目次

[Module 1. 機器學習的常見概念 2](#_Toc180455924)

[資料基本用語 2](#_Toc180455925)

[機器學習的種類 2](#_Toc180455926)

[機器學習的流程 3](#_Toc180455927)

[垃圾進，垃圾出（Garbage In, Garbage Out） 4](#_Toc180455928)

[套件安裝 4](#_Toc180455929)

[Module 2. 資料前處理 5](#_Toc180455930)

[數值型資料 5](#_Toc180455931)

[StandardScaler 5](#_Toc180455932)

[MinMaxScaler 7](#_Toc180455933)

[RobustScaler 8](#_Toc180455934)

[Normalizer 9](#_Toc180455935)

[類別型資料 11](#_Toc180455936)

[名目特徵（Nominal Features） 11](#_Toc180455937)

[有序特徵（Ordinal Features） 12](#_Toc180455938)

[獨熱編碼（One-Hot Encoding） 12](#_Toc180455939)

[二進制編碼（Binary Encoding） 12](#_Toc180455940)

[LabelBinarizer 13](#_Toc180455941)

[LabelEncoder 14](#_Toc180455942)

[遺漏值（又稱缺失值，missing value） 14](#_Toc180455943)

[SimpleImputer 15](#_Toc180455944)

[KNNImputer 15](#_Toc180455945)

[K-fold Cross Validation 16](#_Toc180455946)

[異常值 17](#_Toc180455947)

