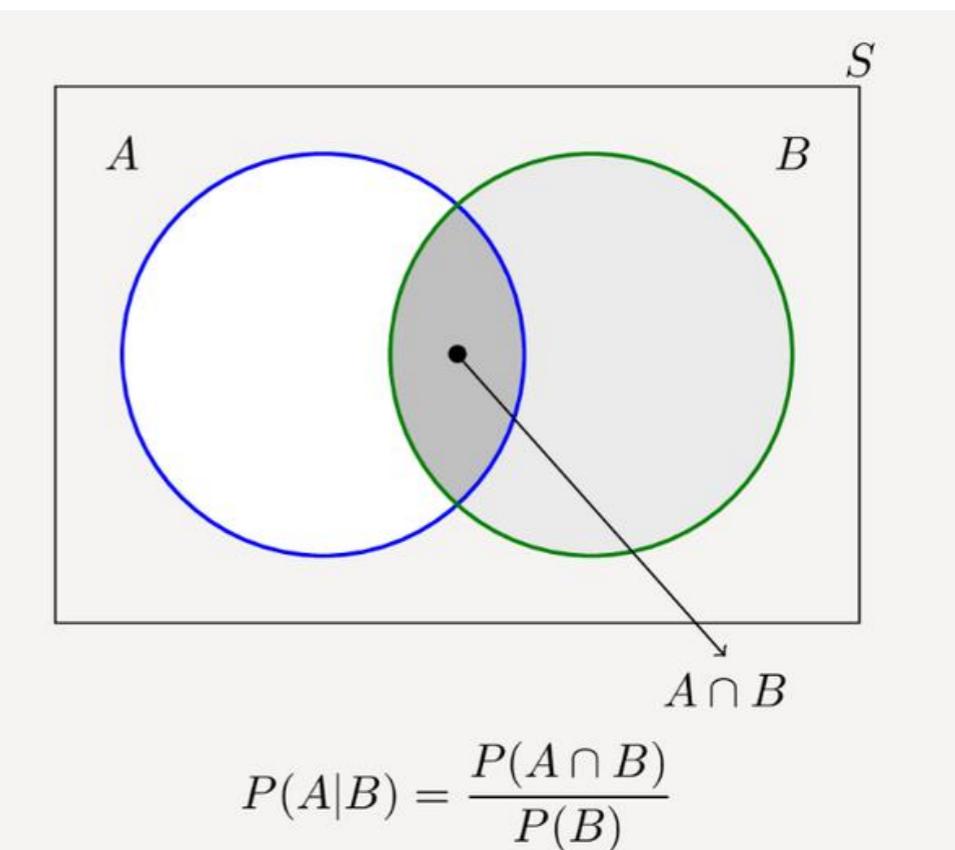
Conditional Probability Formula

$$P(A | B) = \frac{\text{Probability of } A \text{ and } B}{\text{Probability of } A \text{ given } B}$$

Probability of B

onlinemathlearning.com

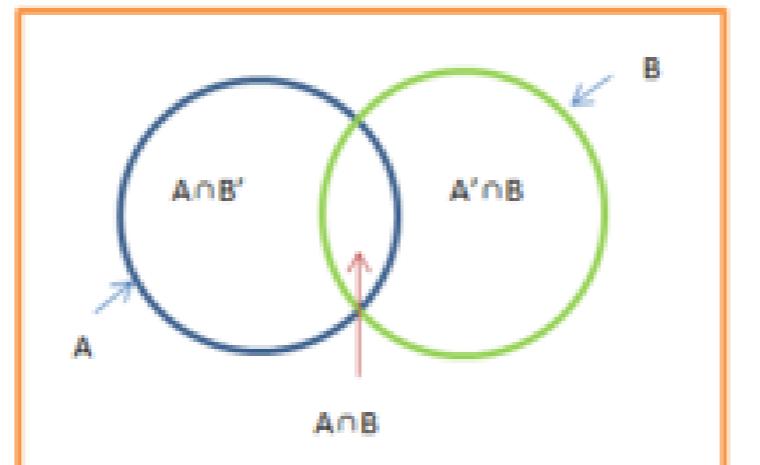
以圖示之



Pic from: Quora.com

何謂獨立事件？

- 我們知道當B發生時A發生的條件機率是 $P(A|B) = P(A \cap B)/P(B)$
- 獨立事件指的是無論B是否發生，都不影響A事件發生：
- $P(A|B) = P(A \cap B)/P(B) = P(A)$
- 換句話說 $P(A \cap B) = P(A) * P(B)$



貝氏定理推導：

- $P(A|B) = P(A \cap B)/P(B)$
- $P(A|B) * P(B) = P(A \cap B)$ (2)
- $P(B|A) = P(B \cap A)/P(A) = P(A \cap B)/P(A)$
- $P(B|A) * P(A) = P(A \cap B)$ (4)
- $P(A|B) * P(B) = P(B|A) * P(A)$ from (2),(4)

得知：

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}, \text{ 同理} \quad P(B|A) = \frac{P(A|B) * P(B)}{P(A)}$$

Bayes' Theorem

$$P(A|B) = \frac{P(B|A) P(A)}{P(B)}$$

教室裡有24人，8人熱愛運動，14人是男生，男生且熱愛運動有5人，假設某A喜歡運動，此人為男生的機率是多少？

Bayes' Theorem

$$P(A|B) = \frac{P(B|A) P(A)}{P(B)}$$

$$P(\text{熱愛運動}|男) = \frac{P(\text{男} \cap \text{熱愛運動})}{P(\text{男})} = \frac{5/24}{14/24} = 5/14$$

$$P(\text{男}|\text{熱愛運動}) = \frac{P(\text{熱愛運動}|男) P(\text{男})}{P(\text{熱愛運動})} = \frac{5/14 * 14/24}{8/24} = (5/24)/(8/24) = 5/8$$

Module 6. 樸素貝葉斯 Naïve Bayes

學習目標：

- 認識貝氏定理在機器學習的應用

樸素貝葉斯
(Naïve Bayes)

我們已學過貝氏定理，而樸素貝葉斯就是：
以貝氏定理為基礎，進一步假設各事件為獨立所得到的演算法

假設一群樣本有三特徵 F_0, F_1, F_2 ，想分辨是哪種類別 C_0 或 C_1
判別準則定為：

If $p(C_0|F_1, F_2, F_3) > p(C_1|F_1, F_2, F_3) \rightarrow$ 判為 C_0

If $p(C_1|F_1, F_2, F_3) > p(C_0|F_1, F_2, F_3) \rightarrow$ 判為 C_1

F_0	F_1	F_2	Class
1	1	1	C_0
1	1	0	C_1
1	0	1	C_0
0	1	1	C_1
0	0	0	C_0

假設各 F 獨立(樸素)，此時

$$\begin{aligned} p(F_1, F_2, F_3 | C_0) &= p(F_1 | C_0)p(F_2 | C_0)p(F_3 | C_0) \\ p(F_1, F_2, F_3 | C_1) &= p(F_1 | C_1)p(F_2 | C_1)p(F_3 | C_1) \end{aligned}$$

所以

$$p(C_0 | F_1, F_2, F_3) = p(F_1, F_2, F_3 | C_0) * p(C_0) / p(F_1, F_2, F_3) = p(F_1 | C_0)p(F_2 | C_0)p(F_3 | C_0) * p(C_0) / p(F_1, F_2, F_3)$$

同時

$$p(C_1 | F_1, F_2, F_3) = p(F_1, F_2, F_3 | C_1) * p(C_1) / p(F_1, F_2, F_3) = p(F_1 | C_1)p(F_2 | C_1)p(F_3 | C_1) * p(C_1) / p(F_1, F_2, F_3)$$

現在要判斷是否 $p(C_0 | F_1, F_2, F_3) > p(C_1 | F_1, F_2, F_3)$ 從而決定是 C_0 就變成判斷是否

$$p(F_1 | C_0)p(F_2 | C_0)p(F_3 | C_0) * p(C_0) / p(F_1, F_2, F_3) > p(F_1 | C_1)p(F_2 | C_1)p(F_3 | C_1) * p(C_1) / p(F_1, F_2, F_3)$$

簡化之

$$p(F_1 | C_0)p(F_2 | C_0)p(F_3 | C_0) * p(C_0) > p(F_1 | C_1)p(F_2 | C_1)p(F_3 | C_1) * p(C_1) \rightarrow \text{判為 } C_0$$

反之

$$p(F_1 | C_0)p(F_2 | C_0)p(F_3 | C_0) * p(C_0) > p(F_1 | C_1)p(F_2 | C_1)p(F_3 | C_1) * p(C_1) \rightarrow \text{判為 } C_1$$

範例

以下文件集(corpus)有8個例句2個類別('stmt' , 'question')我們以樸素貝葉斯判斷 'What is the price of the book' 是' stmt' 還是' question'

Out[97]:

	sent	class
0	This is my book	stmt
1	They are novels	stmt
2	have you read this book	question
3	who is the author	question
4	what are the characters	question
5	This is how I bought the book	stmt
6	I like fictions	stmt
7	what is your favorite book	question

要判斷：'What is the price of this book?' 是 question 還是 stmt，就是去比較

$p('question' | 'what is the price of this book')$ 和 $p('stmt' | 'what is the price of this book')$

也就是

$p('what is the price of this book' | 'question')^*p('question')$ 和
 $P('what is the price of this book' | 'stmt')^*p('stmt')$

的大小

樸素貝葉斯轉換

$$\begin{aligned} P(\text{What is the price of the book}|\text{Stmt}) &= P(\text{What}|\text{Stmt}) \times P(\text{is}|\text{Stmt}) \times P(\text{the}|\text{Stmt}) \\ &\quad \times P(\text{price}|\text{Stmt}) \times P(\text{of}|\text{Stmt}) P(\text{the}|\text{Stmt}) \\ &\quad \times P(\text{book}|\text{Stmt}) \end{aligned}$$

$$\begin{aligned} P(\text{What is the price of the book}|\text{Question}) &= P(\text{What}|\text{Question}) \times P(\text{is}|\text{Question}) \times P(\text{the}|\text{Question}) \\ &\quad \times P(\text{price}|\text{Question}) \times P(\text{of}|\text{Question}) P(\text{the}|\text{Question}) \\ &\quad \times P(\text{book}|\text{Question}) \end{aligned}$$

Out[97]:

	sent	class
0	This is my book	stmt
1	They are novels	stmt
2	have you read this book	question
3	who is the author	question
4	what are the characters	question
5	This is how I bought the book	stmt
6	I like fictions	stmt
7	what is your favorite book	question

$$P(\text{Stmt}) = \frac{\text{Number of sentences in Stmt Class}}{\text{Total number of sentences in the training set}} = \frac{4}{8} = 0.5$$

$$P(\text{Question}) = \frac{\text{Number of sentences in Question Class}}{\text{Total number of sentences in the training set}} = \frac{4}{8} = 0.5$$

範例

在stmt條件下的每個字數

```
Out[31]: {'are': 1,
          'book': 2,
          'bought': 1,
          'fictions': 1,
          'how': 1,
          'is': 2,
          'like': 1,
          'my': 1,
          'novels': 1,
          'the': 1,
          'they': 1,
          'this': 2}
```

在question條件下的每個字數

```
Out[39]: {'are': 1,
          'author': 1,
          'book': 2,
          'characters': 1,
          'favorite': 1,
          'have': 1,
          'is': 2,
          'read': 1,
          'the': 2,
          'this': 1,
          'what': 2,
          'who': 1,
          'you': 1,
          'your': 1}
```

Out[97]:

	sent	class
0	This is my book	stmt
1	They are novels	stmt
2	have you read this book	question
3	who is the author	question
4	what are the characters	question
5	This is how I bought the book	stmt
6	I like fictions	stmt
7	what is your favorite book	question

每個單字在stmt class 下的條件機率

```
{'are': 0.0666666666666667,  
'book': 0.1333333333333333,  
'bought': 0.0666666666666667,  
'fictions': 0.0666666666666667,  
'how': 0.0666666666666667,  
'is': 0.1333333333333333,  
'like': 0.0666666666666667,  
'my': 0.0666666666666667,  
'novels': 0.0666666666666667,  
'the': 0.0666666666666667,  
'they': 0.0666666666666667,  
'this': 0.1333333333333333}
```

每個單字在question class 下的條件機率

```
{'are': 0.0555555555555555,  
'author': 0.0555555555555555,  
'book': 0.1111111111111111,  
'characters': 0.0555555555555555,  
'favorite': 0.0555555555555555,  
'have': 0.0555555555555555,  
'is': 0.1111111111111111,  
'read': 0.0555555555555555,  
'the': 0.1111111111111111,  
'this': 0.0555555555555555,  
'what': 0.1111111111111111,  
'who': 0.0555555555555555,  
'you': 0.0555555555555555,  
'your': 0.0555555555555555}
```

Laplace Smoothing:

樸素貝葉斯利用了新的句子在不同條件下的機率相乘之結果比大小。但如果新句中有字的機率為0，整個相乘結果變0，就沒法比大小了

所以要引進一個近似法叫做Laplace Smoothing來調整最後的 $p(\text{word}|\text{stmt})$ 或 $p(\text{word}|\text{question})$

$$P(\text{word}|\text{stmt or question}) = \frac{\text{該字出現次數} + 1}{\text{該條件總字數} + \text{文件集獨特字總數}}$$

範例：

計算 $P(\text{Stmt}|\text{What is the price of the book})$:

$$\begin{aligned} P(\text{What is the price of the book}|\text{Stmt}) &= P(\text{What}|\text{Stmt}) \times P(\text{is}|\text{Stmt}) \times P(\text{the}|\text{Stmt}) \\ &\quad \times P(\text{price}|\text{Stmt}) \times P(\text{of}|\text{Stmt}) P(\text{the}|\text{Stmt}) \\ &\quad \times P(\text{book}|\text{Stmt}) \end{aligned}$$

Laplace Smoothing 之後的條件機率

```
{'book': 0.08333333333333333,  
 'is': 0.0833333333333333,  
 'of': 0.02777777777777776,  
 'price': 0.02777777777777776,  
 'the': 0.0555555555555555,  
 'what': 0.02777777777777776}
```

$$\begin{aligned} P(\text{what is the price of the book}|\text{stmt}) &= 0.0277 * 0.0833 * 0.0555 * 0.0277 * 0.0277 * \\ &0.0555 * 0.0833 = 4.5939365799778324e-10 \end{aligned}$$

$$\begin{aligned} P(\text{Stmt}|\text{What is the price of the book}) &= P(\text{what is the price of the book}|\text{Stmt}) * P(\text{Stmt}) \\ &= 4.5939365799778324e-10 * 0.5 = 2.2969682899889162e-10 \end{aligned}$$

範例：

計算 $P(\text{Question}|\text{What is the price of the book})$:

$$\begin{aligned} P(\text{What is the price of the book}|\text{Question}) &= P(\text{What}|\text{Question}) \times P(\text{is}|\text{Question}) \times P(\text{the}|\text{Question}) \\ &\quad \times P(\text{price}|\text{Question}) \times P(\text{of}|\text{Question}) \times P(\text{the}|\text{Question}) \\ &\quad \times P(\text{book}|\text{Question}) \end{aligned}$$

Laplace Smoothing 之後的條件機率

Out[40]: {'book': 0.07692307692307693,
'is': 0.07692307692307693,
'of': 0.02564102564102564,
'price': 0.02564102564102564,
'the': 0.07692307692307693,
'what': 0.07692307692307693}

$$\begin{aligned} P(\text{what is the price of the book}|\text{question}) &= 0.0769 * 0.0769 * 0.0769 * 0.0256 * 0.0256 \\ &\quad * 0.0769 * 0.0769 = 1.7624289971722582e-09 \end{aligned}$$

$$\begin{aligned} P(\text{Question}|\text{What is the price of the book}) &= P(\text{what is the price of the book}|\text{Question}) * P(\text{Question}) \\ &= 1.7624289971722582e-09 * 0.5 = \\ &= 8.812144985861291e-10 > 2.2969682899889162e-10 \quad \text{It is a question} \end{aligned}$$

