

# 機器學習模型的簡介

群創光電 智能推進處 idd

貢獻者們：黃齡誼

張訓漢

盧提文

吳彥霖

發佈日期：2024/ 04/ 12

# 前言

大家好，我們是群創光電 智能推進處 idd 的數據科學團隊。

此份「機器學習模型的簡介」係我們團隊精心編製的繁中說明文件，旨在希望能藉由白話且簡單的敘述，協助大家能快速理解機器學習的基礎知識，且建立正確的應用概念。

此份文件中，將介紹機器學習的基礎知識，包括：損失函數與評估指標，並進一步探討其重要的概念，接著介紹常見的監督式學習模型，解釋模型如何從有標記的資料中學習且做出預測，包括：線性迴歸、支援向量機、樹模型等，另外，最後將會說明集成學習(Ensemble Learning)的概念，如何結合多個基礎模型，來提高整體模型的表現。

我們由衷希望，此份文件能幫助大家，建立機器學習模型基本的概念，且探索如何應用機器學習模型，解決真實世界的問題。無論您是這領域的初學者或是已有經驗的專業大大，此文件皆能為您提供有價值的參考資訊。

# 機器學習

## Machine Learning

# Agenda

## ➤ 機器學習的基礎知識 ..... P. 4 - 11

Loss Function、Evaluation Metrics

## ➤ 線性迴歸與正則化 ..... P. 12 - 18

Linear Regression、Huber Regression、LASSO、Ridge Regression、Elastic Net、Bayesian Ridge Regression

## ➤ 支援向量機 ..... P. 19 - 24

Support Vector Classification、Support Vector Regression

## ➤ K 近鄰 ..... P. 25 - 28

K-nearest Neighbors

## ➤ 梯度下降 ..... P. 29 - 31

Batch Gradient Descent、Stochastic Gradient Descent、Mini Batch Gradient Descent

## ➤ 樹模型 ..... P. 32 - 38

Decision Tree、Random Forest、Extra Trees、Adaptive Boosting、Gradient Boosting、Extreme Gradient Boosting

## ➤ 集成學習 ..... P. 39 - 42

Bagging、Boosting、Stacking

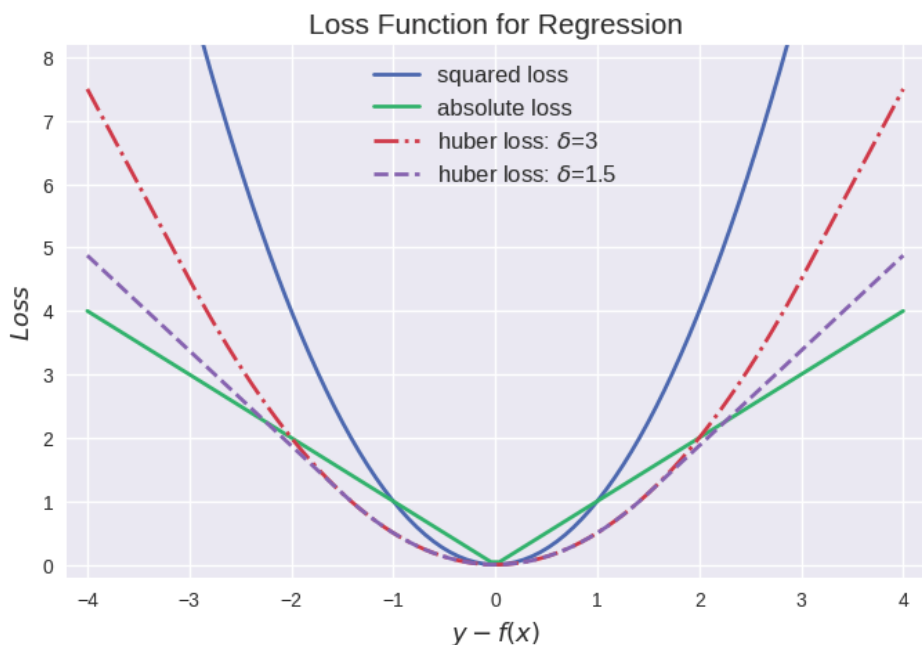
# 機器學習的基礎知識

---

- 損失函數
- 評價指標

# 損失函數 Loss Function

- 在機器學習與深度學習的應用，損失函數用來作為評估預測誤差與表現的函數。在訓練模型的過程中，透過迭代尋找最佳的模型參數，使得模型在這組參數中，能達到最小的損失誤差。
- 迴歸問題中，常見的損失函數：



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

平均絕對誤差  
Mean Absolute Error

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

均方誤差  
Mean Squared Error

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

均方根誤差  
Root Mean Squared Error

$$Huber\ Loss = \begin{cases} \frac{1}{2} (y_i - \hat{y}_i)^2, & \text{for } |y_i - \hat{y}_i| \leq \delta, \\ \delta |y_i - \hat{y}_i| - \frac{1}{2} \delta^2, & \text{otherwise.} \end{cases}$$

Huber Loss

## 以預測天氣型態為例

- 分類問題中，常見的損失函數：

$$\text{Cross Entropy Error} = - \sum_{c=1}^C \sum_{i=1}^N y_{c,i} \times \ln(\hat{y}_{c,i})$$












交叉熵誤差  
Cross Entropy Error

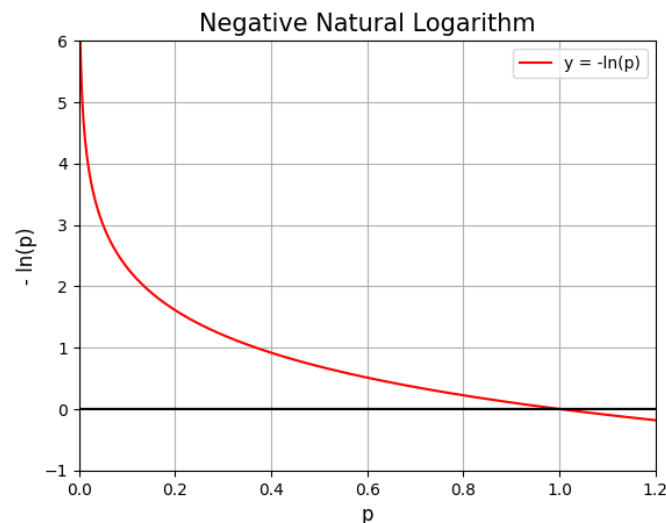
其中，

$y_{c,i}$  為真實的類別標籤且以 One-Hot 形式表示，  
 $\hat{y}_{c,i}$  為模型預測樣本所屬類別的各類別機率值。

說明：

- 當模型預測正確類別的機率值越大，交叉熵誤差就會越小。

| 預測機率值   |   |   | 實際情況   |
|---|---|---|--|
|    |  |  |  |
| 0.7   | 0.2   | 0.1   |   |
| 0.6   | 0.3   | 0.1   |   |
| 0.4   | 0.5   | 0.1   |   |
| 0.1   | 0.3   | 0.6   |   |
| 0.3   | 0.5   | 0.2   |  |
|    |   |   |  |
| $-(1 \times \ln 0.7 + 1 \times \ln 0.6)$<br>$-(1 \times \ln 0.5 + 1 \times \ln 0.3)$<br>$-(1 \times \ln 0.2)$   |   |   |  |
| + )   |   |   |  |
| $\cong$   |   |   | 4.37406  |



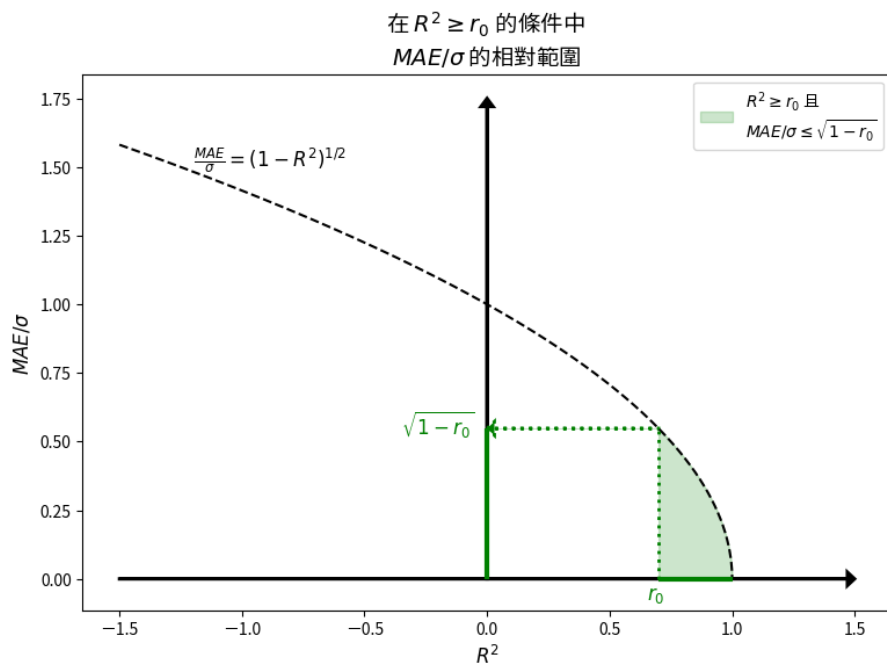
$$\because \lim_{p \rightarrow 1} \ln(p) = 0$$

# 評價指標 Evaluation Metrics



# 迴歸問題的評價指標

- 在機器(深度)學習的應用，迴歸問題中常見的評價指標有：



指標之間的關係

推導



$$\frac{MAE}{\sigma} \leq \frac{\sqrt{MSE}}{\sigma} = \sqrt{1 - R^2}$$

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

平均絕對誤差  
Mean Absolute Error

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

均方誤差  
Mean Squared Error

$$MAPE = \frac{1}{N} \times \sum_{i=1}^N \left| \frac{y_i - \hat{y}_i}{y_i} \right|$$

平均絕對百分比誤差  
Mean Absolute Percentage Error

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

決定係數  
Coefficient of Determination

# 分類問題的評價指標

## 二元分類的混淆矩陣

|    |   | 預測 |   |
|----|---|----|---|
|    |   | T  | N |
| 實際 | T | 7  | 4 |
|    | N | 1  | 9 |

●  $Accuracy = \frac{7 + 9}{7 + 9 + 1 + 4}$

●  $Precision = \frac{7}{7 + 1}$

●  $Recall = \frac{7}{7 + 4}$

●  $Specificity = \frac{9}{9 + 1}$

- 在機器(深度)學習的應用，分類問題中常見的評價指標有：

- True Positive (TP)：實際為 T，預測為 T
- True Negative (TN)：實際為 N，預測為 N
- False Positive (FP)：實際為 N，預測為 T
- False Negative (FN)：實際為 T，預測為 N

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad \begin{array}{l} \text{正確率} \\ \text{Accuracy} \end{array}$$

$$Precision = \frac{TP}{TP + FP} \quad \begin{array}{l} \text{精確率} \\ \text{Precision} \end{array}$$

$$Recall = \frac{TP}{TP + FN} \quad \begin{array}{l} \text{召回率} \\ \text{Recall} \end{array}$$

$$Specificity = \frac{TN}{TN + FP} \quad \begin{array}{l} \text{特異度} \\ \text{Specificity} \end{array}$$

$$F1 \text{ score} = 2 \times \frac{Precision + Recall}{Precision \times Recall}$$

# 分類問題的評價指標

- 在機器(深度)學習的應用，分類問題中常見的評價指標有：

## 三元分類的混淆矩陣

|    |   | 預測 |   |   |
|----|---|----|---|---|
|    |   | A  | B | C |
| 實際 | A | 7  | 4 | 5 |
|    | B | 1  | 9 | 6 |
|    | C | 2  | 3 | 8 |

➔

| A 類別的<br>混淆矩陣 |   |
|---------------|---|
| 實際 \ 預測       |   |
| A             | 7 |
| N             | 9 |

- True Positive (TP): 實際為 T，預測為 T
- True Negative (TN): 實際為 N，預測為 N
- False Positive (FP): 實際為 N，預測為 T
- False Negative (FN): 實際為 T，預測為 N