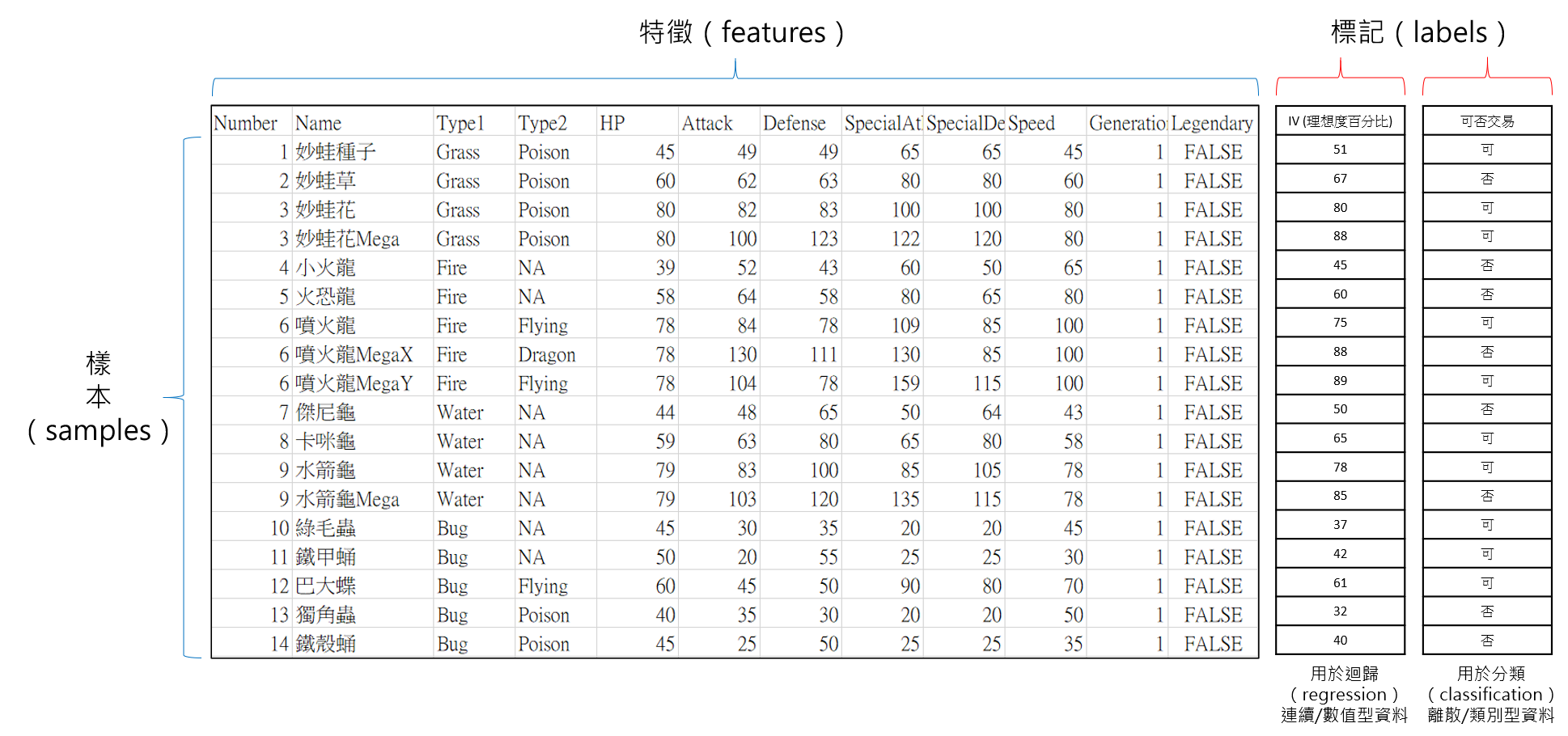
* GitHub 專案連結

# Module 1. 機器學習的常見概念

能夠從現有的資料（data）當中，不斷地從中學習經驗（experience），找出共通的模式（pattern），以便引導未來的決策。在學習這門知識之前，強烈建議先行研讀統計學（含機率）、微積分以及線性代數，可以較容易地理解模型運作過程，以及它們背後的原理。

## 資料基本用語

資料（有時稱之為「數據」）的基本用語如下：



圖：資料的基本用語

註：樣本有時也會被稱為「觀察值」。

## 機器學習的種類

根據解決的任務類型不同，所採用的機器學習方法，可粗略地分為三種：

* 監督式學習（supervised learning）：  
  透過已知的標記（label）來建立模型，可以對未見的樣本資訊，來預測出它可能的答案（預測未見樣本的標記）。比較常見的監督式學習任務是分類（classification）和迴歸（regression）。在分類任務中，程式必須學習從一個或多個特徵來預測一個或多個離散值，例如預測一支股票的價格會「上漲」或「下跌」，或是決定一篇新聞屬於「政治版」還是「娛樂版」等；在迴歸任務中，程式必須學習從一個或多個特徵來預測一個或多個連續型數值，例如預測一個新產品的銷售收入，或是根據一個職務的描述來預測其薪水。
* 非監督式學習（unsupervised learning）：  
  在樣本沒有標記或不考慮標記的情況下，透過既有的特徵（feature）來探索資料的結構，進而發現有用的資訊。比較常見的非監督式學習任務，就是在資料集當中，發現互相關聯的樣本，基於一些相似性衡量標準，把樣本放在和其它群組（group）相比，更加類似的群組當中，這樣的過程稱之為分群（cluster），例如在市場中為某項產品發現客戶群體，透過了解特定客戶群體的共同屬性，銷售人員可以決定採用何種銷售策略，用於該客戶群體。
* 強化學習（reinforcement learning）：  
  經由與環境互動產生的獎勵（reward）來改善模型的效能，驅使模型朝既定的目標前進。例如下象棋，每吃掉對方一個棋子作為正面獎勵，被吃掉為負面獎勵，根據棋盤狀態以及先前落子的順序，來決定下一步的程動行為（action），透過不斷地反饋與獎勵進行學習。

## 機器學習的流程

1. 收集資料：  
   明確地定義問題，同時收集相關的資料，是機器學習的第一步。資料的完整性對模型的預測效能有著關鍵性的影響，除了資料量要足夠大以外，代表性也很重要。資料的收集夠完整，才能順利進行後續的分析，這部分往往都是最耗時間。
2. 資料前處理：  
   真實世界的資料是非常雜亂的，經常會有缺漏、雜訊，或是資料不平衡的情況發生，於是需要進行探索性資料分析（exploratory data analysis，EDA），大致上瀏覽資料的樣式與分佈程度，依需求進行資料清理（data cleaning），將資料整理成特定的格式，也可以透過特徵工程（feature engineering）來抽取、轉換或產生重要的特徵，方便之後的建模工作。
3. 訓練模型與調校：  
   在訓練模型之前，會將整個資料集分成三個部分，一個是訓練資料集（training dataset），一個是驗證資料集（validation dataset），一個是測試資料集（testing dataset）。一個完整的資料集，通常訓練資料集佔50% ~ 75%，測試資料集佔 10% ~ 25%，剩餘的部分，就是驗證資料集。資料切分完成後，依照問題的類型，選擇一個或數個機器學習模型來擬合（fit）訓練資料，藉由擬合過程來學習模型的參數，以建立最終版本的模型。大部分機器學習的模型，都需要使用者預先提供的超參數（hyperparameters），這可以透過調校來取得，有效地發揮模型潛力。
4. 測試模型：  
   模型建立完成後，會使用先前切分出來的測試資料集，依照效能量測指標進行測試，來評估（evaluation）模型擬合結果的好壞，再從幾個模型當中挑選相對合適的模型。
5. 改善效能：  
   不斷地重複觀察與調整 3、4 步驟的過程來進行改善；必要時，可以重新進行 1、2 步驟，來取得更多有利於資料規律的特徵。

