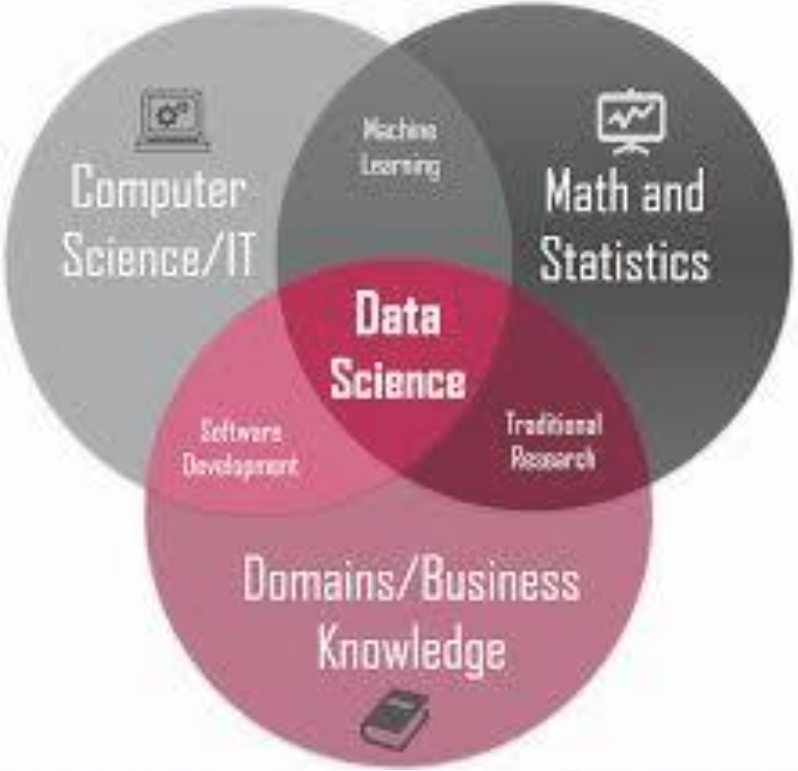
AI & DS



U09-機器學習:監督式學習迴歸演算法

2023.05_V1.0

Data Science | Artificial Intelligence | Machine Learning | Deep Learning | Statistics

單元大綱

- 線性迴歸演算法
- 邏輯迴歸演算法
- 支持向量機演算法

Part 1

線性迴歸演算法

線性迴歸演算法介紹

- 迴歸分析（Regression Analysis）是一種統計學上分析數據的方法，目的在於了解兩個或多個變數間是否相關、相關方向與強度，並建立數學模型以便觀察特定變數來預測研究者感興趣的變數。一般來說，通過迴歸分析我們可以由給出的自變數估計應變數的條件期望。
- 迴歸分析是建立被解釋變數Y（或稱應變數、依變數、反應變數）與解釋變數X（或稱自變數、獨立變數）之間關係的模型。

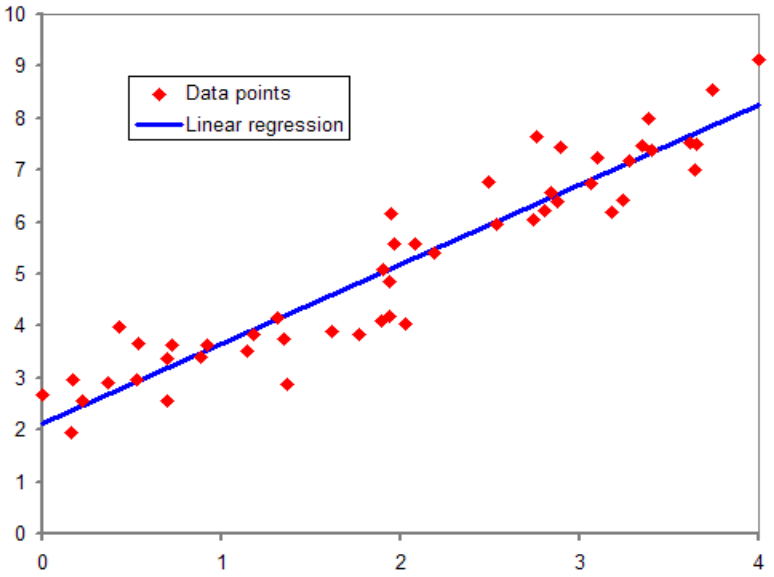
$$y = Wx + b$$ (W: weight 權重　b:bias 偏誤)

- 在迴歸的問題中，通常希望預測出來的結果可以跟實際的值一樣。但現實是預測出來的東西基本上跟實際值都會有落差，這個落差在統計上稱為「殘差(residual)」。
- **損失函數**中的損失就是「實際值和預測值的殘差」，計算每一個預測值與真實值的平方總和，稱為「平均方差」。

$$V = \sum_{i=1}^{n} (Xpred - Xreal)^2$$

V：平均方差，X_{pred}: 預測值，X_{real}: 真實值

- 不斷改變W與b值，平均方差最小時的W與b值，就是最佳直線。



[Ref]: https://pse.is/4wfr8n

正規方程式(Normal Equation)線性迴歸

- 線性迴歸演算法找到最小平均方差的權重(W)及偏誤值(b)的方法有:
  正規方程式法 與 **梯度下降法**
- **Multiple linear regression （多元線性迴歸）與正規方程式**