Module 6. 樸素貝葉斯 Naïve Bayes

學習目標：

- 認識Naïve Bayes 在離散型連續型特徵使用法

實作 – 使用Naïve Bayes 建立分類模型

sklearn.naive_bayes.MultinomialNB (離散型特徵)

$$p(\text{book}|\text{stmt}) = (2+1)/15+21) = 0.0833$$

Sklearn.naive_bayes.GaussianNB (連續型常態分布特徵)

$$p(\text{feature1} | \text{class1}) = N(\text{feature1} | \mu, \sigma)$$

Out[97]:

	sent	class
0	This is my book	stmt
1	They are novels	stmt
2	have you read this book	question
3	who is the author	question
4	what are the characters	question
5	This is how I bought the book	stmt
6	I like fictions	stmt
7	what is your favorite book	question

Module 7. 類神經網路

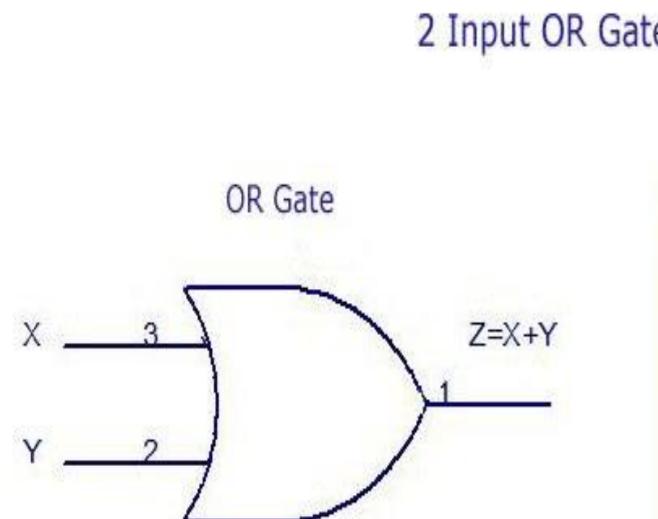
類神經網路介紹

學習目標：

- 認識Perceptron
- 認識類神經網路

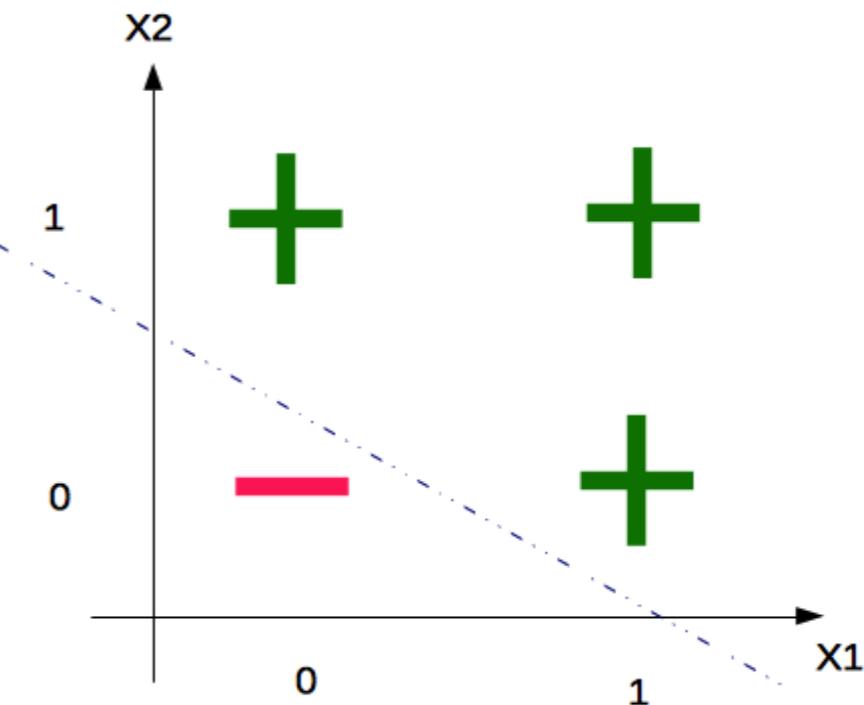
線性神經元的數學定義

起先用線性神經元來模擬邏輯閘，實現邏輯電路功能
以下是OR電路的真值表和座標圖示

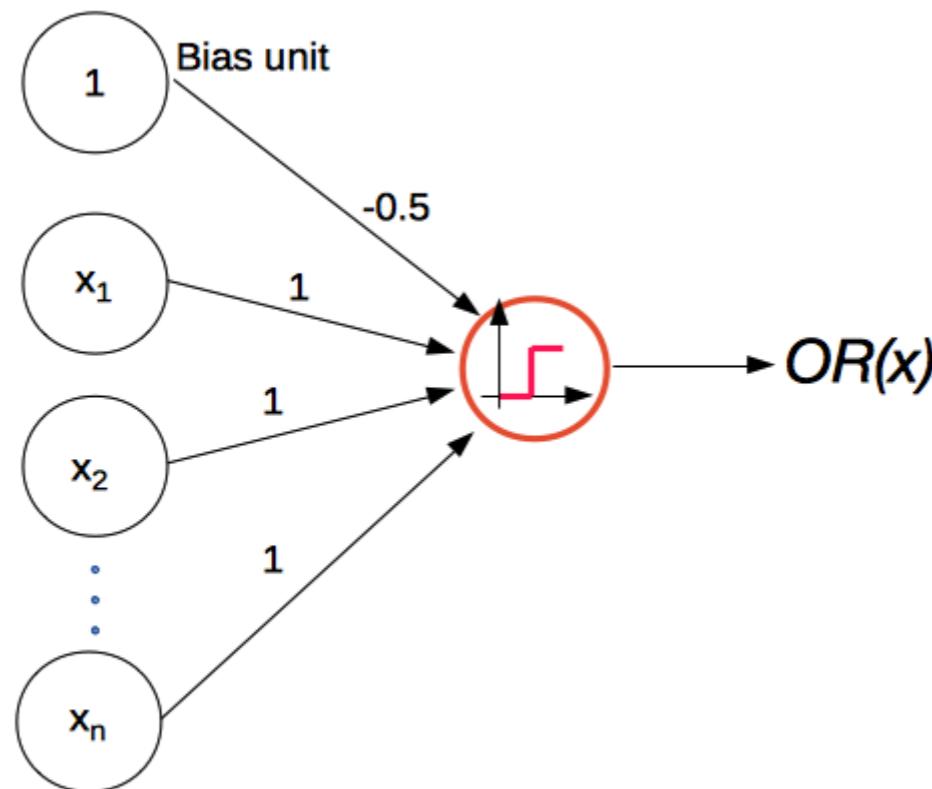


TRUTH TABLE

INPUTS		OUTPUT
X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1



線性神經元形成OR 閘的電路示意與Python程式



```
def forwardOR(operands):
    m = len(operands) # number of inputs
    ORweights = [-0.5] + [1]*m // hard-coded
    return booleanActivation(operands,
                            ORweights)
```

數學式：

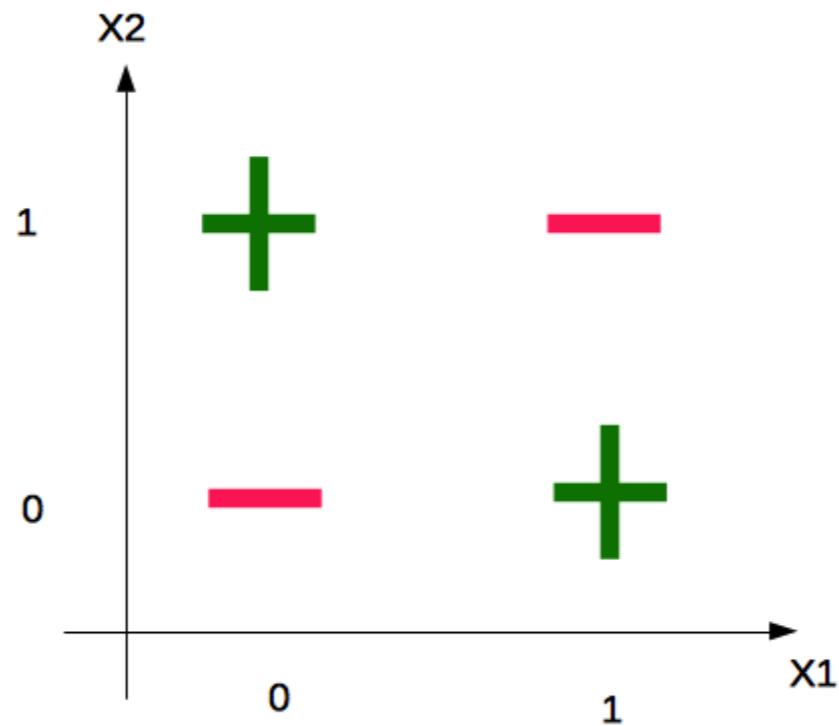
If $x_1 \cdot w_1 + x_2 \cdot w_2 + b < \text{threshold}$ then 0

If $x_1 \cdot w_1 + x_2 \cdot w_2 + b > \text{threshold}$ then 1

<https://mashimo.wordpress.com>

線性神經元的數學定義

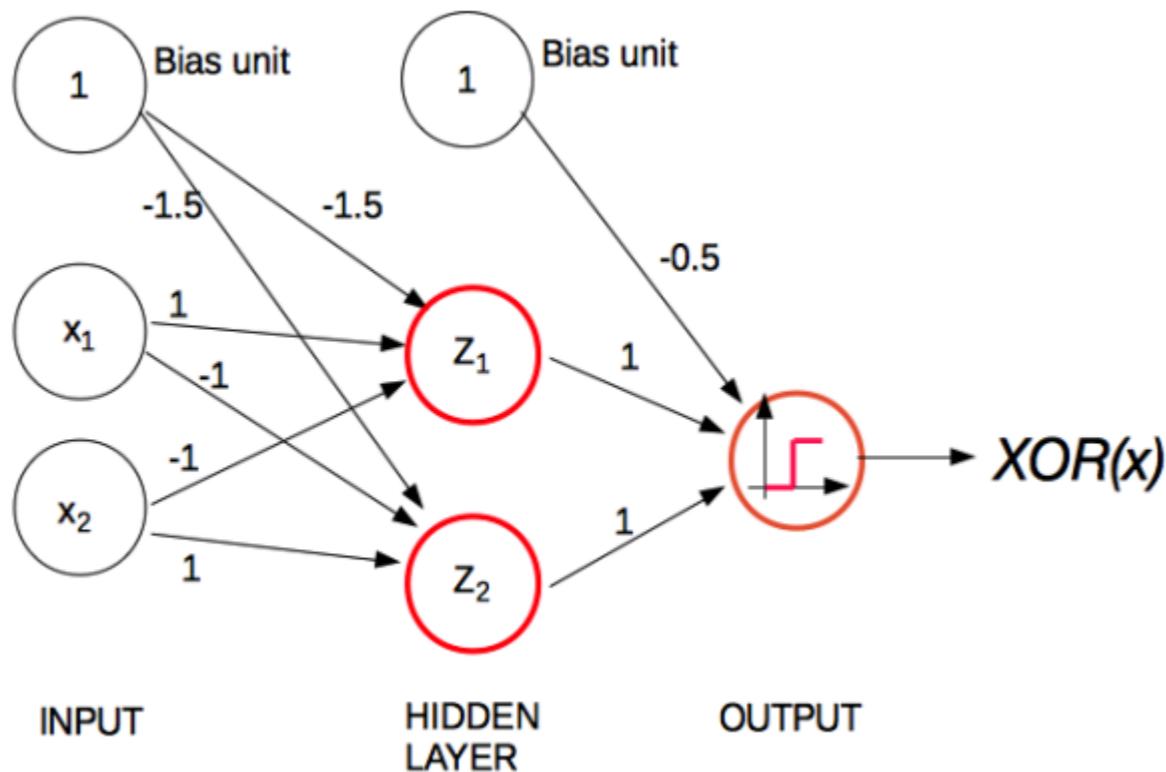
但是對非線性的閘，像是XOR，就無法以單層線性神經元表達



線性神經元

這時就必須要用多層邏輯閘完成

$$x_1 \text{ XOR } x_2 = (x_1 \text{ AND NOT } x_2) \text{ OR } (\text{NOT } x_1 \text{ AND } x_2)$$



```

def forwardXOR(x1,x2):
    # this is a network combining existing neurons,
    # no weights to prepare
    x1not = forwardNOT([x1])
    x2not = forwardNOT([x2])
    z1 = forwardAND([x1, x2not])
    z2 = forwardAND([x1not, x2])

    return forwardOR([z1,z2])

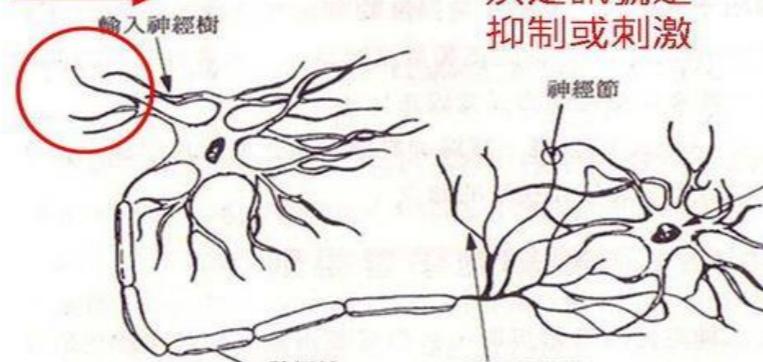
print("*** Simulation of XOR network of neurons")
print("X1 | X2 | Y = X1 XOR X2")
print("-----")
for x1 in (0,1):
    for x2 in (0,1):
        print(x1, " | ", x2, " | ", forwardXOR(x1,x2))
    
```