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

正確率  
Accuracy

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

精確率  
Precision

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

召回率  
Recall

$$Specificity = \frac{TN}{TN + FP}$$

特異度  
Specificity

$$F1\ score = 2 \times \frac{Precision + Recall}{Precision \times Recall}$$

- Accuracy of A =  $\frac{7 + 26}{7 + 26 + 3 + 9}$

- Precision of A =  $\frac{7}{7 + 3}$

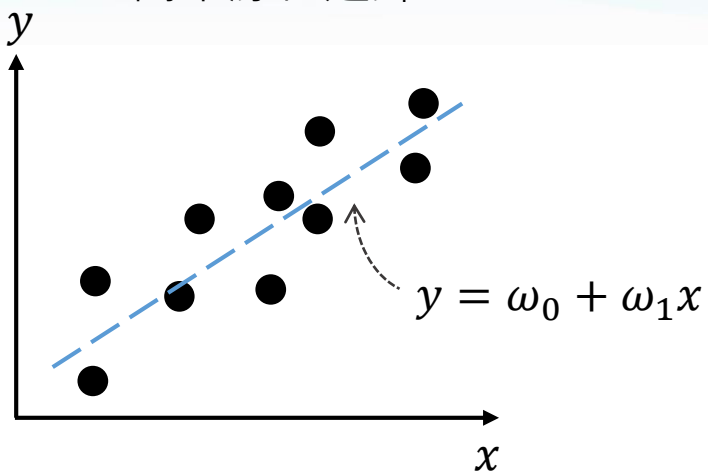
- Recall of A =  $\frac{7}{7 + 9}$

- Specificity of A =  $\frac{26}{26 + 3}$

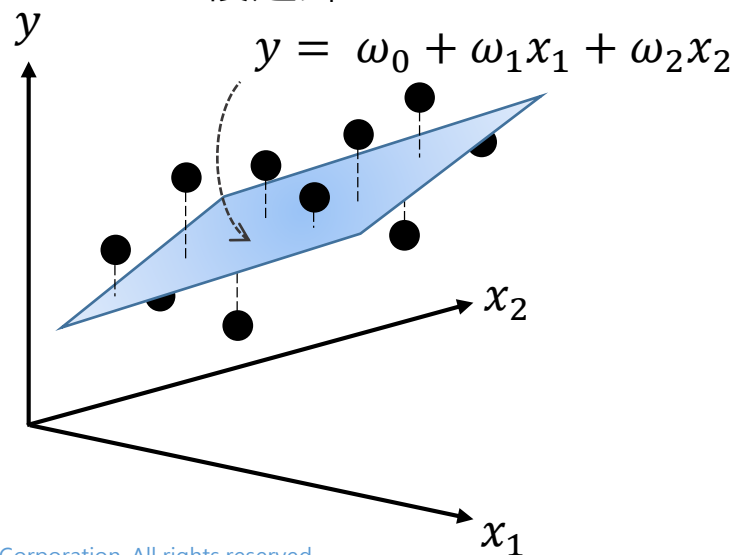
# 線性迴歸 Linear Regression

# 線性迴歸 Linear Regression

簡單線性迴歸



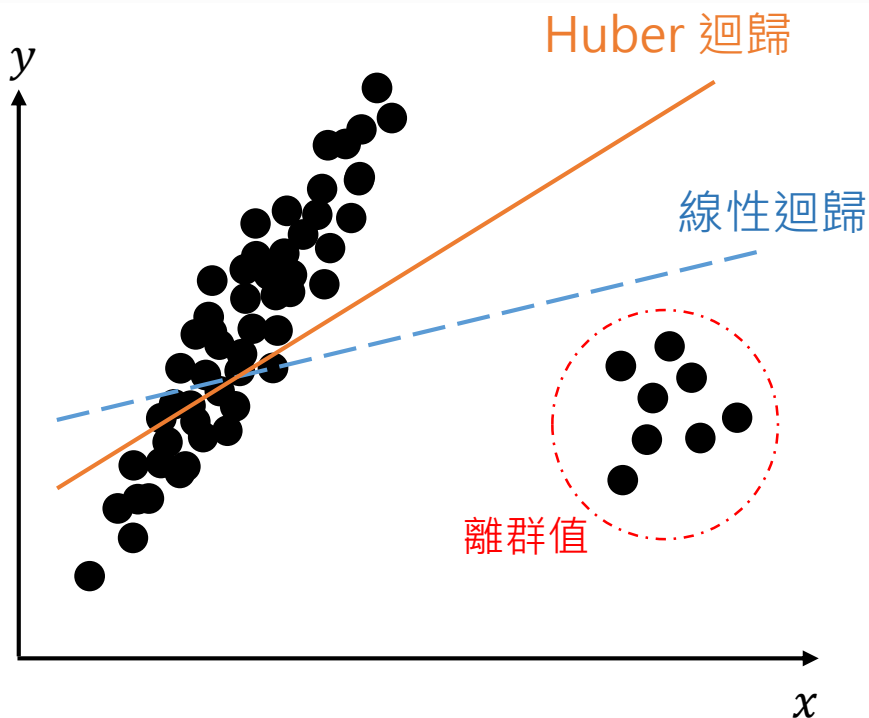
複迴歸



- 線性迴歸是一種基於統計的機器學習演算法，用來處理迴歸預測的問題。
- 基本思維：
 

用來描述「隨著某個特徵變數(Feature Variable)  $x$  的增加，目標變數(Target Variable)  $y$  也會增加或減少」的關聯性。若只有一個特徵變數時，線性迴歸模型稱為簡單線性迴歸 Simple Linear Regression，若特徵變數的數量大於 2 個時，則稱為複迴歸 Multiple Regression。
- 優點：
  - 模型結構單純簡單，容易理解與解讀。
  - 運算資源需求低，執行速度快。
- 注意事項：
  - 藉由殘差分析 Residual Analysis，可進一步提升線性迴歸的表現。
  - 線性迴歸無法解釋變數間的非線性關係。
  - 線性迴歸易受離群值 Outlier 的影響，降低預測能力。
  - 當特徵變數之間存有共線性(Multicollinearity)的現象時，線性迴歸的權重參數，容易因數據微小的變化，卻產生劇烈的變化，因此，共線性會大幅降低線性迴歸參數的可信度。

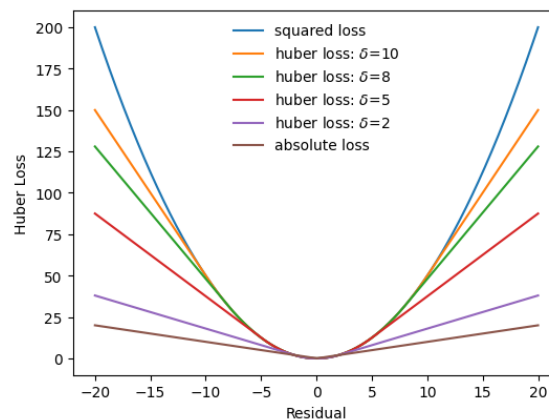
# Huber 迴歸 Huber Regression



- Huber 迴歸是線性迴歸的一種優化，主要目的係要消弭離群值的影響。

- 核心概念：

相較於線性迴歸最主要的差別，係在於 Huber 迴歸的損失函數採用 Huber Loss，Huber Loss 結合絕對誤差與均方誤差，對於造成預測誤差小的預測值採用均方誤差，對於預測誤差大的預測值採用絕對誤差。



$$Huber Loss = \begin{cases} \frac{1}{2}(y_i - \hat{y}_i)^2, & \text{for } |y_i - \hat{y}_i| \leq \delta, \\ \delta|y_i - \hat{y}_i| - \frac{1}{2}\delta^2, & \text{otherwise.} \end{cases}$$

註：當  $\delta$  越小，Huber 迴歸受離群值的影響越小。

- 優點：

- 對於有離群值的資料，Huber 迴歸不會像線性迴歸過於敏感。

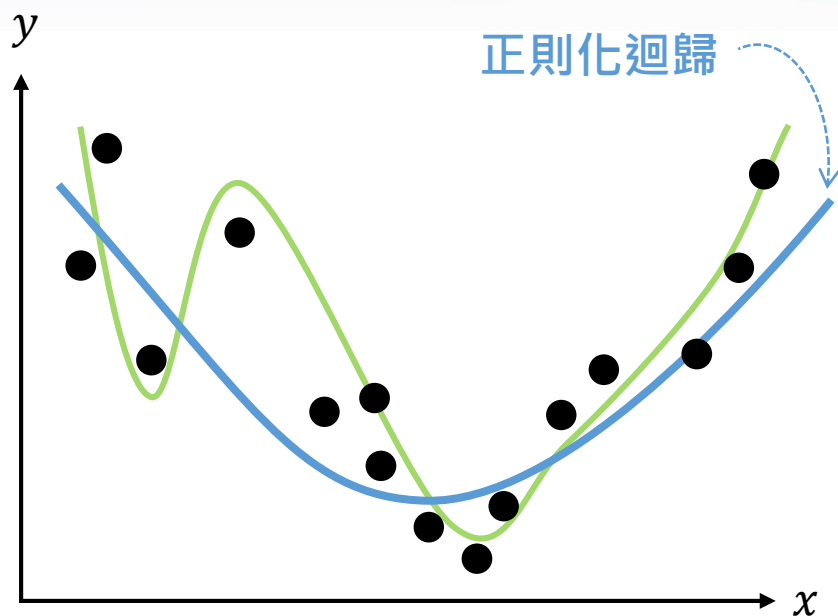
- 注意事項：

- 雖然，Huber 迴歸對離群值的處理，有相當程度的穩健性(Robustness)，但是，針對極端的離群值，Huber 迴歸的處理能力仍有限，建議仍須從資料的理解與清理著手，以排除極端離群值的影響。

# 正則化迴歸 Regularized Regression

- 係一種預防過擬合的建模技巧。
- 降低模型的複雜度。
- 常見的三種正則化技巧 : LASSO、Ridge、Elastic Net。

# 正則化迴歸 Regularized Regression



- 正則化迴歸係一種優化擬合過程的機器學習技巧。

- 核心概念：

在機器學習模型的訓練過程中，目的係要逐漸降低且收斂損失函數的值(預測值與實際值的誤差)，從中尋找出一組模型的最佳參數，使得模型在套入這組參數時會得到最小的損失。

正則化迴歸係在模型的損失函數中，添加一個懲罰項(Penalty Term)來限制模型的參數，對於訓練過程「參數變大，損失也會變大」的情況給予懲罰，藉此抑制模型參數，以降低模型的複雜度。

正則化的主要方法有下列三種：

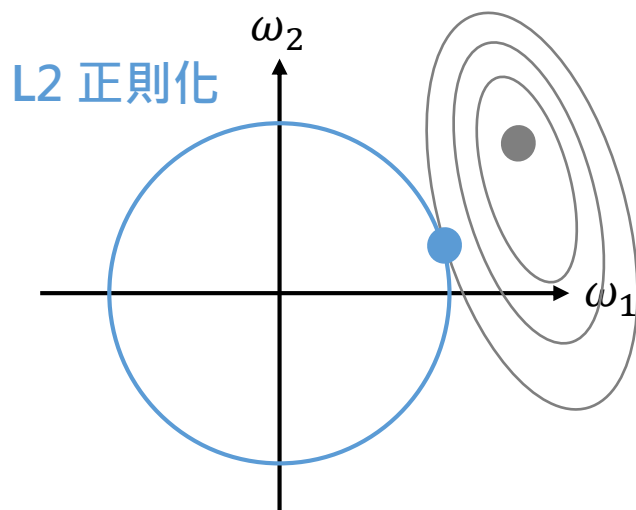
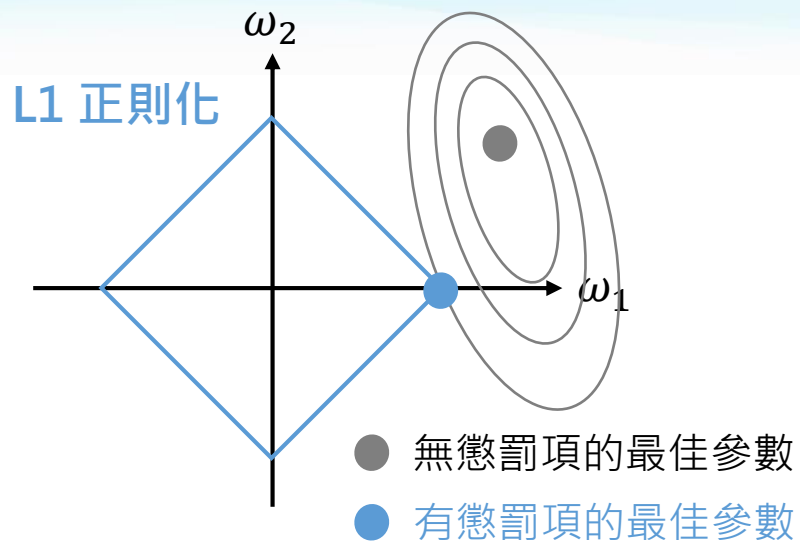
- (1) L1 正則化：LASSO
- (2) L2 正則化：嶺迴歸 Ridge Regression
- (3) 彈性網絡 Elastic Net

- 優點：

- 能有效降低模型的複雜度，降低過擬合(Overfitting)的情況發生。
- 能提升模型的泛化能力(Generalization Ability)。



# 正則化迴歸 Regularized Regression



**L1 正則化**  
**LASSO**  
Least Absolute Shrinkage and Selection Operation

**L2 正則化**  
**嶺迴歸**  
Ridge Regression

**L1+L2 正則化**  
**彈性網絡**  
Elastic Net

## 正則化損失函數

$p$  為特徵欄位的數量  
 $N$  為樣本資料的筆數

$$\sum_{i=1}^N (y_i - \hat{y})^2 + \alpha \sum_{j=1}^p |\omega_j|$$

$$\sum_{i=1}^N (y_i - \hat{y})^2 + \alpha \sum_{j=1}^p \omega_j^2$$

$$\sum_{i=1}^N (y_i - \hat{y})^2 + \alpha_1 \sum_{j=1}^p |\omega_j| + \alpha_2 \sum_{j=1}^p \omega_j^2$$