多元線性迴歸被用來探討一個以上的 independent variable（自變數）及一個 dependent variable （應變數）之間的線性關係。假設有 n 個自變數 x_i（或 features 特徵資料），及一個應變數 y，我們可以將兩變數之間的關係用下列式子來表示（其中 $\theta_0 \ldots \theta_n$ 為迴歸係數）：

$$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \cdots + \theta_n x_n$$

若有 m 組 training data （訓練資料），則可以進一步將 θ、x 及 y 的組合以陣列的方式表示如下：

$$\begin{bmatrix} \theta_0 x_0^{(1)} & \theta_1 x_1^{(1)} & \theta_2 x_2^{(1)} & \theta_3 x_3^{(1)} & \cdots & \theta_n x_n^{(1)} \\ \theta_0 x_0^{(2)} & \theta_1 x_1^{(2)} & \theta_2 x_2^{(2)} & \theta_3 x_3^{(2)} & \cdots & \theta_n x_n^{(2)} \\ \theta_0 x_0^{(3)} & \theta_1 x_1^{(3)} & \theta_2 x_2^{(3)} & \theta_3 x_3^{(3)} & \cdots & \theta_n x_n^{(3)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \theta_0 x_0^{(m)} & \theta_1 x_1^{(m)} & \theta_2 x_2^{(m)} & \theta_3 x_3^{(m)} & \cdots & \theta_n x_n^{(m)} \end{bmatrix} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ y^{(3)} \\ \vdots \\ y^{(m)} \end{bmatrix}$$

簡化為: $\mathrm{X}\Theta = \mathrm{y}$

其中 $x_0 = 1$， X 和 Θ 分別為

$$\mathrm{X} = \begin{bmatrix} 1 & x_1^{(1)} & x_2^{(1)} & x_3^{(1)} & \cdots & x_n^{(1)} \\ 1 & x_1^{(2)} & x_2^{(2)} & x_3^{(2)} & \cdots & x_n^{(2)} \\ 1 & x_1^{(3)} & x_2^{(3)} & x_3^{(3)} & \cdots & x_n^{(3)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_1^{(m)} & x_2^{(m)} & x_3^{(m)} & \cdots & x_n^{(m)} \end{bmatrix}, \Theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix}$$

最後得到 $\Theta = (X^T X)^{-1} X^T y$

[Ref] https://pse.is/4w8v3d

[說明]

- 於自變項X與依變項Y的座標分布圖中，可配置一最理想直線(迴歸直線)來進行依變項數值Y的估計，以 y=ax+b 表示。其中a為迴歸直線斜率，又稱迴歸係數(regression coefficient)。
- 當自變項數值固定時，依變項的實際值Y與預測值y之距離(Y-y)，稱為殘差(residual)。

※**[注意]**

1. 正規方程式迴歸法因利用特徵值矩陣的轉置與反矩陣進行複雜計算。僅適用於:**特徵數量較少**的狀況!
2. 使用線性迴歸演算法時，各特徵數值大小會影響結果，訓練前需要進行**特徵標準化**。

Scikit-Learn 的正規方程法線性迴歸模組

1. 載入Scikit-Learn的正規方程法線性迴歸模組

```
from sklearn. linear_model import LinearRegression
```

2. 建立LinearRegression物件

```
正規變數= LinearRegression ()
```

3.利用fit ()方法進行訓練

```
正規變數. fit (訓練資料, 訓練目標值)
```

4. 使用predict方法對未知資料進行預測

```
預測變數= 正規變數.predict(預測資料)
```

評估正規方程式線性迴歸模組的性能

迴歸演算法評估模型的性能是以誤差大小判斷, 平均方差越小越好。

1. 載入Scikit-Learn的計算平均方差的模組: mean_squared_error

```
from sklearn. metrics import mean_squared_error
```

2.計算平均方差的語法:

```
mean_squared_error(真實目標值, 預測值)
```

梯度下降(Gradient Descent)法線性迴歸

- 梯度下降法就像下山，先走一步查看是否較低，若有就繼續向前一步，若無就倒退。
- **梯度下降法無論特徵值數量多寡都適用。**
- **[說明]**

S1.以亂數隨機設定 W 及b, 計算平均方差。

S2.將W值略微增加後, 計算平均方差。若:

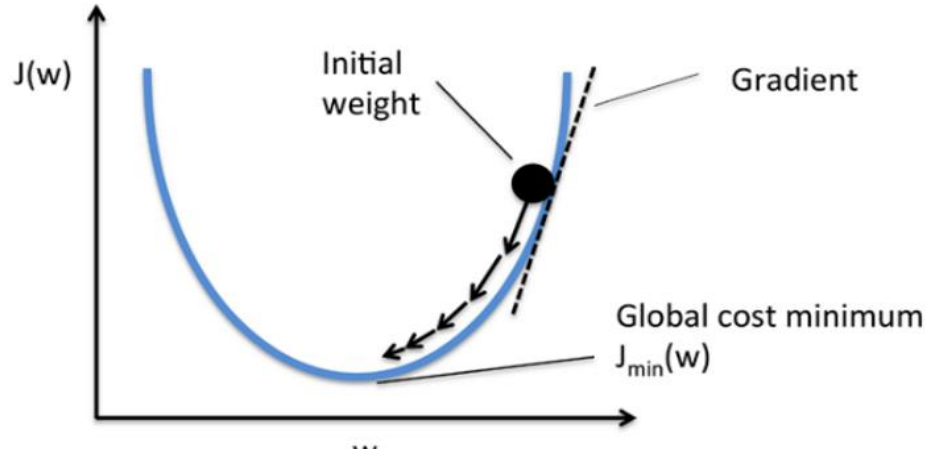
平均方差變大, 將原W略減小做為新的W值

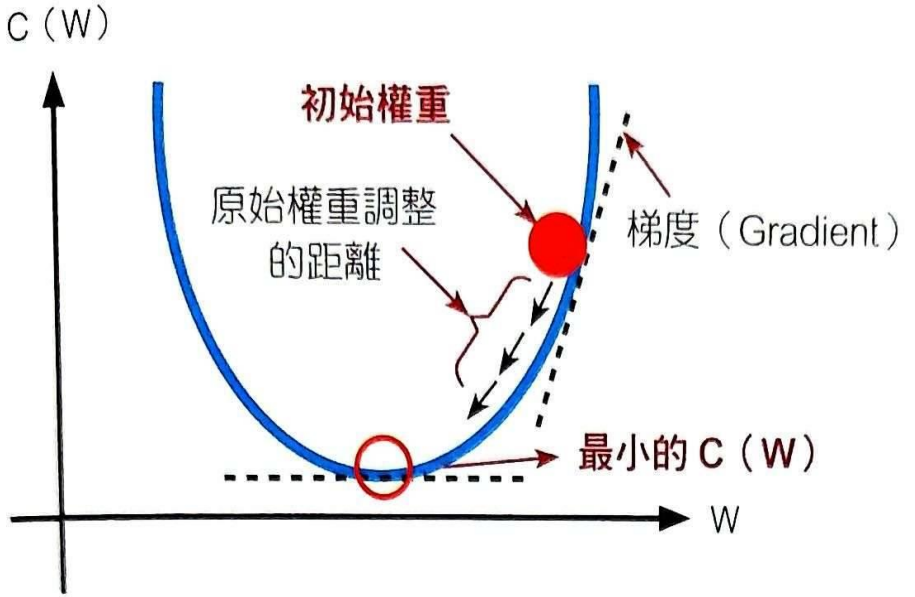
b值也如是操作。

S3.以新的W值與b值, 計算平均方差。

S4.重複S1-S3, 直到平均方差到無法變小為止。

此時w值與b值, 就是最佳權重值與偏誤值。





- C（W）：成本函數（Cost Function）
- W：權重（Weight），自變數前方的係數
- 原始權重調整的距離：調整自變數前方的係數，以獲得更小的 C（W）
- 梯度：往哪個方向會減少 C（W）

Scikit-Learn 的梯度下降法線性迴歸模組

1. 載入Scikit-Learn的梯度下降線性迴歸模組

```
from sklearn. linear_model import SGDRegressor
```

2. 建立SGDRegressor物件

```
梯度變數= SGDRegressor ()
```

3.利用fit ()方法進行訓練

```
梯度變數. fit (訓練資料, 訓練目標值)
```

coef_ : 權重值　　intercept_: 偏誤值(截距)

4. 使用predict方法對未知資料進行預測

```
預測變數= 梯度變數.predict(預測資料)
```

Part 2

邏輯迴歸演算法

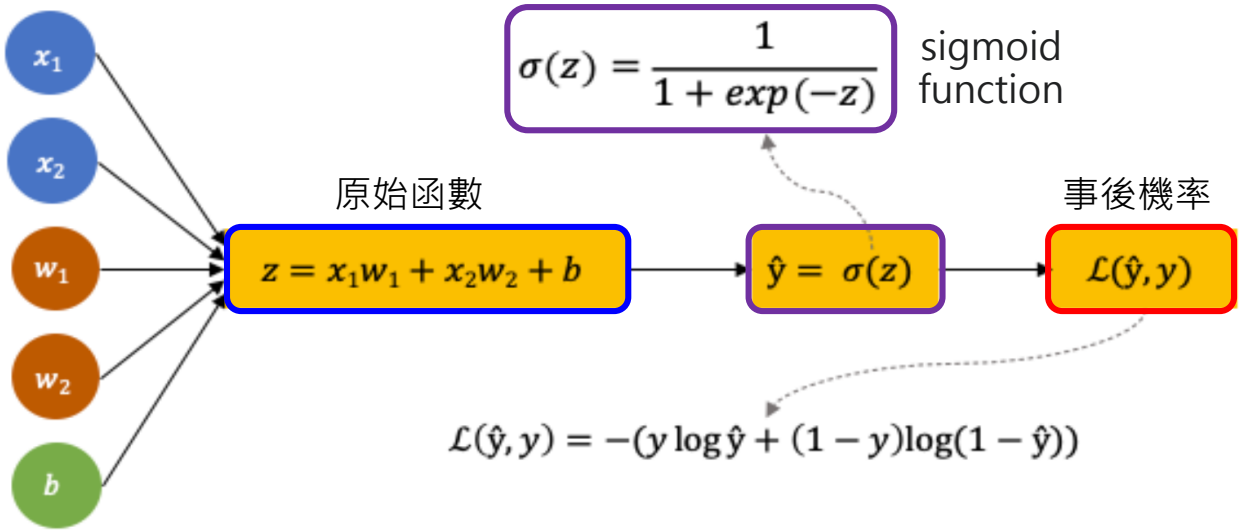
邏輯迴歸演算法介紹(1/2)