終於發現，需要非線性功能的神經元解決或學習更複雜的問題

神經元 V.S. 人工神經元

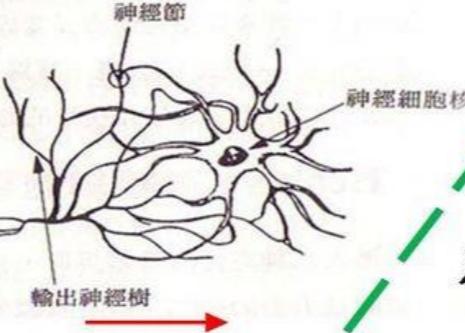
神經樹

由其它神經元經由
神經節輸入訊號



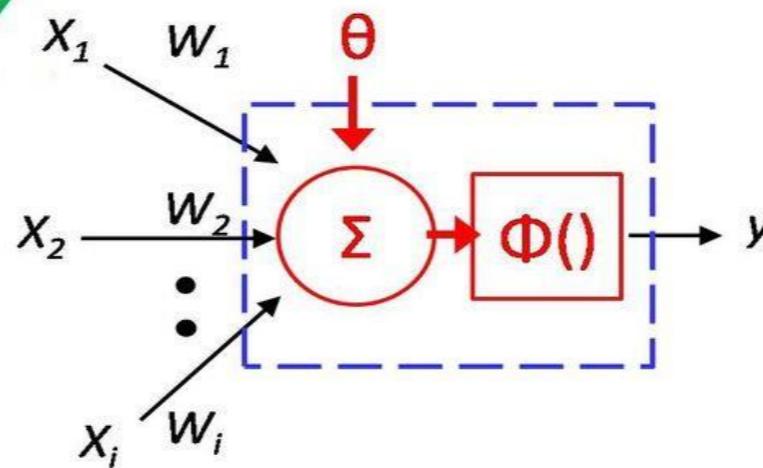
神經節

決定訊號是
抑制或刺激

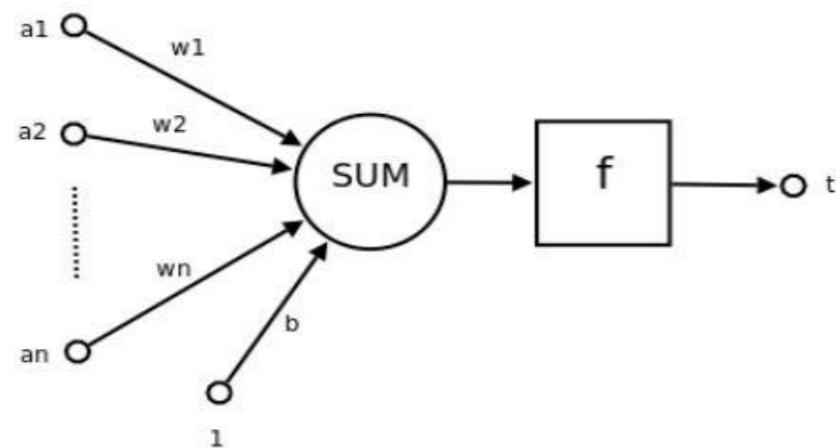


Ex. 嬰兒學走、中風患者

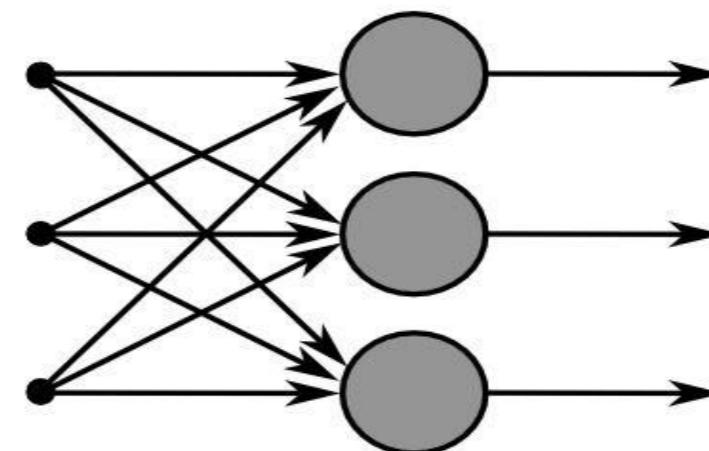
x_i : 訊號輸入
 w_i : Weight
 θ : Bias
 $\Phi()$: Active Function
 y : 結果輸出



下圖這個 f 叫做激活函數，將神經元從線性帶到到神經網路的領域



(a) 單一神經元的模型



(b) 單層神經網路

Module 7. 類神經網路

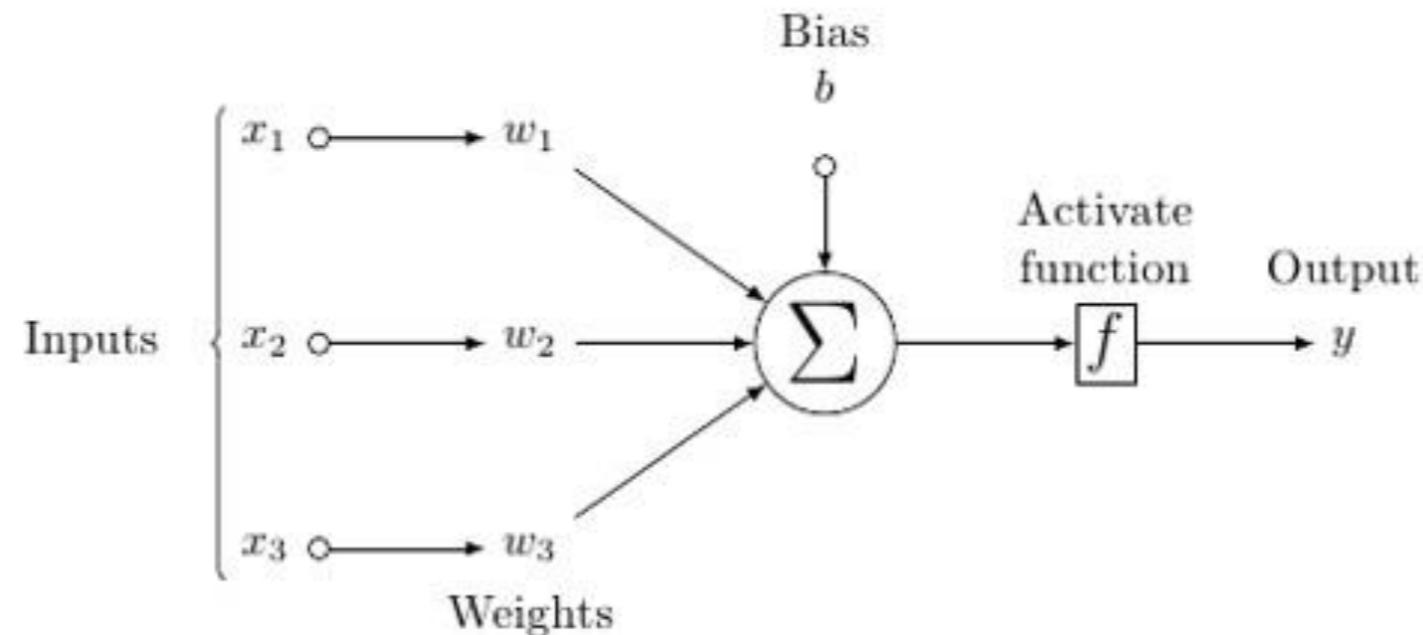
Activation Function 介紹

學習目標：

- 認識激活函數

Activation Function

何謂激活函數：通常是一個非線性的函數，因為非線性，所以能賦予神經網路模型更容易學習



$$Y = f(x_1 * w_1 + x_2 * w_2 + x_3 * w_3 \dots + b)$$

矩陣式

$$\hat{Y} = f(\mathbf{W}^T * \mathbf{X} + \mathbf{b})$$

激活函數 (Activation Functions)

激活函數使得神經元非線性，完成分類群聚等功能，比方 Sigmoid() 和 Tanh()，但由於它的存在，以下問題必須克服：

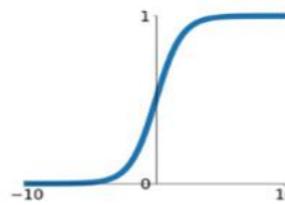
- ▶ 標準化 (Standardization) 問題。
- ▶ 消失梯度 (Vanishing Gradient) 問題。

Activation Function

激活函數 (Activation Functions) : 消失的梯度與標準化

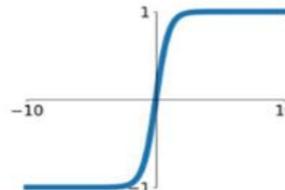
Sigmoid

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



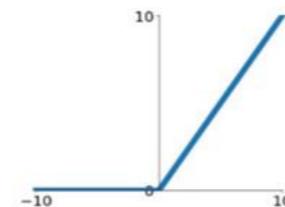
tanh

$$\tanh(x)$$



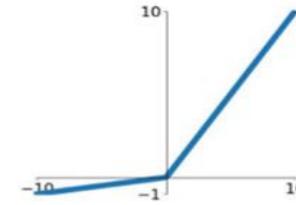
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

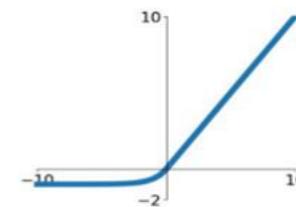


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



Module 7. 類神經網路

實作 – 使用感知器建 立分類模型

學習目標：

- 認識感知器的
sklearn API

sklearn.linear_model.Perceptron : Perceptron

Parameters: (參數)

penalty : 正則化方法{ 'l2' , 'l1' , 'elasticnet' }, default=None

tol : 停止計算的判準 , float, default=1e-3

Attributes(屬性)

classes_ : 多少類別

coef_ : 各項特徵係數

Intercept_ : 截距

loss_function:損失函數

Module 8. 集成學習

集成學習簡介
(Bagging,
Boosting,
Blending)

學習目標：

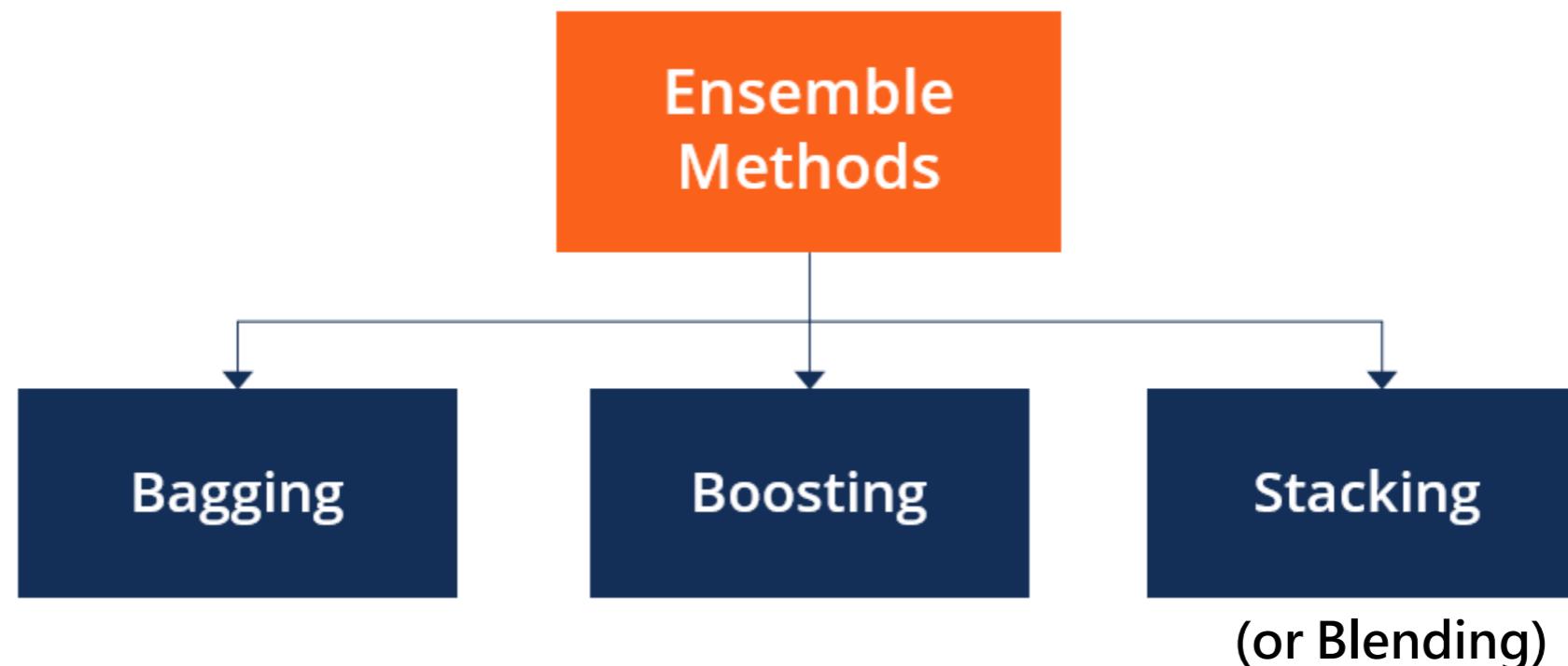
- 認識集成學習
- 認識Bagging,
Boosting, Blending)

- 什麼是集成
 - 集成是使用不同方式，結合多個 / 多種不同分類器，作為綜合預測的做法統稱
 - 將模型截長補短，也可說是機器學習裡的和議制 / 多數決
- 集成的分類
 - 以集合的時間點分類
 - 以結合的方式分類

	Blending	Aggregation Learning
	aggregation after getting g_t	aggregation as well as getting g_t
Uniform Combination	Averaging	Bagging
Linear Combination	Weighted	AdaBoost GradientBoost
Non-linear Combination	Stack	Decision Tree