## 垃圾進，垃圾出（Garbage In, Garbage Out）

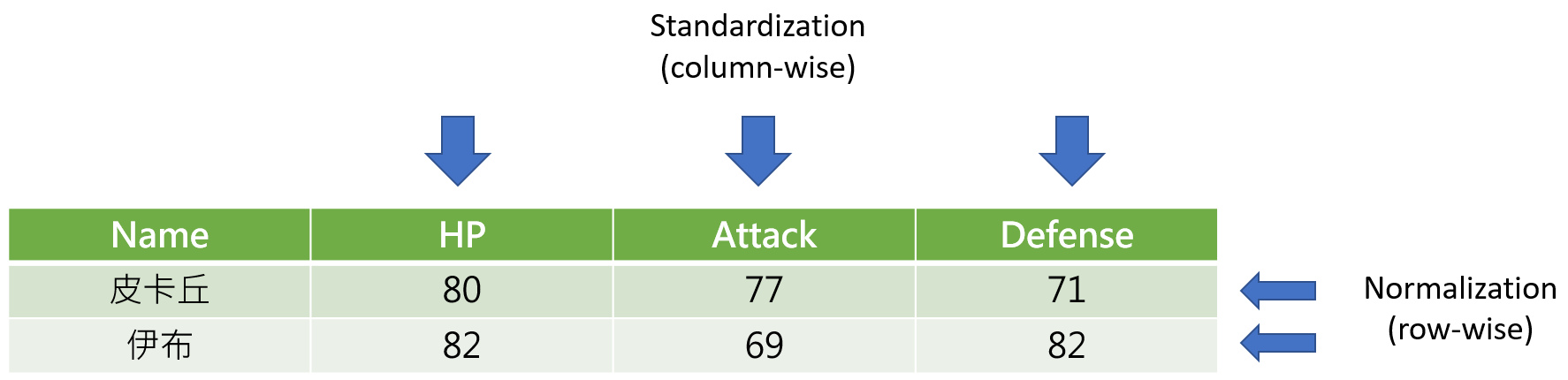
許多模型預測能力的提升，有賴於訓練資料數量的增加。然而，機器學習演算法（algorithm）遵循著格言「Garbage In, Garbage out」，假如一個學生透過閱讀一本錯誤百出、令人困惑的教材來準備考試，他的成績，並不會比閱讀教材篇幅短少，但內容質量較高的學生的成績來得好。在現實世界裡，如果在一個包括噪音（noise）、不相關或是帶有錯誤標記的資料集上進行訓練，其表現不會比一個在更能代表問題的小型資料集上所訓練的模型來得好。

## 套件安裝

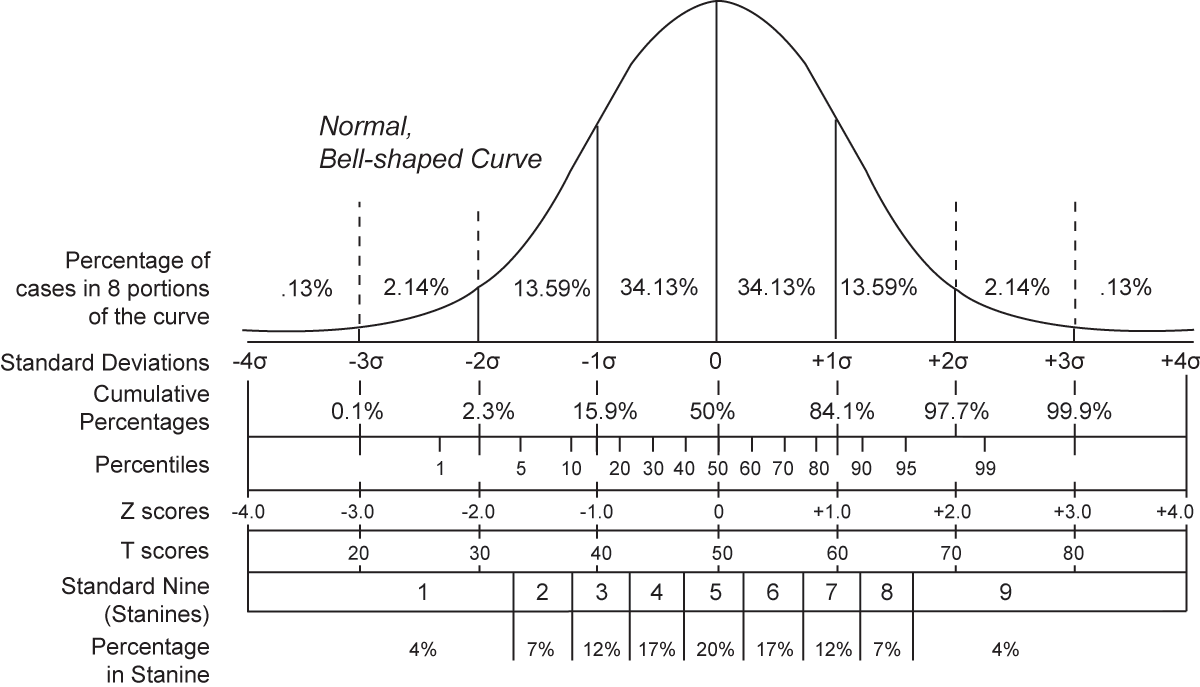
|  |
| --- |
| 指令 |
| pip install -U scikit-learn matplotlib |