- 邏輯迴歸 (Logistic regression) 是由線性迴歸變化而來的，它是一種分類的模型。其目標是要找出一條直線能夠將所有數據清楚地分開並做分類，可以稱迴歸的線性分類器。Ex:找一個機率 (posterior probability) 當機率 P(C1|x) 大於 0.5 時則輸出預測 Class 1，反之機率小於 0.5 則輸出 Class 2。
- 邏輯迴歸是在說明一個機率的意義，透過一個 function 去訓練得到的一組參數，不同的 w, b 就會得到不同的 function。可以說 fw, b(x) 即為「事後機率」(posteriror probability)。
- 線性迴歸與邏輯迴歸的差異:

  1.邏輯迴歸是用來處理分類問題，目標是找到一條直線可以將資料做分類。主要是利用 sigmoid function 將輸出轉換成 0~1 的值，表示可能為這個類別的機率值。

  2.線性迴歸是用來預測一個連續的值，目標是想找一條直線可以逼近真實的資料。

- 邏輯迴歸的運作機制(如右圖)，function 會有兩組參數，一組是 w: weight，另一個常數 b:bias。假設有兩個輸入特徵，並將這兩個輸入分別乘上 w 再加上 b 就可以得到 z，然後通過一個 sigmoid function 得到的輸出就是 posterior probability。