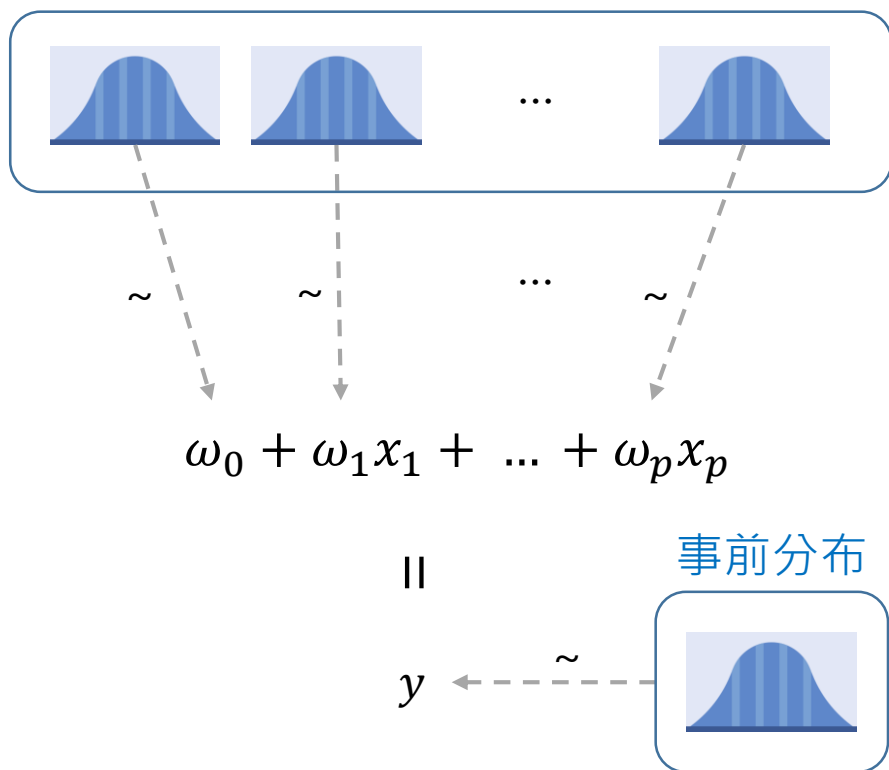
- 當  $\alpha$  越大時，模型更簡化且參數更加縮小，但，過大的  $\alpha$  會使模型過於簡單，導致欠擬合(Underfitting)的發生。

延伸閱讀：

- Allen Tzeng. Sep. 6, 2020. [L1, L2 Regularization 到底正則化了什麼？](#)

# 貝氏嶺迴歸 Bayesian Ridge Regression

事前分布



- 貝氏嶺迴歸為複線性迴歸的一種衍生，結合正則化與貝氏機率的思維，藉由事前機率(Prior Probability)來估計線性迴歸的參數，常應用於當特徵變數高度相關的情況。

- 核心概念：

貝氏嶺迴歸須先假設複迴歸的各項參數服從某種先驗分佈，再藉由樣本資料的數據與貝氏定理，計算各項參數的事後機率(Posterior Probability)。

$$\text{事後機率} \quad Prob(\omega|y) = \frac{Prob(y|\omega) \times \text{事前機率} \quad Prob(\omega)}{Prob(y)}$$

事後機率

事前機率

- 優點：

- 相較於複迴歸，貝氏嶺迴歸能降低共線性(Multicollinearity)的影響，更佳的泛化能力(Generalization Ability)。

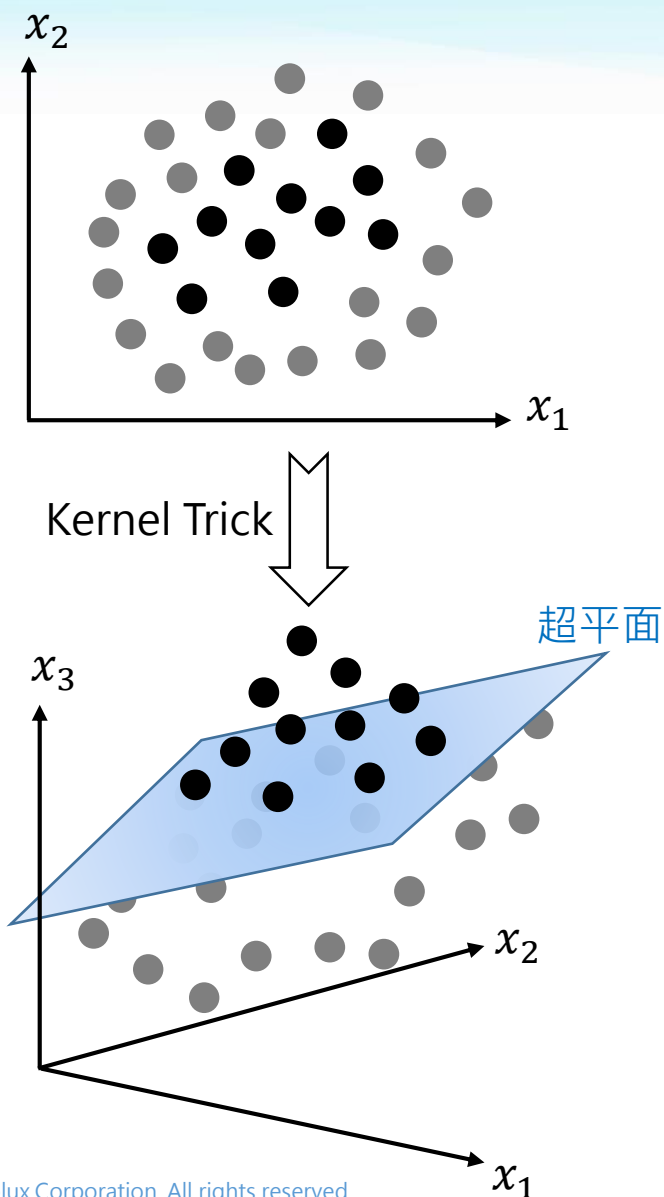
- 注意事項：

- 參數的估計易受主觀事前機率的假設所影響。因此，當樣本蒐集不易或樣本數不足時，如果我們已對各特徵變數有高度的了解與掌握，則貝氏嶺迴歸適合被應用。

# 支援向量機 Support Vector Machine

- 將樣本映射至更高維度的空間，在此空間中尋找一個能完美分割樣本的超平面。
- 善於處理高維度與非線性的複雜資料。

# 支援向量機 Support Vector Machine



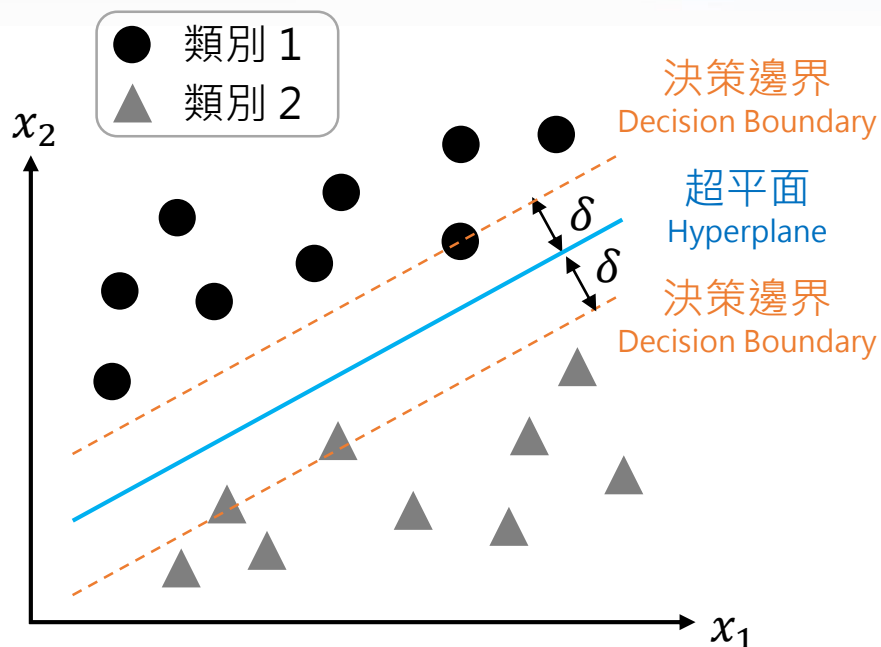
- 支援向量機是一種監督式的機器學習演算法，可用來處理迴歸與分類問題。
- 當支援向量機在處理迴歸問題時，則也可稱為支援向量迴歸 Support Vector Regression, SVR，同樣地，當運用在分類問題時，支援向量機則可稱為支援向量分類 Support Vector Classification, SVC。
- 核心概念：
  - 線性支援向量機：
 

在訓練學習的過程中，分類問題以最大化間距(Margin)為目標，尋找一超平面(Hyperplane)使得不同類別之間間距最大化，另外，迴歸問題以最大化決策邊界(容忍誤差  $\epsilon$ )內的樣本數，且最小化決策邊界間外樣本的預測誤差為基礎。
  - Kernel 法的支援向量機：
 

藉由核函數 Kernel Function，將樣本資料投影至更高維度的空間中，例如：把兩個特徵變數從二維平面轉換到三維空間，在更高維度的空間中，尋找一個最佳的超平面將樣本數據明確地區隔開，同時，滿足在這超平面的訓練預測誤差能達最小化。
- 優點：
  - 可處理高維度的複雜資料。
- 注意事項：
  - 支援向量機的訓練結果，對於超參數的設定敏感，尤其是核函數 Kernel Function 的選擇。

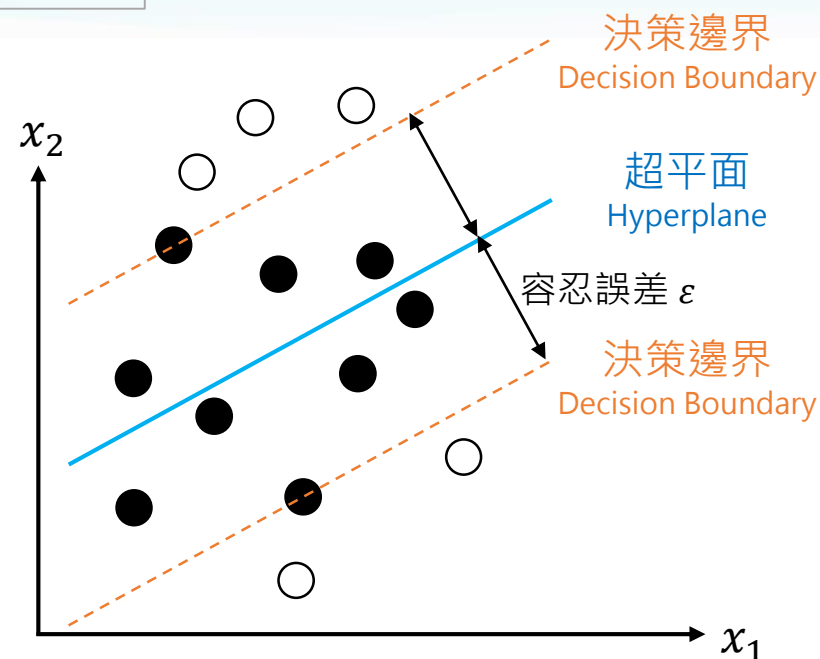
# SVC 與 SVR 的基本概念

以線性支援向量機為例



SVC 分類問題

- 藉由超平面將隸屬於不同類別的樣本區隔開。
- 最大化超平面到最近樣本的距離  $\delta$ 。
- 上面圖例的情況稱**硬性間距(Hard-Margin)**，即在決策邊界內不容許有資料在其當中。



SVR 迴歸問題

- 計算損失殘差時，決策邊界內(小於過容忍誤差)的樣本不列入計算。
- 最大化決策邊界內的樣本數，同時，最小化決策邊界外的預測殘差。

# 支援向量機的重要超參數

## ● Kernel Function

➤ 核函數 Kernel Function 係將樣本數據從原始空間中，映射到更高維度空間的一種技巧，目的係要使得在原始空間中無法以線性分割的資料，在更高維度的空間變成線性可分割。

