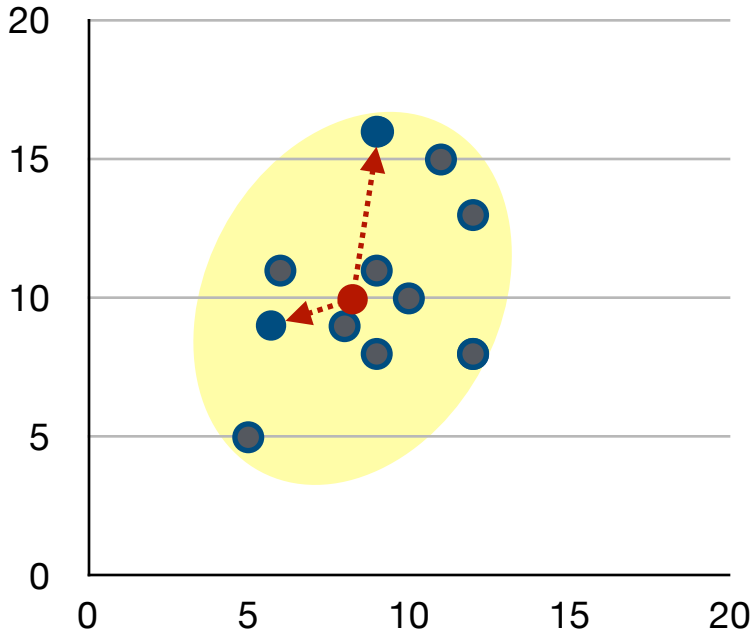
特徵工程

鄭年亨

通識教育中心

臺北醫學大學
TAIPEI MEDICAL UNIVERSITY

複習變異數的概念



標準差 SD

變異數 Variance

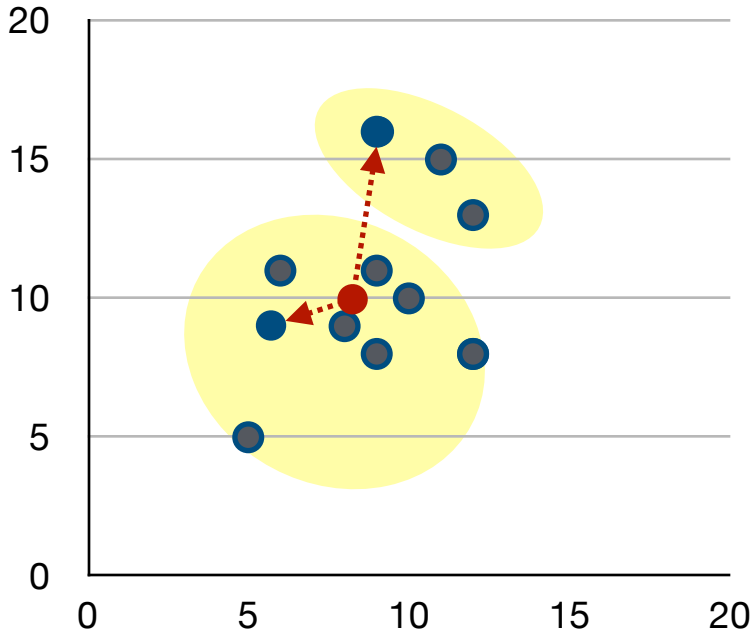
重要因素

(性別、年紀、疾病、治療...)

不重要因素

(個人差異、測量誤差...)

複習變異數的概念



標準差 SD

變異數 Variance

重要因素

(性別、年紀、疾病、治療...)

不重要因素

(個人差異、測量誤差...)