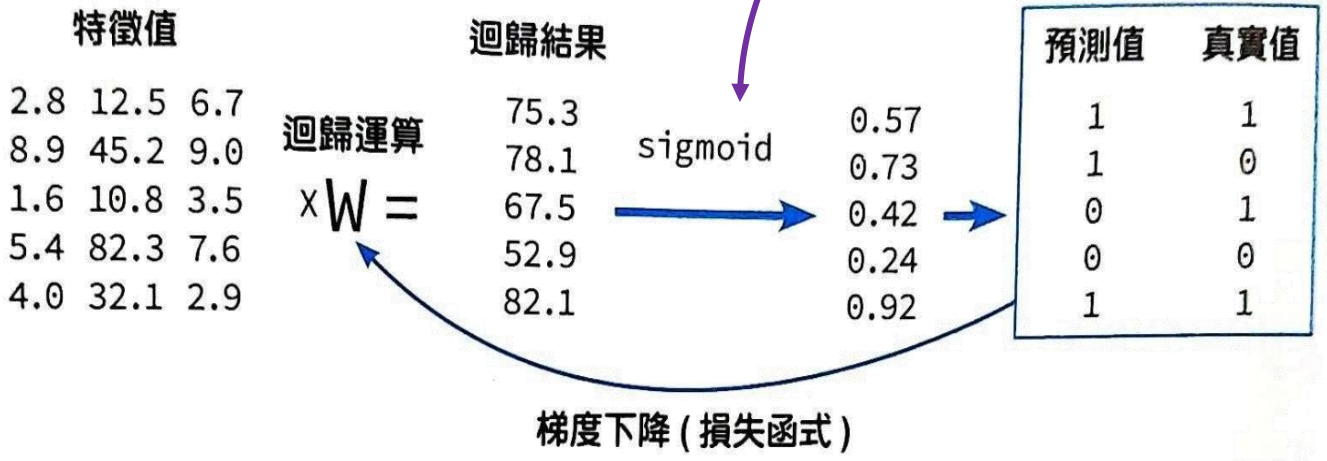
[Ref]: https://pse.is/4xspv7

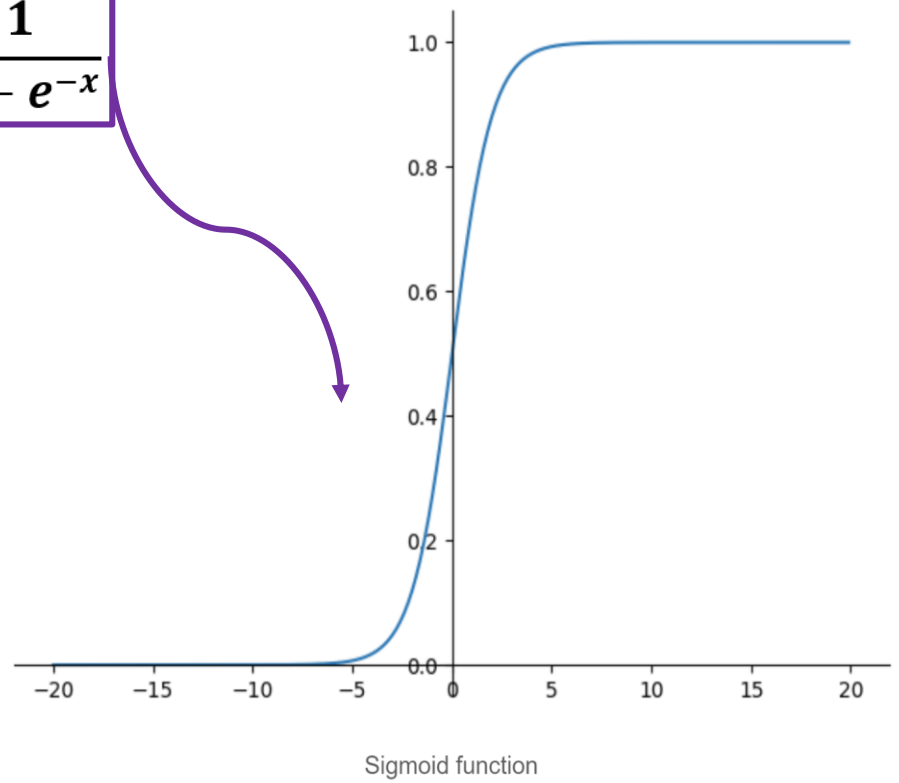
邏輯迴歸演算法介紹(2/2)

- Sigmoid() 函數簡單來講就是個映射函數，將任何變量（這些先寫成 x）映射到 [0, 1] 之間。通常被用來當作機器學習領域 (Machine Learning) 神經網路的激活函數 (Activation Function)。
- 常見應用情境，就是在訓練模型做二分類的時候。將模型的最後一層神經網路設定為只有一個神經元，再將最後神經元所輸出的值輸入 sigmoid() 函數。就會得到一個介於 [0, 1] 之間的數值。Ex:只需要設定閾值，將小於 0.5 的值通通判定為 0、大於 0.5 的值通通判斷為 1 ，就可以做出二分類的預測。
- Sigmoid 函式公式:

$$\text{Sigmoid output} = \frac{1}{1 + e^{-x}}$$

- 利用邏輯迴歸演算法的過程,整理如下:





Sigmoid function

[Ref]: https://pse.is/4wfr8n

Scikit-Learn 的邏輯迴歸模組

1. 載入Scikit-Learn的邏輯迴歸模組

```
from sklearn. linear_model import LogisticRegression
```

2. 建立LogisticRegression物件

```
迴歸變數= LogisticRegression()
```

3.利用fit ()方法進行訓練

```
迴歸變數. fit (訓練資料, 訓練目標值)
```

4. 使用predict方法對未知資料進行預測

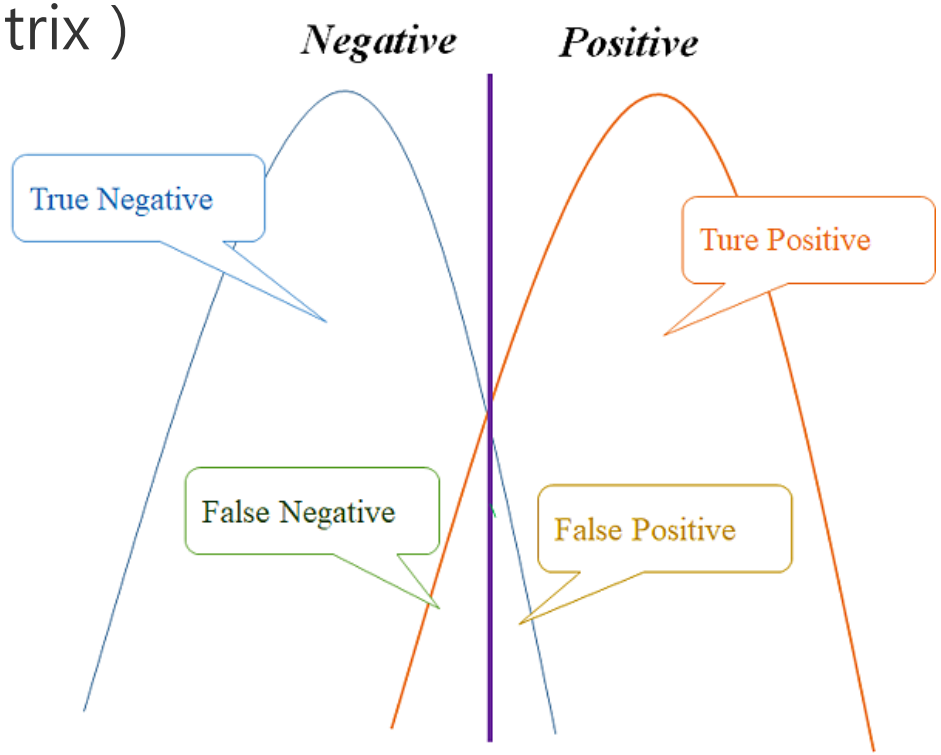
```
預測變數= 迴歸變數.predict(預測資料)
```

5.使用score ()方法對未知資料進行預測, 並計算準確率

```
準確率變數 = 迴歸變數. score (預測資料, 預測目標值)
```

精準率與召回率(1/2)