➤ SVM 中可使用的核函數有：

### (1) 線性 Linear

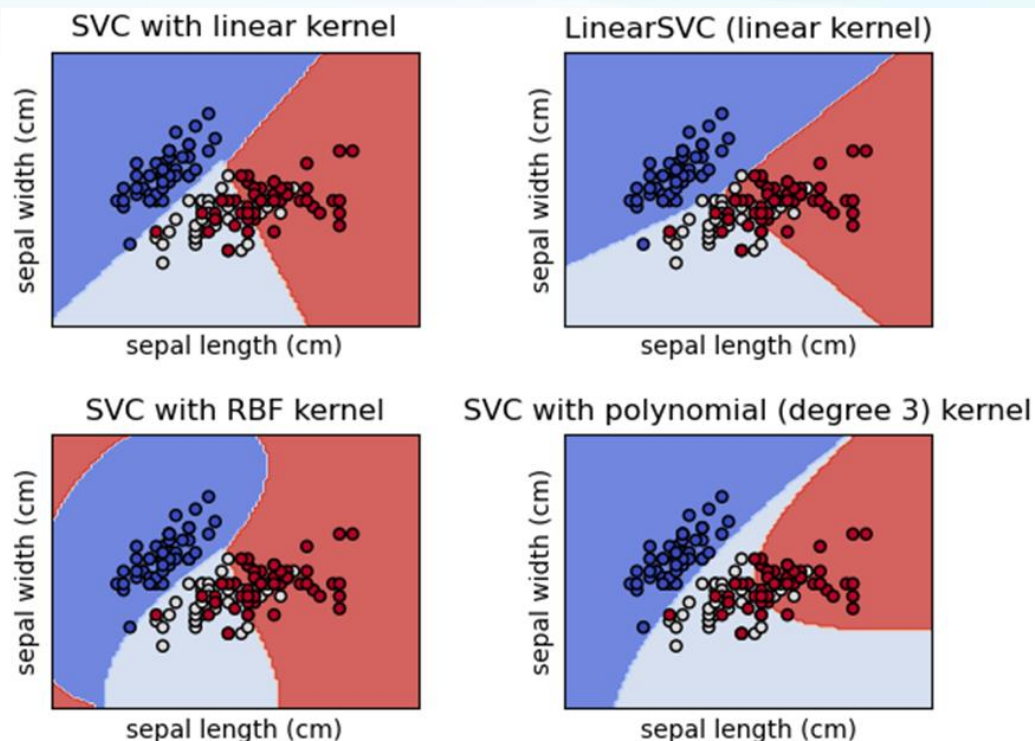
- ◆ 適合用來處理線性可分割的樣本數據；
- ◆ 無法處理非線性的樣本數據。

### (2) 多項式 Polynomial

- ◆ 將樣本數據映射至更高維度的空間，例如：兩特徵欄位經由 2 次多項式核函數，可將其映射至三維空間中；
- ◆ 能處理非線性的樣本數據。

### (3) 徑向核函數 [Radial Basis Function](#), RBF

- ◆ 將樣本數據映射至更高維度的空間，
- ◆ 能處理更複雜的非線性樣本數據。



說明：上圖為 SVC 搭配不同的核函數，應用在鳶尾花資料集 Iris Dataset 中，所產生不同的決策邊界結果。

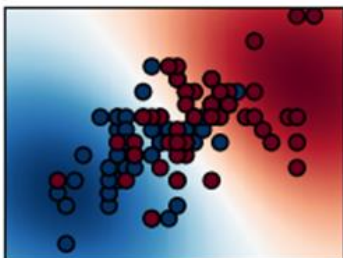
註：鳶尾花資料集 Iris Dataset

- 4 個花卉特徵欄位，分別為：花萼長寬、花瓣長寬；
- 3 種花卉的屬種，分別為：'setosa'、'versicolor'、'virginica'；
- 屬於分類問題的資料集。

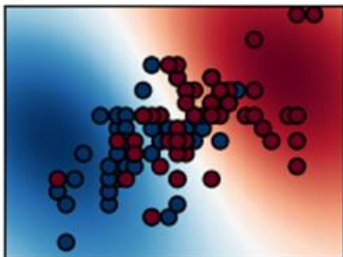
Source :  
[Plot different SVM classifiers in the iris dataset.](#)



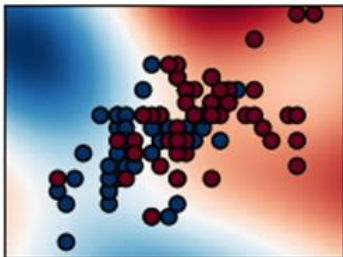
gamma= $10^{-1}$ , C= $10^{-2}$



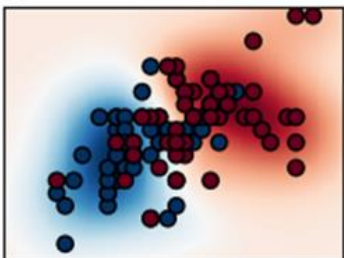
gamma= $10^{-1}$ , C= $10^0$



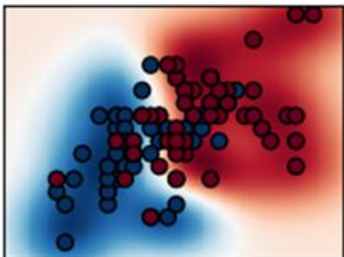
gamma= $10^{-1}$ , C= $10^2$



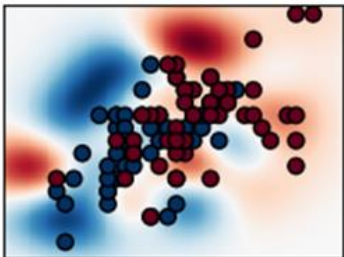
gamma= $10^0$ , C= $10^{-2}$



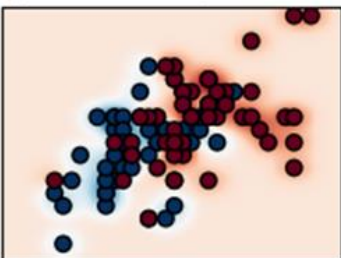
gamma= $10^0$ , C= $10^0$



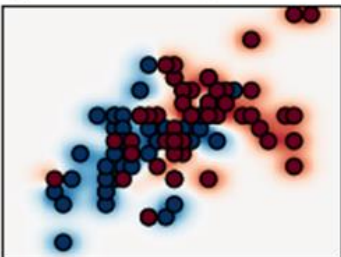
gamma= $10^0$ , C= $10^2$



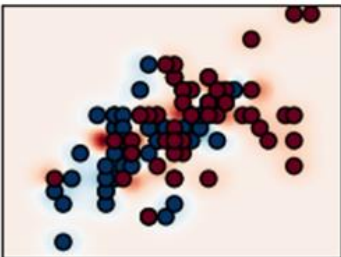
gamma= $10^1$ , C= $10^{-2}$



gamma= $10^1$ , C= $10^0$



gamma= $10^1$ , C= $10^2$



## ● Gamma

- 當核函數 Kernel Function 係選用 poly、rbf 或 sigmoid 三者其中之一時，才須設定 Gamma。
- Gamma 表示每個單一樣本對於超平面的影響力；
- 當 Gamma 越大時，則距離超平面越近的樣本，其影響力越大，越能勾勒出複雜的決策邊界。

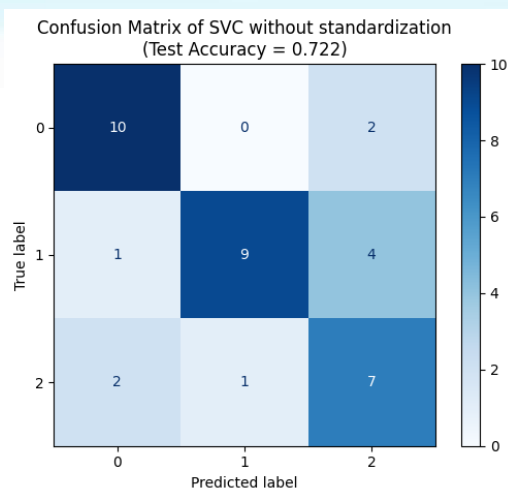
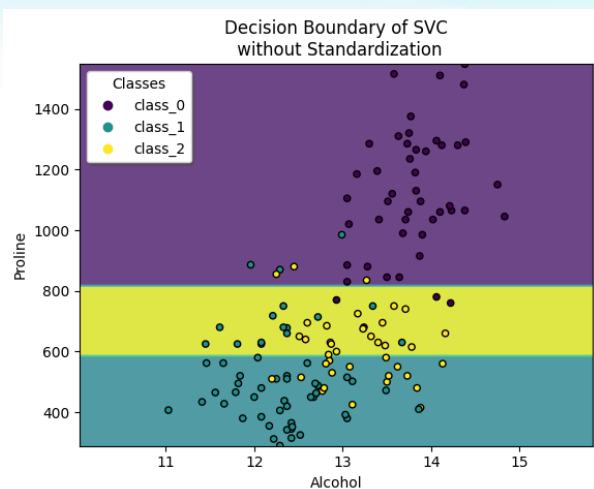
## ● Penalty Coefficient (C)

- Penalty Coefficient 表示預測誤差的容忍程度；
- 當 C 越大時，決策邊界的寬度越窄，容許訓練誤差的條件越嚴苛，越無法接受訓練時的預測誤差。

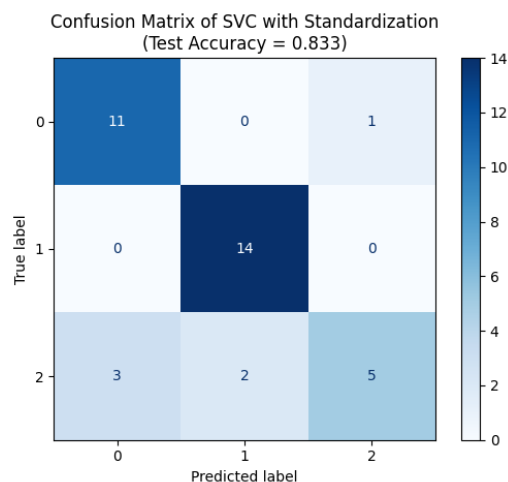
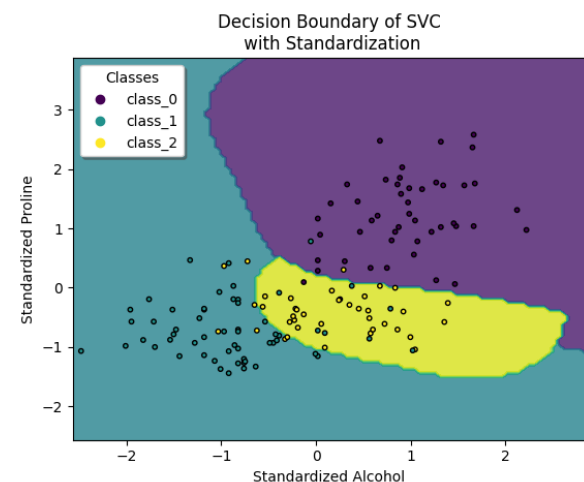
## ● 注意事項：

- 當 Gamma = 'auto' 時，表示  
 $Gamma = 1/\text{特徵欄位的數量}$ 。
- 當 Gamma 或 Penalty Coefficient 設定過大時，易造成過擬合(Overfitting)的發生。

# 支援向量機的資料前處理



標準化  
[Standardization](#)



- 特徵縮放(Scaling)對於支援向量機的結果，有顯著的影響。
- 特徵縮放的必要性有二：
  - 避免因**特徵尺度的不同**，而導致較大數值範圍的特徵支配較小數值範圍的特徵，例如：特徵 A 的範圍介於 0~1，特徵 B 的範圍介於 1,000~10,000，則，支援向量機對於特徵 B 的關注將遠遠超過特徵 A。換句話說，特徵尺度間的差異，會影響到支援向量的選擇，進而影響模型的泛化能力。
  - 可提升模型的收斂速度。
- 左圖係以紅酒資料集為範例，以 8:2 隨機切分成訓練集與測試集，在訓練集中訓練 SVC，且在測試集中檢視 SVC 的分類能力。
  - 當沒有對特徵進行縮放時，SVC 在測試集的整體準確率 (Accuracy)約 0.722，且決策邊界主要取決於數值較大的特徵 Proline。
  - 在對特徵進行標準化縮放後，SVC 的整體準確率提升到約 0.833，且決策邊界的選擇不會因尺度的差異而造成偏頗。

## 衍生閱讀：

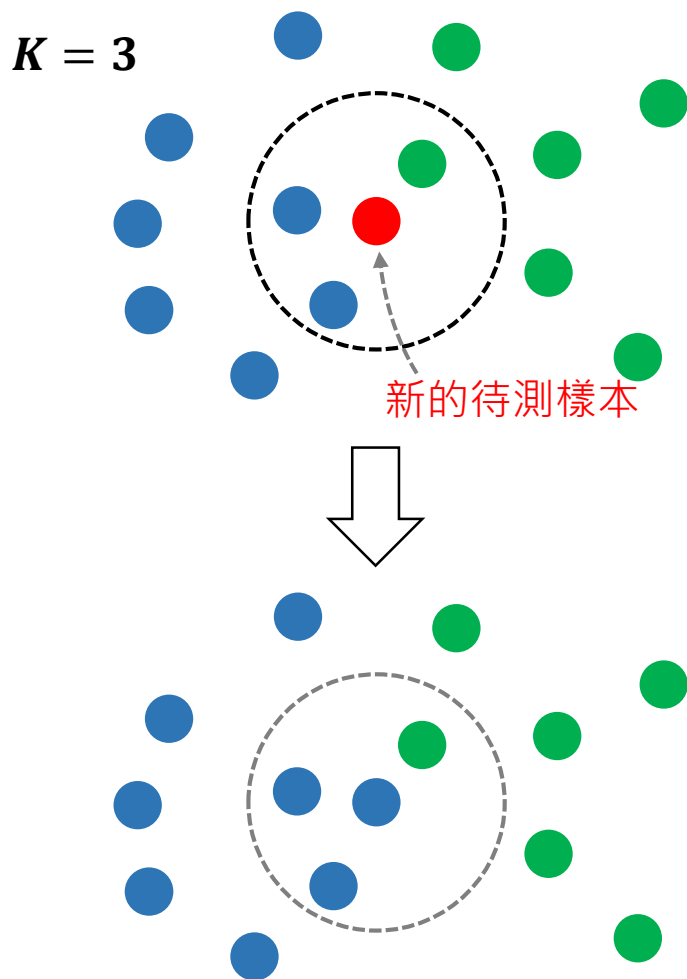
- [When using SVMs, why do I need to scale the features?](#). Stack Exchange.
- Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. May 19, 2016. [A Practical Guide to Support Vector Classification](#).
- Dave Sotelo. Jul 26, 2017. [Effect of Feature Standardization on Linear Support Vector Machines](#).