# Module 2. 資料前處理

## 數值型資料



圖：對欄位進行標準化，對樣本進行正規化



圖：取自 https://zh.wikipedia.org/zh-tw/標準分數

### StandardScaler

一種常見的標準化方法，用來將資料進行縮放和中心化處理，使資料的平均值（mean）變為 0，標準差（standard deviation）變為 1，大部分的資料會集中在 [-1, 1] 的範圍內，通常大約 68% 的資料點會落在這個範圍。這種方法通常應用於需要使不同特徵保持一致資料範圍的情境，例如在機器學習模型中，確保不同特徵不因為範圍不同而對模型產生不均衡的影響。

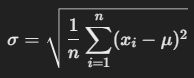
平均值為 0、標準差為 1 的資料通常指的是「**標準化**」後的資料，也稱為「**Z-score 標準化**」或「**Z 標準化**」。這是一種常用的資料預處理方法，尤其在機器學習模型訓練中很常見。讓我們深入了解這兩個概念的具體意義。

「平均值為 0」的意思是，資料經過轉換後，其所有資料點的**平均值**被調整到 0。計算平均值的公式是：



經過標準化後，資料的整體中心被平移到 0，表示資料的中心點與原點對齊。換句話說，經過標準化後，大部分資料會集中在零附近，資料點大約一半位於零的左邊，另一半位於右邊。

「標準差為 1」的意思是，資料經過標準化後，其**離散程度**（分散程度）調整到 1。標準差衡量資料點相對於平均值的偏離程度，公式為：



當標準差為 1 時，資料的離散程度被縮放，使得資料點的變化範圍在 1 個標準差內。也就是說，絕大多數資料點都會落在 0 的附近，並在 -1 到 1 之間波動。

標準化資料的公式如下：



* 是標準化後的數值。
* 是原始數值。
* 是資料的原始平均值。
* 是資料的原始標準差。

為什麼要標準化？

* **使模型對不同特徵同等重視**：  
  如果資料中的不同特徵有著不同的範圍（例如收入用元表示，年齡用年表示），那麼在模型訓練過程中，某些數值較大的特徵可能會過分影響模型。標準化可以解決這個問題，讓所有特徵處於相同的範圍。
* **加速模型收斂**：  
  某些機器學習演算法（如梯度下降法）在資料標準化後，收斂速度會更快，從而提高訓練效率。
* **有利於距離度量**：  
  在需要計算距離的模型（如 KNN 或 SVM）中，標準化確保距離計算不會被數值較大的特徵主導。

注意事項

* **異常值的影響**：

StandardScaler 使用均值和標準差進行標準化，因此對於包含極端異常值的資料集，這些異常值可能會對均值和標準差產生顯著影響，從而影響資料的標準化效果。如果資料集中存在異常值，應該考慮使用對異常值更為穩健的標準化方法，如 RobustScaler。

* **資料分佈的假設**：

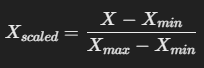
StandardScaler 假設資料接近常態分佈。在資料分佈非常偏態（skewed）的情況下，標準化後的資料仍然可能存在一些問題，這時可能需要進行額外的資料處理。

StandardScaler 是將資料縮放為均值 0、標準差 1 的標準化方法，特別適合資料接近正態分佈的情況。它有助於機器學習模型的訓練過程，尤其是對於需要統一特徵範圍、提升收斂速度以及確保特徵影響均衡的模型。在資料集不包含異常值且資料接近正態分佈時，StandardScaler 是一個非常有效的標準化工具。

### MinMaxScaler

一種常見的資料標準化方法，它將資料縮放到指定的區間（通常是 [0, 1] 之間）。MinMaxScaler 的核心概念是**線性縮放**，即將資料中的最小值轉換為 0，最大值轉換為 1，其餘資料根據其相對位置映射到這個範圍內。

MinMaxScaler 的標準化公式如下：



* 是原始資料值。
* 是資料集中該特徵的最小值。
* 是資料集中該特徵的最大值。
* 是經過標準化後的資料值。

注意事項

* 異常值的影響：  
  MinMaxScaler 對異常值較為敏感，因為它的縮放是基於最大值和最小值。如果資料中有極端值，這些異常值會對縮放範圍產生很大的影響。例如，如果資料中存在非常大的異常值，則其他正常資料可能會被壓縮得非常小。因此，如果資料集中存在異常值，應考慮先去除異常值或使用對異常值更為穩健的標準化方法，如 RobustScaler。
* 資料分佈的假設：  
  MinMaxScaler 假設資料的分佈是均勻的。當資料的分佈非常不均勻時（例如，大部分資料集中在某個範圍，而少數資料遠離中心），可能會導致縮放後資料分佈過於壓縮。因此，對於這類資料分佈，應考慮其他標準化技術。