- 通常分類模型的效能是以「**準確率**」(accuracy)來評估，但某些狀況會有特殊考量。
- 在攸關生命的病況篩選過程中，將罹患某種疾病的患者全部篩選出來，比準確率更重要。
- **精準率(precision)**與**召回率(recall)**是另外兩種評估模型優劣的指標, 尤其是**召回率**。
- **混淆矩陣**
- 面對二分類問題（陰性/陽性、正確/錯誤）時，常用的指標稱為混淆矩陣（Confusion Matrix）



真實 / 預測	實際正向	實際負向
預測正向	True Positive (TP)	False Positive (FP)
預測負向	False Negative (FN)	True Negative (TN)

[Ref]: https://pse.is/4xspv7

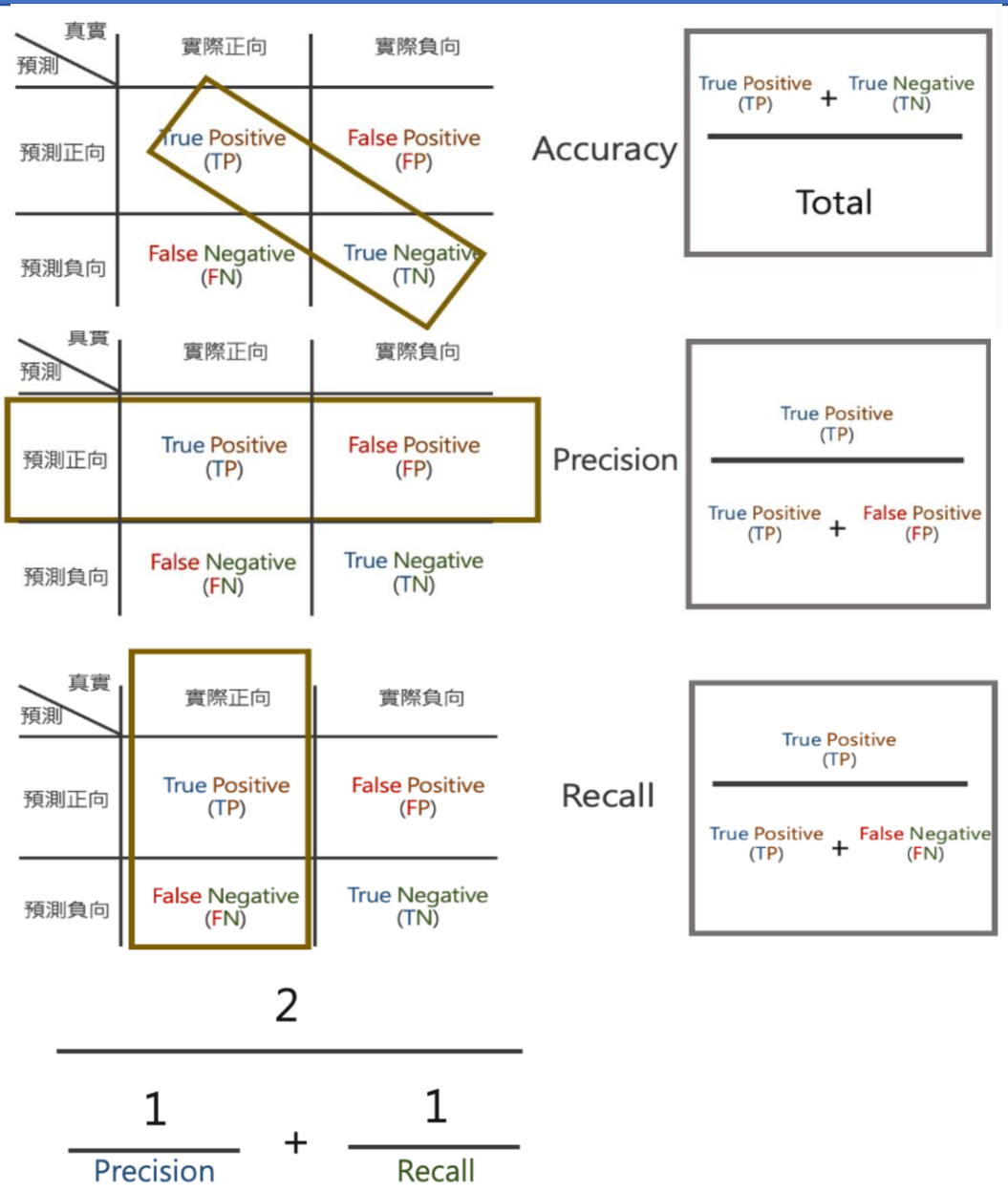
精準率與召回率(2/2)

- **準確率（Accuracy）**：
  最常用的指標，也就是將所有預測與實際相同的情況相加，並除以所有預測情形個數，也就是評估一模型，能成功預測到結果的準確度。

- **精準率（Precision）**：
  關注在True Positive身上，在預測正向(Positive)的情況下，成功預測到結果的比例。

- **召回率（Recall）**：
  關注在True Positive身上，但其看重的是實際情況在正向(Positive)的情況下，預測也是正向(Positive)的比率。

- **F1 Score**：
  當覺得Precision與Recall指標同等重要時，就用F1 Score來表示。



[Ref]: https://pse.is/4tzdb7

Scikit-Learn 的精確率、召回率與準確度模組

1. 載入Scikit-Learn的精確率、召回率與準確度模組

```
from sklearn. linear_model import classification_report
```