# K 近鄰 K-nearest Neighbors, KNN

- 基於樣本間彼此的距離進行預測。

## K 近鄰二元分類的示意圖



- K 近鄰是一種監督式的機器學習演算法，可用來處理迴歸與分類問題。

- **基本概念：**

相較於其他的監督式演算法，K 近鄰演算法最特別的地方，在於它並沒有經由學習訓練資料尋找出最佳參數，在預測新的測試樣本之前，完全不會進行任何的運算。

對於新的測試樣本，先計算新樣本與每個訓練集樣本之間的距離，且找出與新樣本最近的 K 個訓練樣本(近鄰的意思)，最後，以這些近鄰 K 個訓練樣本的平均(或投票)來求得預測結果。

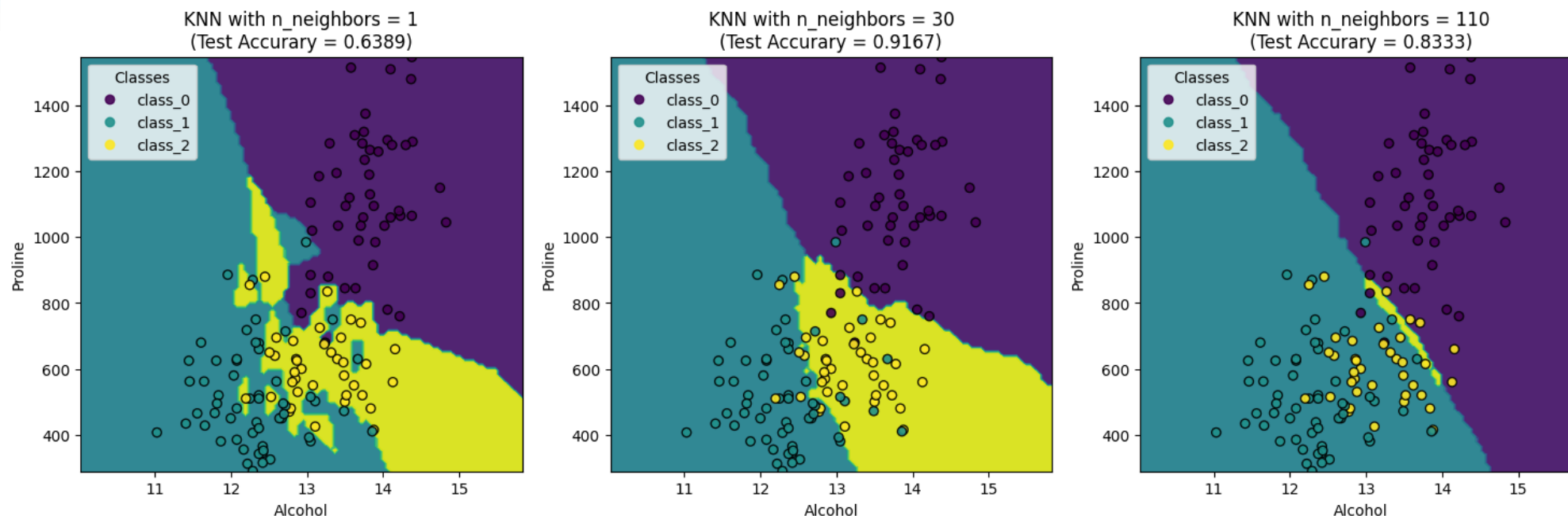
- **優點：**

- 運算邏輯簡單易懂。
- 不須經訓練學習過程，非常適合應用在即時產出的數據。
- 適合應用在樣本數據量較少或特徵較少的資料。

- **注意事項：**

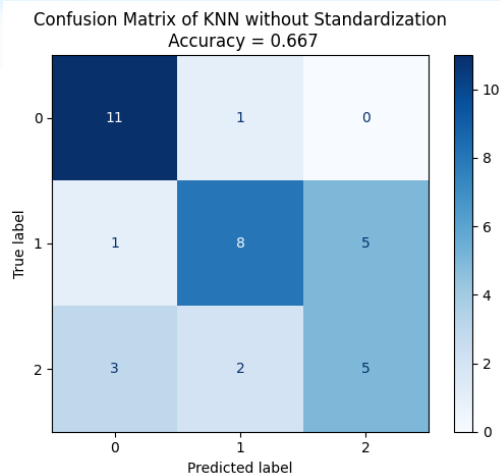
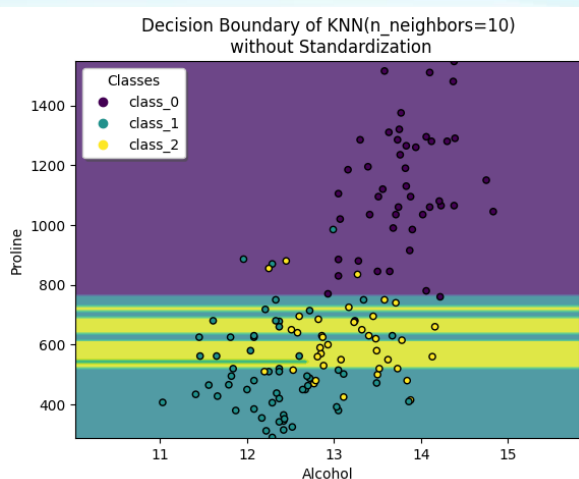
- 若樣本數據量龐大，計算新樣本與訓練樣本彼此的距離將會非常耗時，此時建議改採用其他的演算法。
- 預測結果對於超參數 K 的選擇敏感，建議先多方嘗試不同的 K，再選擇一個最適的 K 值。

# KNN 的重要超參數 n\_neighbors

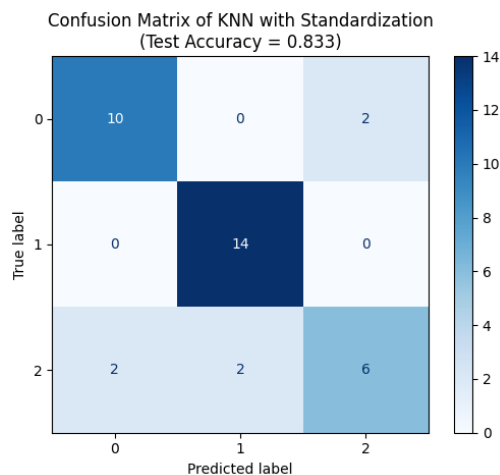
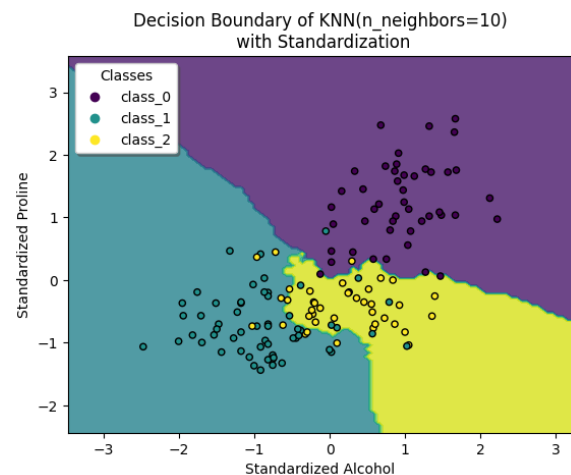


- 超參數  $n\_neighbors$  為近鄰樣本的數量  $K$ ，預設為 5。
- 超參數  $n\_neighbors$  會影響決策邊界的形狀，過小的  $n\_neighbors$  易導致過擬合(Overfitting)，另外，過大的  $n\_neighbors$  易造成決策邊界過於平滑寬鬆，導致欠擬合(Underfitting)。
- 上圖係以紅酒資料集為範例，以 8:2 對目標類別分層抽樣成訓練集與測試集：
  - 當  $n\_neighbors = 1$  時，存在幾個孤島般的決策邊界；
  - 當  $n\_neighbors = 110$  時，決策邊界太過簡單且平滑。

# KNN 的資料前處理



標準化  
Standardization



- 特徵縮放(Scaling)對於 KNN 的結果，有顯著的影響。
- 特徵縮放的必要性：
  - 在計算樣本彼此之間的距離時，距離的大小易受較大尺度的特徵所決定，另外，就幾何觀點而言，鄰近的 K 個訓練樣本在較大尺度特徵中的分布，相較於在較小尺度特徵中來得廣泛，因此，**在沒有對各個特徵進行縮放的情況下，KNN 的決策邊界往往取決於尺度相對大的特徵，進而影響模型的泛化能力。**
- 左圖係以紅酒資料集為範例，以 8:2 對目標類別分層抽樣成訓練集與測試集，且在測試集中評量 KNN 的分類能力。
  - 當沒有對特徵進行縮放時，KNN 在測試集的整體準確率 (Accuracy) 約 0.667，且決策邊界主要取決於數值較大的特徵 Proline 做分割。
  - 在對特徵進行標準化縮放後，KNN 在測試集的整體準確率提升到約 0.833，且決策邊界的選擇不會因尺度的差異而造成偏頗。

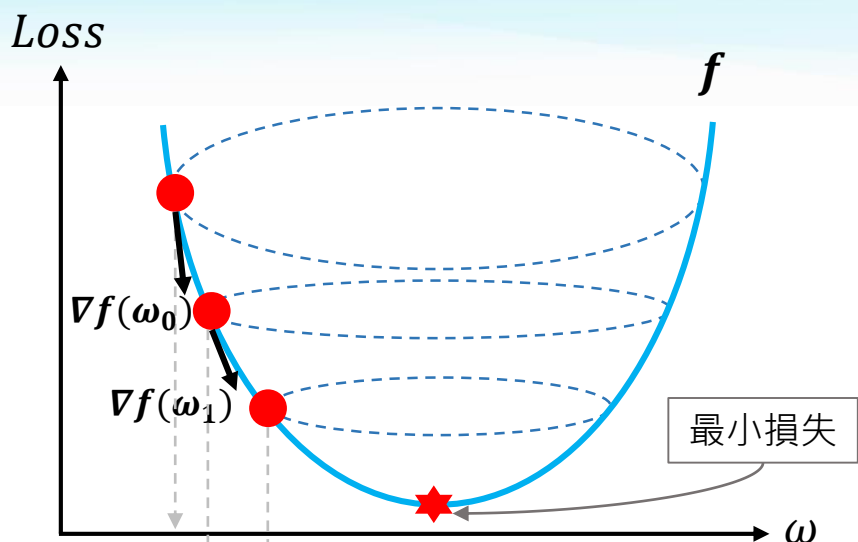
衍生閱讀：

- [Why do you need to scale data in KNN](#). Stack Exchange.
- Mario Filho. March 25, 2023. [Is Feature Scaling Required for the KNN Algorithm?](#).