對於每一個特徵，MinMaxScaler 都會找到該特徵的最小值和最大值，然後將這些資料值縮放到指定的區間內。預設的區間是 [0, 1]，但也可以自定義範圍，比如 [-1, 1]。計算過程是將資料點相對於最小值和最大值的位置映射到 0 和 1 之間，從而保證資料不超出這個範圍。

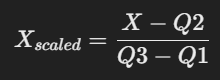
### RobustScaler

一種資料標準化的方法，它的目的是對資料進行縮放和中心化處理，但與傳統的標準化（如 StandardScaler）不同，RobustScaler 對異常值（outliers）的影響較小。這使得它特別適合處理具有極端數值的資料集。

核心思想是使用中位數（median）和四分位距（IQR，Interquartile Range）來進行縮放，這與 StandardScaler 使用的均值和標準差不同。

* 中位數：是資料集中間的數字，對極端資料點（異常值）不敏感。
* 四分位距（IQR）：是上四分位數（75% 分位數）與下四分位數（25% 分位數）之間的距離。它是資料分布範圍的一個穩健度量，同樣對異常值的影響較小。

RobustScaler 的標準化公式是：



* 是原始資料。
* 是中位數。
* 是第 75 個百分位數（上四分位數）。
* 是第 25 個百分位數（下四分位數）。
* 是四分位距。

為什麼使用 RobustScaler？

* 對異常值（outliers）的抗性：

傳統的標準化方法，如 StandardScaler，是基於均值和標準差，這意味著資料集中存在的異常值會極大地影響均值和標準差，從而影響資料的標準化效果。而 RobustScaler 基於中位數和四分位距進行縮放，這兩個指標不受極端值的影響，因此能夠更加穩健地處理資料中的異常值。

* 適合具有異常值的資料集：

如果資料集中存在極端值，使用 StandardScaler 會導致縮放後的資料不均衡，影響模型的表現。而 RobustScaler 可以有效減少這些極端值對資料標準化的影響，保證資料縮放後更加穩定。

RobustScaler 的應用場景

* 帶有極端資料點的資料集：  
  當你的資料集包含一些異常值，而這些異常值可能會影響模型性能時，RobustScaler 是一個很好的選擇。
* 資料範圍變動較大：  
  如果你的資料中存在大量分散且範圍寬廣的資料，RobustScaler 能夠在保持穩定性的同時，進行合理的縮放。

### Normalizer

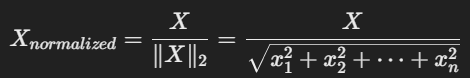
主要目的是將每個資料點（通常是向量）進行向量範數正規化，使得每個資料點的向量長度（即範數）為 1。與其他標準化方法（如 StandardScaler 或 MinMaxScaler）不同，Normalizer 是針對每一個資料點（樣本）進行處理，而不是針對特徵（行）。

Normalizer 的概念

Normalizer 通過將每個資料點縮放，使得該資料點的範數（norm）變為 1，這樣每個樣本都被縮放到單位範圍內。常用的範數有：

* L2 範數（歐幾里得範數）：也稱為平方和範數，通常用於標準的向量正規化，表示向量的平方和的平方根為 1。
* L1 範數（絕對值範數）：表示向量的元素絕對值之和為 1。
* Max 範數：即取向量中的最大絕對值進行縮放，使其最大值為 1。

對於一個資料點（向量），L2 範數的正規化公式為：



這意味著，資料點中的每個元素都會除以該資料點的 L2 範數，使得結果向量的總長度為 1。

Normalizer 的工作原理

* 作用對象是資料點（樣本）：

Normalizer 是針對每個資料點進行處理的，不像其他標準化方法那樣針對整個特徵列。每個資料點都被縮放，使其向量範數為 1。

* 使用範數進行縮放：

Normalizer 根據指定的範數（L1、L2 或 Max）將每個樣本進行縮放，使得資料的整體方向保持不變，但範數為 1。

* 保持資料方向：

Normalizer 只改變資料的範數，保持資料的方向。這意味著原資料中兩個資料點之間的方向關係不變，只是資料點的長度被縮放到單位範圍內。

什麼時候使用 Normalizer？

* 稀疏資料：

Normalizer 常用於稀疏資料集，如文本資料或詞袋模型中的詞頻資料。當你有一組向量，並且更關心資料的方向或模式而不是其大小時，Normalizer 是理想選擇。