2. 建立classification_report物件

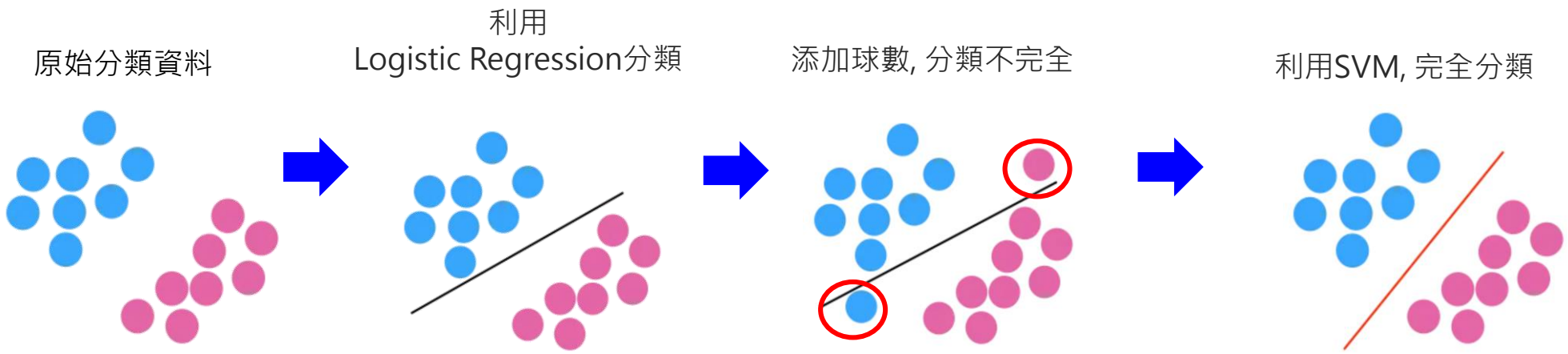
```
classification_report(目標值, 預測值, labels = 類別值,
                      target_names = 類別顯示值)
```

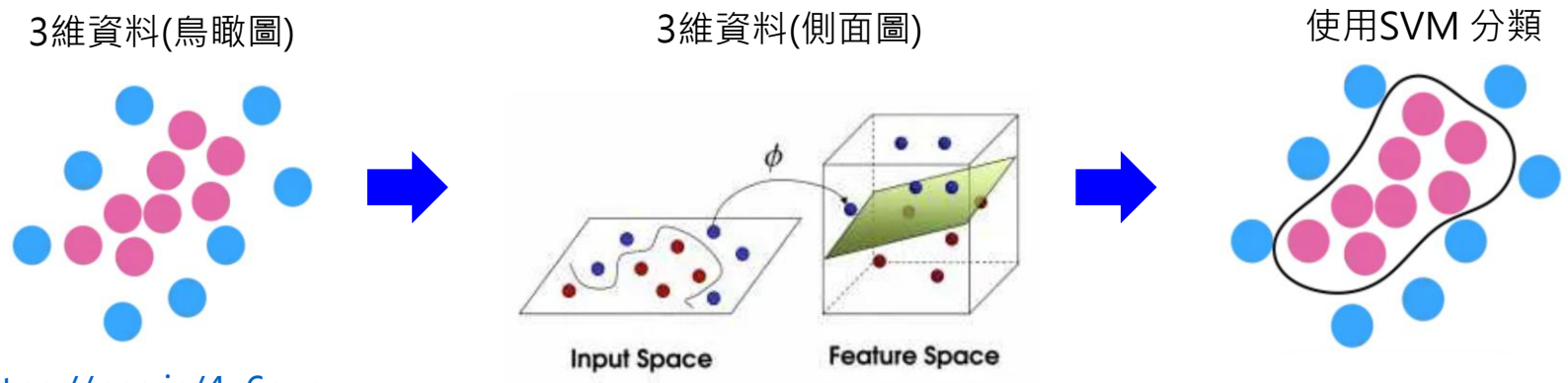
- **目標值**: 資料集的目標值
- **預測值**: 使用模型進行預測得到的結果值
- **labels**: 資料集中的類別值(Ex: 罹癌為1, 否則為否)
- **target_names**: 傳回值的類別名稱(搭配 labels)

Part 3 支持向量機演算法

支持向量機(Support Vector Machine) 介紹

- **支持向量機(Support Vector Machine, SVM):**
  是一種可以用於分類及迴歸的演算法,不僅適用於線性迴歸, 也可用於非線性迴歸。
- 原理:
  SVM對於二維(特徵值)線性資料,可以找到一條最佳的直線,進行分割...若是N維資料, 可以找到(N-1)維的超平面資料分離。
- Ex:





[Ref]: https://pse.is/4x6evc

Scikit-Learn 的支持向量機模組(1/2)

1. 載入Scikit-Learn的精確率、召回率與準確度模組

```
from sklearn. svm import SVC
```

2. 建立SVC物件

```
分類變數=SVC(kernel=核函式, C=數值, gamma =核函式係數,
                                degree= 數值)
```