集成(Ensemble)的基本精神就是集眾分類器的力量，達到單一分類器無法達成的成效

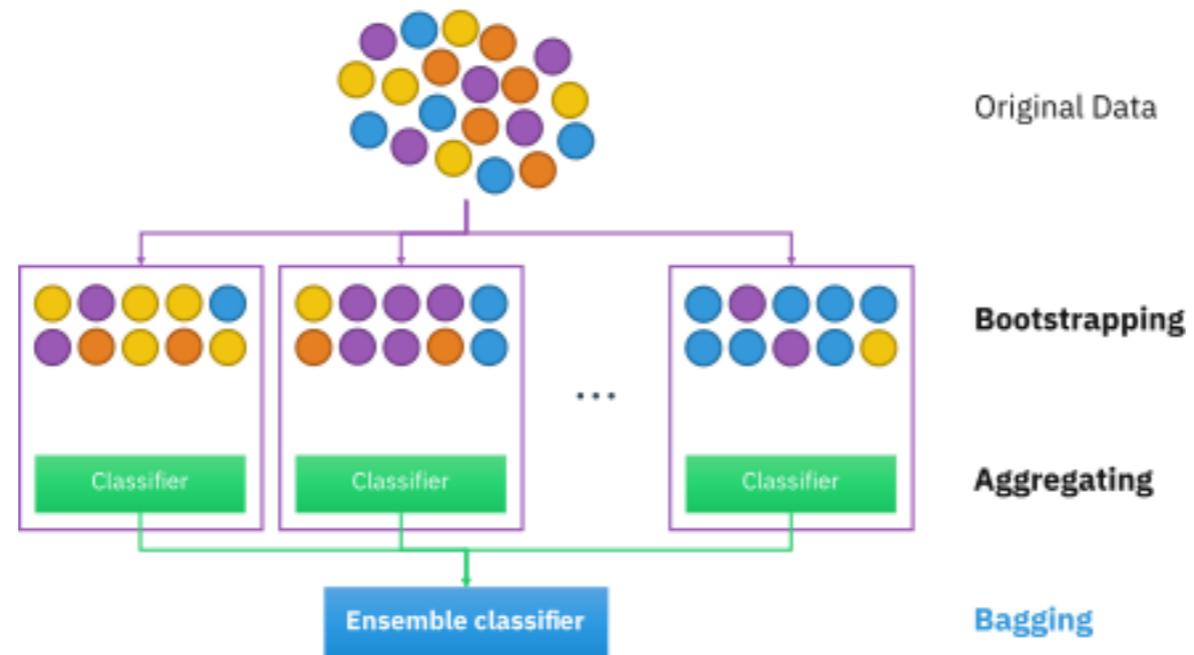
集成可大分為以下三類



Bagging

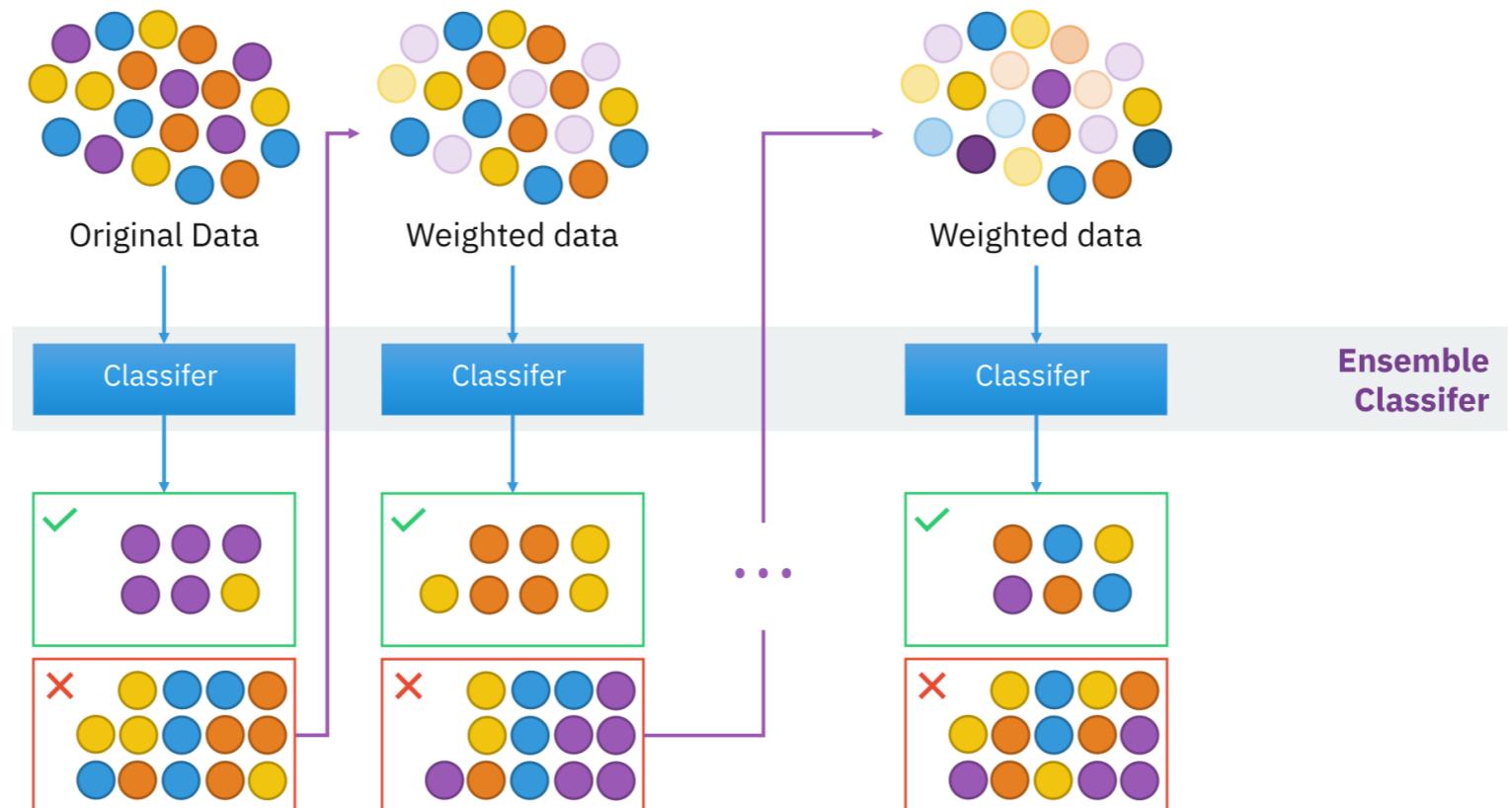
Bagging其實是 Bootstrap Aggregating的縮寫，也就是自舉聚合

資料分幾份，挑出的資料再擺回去供下一個訓練繼續使用，隨機森林以Bagging採樣即為一例



Boosting

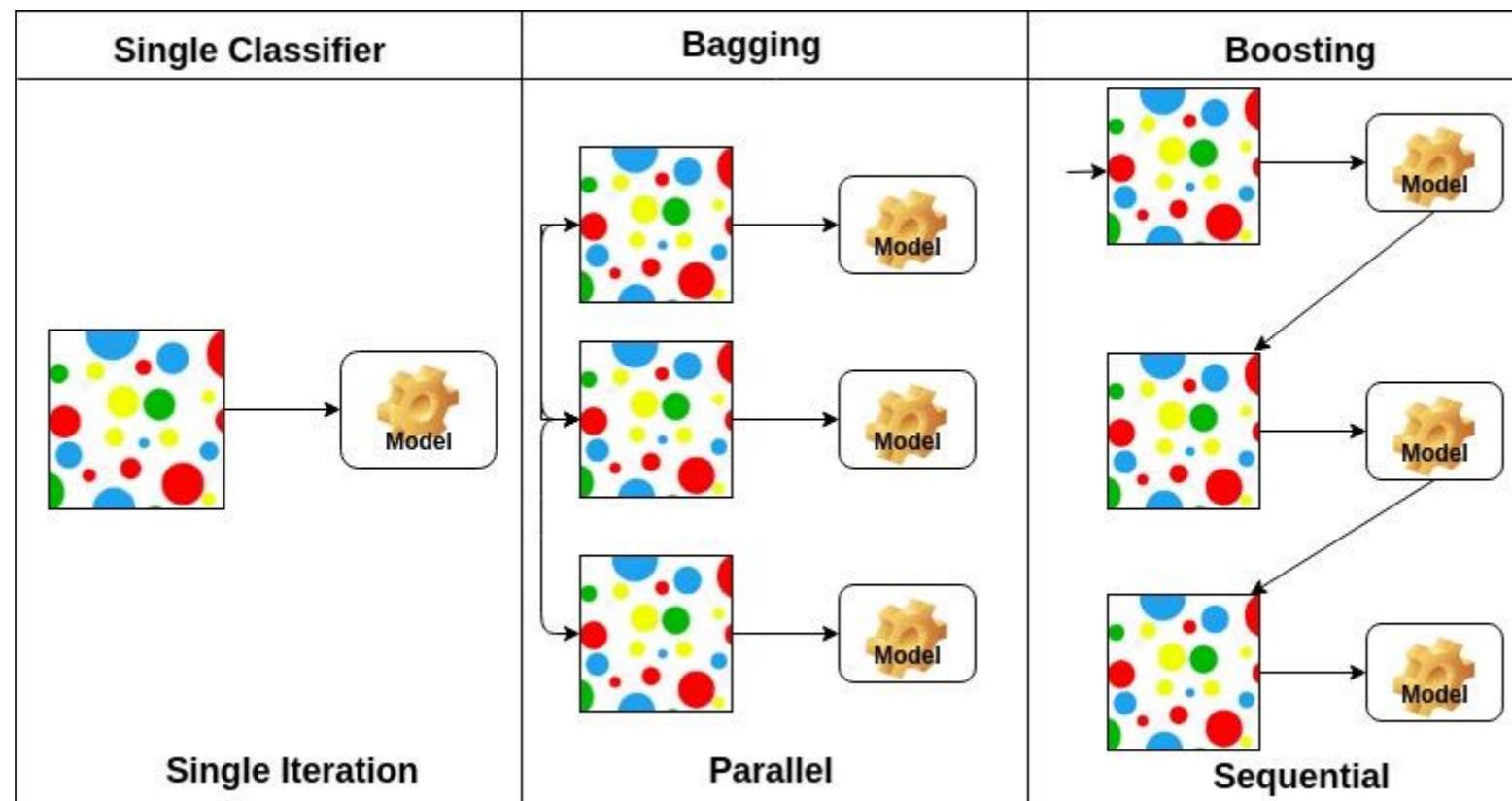
後面的模型，逐漸補救前面分類模型的弱點
前模型預測錯誤結果改變後來模型資料集的權重繼續預測
整個模型逐漸完善，最終的預測結果是前後各模型的加權平均



wikipedia.com

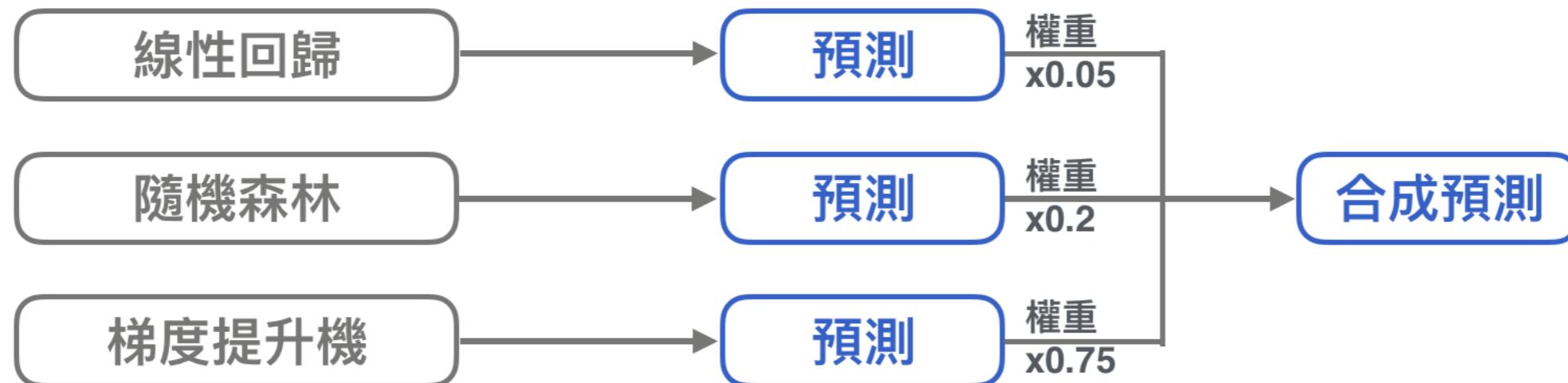
Ensemble

Bagging 和 Boosting 主要不同是時序的關係



Datacamp.com

- 混合泛化(Blending)是以不同的模型對相同的資料集個別做訓練與預測
- 產生的預測值當作特徵值
- 最後再由元模型(Meta Model) 根據產生的新特徵值做訓練和預測
- 混合泛化和堆疊泛化(Stacking)十分接近，不同之處僅是堆疊泛化元模型不用保留資料(holdout data)做訓練，而混合泛化依賴保留資料。



Module 8. 集成學習

實作 - Adaboost

學習目標：

- 認識 Adaboost
- 認識 Adaboost API

Adaboost演算法說明

- Adaboost是由一群相同的m個基本(弱)分類器所組成
- 基本分類器往往是單層決策樹(Decision Stumps)
- 前分類器m的失誤被計算
- m失誤的資料將在m+1時被加權
- 後分類器m+1根據前面被加權資料分類
- 最後預測結果是各分類器加權後的分類結果

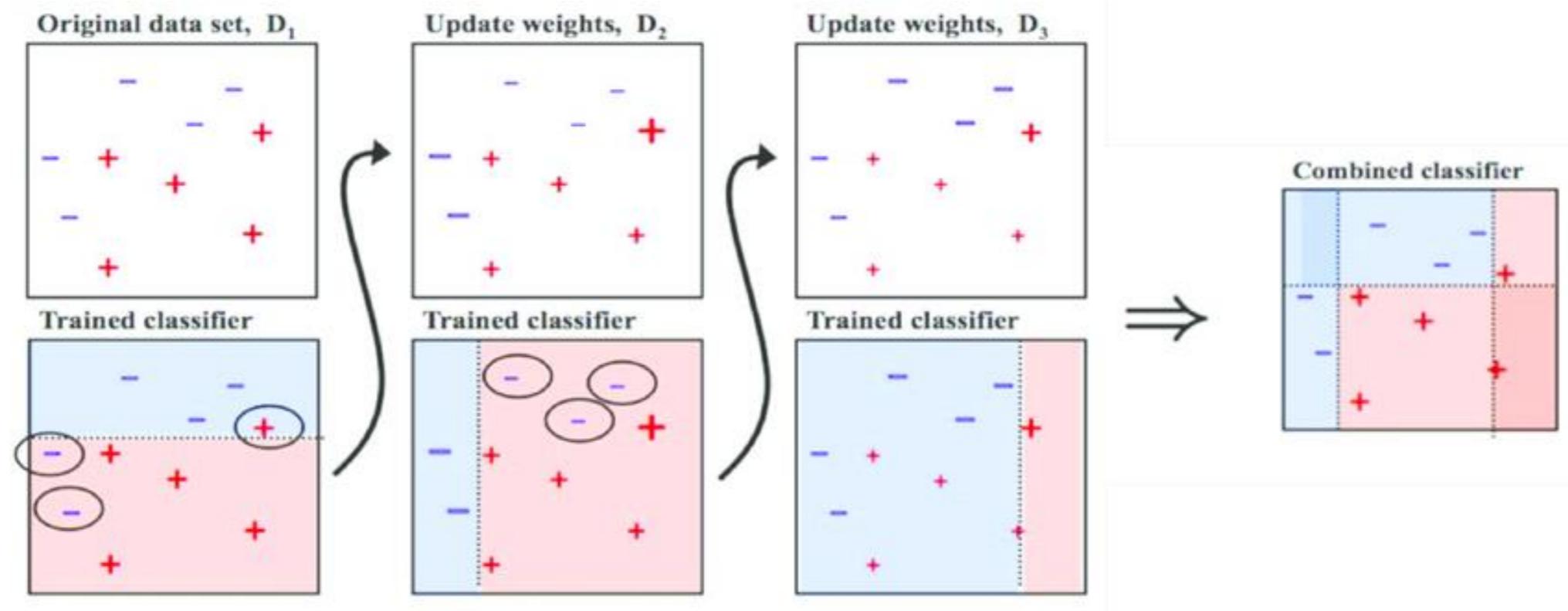
$$\epsilon_m = \sum_{i=1}^N W_{m,i} | h_m(X_i) \neq y_i$$

$$W_{m+1,i} = \frac{1}{Z_{m,i}} w_{m,i} * e^{I_m * \alpha_m}$$

(Z:正規化因子，I=+-1)

$$\alpha_m = \frac{1}{2} \ln \left(\frac{1-\epsilon_m}{\epsilon_m} \right)$$

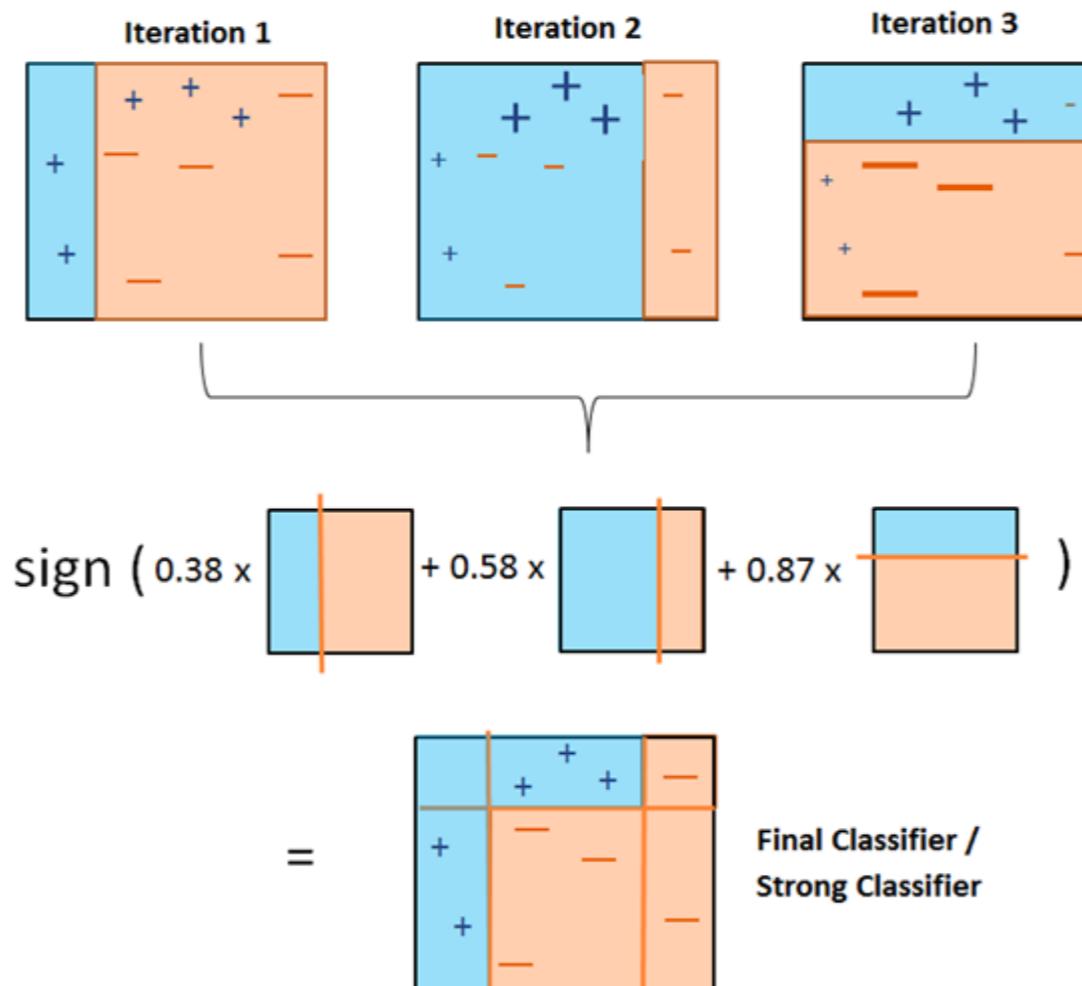
運算至指定的迭代數，形成合併的分類器



Adaboost

合併之Adaboost分類器，是根據各分類器之 α 加權計算

AdaBoost Classifier Working Principle with Decision Stump as a Base Classifier



sklearn.ensemble.AdaBoostClassifier : AdaBoost

- 其基礎分類器 : DecisionTreeClassifier (max_depth=1)
- 常用參數：
 - n_estimators \Rightarrow 決策樹的數量