* 保持資料的幾何形狀：

如果你的模型（例如基於距離的模型，如 KNN 或餘弦相似度）更關心向量之間的相對幾何形狀（方向）而不是數值的絕對大小，Normalizer 能幫助統一向量的大小。

* 特徵向量的處理：

在處理特徵向量時，尤其是高維度資料如圖像特徵、文本向量表示時，使用 Normalizer 來使每個資料點的向量保持一致的長度，可以讓模型更關注於特徵之間的相對關係，而不是數值本身的大小。

注意事項

* 範數的選擇：
  + 默認的範數是 L2，但在某些應用中，如當資料具有稀疏性時，L1 範數有時會更合適，因為它強調非零特徵的重要性。
  + Max 範數對於有極端值的資料可能更適合，因為它只考慮最大值來進行縮放。
* 與其他標準化技術的區別：
  + Normalizer 是針對每個樣本進行正規化，而其他標準化技術（如 StandardScaler 和 MinMaxScaler）則是針對每個特徵列進行處理。
  + Normalizer 保證了每個樣本向量的範數相同，但它不會改變資料點之間的相對大小（與標準化特徵的技術不同）。

Normalizer 是一種針對向量進行範數正規化的技術，主要用於保持資料點的方向一致，並使每個資料點的範數變為 1。它常用於需要保持資料相對幾何關係的場景，特別是當我們更關心資料點之間的方向或相似度時。

## 類別型資料



圖：樣本資訊透過 1 和 0 組合而成的稀疏向量作為表示

### 名目特徵（Nominal Features）

名目特徵是沒有內在順序的類別型數據。例如，血型就是名目特徵，因為「A型」、「B型」、「AB型」、「O型」之間沒有大小、次序的關係。這些值只是表示不同的類別而已。

### 有序特徵（Ordinal Features）

有序特徵則是帶有內在順序的類別型數據。例如，如果我們在某個場景中有評分等級（如「差」、「中」、「好」），這些數值是有順序的。這張圖片中的數據似乎不涉及有序特徵，因為血型沒有順序關係。

### 獨熱編碼（One-Hot Encoding）

獨熱編碼是一種常用來將類別型數據轉換為數值的方法。對於每一個類別，獨熱編碼會創建一個二進制向量，在對應類別的位置上設置為1，其他位置設置為0。例如，血型「A」的獨熱編碼是1 0 0 0，而「B」的獨熱編碼是0 1 0 0。

獨熱編碼的缺點：

* 高維度問題：  
  如果類別數非常多，獨熱編碼會導致數據的維度急劇增加，引發「維度災難」（curse of dimensionality），這會帶來內存和計算資源的消耗問題。
* 稀疏矩陣：  
  獨熱編碼產生的矩陣通常是稀疏的（大多數元素為0），在機器學習模型中處理稀疏矩陣可能不夠高效。

### 二進制編碼（Binary Encoding）

二進制編碼將類別轉換成整數，然後將這些整數轉換為二進制表示。例如血型A 對應的二進制是00，血型 B 對應的是01，血型 AB 是10，血型 O 是11。這種編碼方式可以有效地減少維度，因為我們用更少的位數表示類別（對比獨熱編碼的高維度）。

獨熱編碼和二進制編碼各有其適用場景：

* 獨熱編碼：  
  適合於類別數目較少的數據，並且適合模型能處理高維度的情況。
* 二進制編碼：  
  適合於類別數量龐大的場景，因為它可以大大減少編碼維度。

選擇使用哪種編碼方式取決於數據的性質和模型的需求，獨熱編碼容易理解且適用範圍廣，但當類別數較多時，它的維度會變得過高。

### LabelBinarizer

專門用來將類別型數據轉換成二進制表示。它常用於對類別型標籤進行處理，尤其是在多分類問題中。這個方法與獨熱編碼（One-Hot Encoding）類似，但也可以進行多重標籤（multi-label）的二進制化處理。

LabelBinarizer 的功能

* 將多類別標籤轉換為二進制數組。
* 在二分類情況下，則只返回一個二進制列。

主要特點：

* 二分類模式：  
  如果只有兩個類別，LabelBinarizer 會將數據轉換為單列的二進制數組。例如，對於標籤 ['A', 'B']，它會輸出 [0, 1] 或 [1, 0]。
* 多類別模式：  
  對於多個類別（如三個或更多），它會產生與獨熱編碼類似的結果，但不同於獨熱編碼，LabelBinarizer 更加靈活，特別是在處理多標籤問題時。
* 適用於多標籤數據：  
  LabelBinarizer 也支持將多標籤數據轉換為二進制表示。這與多標籤分類有關（比如一個樣本可以屬於多個類別），在這種情況下，它可以將多標籤數據表示成一個二維二進制矩陣。