- **kernel**: 設定核函式種類, 常用如下:
  lineaer: 線性函式
  poly: 多項式函數, 多項式的次方由degree參數設定。
  rbf(預設值): 高斯函式。
- **gamma**:核函式係數, 常用如下:
  scale(預設值): 值為特徵數量與標準差乘績的倒數
  auto: 值為特徵數量的倒數
  浮點數:直接設定數值
- **degree**:多項式次方, 此參數只有在Kernel參數值為poly時才有效。預設值=3。

Scikit-Learn 的支持向量機模組(1/2)

3.利用fit ()方法進行訓練

```
分類變數. fit (訓練資料, 訓練目標值)
```

4. 使用predict方法對未知資料進行預測

```
預測變數= 分類變數.predict(預測資料)
```

5.使用score ()方法對未知資料進行預測, 並計算準確率

```
準確率變數 = 分類變數. score (預測資料, 預測目標值)
```

Scikit-Learn 的支持向量機應用-人臉辨識

關於Scikit-Learn的人臉資料集

- Scikit-Learn的人臉資料集包含數十人的人臉資料，美人的人臉圖形數量不等。
- 資料集位址：
  https://scikit-learn.org/0.19/modules/generated/sklearn.datasets.fetch_lfw_people.html

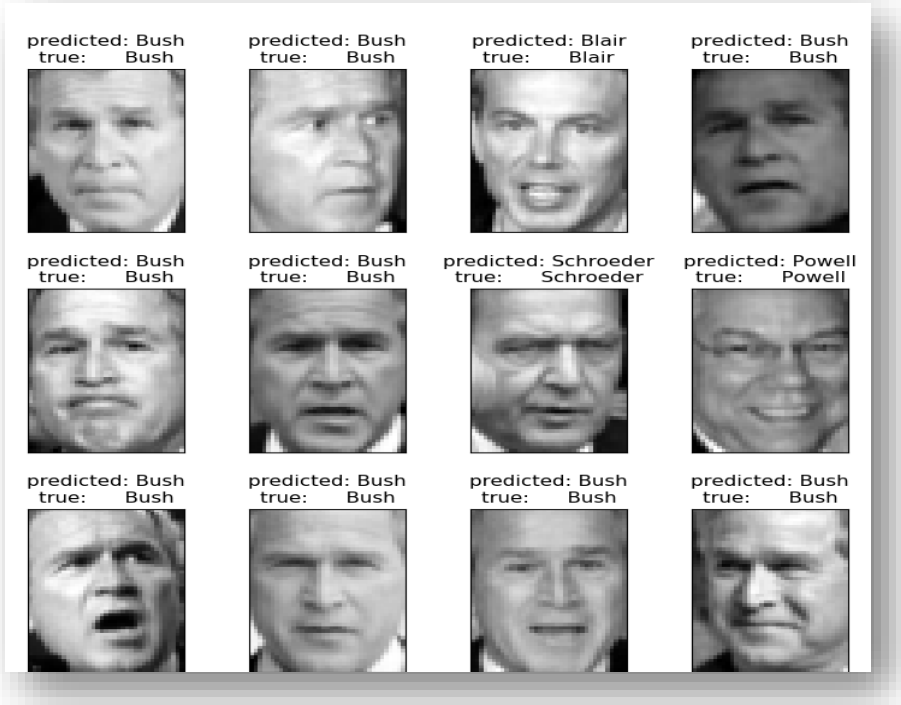
※第一次執行下載,會將資料集存於:雲端硬碟根目錄的<lfw_home>

- 載入語法:

```
from sklearn. datasets import fetch_lfw_people

人臉變數=fetch_lfw_people(min_faces_per_person=數值,
                          resize =數值, color= 布林值)
```

- min_faces_per_person: 取得人臉最小數目。預設值=0，即預設取得資料集所有圖片。
- resize: 圖片縮放比例。預設值=0.5，即長與寬縮小一半。
- color: True為彩色圖片，False為黑白圖片。預設值為False。



- 本例中使用的數據集是“Labeled Faces in the Wild”（又名LFW）的預處理摘錄：
  http://vis-www.cs.umass.edu/lfw/lfw-funneled.tgz (233MB)
- 總 數據集 大小：
- n_samples ： 1288
- n_features ： 1850
- n_classes ： 7
- 在0.233 秒內完成從 966 張面孔中提取前150個特徵臉在 0.024 秒內完成將輸入數據投影到特徵臉正交基礎上將分類器擬合到訓練集在23.793秒內完成

支持向量機在迴歸分析上的應用:廣告效益預測

支持向量機也可應用於迴歸問題：需載入SVR模組。

1.載入Scikit-Learn的SVR模組

```
from sklearn. svm import SVR
```

2. 建立SVR物件(參數語法與SVC相同)

```
迴歸變數=SVR(kernel=核函式, C=數值, gamma =核函式係數,
                                degree= 數值)
```

- **kernel**: 設定核函式種類, 常用如下:
  lineaer: 線性函式
  poly: 多項式函數, 多項式的次方由degree參數設定。
  rbf(預設值): 高斯函式。
- **gamma**:核函式係數, 常用如下:
  scale(預設值): 值為特徵數量與標準差乘績的倒數