Module 8. 集成學習

實作 – Blending Model

學習目標：

- 了解如何實作
Blending

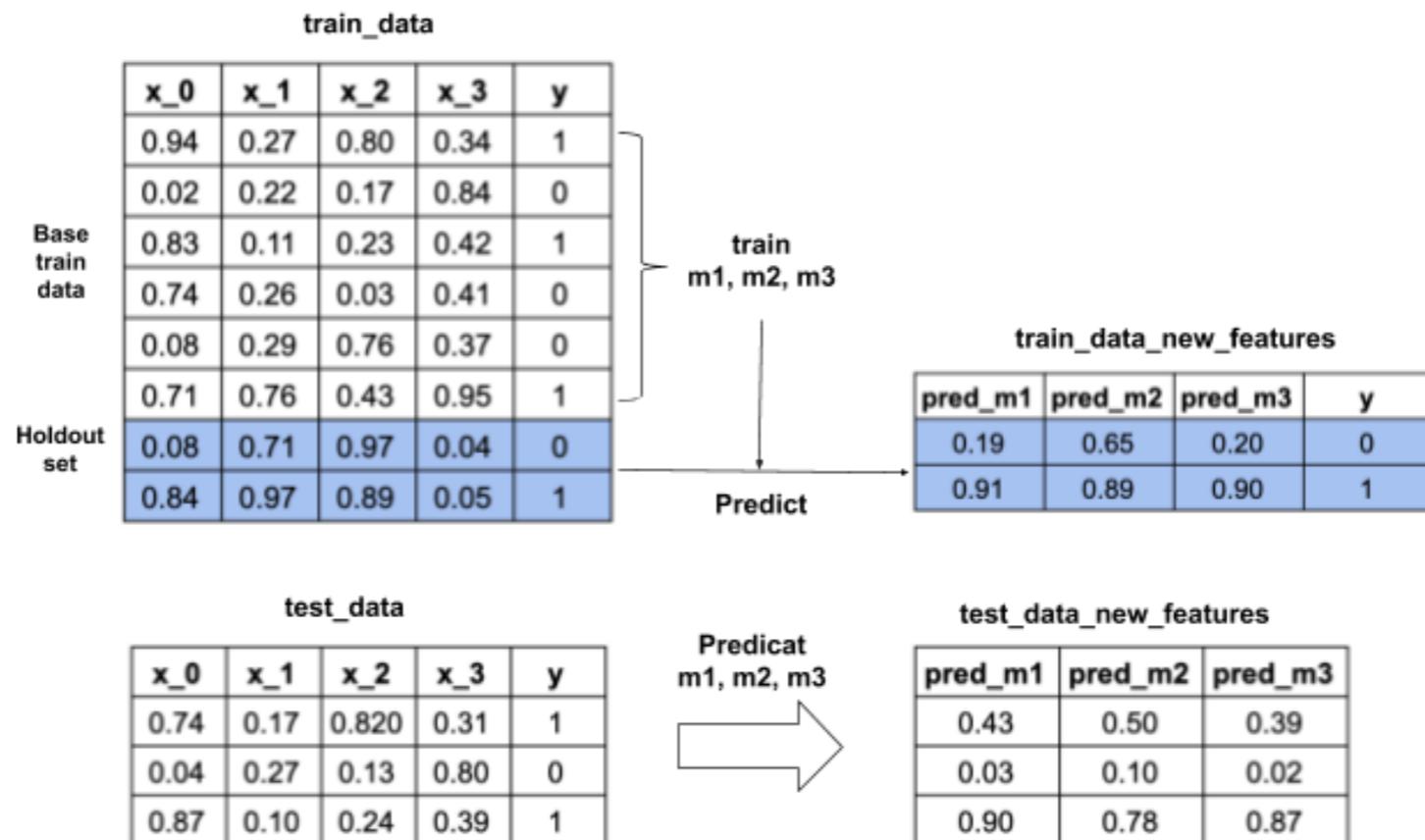
步驟1: 將訓練資料分割為基本訓練資料和保存資料(Holdout Data)

	train_data					test_data				
	x_0	x_1	x_2	x_3	y	x_0	x_1	x_2	x_3	y
base_train_data	0.94	0.27	0.80	0.34	1	0.74	0.17	0.820	0.31	1
	0.02	0.22	0.17	0.84	0	0.04	0.27	0.13	0.80	0
	0.83	0.11	0.23	0.42	1	0.87	0.10	0.24	0.39	1
	0.74	0.26	0.03	0.41	0					
	0.08	0.29	0.76	0.37	0					
	0.71	0.76	0.43	0.95	1					
	0.08	0.71	0.97	0.04	0					
	0.84	0.97	0.89	0.05	1					

medium.com

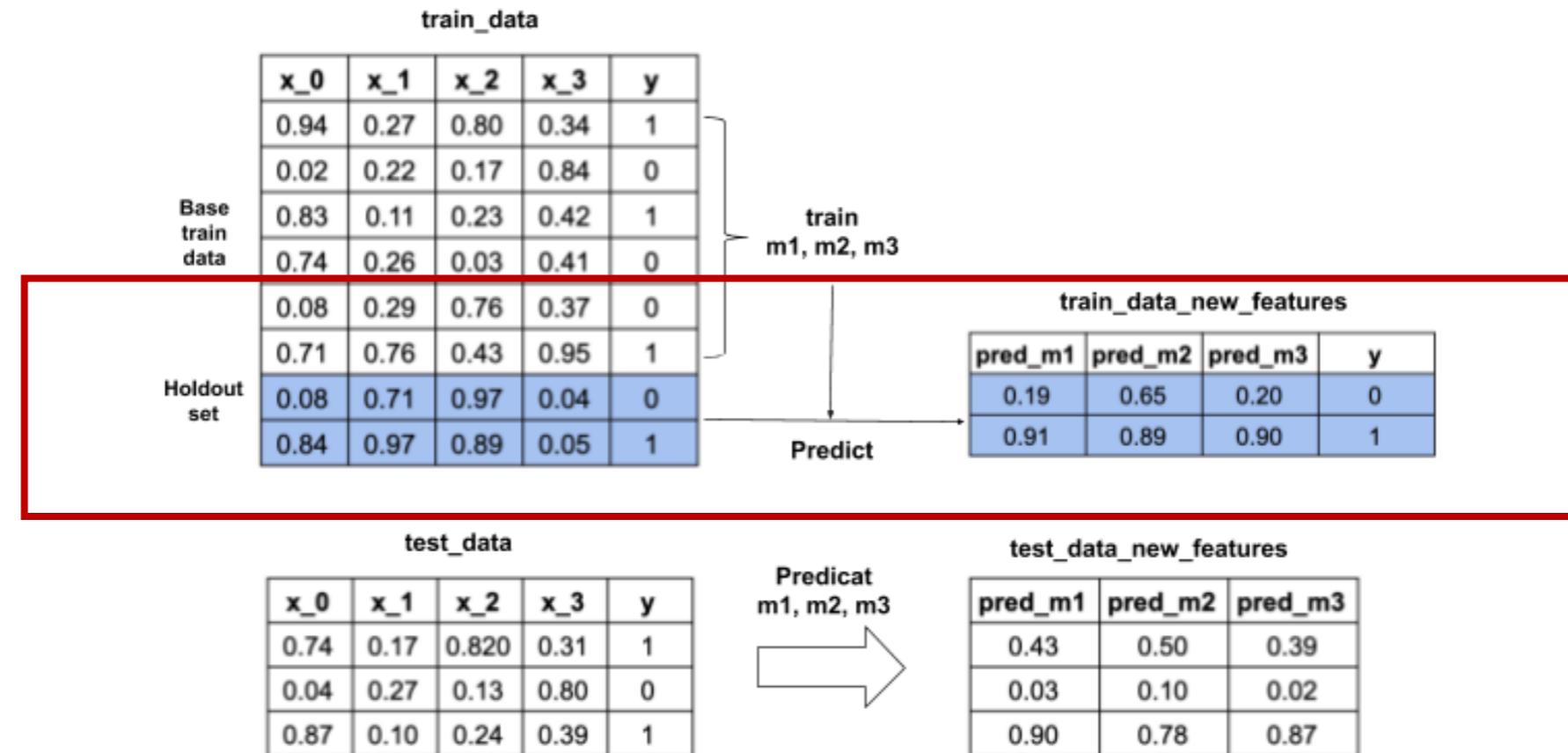
Blending

步驟 2: 用基本訓練資料訓練基本模型, 然後對保存資料與測試資料做預測. 此時將產生新的預測特徵值

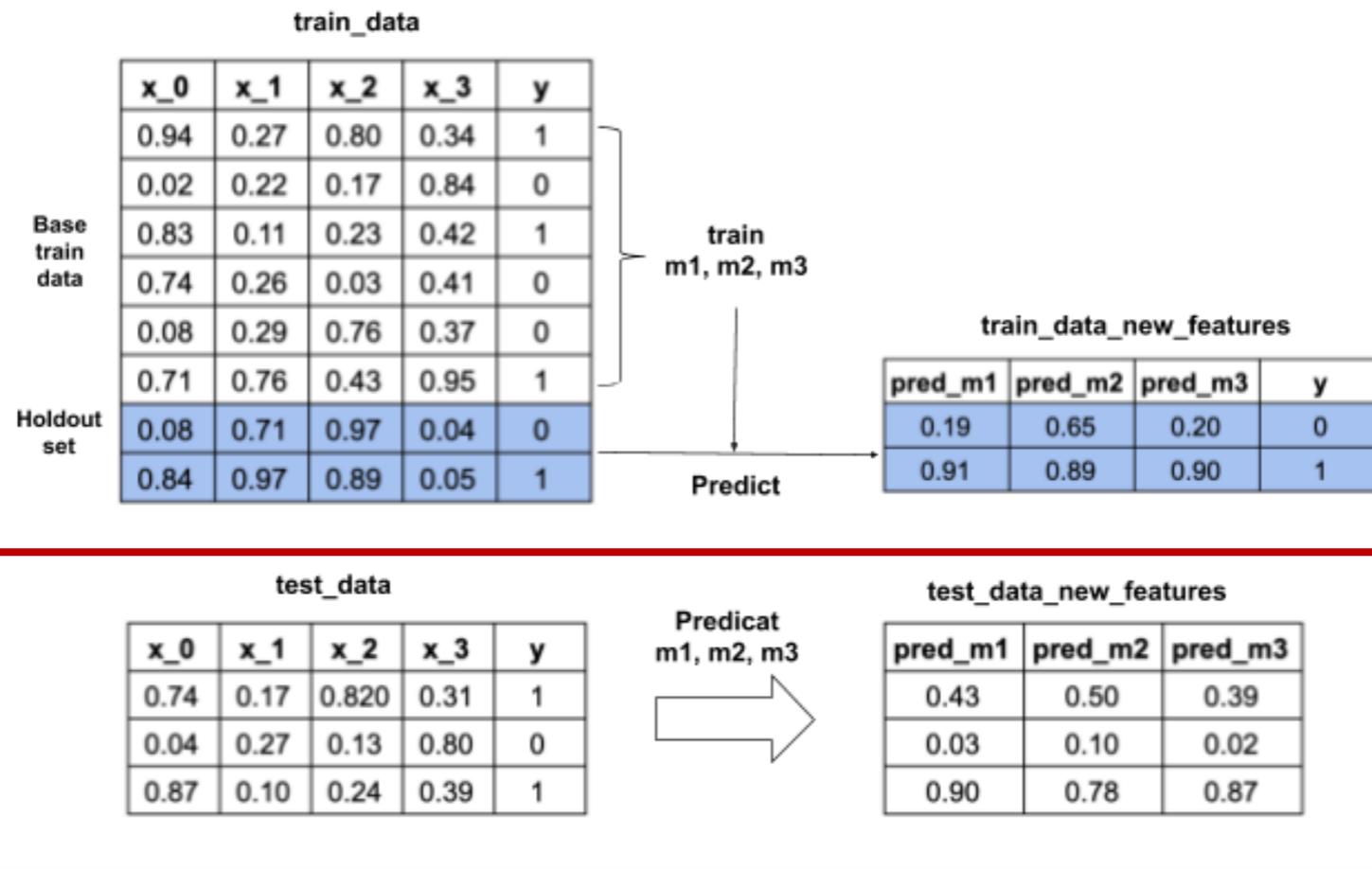


Blending

步驟 3: 利用保存資料新產生的特徵值來訓練一個新的元模型，
原始的保存資料和新產生的保存資料特徵值都將被使用。



步驟 4：將被訓練好的元模型拿來對測試資料，使用原始和新的元特徵值做最後的預測



Module 9.

分類問題案例實作

– 經典安隆案預測模型

案例分析

學習目標：

- 了解案例背景

安隆公司曾是一間能源公司，2001年破產前是世界上最大的電力、天然氣及電信公司之一。這間擁有上千億資產的公司竟然在短短幾周內宣告破產，才揭露其財報在多年以來均是造假的醜聞。在本資料集中你將會扮演偵探的角色，透過高層經理人內部的email來往的情報以及薪資、股票等財務特徵，訓練出一個機器學習模型來幫忙你找到可疑的詐欺犯罪者！

參考：他們靠著作假帳，撐起一間600億美元的公司，最後卻一夕崩塌！
安隆事件始末。

財務相關

salary 95 non-null float64
bonus 82 non-null float64
long_term_incentive 66 non-null float64
deferred_income 49 non-null float64
deferral_payments 39 non-null float64
loan_advances 4 non-null float64
other 93 non-null float64
expenses 95 non-null float64
director_fees 17 non-null float64
total_payments 125 non-null float64
exercised_stock_options 102 non-null float64
restricted_stock 110 non-null float64
restricted_stock_deferred 18 non-null float64
total_stock_value 126 non-null float64

通信相關

to_messages 86 non-null float64
from_messages 86 non-null float64
from_poi_to_this_person 86 non-null float64
from_this_person_to_poi 86 non-null float64
shared_receipt_with_poi 86 non-null float64

標記

poi 146 non-null bool

Module 9.

分類問題案例實作

– 經典安隆案預測模型

模型建立

學習目標：

- 複習之前單元的分類模型

在此之前學過的所有分類模型都可以用：

- 決策樹/隨機森林 (decision tree/random forest)
- 羅吉斯回歸 (logistic regression)
- 樸素貝葉斯 (naïve bayes)
- 感知線性神經元 (perceptron)
- 支持向量機器 (support vector machine)
- 集成預測 (ensemble)

sklearn.tree.DecisionTreeClassifier

sklearn.ensemble.RandomForestClassifier

sklearn.linear_model.LogisticRegression

sklearn.svm.SVC

sklearn.naive_bayes.GaussianNB

sklearn.linear_model.Perceptron

sklearn.ensemble.AdaBoostClassifier

Module 9.