# 梯度下降 Gradient Descent

- 係用來尋找最小(或最大)損失的一種演算法。
- 藉由迭代調整參數，逐步達到損失最小化。
- 梯度下降常見的三種型態：
  - (1) 批量梯度下降 Batch Gradient Descent
  - (2) 隨機梯度下降 Stochastic Gradient Descent
  - (3) 小批量梯度下降 Mini Batch Gradient Descent

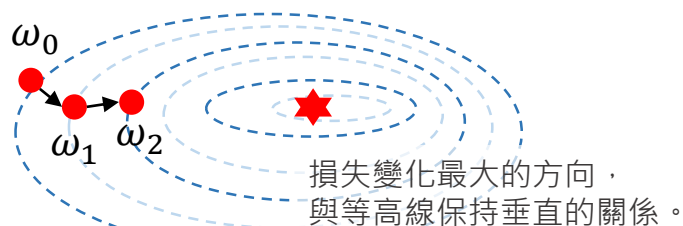
# 梯度下降 Gradient Descent



$$\omega_1 = \omega_0 - \alpha \times \nabla f(\omega_0)$$

$$\omega_2 = \omega_1 - \alpha \times \nabla f(\omega_1)$$

⋮



損失函數的等高線圖

- 在機器學習的領域中，梯度下降係一種常用於找尋最小損失的演算法。

- 核心思維：

在訓練模型的過程中，損失函數  $f$  係用來評估模型當下的學習表現，透過迭代更新參數  $\omega$  的機制，如下列公式，從中找尋最小損失的模型參數。

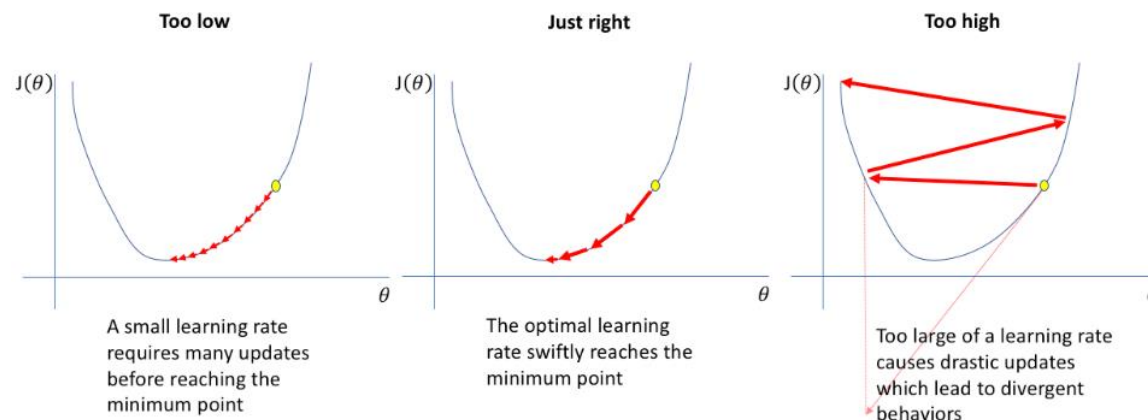
$$\omega_{t+1} = \omega_t - \alpha \times \nabla f(\omega_t) \quad t \text{ 為迭代次數}$$

- 更新參數  $\omega$ ：

損失函數的負梯度方向  $-\nabla f$ ，即為損失降幅最大最陡峭的路徑方向。參數的更新係依循著損失函數負梯度的方向進行，使得每次更新參數後的損失能更小，且不斷重複這迭代更新的步驟，直到損失達到收斂條件才停止。

- 學習率 Learning Rate,  $\alpha$ ：

控制每次更新參數的下降步伐大小，過大或太小的學習率，皆容易造成參數更新無法收斂的情況。

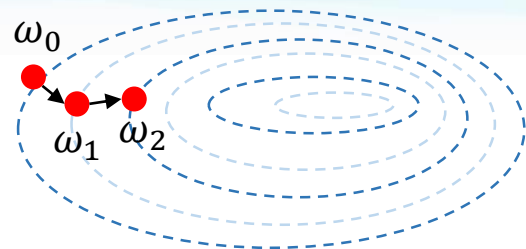


Source:  
[Gradient descent.](#)

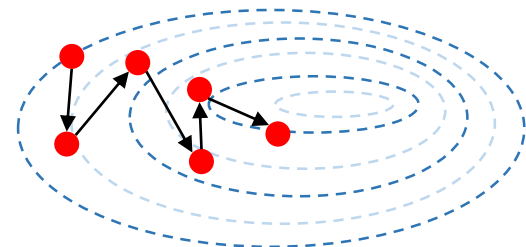


# 梯度下降的三種型態

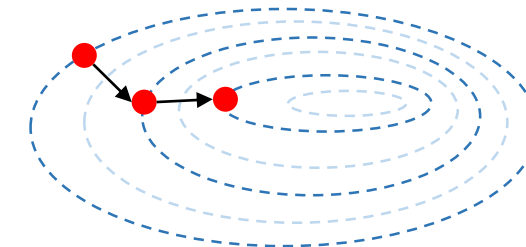
批量梯度下降



隨機梯度下降



小批量梯度下降



損失函數的等高線示意圖

## ● 批量梯度下降 Batch Gradient Descent

- 運用所有的訓練集樣本來計算損失，且沿著損失遞減最迅速劇烈的方向，進行下一次的迭代計算。
- 當訓練樣本數很大量時，運算成本將會很高。

## ● 隨機梯度下降 Stochastic Gradient Descent

- 在每一次迭代計算損失時，皆只藉由單一個樣本進行計算，損失函數的曲面會隨著迭代動態變化，可避免陷入局部最小的情況。
- 相較於批量梯度下降，SGD計算複雜度較低，訓練速度較快，但是，損失下降收斂的過程也較為震盪，穩定性較差。
- 當訓練樣本數很大量時，迭代次數也會隨著變很大。

## ● 小批量梯度下降 Mini Batch Gradient Descent

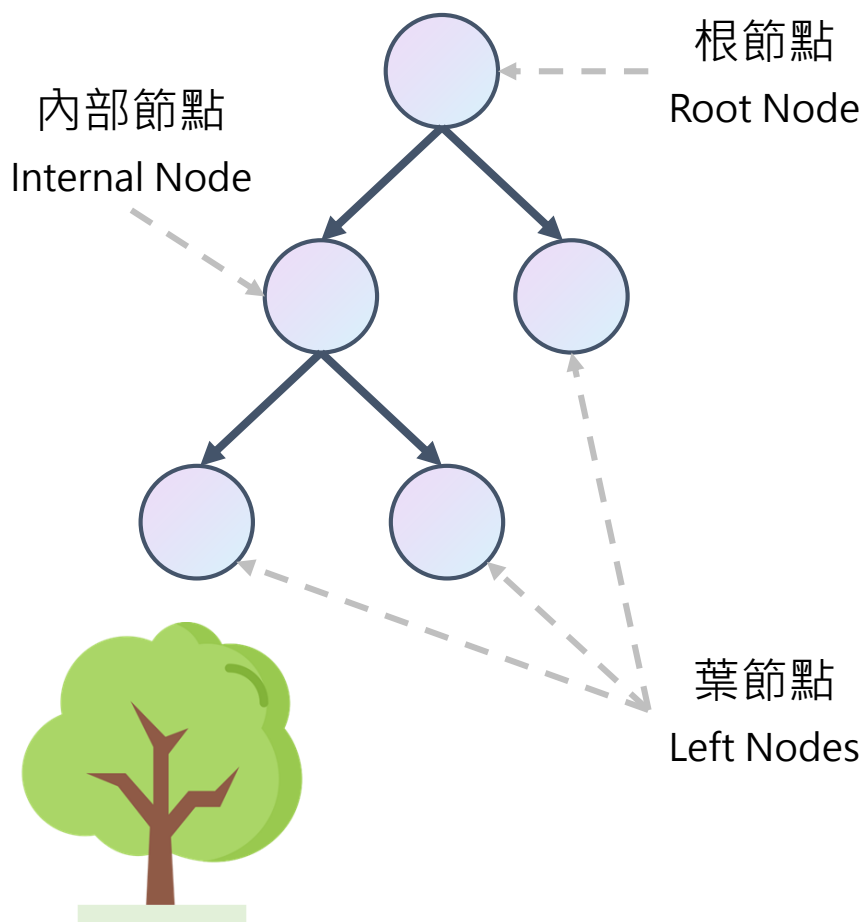
- 在每一次迭代計算損失時，皆從所有樣本中取出部分(非單一個)的樣本進行計算。

# 樹模型 Tree-Based Model

- 利用條件式分割(Conditional Branch)，逐步切分出越來越低雜亂度的子資料集，且將這一序列的分割過程，建立成一顆樹狀結構的決策過程。



## 決策樹的樹狀結構



- 決策樹是一種監督式的機器學習演算法，可用來處理迴歸與分類問題。

### ● 基本概念：

將複雜的資料集，按照一系列的規則，分割成越來越單純的子資料集，這一系列分割的過程從而形成一個樹狀的結構。其中，建構決策樹的重要關鍵，在於如何選擇最適合的特徵作為分割的依據。

另外，衡量子資料集單純程度的指標有：

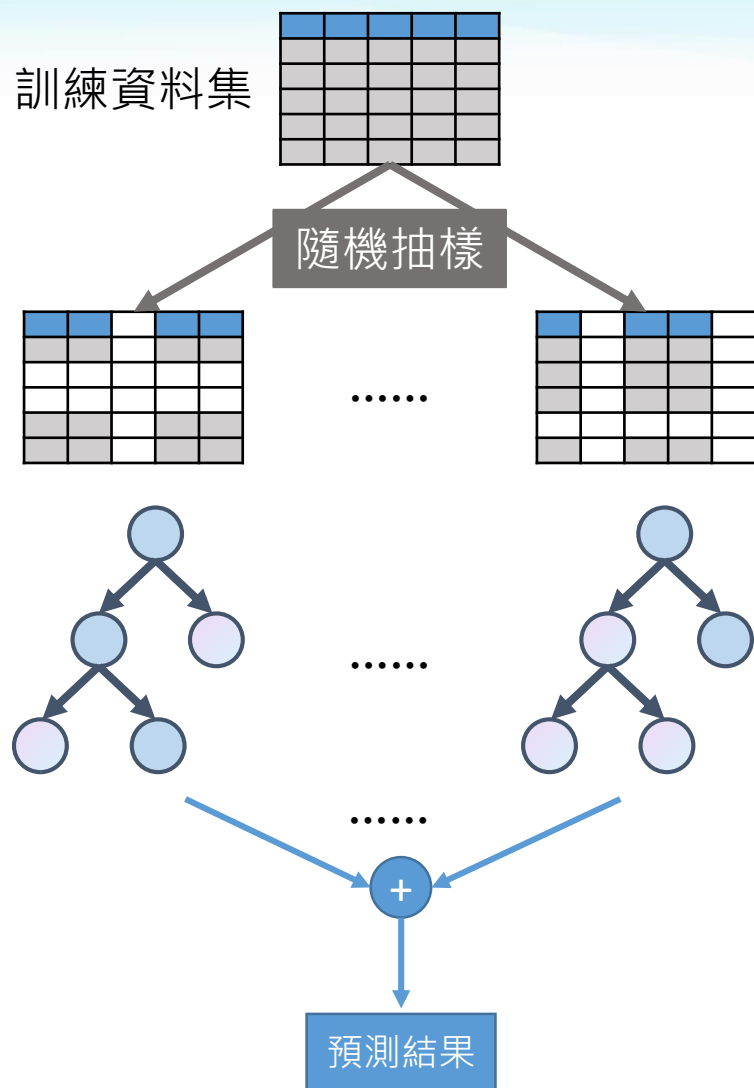
- 資訊增益 [Information Gain, IG](#)
- 資訊增益比 [Gain Ratio](#)
- 吉尼不純度 [Gini Impurity](#)

### ● 優點：

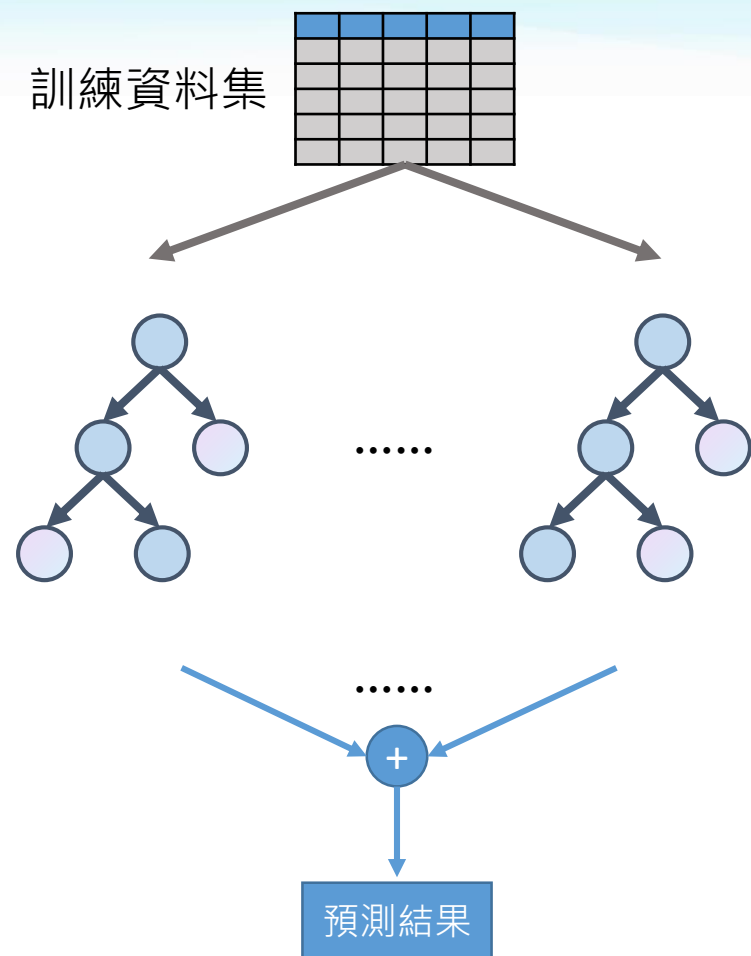
- 數值型與類別型特徵皆可適用於決策樹。
- 決策樹易理解且解釋性高，可從中觀察決策的過程。
- 決策樹能處理非線性的關係。
- 簡單的資料前處理：特徵不要求須標準化 Standardization。