LabelBinarizer 的優點：

* 簡單易用：它能夠簡化標籤轉換過程，尤其在進行分類任務時。
* 靈活性高：能夠處理多標籤問題，並根據不同的需求調整輸出格式。

LabelBinarizer 是一個簡便的工具，可以將類別標籤轉換為二進制格式，尤其適合在需要對標籤進行二進制化的情況下使用，並且它可以處理多標籤和多分類的數據，讓數據更容易被模型所接受。

### LabelEncoder

專門用來將類別型數據轉換成數字標籤，它將每個類別標籤映射到一個唯一的整數值。

LabelEncoder 的功能

* 將類別型標籤（如文字標籤）轉換成對應的數字編碼。
* 適合於那些具有名目特徵的類別型數據。

LabelEncoder 的主要用途是處理類別標籤，使得它們能夠被大多數機器學習模型直接接受，因為大部分模型只接受數值型數據，而不是文字標籤。

工作原理

例如，假設你有一組類別標籤 ['A', 'B', 'C']，LabelEncoder 會將這些類別標籤映射到對應的整數，如：

A -> 0

B -> 1

C -> 2

主要特點

* 單列數據：  
  LabelEncoder 是針對單一特徵進行轉換的，並且它返回的是整數標籤（而不是二進制編碼）。
* 適合名目特徵：  
  LabelEncoder 適用於那些沒有內在順序的類別（如顏色、血型、性別等），但是其編碼結果的數字並沒有順序關係。
* 反轉編碼：  
  使用 LabelEncoder.inverse\_transform() 方法，你可以將編碼過的數字重新轉換回原本的類別標籤。

## 遺漏值（又稱缺失值，missing value）

處理真實數據時常會遇到資料不完整或資料遺漏的情況，遺漏值除了容易造成分析結果的偏誤外，也難以直接交由機器學習模型來擬合。

### SimpleImputer

用來處理數據中的缺失值。當數據集包含有遺漏值（如 NaN、空白或其他缺失標記）時，SimpleImputer 可以自動填補這些缺失值。它提供了多種填補策略，以確保數據完整性，從而能夠進行後續的分析或建模。常見的填補策略包括：

* 平均值填補（mean）：  
  使用每列的平均值來填補缺失值。
* 中位數填補（median）：  
  使用每列的中位數來填補缺失值，適合處理偏態較大的數據。
* 最常出現值填補（most\_frequent）：  
  用每列中最常見的值來填補缺失值。
* 常數填補（constant）：  
  可以指定一個固定的值來填補缺失值。

### KNNImputer

一個用於填補缺失值的類別，它基於 K 近鄰（K-Nearest Neighbors, KNN）演算法來估算和填補缺失的數據。與簡單的填補策略（如使用平均值或中位數）不同，KNNImputer 是通過考慮數據集中與缺失值最接近的觀測點來進行更智能的填補。具體來說，它會基於其他相似樣本的值來推測缺失值。

KNNImputer 的運作原理：

* 對於每一個缺失值，KNNImputer 會找到數據集中與該行最相似的 K 個樣本（即最接近的鄰居）。
* 它會計算這些鄰居的非缺失值，並使用這些值的平均值來填補缺失的數據點。

這種方法在某些情況下比簡單的填補方法更準確，因為它根據樣本的整體結構來推斷缺失值，而不僅僅依賴單個列的統計值，且數據之間具有相似性時表現較好。例如，對於地理數據或時間序列數據，KNNImputer 可能會比簡單的均值或中位數填補方法更有效。