分類問題案例實作

– 經典安隆案預測模型

模型評估與比較

學習目標：

- 了解 recall , precision

各種評估 Metrics

CONFUSION MATRIX		ACTUAL	
PREDICTED	True Positive (TP)	False Positive (FP)	
	False Negative (FN)	True Negative (TN)	

$$\text{Precision} = \frac{TP}{TP+FP}$$

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN}$$

$$\text{F1-Score} = \frac{2*Precision*Recall}{Precision+Recall}$$

$$\text{Recall} = \frac{TP}{TP+FN}$$

勿枉勿縱：Precision 和 Recall

預測\真實	有罪	清白
有罪	TP: 抓出詐欺員工	FP: 冤枉好人
清白	TN: 放過詐欺員工	FN: 證明清白

Precision: $\frac{TP}{(TP+FP)}$ 愈接近1 FP就愈小，不冤枉好人

Recall: $\frac{TP}{(TP+FN)}$ 愈接近1 FN就愈小，不放過詐欺員工

分數	Sklearn API
混淆矩陣	<code>sklearn.metrics.confusion_matrix(Y, y_pred)</code>
Precision	<code>sklearn.metrics.precision_score(Y, y_pred)</code>
Recall	<code>sklearn.metrics.recall_score(Y, y_pred)</code>

Module 10. 階層式分群

資料分群簡介，
如何計算樣本間距離

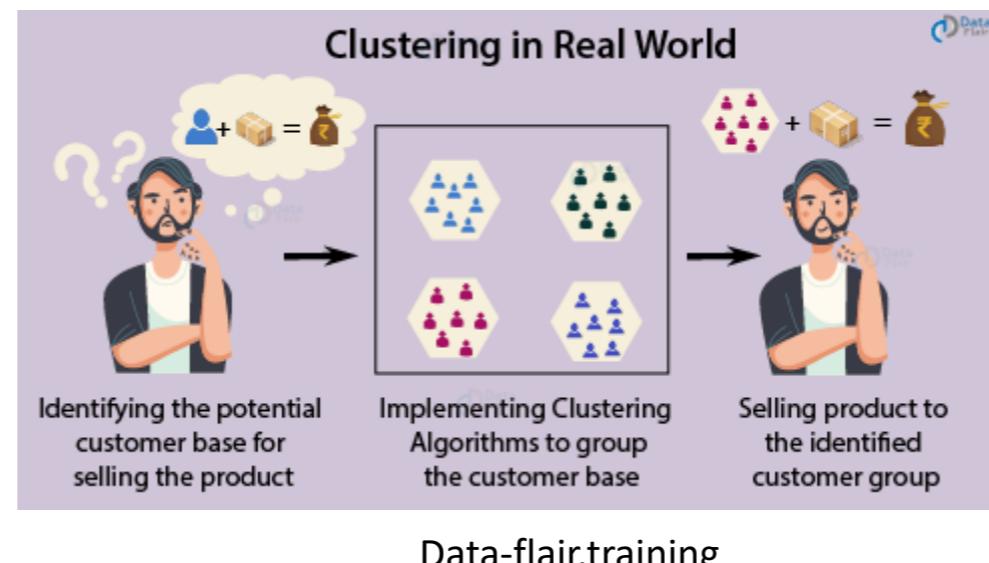
學習目標：

- 資料分群簡介
- 認識距離功氏

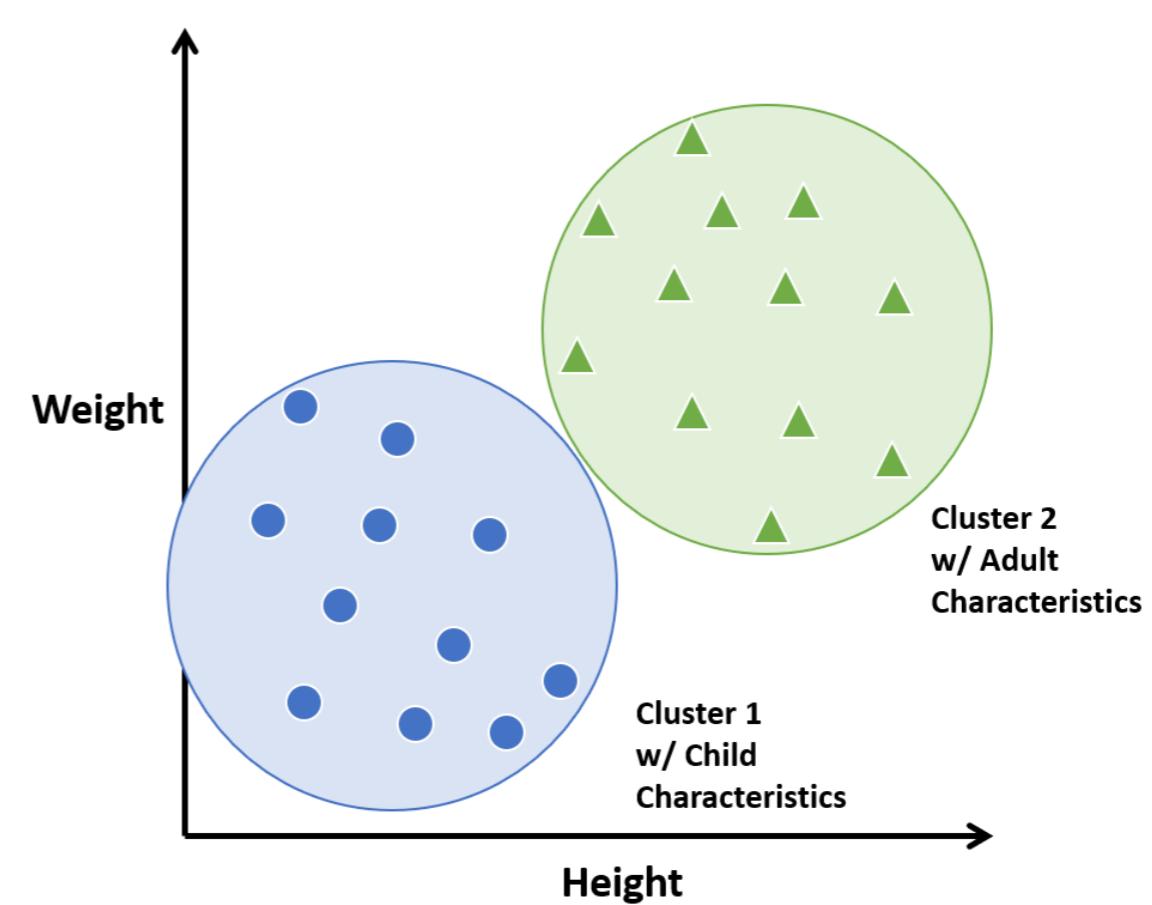
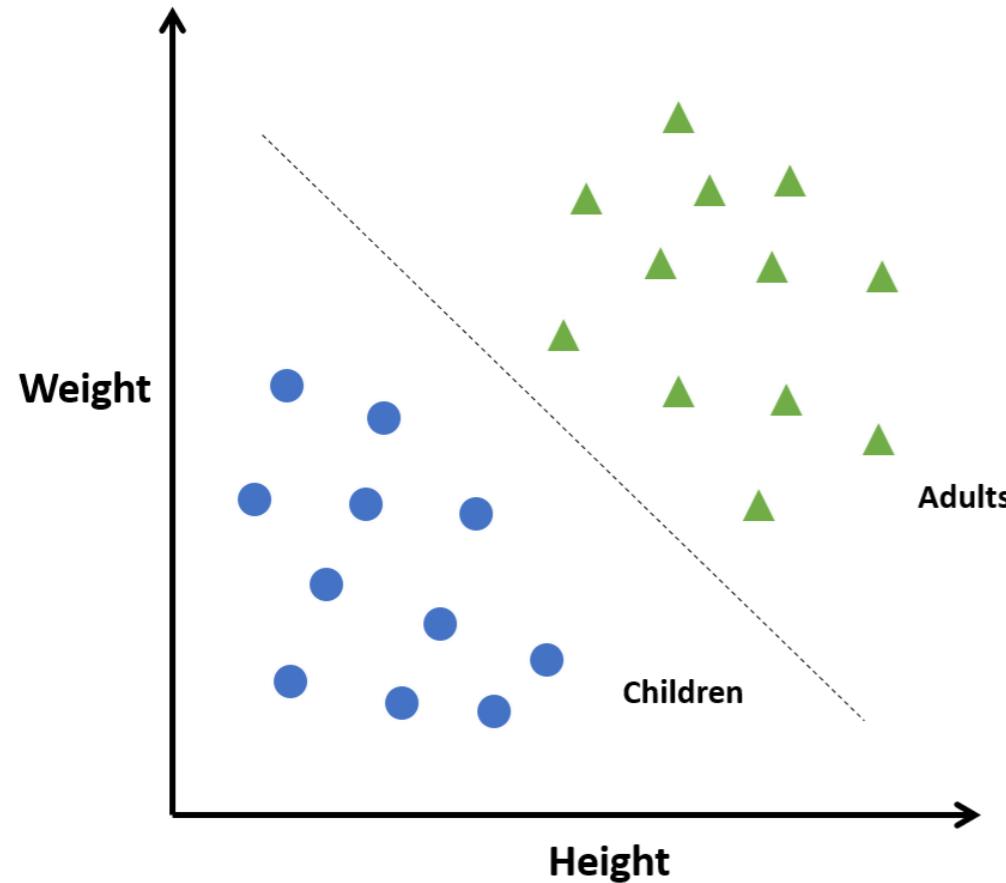
資料分群是一種非監督式學習的演算法，在無標籤情形下自動將資料依相似度結群

- 目標是群內儘可能相似，群間儘可能相異
- 如何定義相似度？將資料點映射到幾合座標再求距離，距離即相似度
- 分群演算法：有 DBSCAN，K-mean，Hierarchical 等

資料分群應用：晶圓瑕疵診斷，客群分析，輿情分析，行為分析，體適能分群等。



Classification vs Clustering



medium.com

Euclidean distance (歐基里德距離)

- $d(i, j) = \sqrt{|x_{i1} - x_{j1}|^2 + |x_{i2} - x_{j2}|^2 + \cdots + |x_{in} - x_{jn}|^2}$

Manhattan (city block) distance (曼哈頓距離)

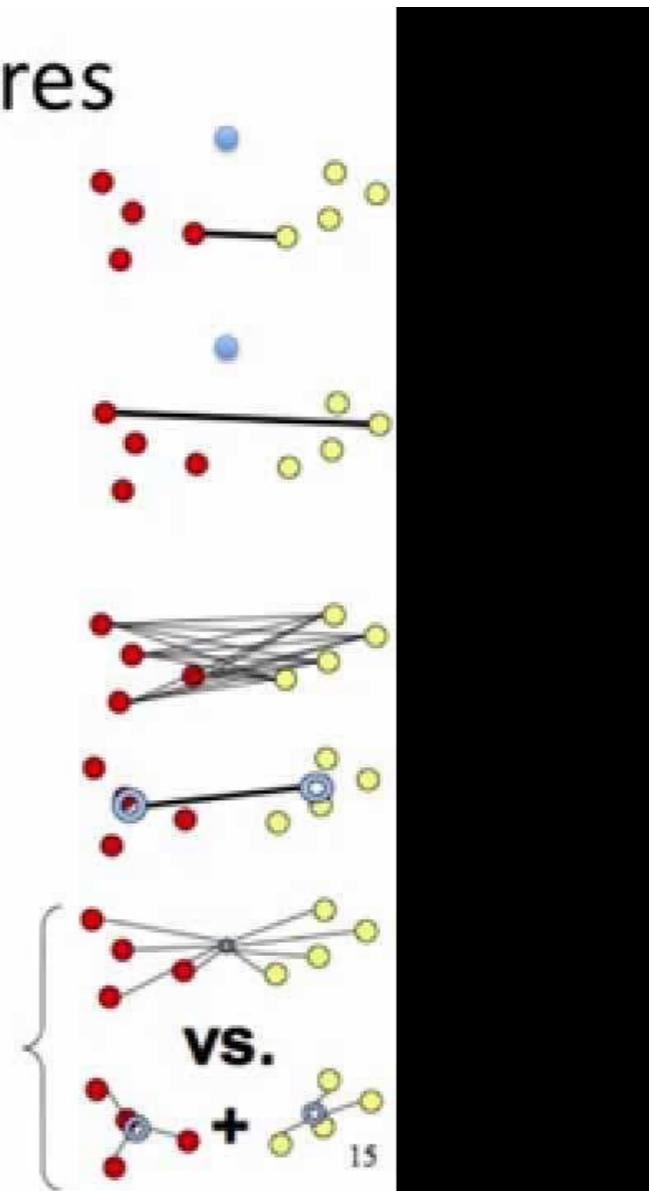
- $d(i, j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \cdots + |x_{in} - x_{jn}|$

Weighted Manhattan distance (加權曼哈頓距離)

- $d(i, j) = \sqrt[q]{w1|x_{i1} - x_{j1}|^q + w2|x_{i2} - x_{j2}|^q + \cdots + w3|x_{in} - x_{jn}|^q}$

Cluster distance measures

- Single link: $D(c_1, c_2) = \min_{x_1 \in c_1, x_2 \in c_2} D(x_1, x_2)$
 - distance between closest elements in clusters
 - produces long chains a→b→c→...→z
- Complete link: $D(c_1, c_2) = \max_{x_1 \in c_1, x_2 \in c_2} D(x_1, x_2)$
 - distance between farthest elements in clusters
 - forces "spherical" clusters with consistent "diameter"
- Average link: $D(c_1, c_2) = \frac{1}{|c_1| + |c_2|} \sum_{x_1 \in c_1} \sum_{x_2 \in c_2} D(x_1, x_2)$
 - average of all pairwise distances
 - less affected by outliers
- Centroids: $D(c_1, c_2) = D\left(\left(\frac{1}{|c_1|} \sum_{x \in c_1} \bar{x}\right), \left(\frac{1}{|c_2|} \sum_{x \in c_2} \bar{x}\right)\right)$
 - distance between centroids (means) of two clusters
- Ward's method: $TD_{c_1 \cup c_2} = \sum_{x \in c_1 \cup c_2} D(x, \mu_{c_1 \cup c_2})^2$
 - consider joining two clusters, how does it change the total distance (TD) from centroids?



Module 10.