### ● 注意事項：

- 若樹的結構太過複雜，易造成過擬合(Overfitting)的情況。
- 當特徵的類別過多時，易建構出不平衡的偏差樹結構。



- 隨機森林係一種採用集成學習裝袋法 Bagging 的演算法，可用來處理迴歸與分類問題。
- 核心概念：
  - 基於樹的集成學習，從多顆決策樹中綜合預測結果來提高準確性，主要的概念有二：
  - (1) 結合多個弱學習器，建構一個強學習器：
    - 預測結果係由多棵決策樹所綜合而來，一般而言，迴歸問題採用平均機制，分類問題採用投票機制。
  - (2) Bootstrapping 隨機抽樣：
    - 在建構每棵樹時，會隨機抽樣部分樣本與特徵，每棵樹皆只會看到訓練資料集的一部分。
- 優點：
  - 相較於決策樹，隨機森林較不易過擬合(Overfitting)。
  - 相較於決策樹，隨機森林的泛化能力(Generalization Ability)較高，避免只因依賴單一顆決策樹所發生的偏差。
- 注意事項：
  - 相較於決策樹，隨機森林需要更多的運算資源。
  - 因為每一棵樹皆是基於訓練資料集中的一部分所建立，因此，隨機森林的預測能力易受限於訓練資料集的範圍。



- 極限隨機樹可用來處理迴歸與分類問題。

- **核心概念：**

與隨機森林相似，同樣係基於樹的集成學習，從多顆決策樹中綜合預測結果來提高準確性，與隨機森林的主要差別有二：

**(1) 使用完整的訓練資料集：**

隨機森林採用 Bagging 的訓練機制，每一顆樹皆只從部分的訓練資料中所建立，然而，極限隨機樹中的每一棵樹，皆使用完整的訓練資料來進行訓練。

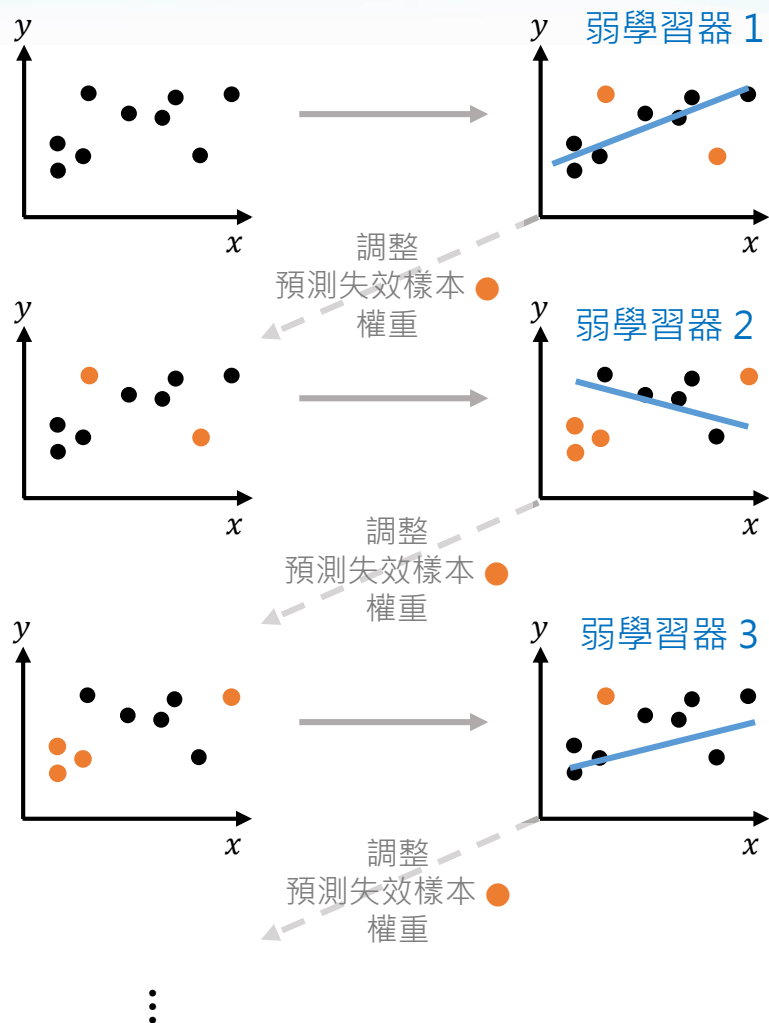
**(2) 節點特徵的選擇：**

隨機森林會從部分特徵中，選擇一個最佳的分割特徵，而，極限隨機樹在分類問題中，係從所有的特徵中隨機尋找到一個特徵進行分割(註：隨機先挑  $k$  個，再從  $k$  個中挑選最佳分割的特徵)，另一方面，在迴歸問題中，先在每個特徵中隨機選擇一個分割點，再從其挑選出能使預測效果最佳(選擇均方誤差最低)之特徵進行分割。

- **注意事項：**

- 雖然極限隨機樹的隨機性機制，能有效抑制過擬合的發生，但是，在節點的特徵選擇上，若隨機選擇了過多不相關的特徵，則也容易造成方差 (Variance) 過高的情況。

# 自適應提升 Adaptive Boosting



- 自適應提升係一種採用集成學習[提升法 Boosting](#) 的演算法，可用來處理迴歸與分類問題。
- 核心思維：
 

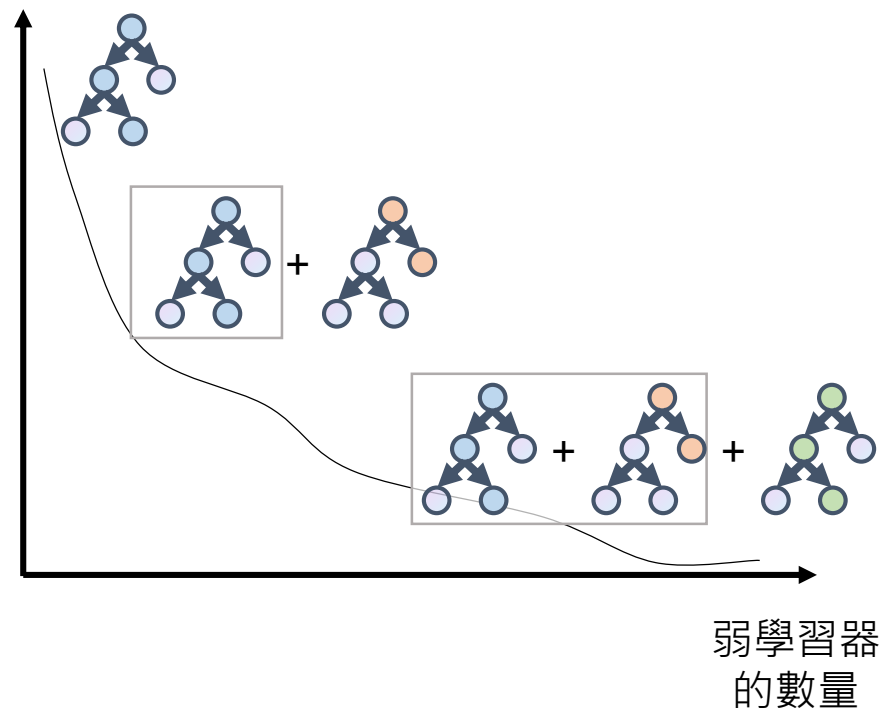
採用序列化的方式，逐一訓練每個弱學習器，最後，綜合這多個弱學習器的預測結果，從而建構出一個強學習器，訓練自適應提升模型的基本概念為：

  - 樣本權重的調整：
 

在每一個新的弱學習器的訓練中，著重於前一個弱學習器預測失效的樣本，調整這些樣本的權重，使得每一個新的弱學習器能更重視前一個弱學習器預測失效的樣本。
- 注意事項：
  - 在自適應提升的訓練過程中，因為會不斷更新樣本權重，提高較難以預測的樣本權重，所以，自適應提升易受離群值(或異常值)的影響，須格外重視離群值(或異常值)的資料預處理。

# 梯度提升 Gradient Boosting

誤差



- 梯度提升係一種採用集成學習提升法 Boosting 的演算法，可用來處理迴歸與分類問題。

- **核心概念：**

採用序列化的方式，逐一訓練每個弱學習器，在訓練每一個新的弱學習器時，嘗試修正前一個弱學習器的預測殘差，最後，綜合這多個弱學習器的預測結果，從而建構出一個強學習器。梯度提升模型的基本概念為：

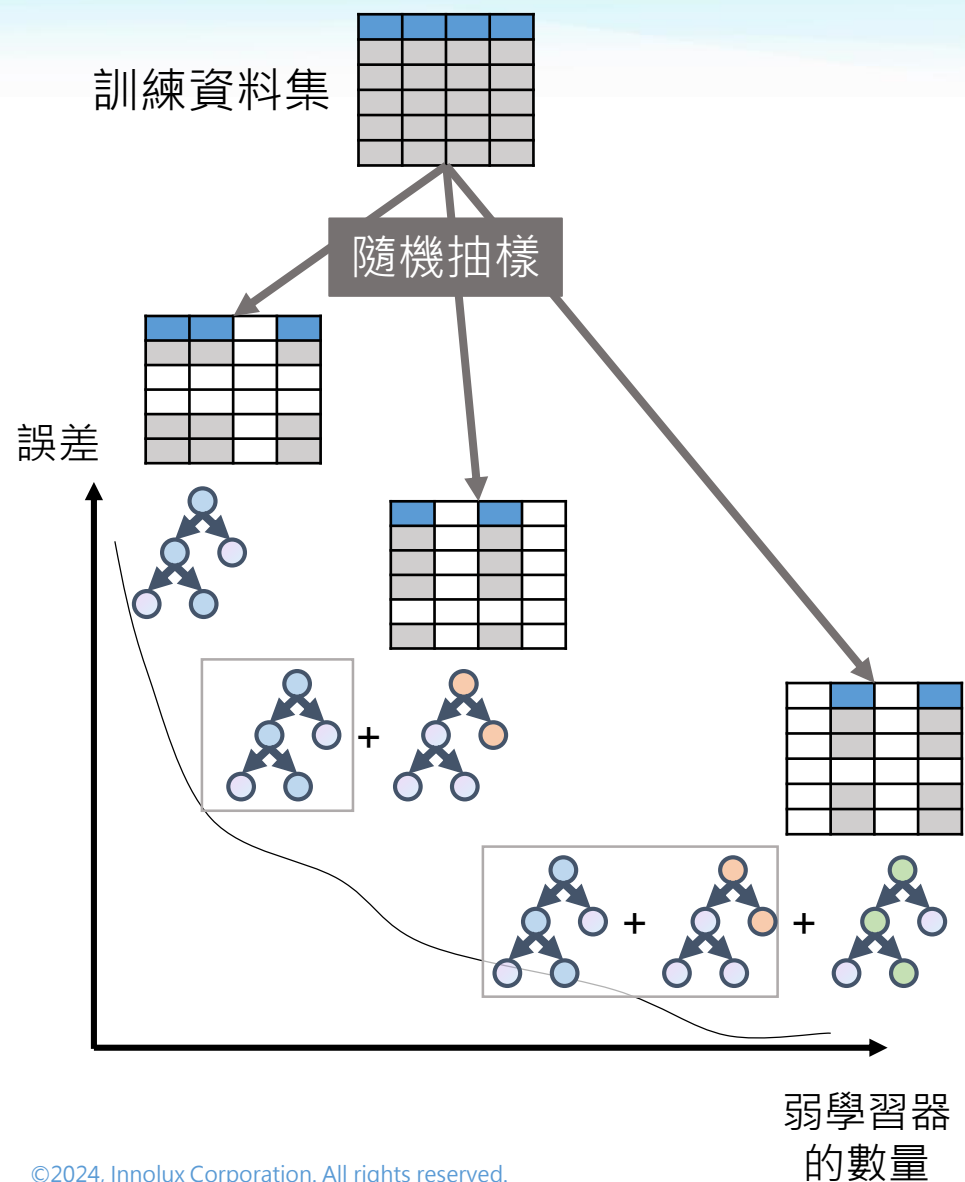
- **透過分布算法 (Stagewise Additive Modeling) 輔以梯度下降 (Gradient Descent)，逐步最小化損失函數：**