## 切割資料集

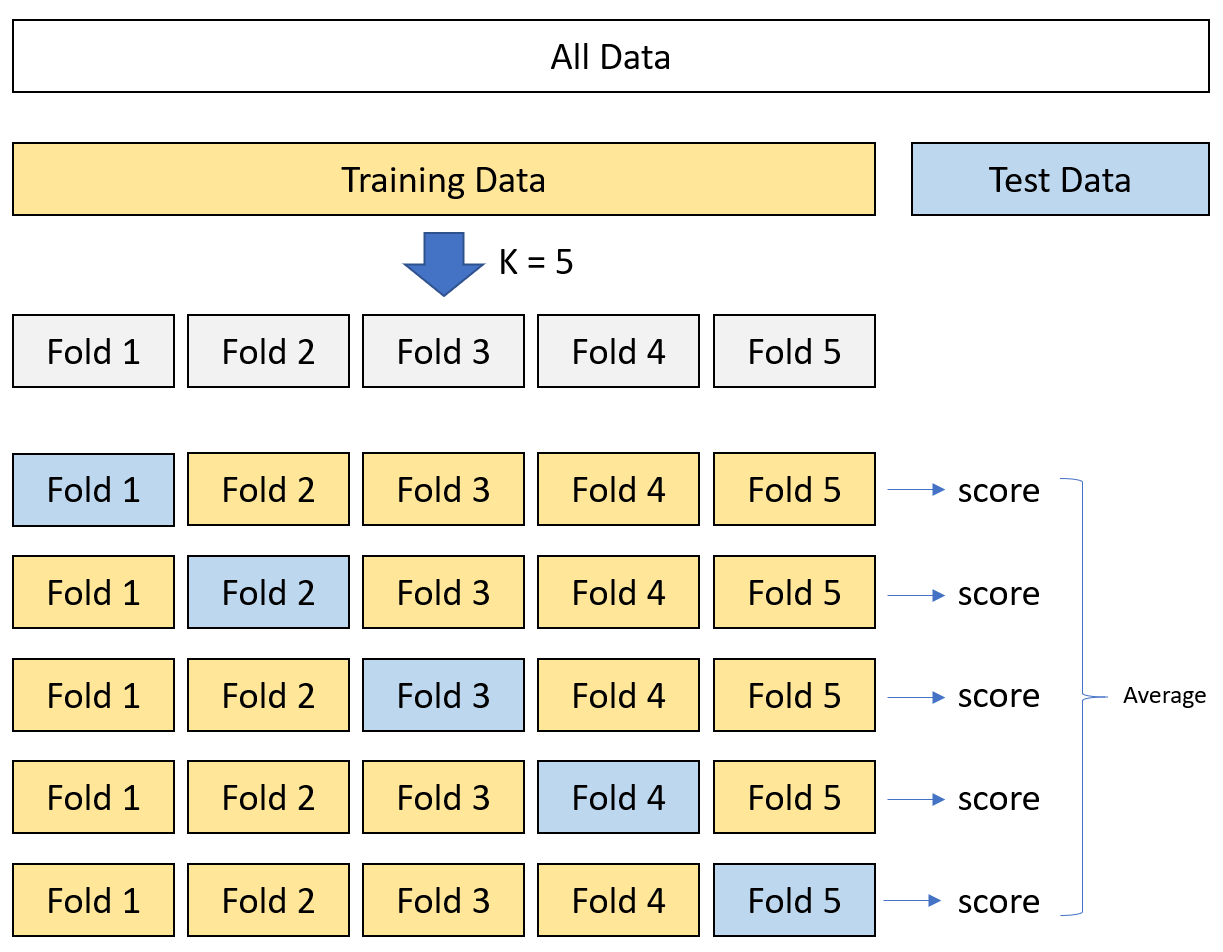
### train\_test\_split

主要目的是將原始數據集劃分為兩部分，一部分用來訓練模型（訓練集），另一部分用來評估模型的表現（測試集）。這樣的分割可以幫助避免模型過度擬合，也能更準確地評估模型在未見數據上的泛化能力。

概念上，train\_test\_split 做了以下幾件事：

* 訓練集（Training Set）：  
  用來訓練模型，讓模型學習數據中的模式或規律。這部分數據是模型在訓練過程中能夠「看到」的。
* 測試集（Test Set）：  
  用來評估模型的準確性和泛化能力。測試集不會在模型訓練過程中使用，而是留到訓練完成後進行模型的最終評估。這能幫助檢查模型在未見數據上的表現。

### K-fold Cross Validation



圖：交叉驗證，K = 5 的情形

是一種用來評估機器學習模型性能的技術，尤其適合於數據量有限的情況。它將數據集劃分為 K 個子集（folds），並進行多次訓練和測試。具體過程如下：

* 數據分割：  
  將整個數據集隨機分為 K 個等份。
* 迭代訓練與測試：
  + 在第 i 次迭代中，選取第 i 個子集作為測試集，其餘的 K-1 個子集作為訓練集。
  + 訓練模型並在該測試集上評估性能（如準確率 Accuracy、F1-score 等）。
* 重複 K 次：  
  每次都使用不同的子集作為測試集，並重複這個過程 K 次。
* 計算平均結果：  
  最後，將 K 次測試的評估指標平均，作為模型的最終性能評估。

這樣的過程有助於模型的評估更加穩健，因為模型在不同的數據劃分下進行了多次測試，有效減少了因數據集劃分不同而造成的隨機誤差，同時也可以當作參數挑選器（從交叉驗證中找到合適的超參數）。

## 異常值

實務上，要判斷一個新的觀察值是否屬於已知樣本的分析，例如判斷信用卡交易是否異常，一般稱為異常偵測（anomaly detection），有時候也會異常值為雜訊（noise），機器學習對這一類的訓練樣本特別敏感，容易造成模型擬合後的效能低落。主要偵測方式分成兩種：

* 離群值（outlier）偵測：  
  找出在訓練資料集當中，與其它樣本有相當大差異的觀察值，通常是非監督式（unsupervised）的作法。
* 新奇值（novelty）偵測：  
  決定一筆新的觀察值是否為離群值，一般為半監督式（semi-supervised）作法（同時使有、無標籤的資料）。