階層式分群

階層式分群演算法

學習目標：

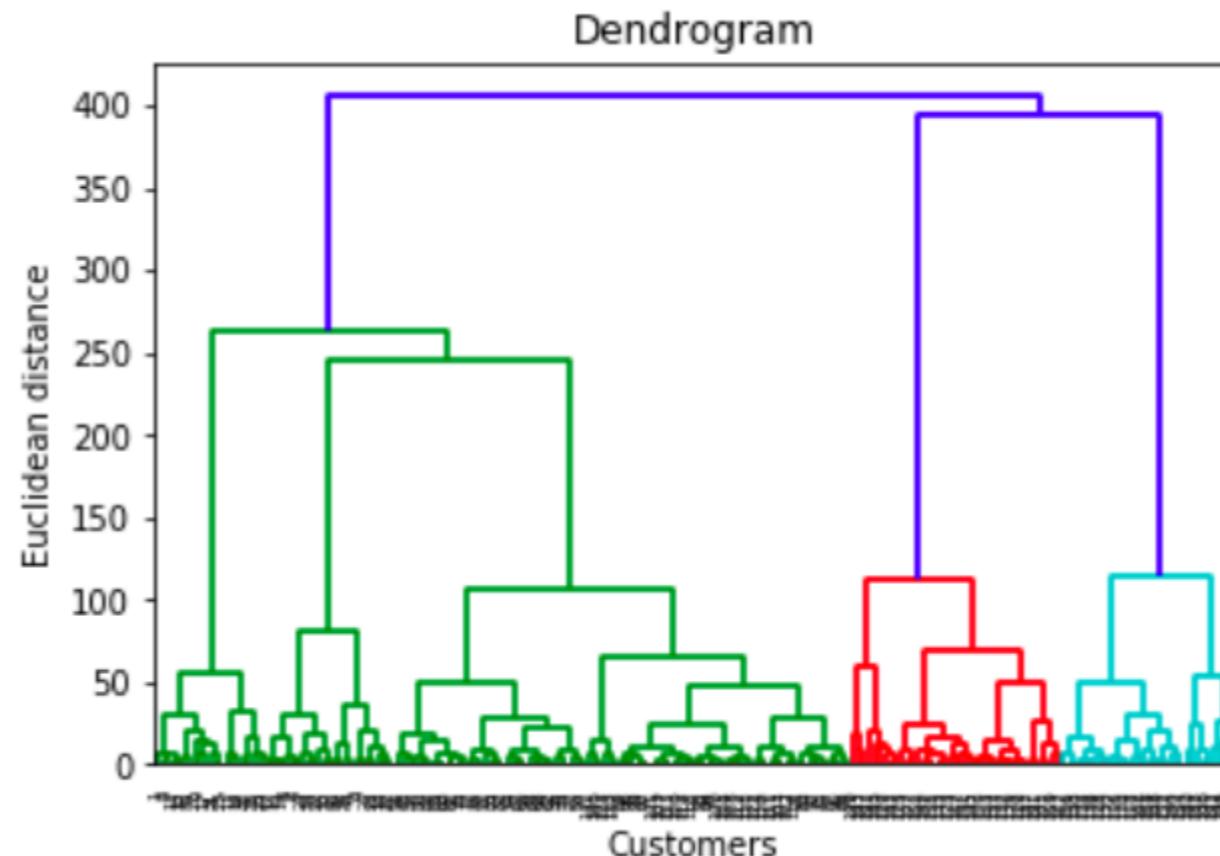
- 認識 Hierarchical Clustering

階層分群演算法流程

1. 每筆資料各自獨立，成為一個群組
2. 透過掃描過整個資料集尋找出最靠近的兩個群組。
3. 把這兩個點綁在一起變成一個群組
4. 重覆步驟2、3，直到所有資料合併成同一群組或是**達到參數設定值為止**。

如何由數據或圖表決定群？

- 在圖表中決定Y的值
- 運算時限制群間最大距離



- 優點
 - 1. 概念簡單，易於呈現
 - 2. 可以不用指定群數
- 缺點
 - 只適用於少量資料，大量資料會很難處理

Module 10. 階層式分群

實作 – 使用階層式分群將資料分群

學習目標：

- Hierarchical Clustering in Sklearn

`sklearn.cluster.AgglomerativeClustering()` 由下(小)而上(大)分群。由上(大)而下(小)叫Divisive，沒有API

重要參數：

決定要分幾群：`n_clusters` (default=2)

群間距離如何決定：`linkage` ('single','complete','average','ward')

群間最大距離：`distance_threshold`

回傳屬性：

每個資料點的所屬群標籤：`Label_`

Module 11.

分裂式分群

K-means 演算法

學習目標：

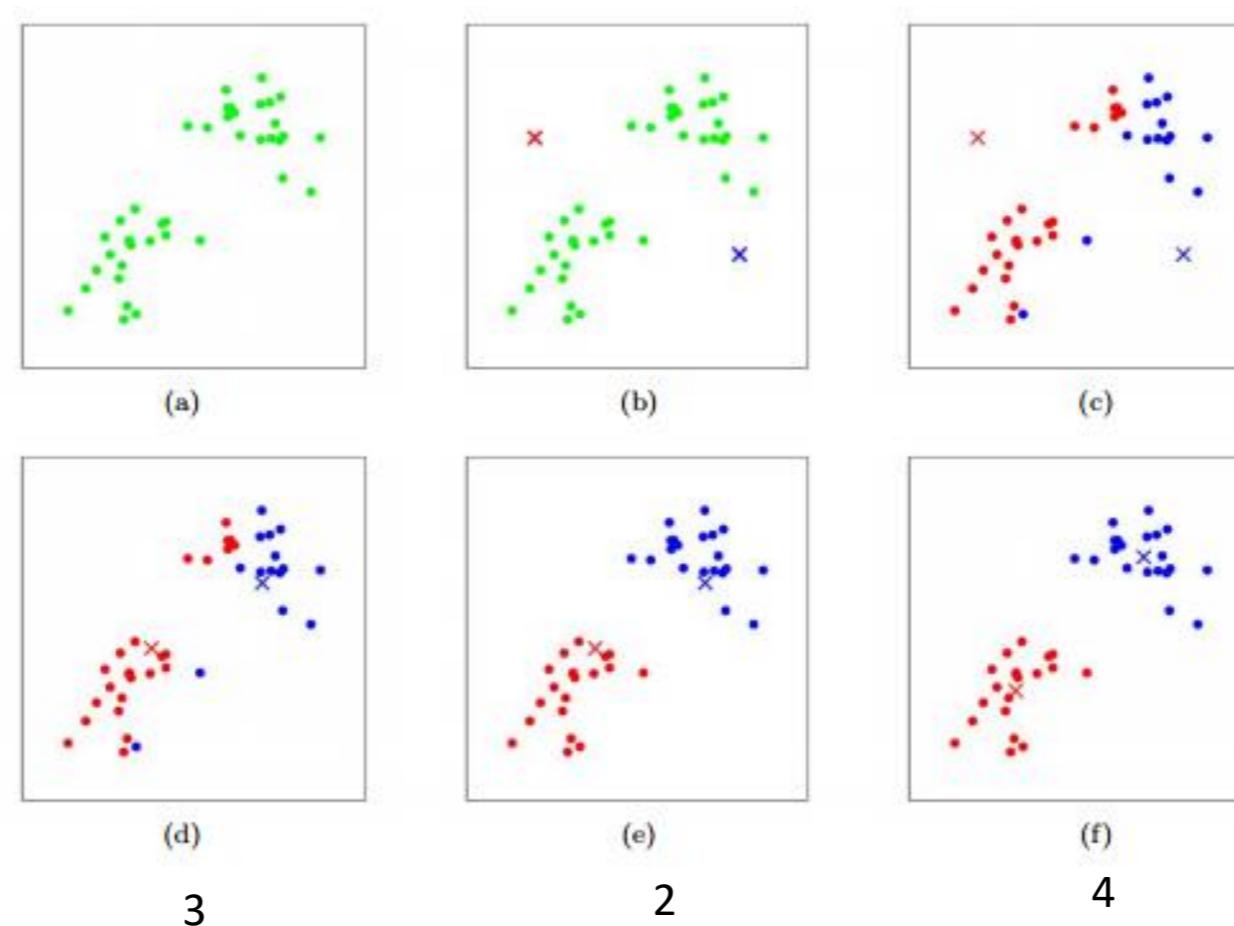
- 認識 K-means

K-means演算法的目標是使總體群內誤差的平方最小，達到分群的目的

1. 設定重心：隨機設定K個群組重心(Cluster Centroid)
2. 樣本入群：為每一個樣本找到距離最近的重心，形成第一迴圈的K群
3. 更新重心：在各群中，重新計算重心位置
4. 重複至結束：重複前面2,3步驟，直到所有樣本都不因重心移動而換組