分布算法係假設理論且完美的模型，由多個基底函數(i.e., 弱學習模型)組合而成，而且，在訓練每一個基底函數時，其參數係藉由梯度下降法來最小化每一次的訓練誤差。

- **注意事項：**

- 當弱學習器採用 CART (Classification and Regression Tree) 決策樹時，則這樣的梯度提升稱為梯度提升決策樹 Gradient Boosting Decision Tree，簡稱 GBDT。

# 極限梯度提升 Extreme Gradient Boosting



- 極限梯度提升屬於集成學習(Ensemble Learning)的一種演算法，可用來處理迴歸與分類問題，簡稱 XGBoost。

- 核心概念：

主要基於梯度提升決策樹 GBDT 的思維，逐一訓練每個弱學習器 – 決策樹，且組合這些弱學習器成一個強學習器，此外，極限梯度提升還有下列二項重要的改進與優化：

(1) 損失函數加入正則項：

損失函數添加正則懲罰項，主要目的係在避免模型過度複雜，降低過擬合(Overfitting)的風險。在訓練模型時，若模型結構過度複雜，則易受雜訊干擾而導致過擬合。

(2) 隨機採樣：

在訓練每棵樹時，會隨機抽樣部分樣本與特徵，每棵樹皆只用部分的訓練資料來進行訓練。

- 優點：

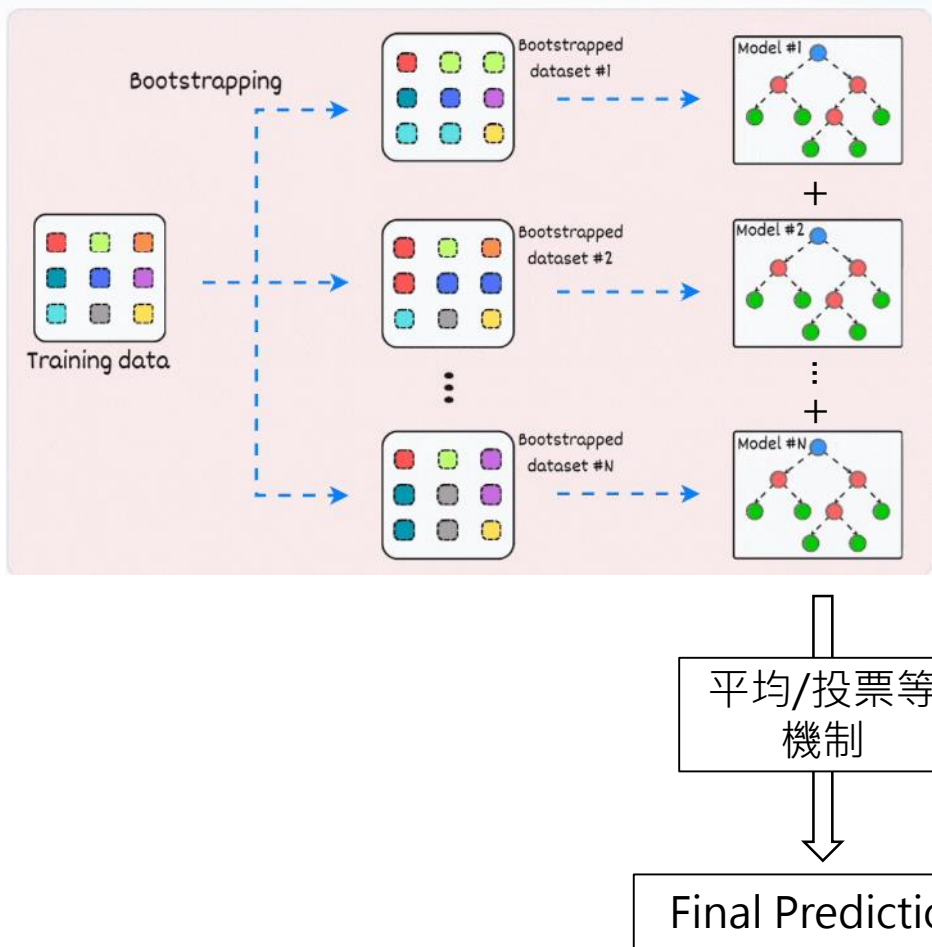
- 泛化能力(Generalization Ability)佳。
- 可平行運算進行訓練，減少運算資源的耗費。

# 集成學習 Ensemble Learning

- 係一種結合多個弱學習器，產生一個較強學習器的建模技巧。
- 能有效提升泛化能力，且降低過擬合的風險。
- 三種集成的方法：裝袋法、提升法、堆疊法。



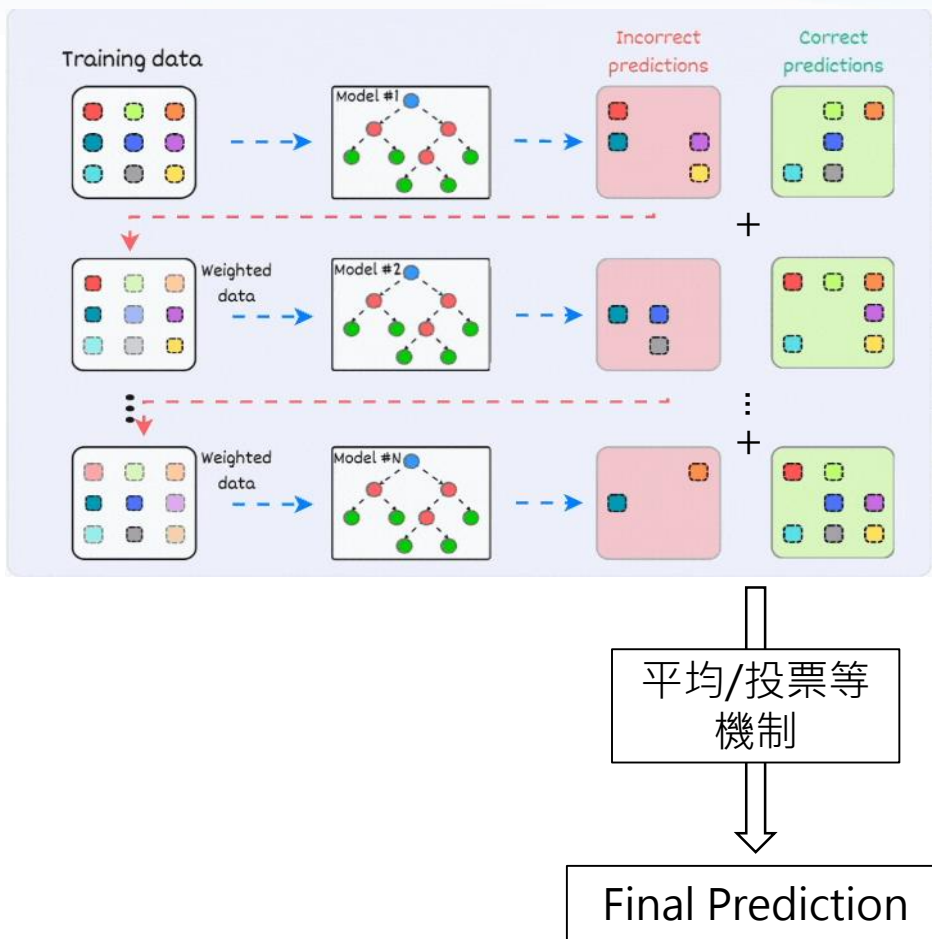
# 集成學習－裝袋法 Bagging



- 裝袋法 Bagging (全名為 Bootstrap Aggregating) 是一種集成學習的方法。
- 基本概念：  
藉由取後放回的隨機抽樣方式 - [Bootstrapping](#)，從原始訓練資料集中取出多個子訓練資料集，且在每個子訓練資料集中獨立訓練一個弱學習器，最終，透過機制綜合每個弱學習器的預測，例如：平均(迴歸問題)、投票(分類問題)等，將這些弱學習器組合成一個強學習器。
- 採用裝袋法的機器學習模型：
  - 隨機森林 Random Forest



# 集成學習－提升法 Boosting



- 提升法 Boosting 是一種集成學習的方法。

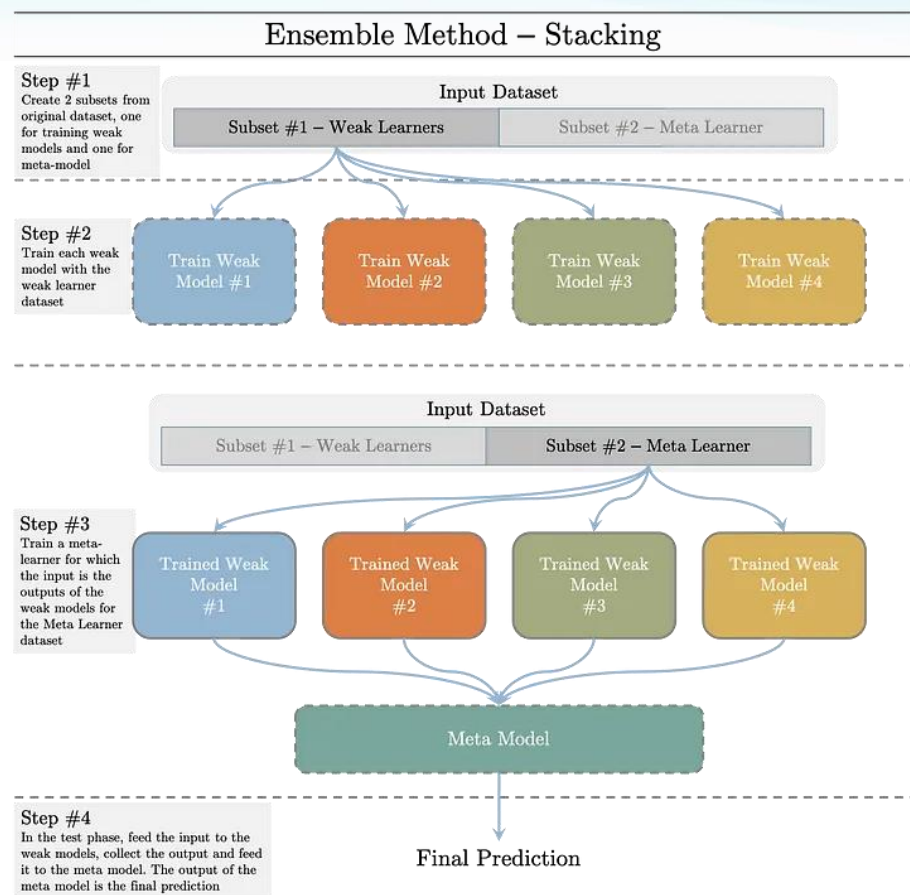
- **基本概念：**

以串聯的方式依序生成多個弱學習器，每個弱學習器皆會與前一個弱學習器有關聯，每個弱學習器會關注前一個弱學習器預測錯誤的樣本，給予更大的權重改進前一個弱學習器的錯誤，嘗試逐步提升後續的弱學習器表現，最終，透過綜合每個弱學習器的預測機制，例如：平均(迴歸問題)、投票(分類問題)等，將這些弱學習器組合成一個強學習器。

- **採用提升法的機器學習模型：**

- 梯度提升 Gradient Boosting
- 自適應提升 Adaptive Boosting, AdaBoost
- 極限梯度提升 Extreme Gradient Boosting, XGBoost

# 集成學習 – 堆疊法 Stacking



- 堆疊法 Stacking 是一種集成學習的方法。
- 相較於裝袋法 Bagging 與提升法 Boosting，堆疊法 Stacking 的架構更複雜，但也較常表現得更好。
- **基本概念：**  
首先，訓練多個弱學習器，例如：決策樹、支持向量機等，再將這多個弱學習器的預測結果作為新的特徵，最後，藉由這新的特徵用其訓練一個元模型 Meta Model 作為最終的預測模型。
- **堆疊法的訓練過程：**  
Step 1：將訓練資料集分成兩個子資料集；  
Step 2：在第一個子資料集中，訓練多個弱學習器；  
Step 3：在第二個子資料集中，取得每個弱學習器的預測結果，且視為新的特徵；  
Step 4：在新的特徵中，訓練元模型 Meta Model，通常元模型是一個簡單的模型，例如：線性迴歸。

**機器學習** 的極限取決於 **數據** 與 **特徵**，

而，模型和演算法僅是逼近極限的方法。

了解如何將 **數據分析** 與 **機器學習** 應用於您的業務！  
請立即與我們聯繫，發掘數據資料中的無限可能！！

## CONTACT US



黃齡誼 [lisa.ly.huang@innolux.com](mailto:lisa.ly.huang@innolux.com)



張訓漢 [xunhan.zhang@innolux.com](mailto:xunhan.zhang@innolux.com)



盧提文 [tiffany.tw.lu@innolux.com](mailto:tiffany.tw.lu@innolux.com)



吳彥霖 [yenlin.wu@innolux.com](mailto:yenlin.wu@innolux.com)