K-means

步驟示範圖：



Stanford.edu的K-means動態演示

主要優點：

- 演算法簡單易行
- 適合大型資料集
- 保證收斂
- 運算時可調整重心

主要缺點：

- 必須事先決定群數K
- 初始值影響分群結果，造成非最佳解(後述)
- 不易處理群的大小和密度相差甚大的狀況
- 結果會被離散值影響

Module 11. 分裂式分群

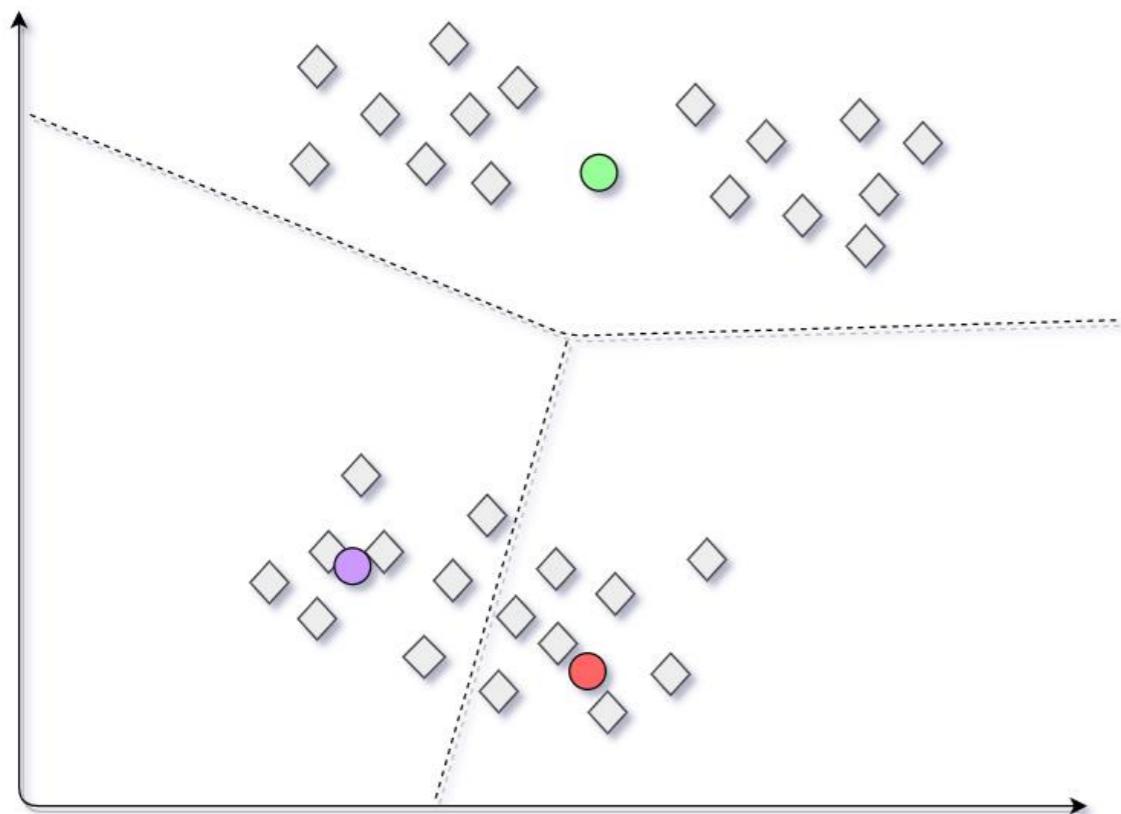
K-means 隨機初始化陷阱

學習目標：

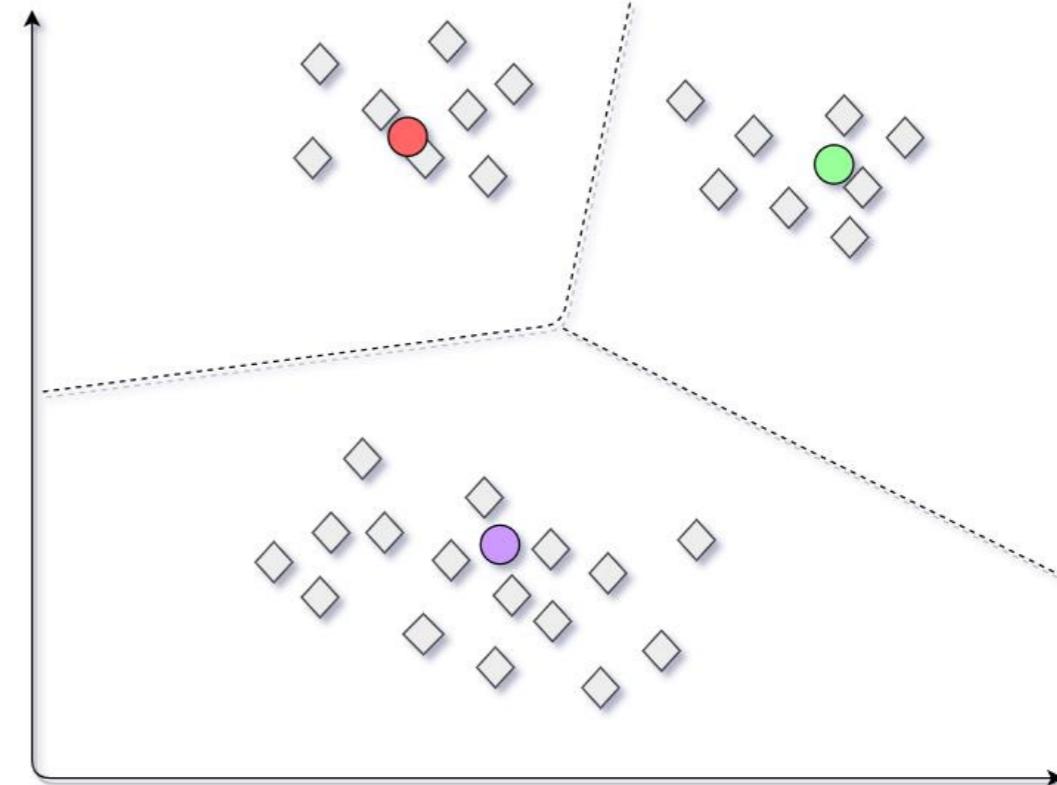
- 認識K-means的初始化問題

K-means 初始點選擇

K-means 隨機選取初始點當重心，會有不同結果



初始化 A

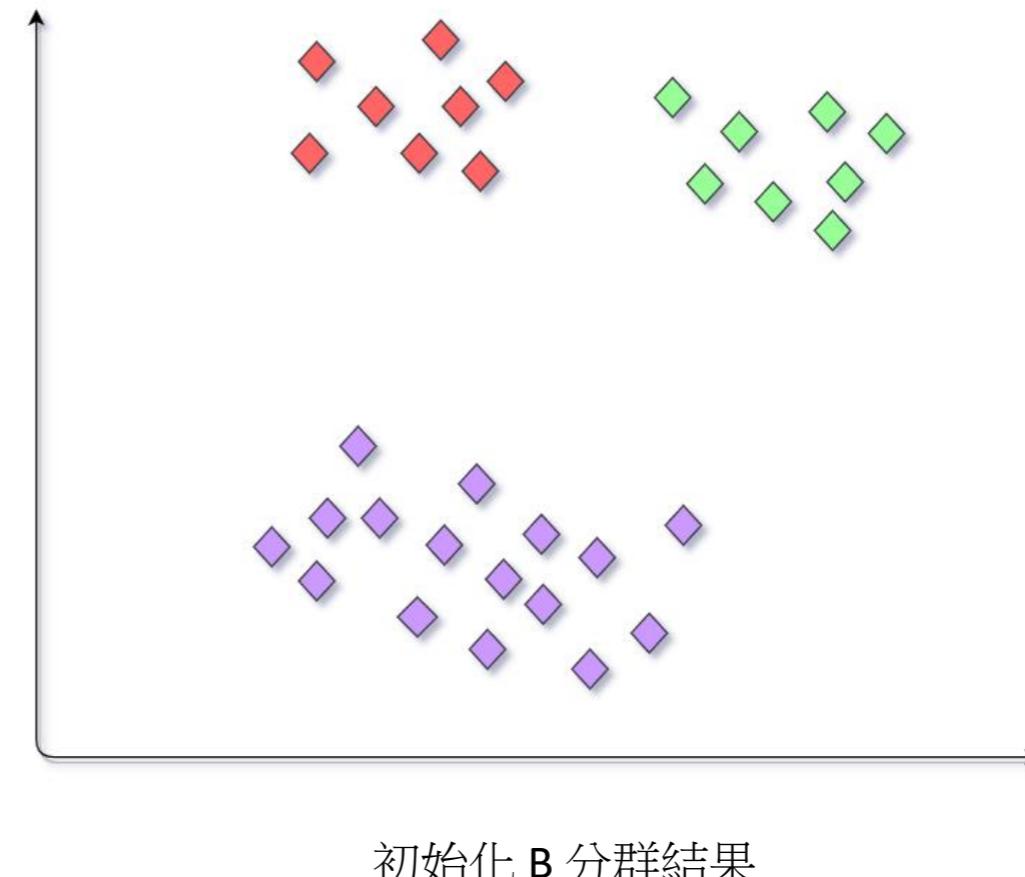
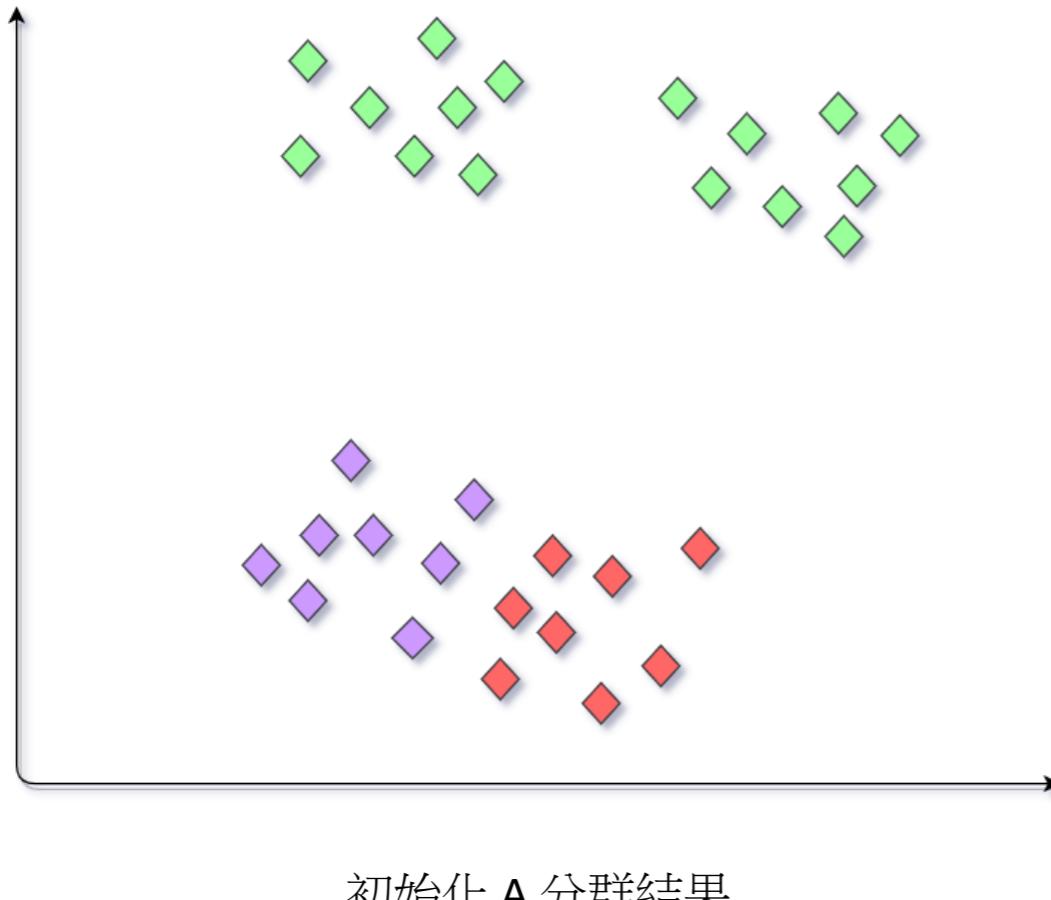


初始化 B

geekforgeeks.com

K-means 初始值陷阱

不同初始化造成在地優化(local optimum)而非全局優化



Module 11. 分裂式分群

實作 – 使用K-means
將資料分群

學習目標：

- K-means in Sklearn

Sklearn.cluster.Kmeans() K-means

主要參數

n_clusters : 事先定義的群數

random_state: 固定初始點的位置

主要屬性 :

Label_ : 分出的群及標籤

範例：

```
from sklearn.cluster import KMeans
import numpy as np
#載入X
x = np.array([[1, 2], [1, 4], [1, 0],
...             [10, 2], [10, 4], [10, 0]])
#k=2 k-mean演算法建模
kmeans = KMeans(n_clusters=2,
random_state=0).fit(X)
#分成k=2
kmeans.labels_
array([1, 1, 1, 0, 0, 0], dtype=int32)
#預測兩點
kmeans.predict([[0, 0], [12, 3]])
array([1, 0], dtype=int32)
#兩群中心在哪
kmeans.cluster_centers_
array([[10.,  2.], [1.,  2.]])
```

Module 12. 密度式分群

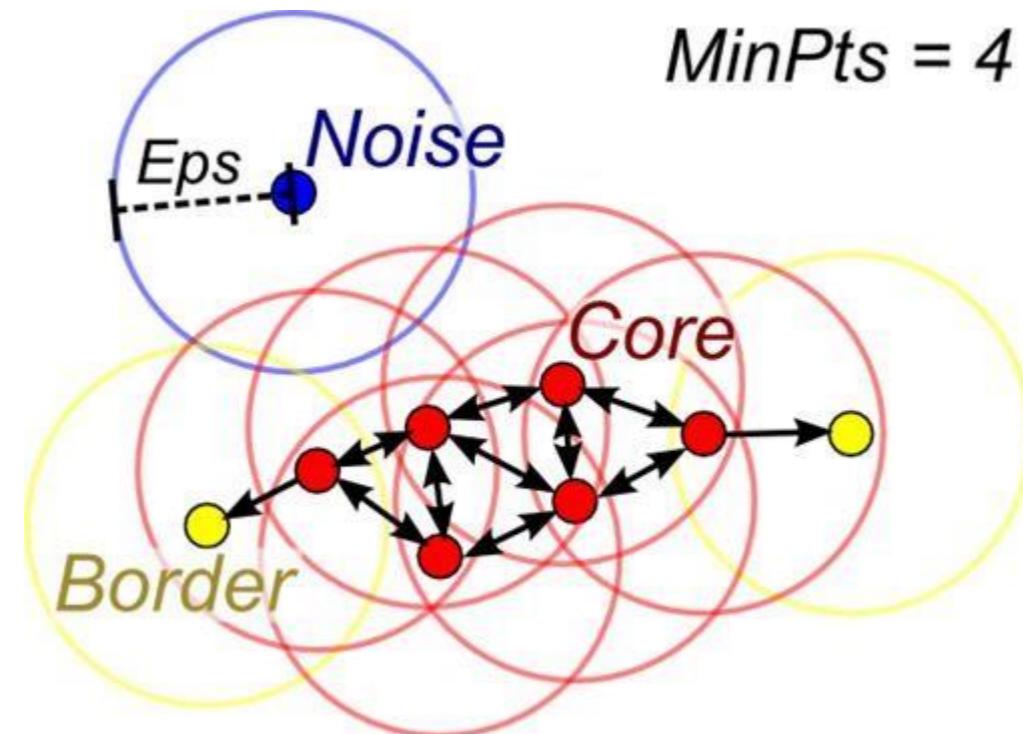
密度式分群 (DBSCAN)簡介

學習目標：

- 認識DBSCAN的定義

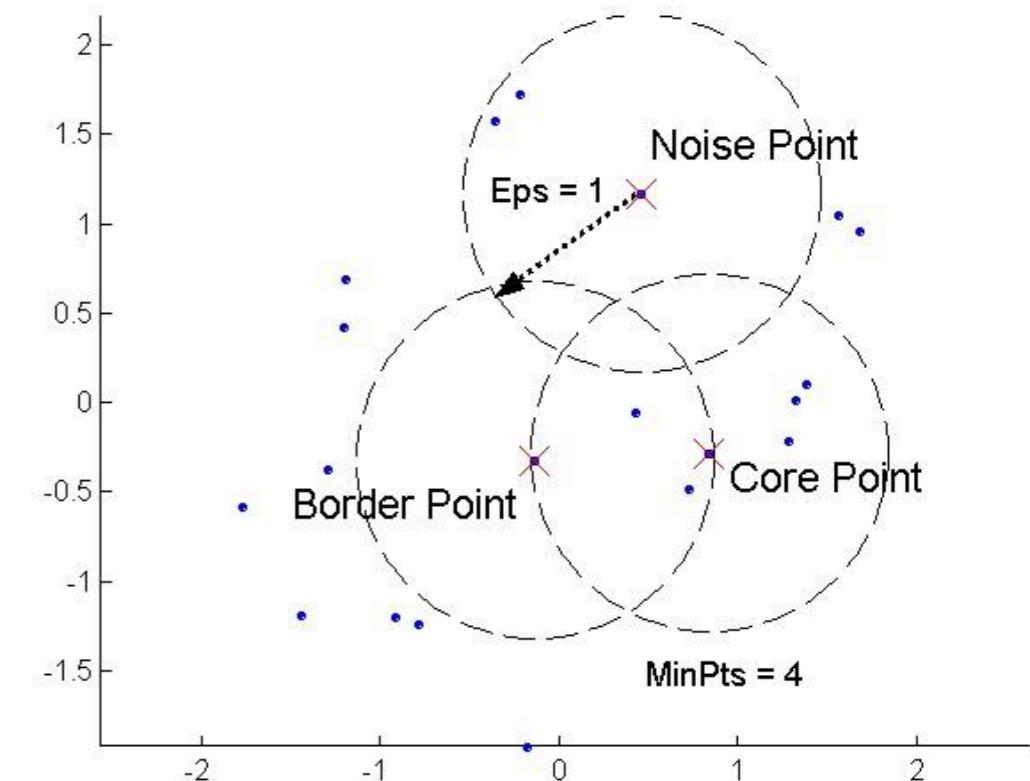
密度式分群簡介

密度式分群(DBSCAN)以點分布密度作為計算基礎，將某半徑(Eps)內的資料分為一群，不斷延伸直至無法增大。



密度式分群簡介

DBSCAN 名詞	說明
Eps	指定之半徑
MinPts	Eps為半徑之圓最小點數
Core Point	該點為中心Eps半徑內點數>MinPts
Border Point	在Core Point為中心Eps半徑內之非CorePoint
Noise Point	既非Core Point 也非 Border Point
Density	Eps為半徑之圓的點數



Module 12.

密度式分群

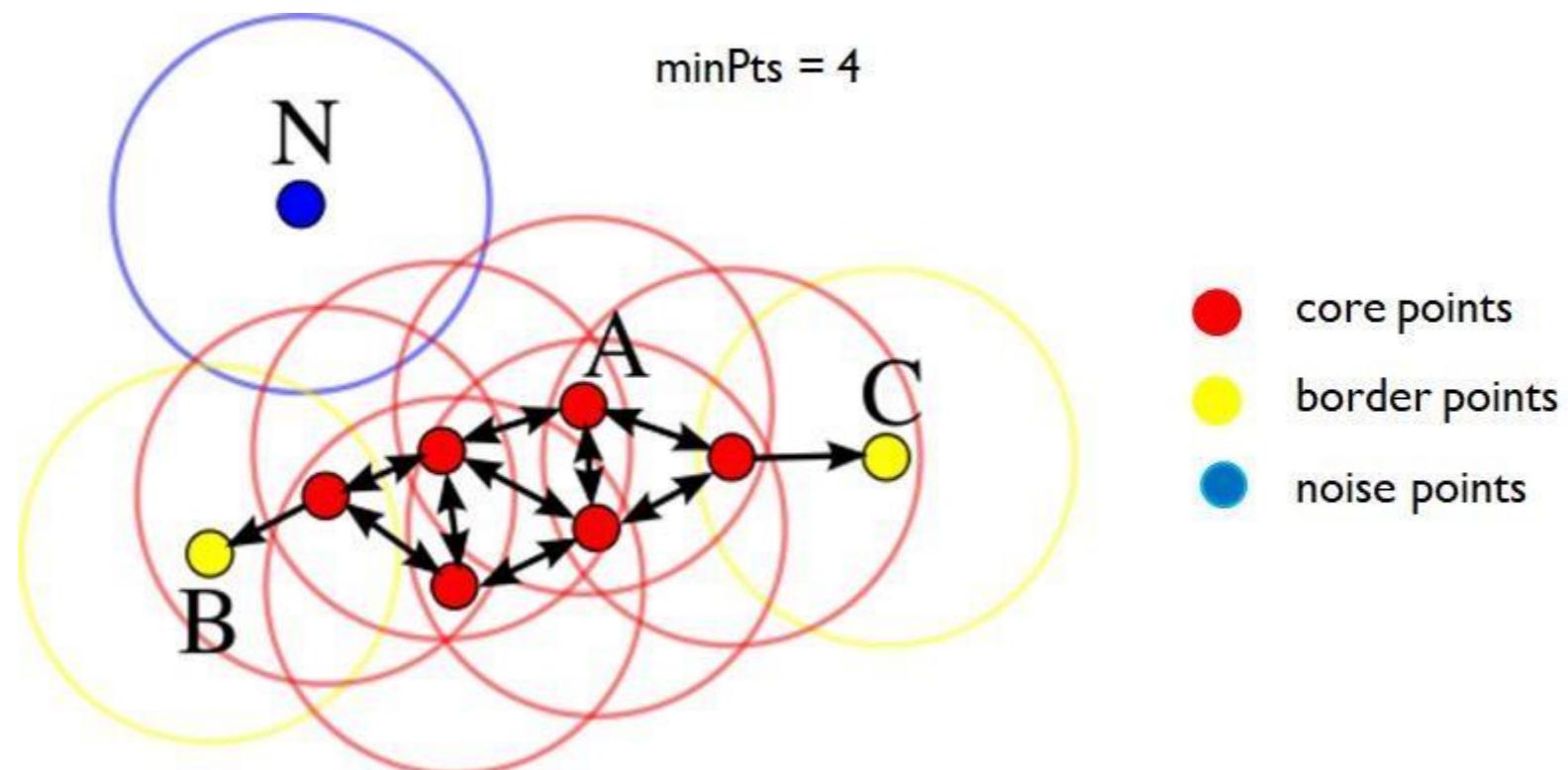
DBSCAN演算法

學習目標：

- 認識DBSCAN演算法

在Eps半徑內同一群內的資料點彼此之間都是直接可達(Directly Reachable)

如下圖中，點B與點C雖然都是邊界點(border point)，但為與A是直接可達，所以同屬一群



優點：

- 無須預先設定群數
- 被離群值影響不大
- 對各種分布形狀的數據集都可勝任

缺點：

- 無法應付密度變化的數據集
- 被參數(Eps , $MinPts$...)影響大

Module 12. 密度式分群

**實作 – 使用DBSCAN
將資料分群**

學習目標：

- 認識DBSCAN演算法

sklearn.cluster.DBSCAN() : DBSCAN

- 常用參數
 - eps : 指定半徑
 - min_samples : 指定 eps 範圍內要有幾個 sample 才算做同一群
- 常用屬性
 - labels_ \Rightarrow Labels of each point

sklearn.cluster.DBSCAN(): DBSCAN 分群演算法

主要參數：

- eps: 指定的半徑
- min_samples: 在eps半徑內最少成群樣本數

主要屬性：

- labels_ : 分出的群及標籤

範例：

```
from sklearn.cluster import DBSCAN
import numpy as np
x = np.array([[1, 2], [2, 2], [2, 3],
...             [8, 7], [8, 8], [25, 80]])
clustering = DBSCAN(eps=3, min_samples=2).fit(X)
clustering.labels_
array([ 0,  0,  0,  1,  1, -1])
clustering
DBSCAN(eps=3, min_samples=2)
```