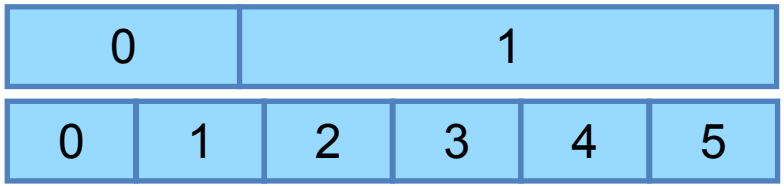
特徵工程

提高模型效能

- 特徵轉換 Feature Transformation
  - **特徵縮放 Feature Scaling**
- 特徵建構 Feature Construction
- **特徵選擇 Feature Selection**
- 特徵萃取 Feature Extraction
  - **降維 Dimensionality Reduction**

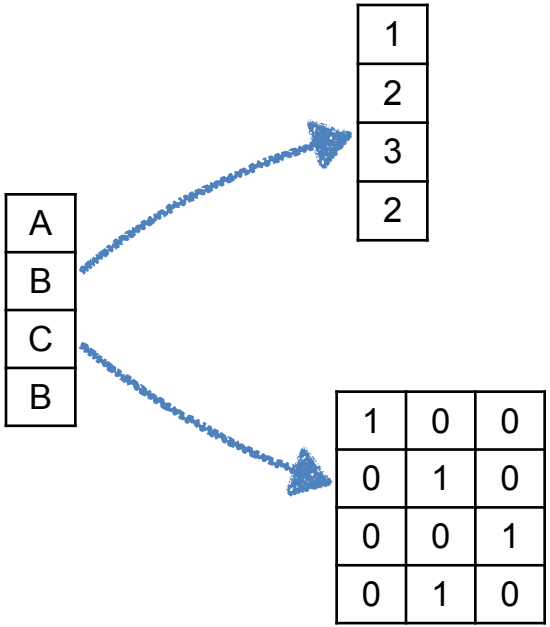
特徵轉換（1/2）

- 特徵縮放
  - 最大最小縮放 Min-Max Scaling：轉為 [0, 1]
  - 標準化（常態化）Standardization：轉為z值（M=0, SD=1）
- 連續數值
  - 對數轉換
- 將連續數值轉為類別
  - 二值化 Binarization
  - 分組 Binning



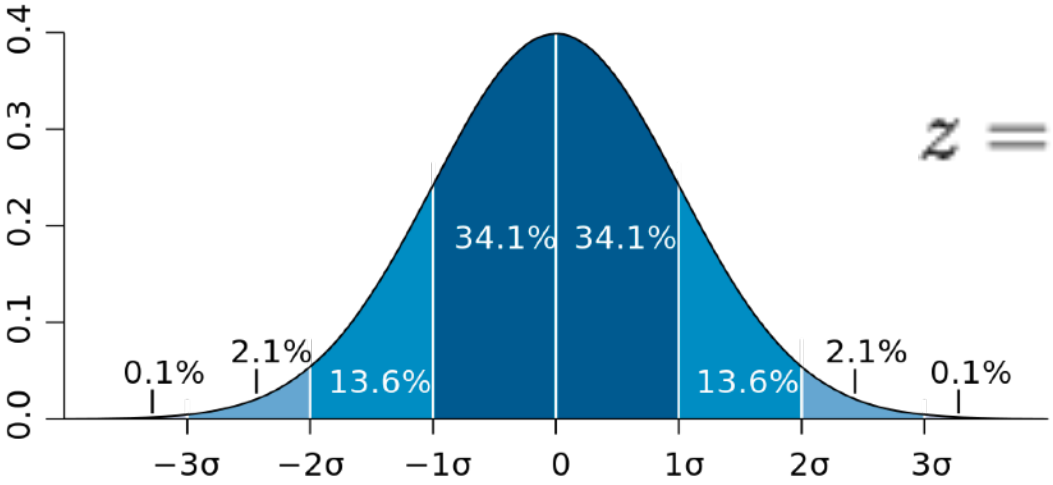
特徵轉換（2/2）

- 將類別轉為數值：編碼
  - Integer Encoding
  - One-hot Encoding (OHE)
  - ……



標準化 standardization

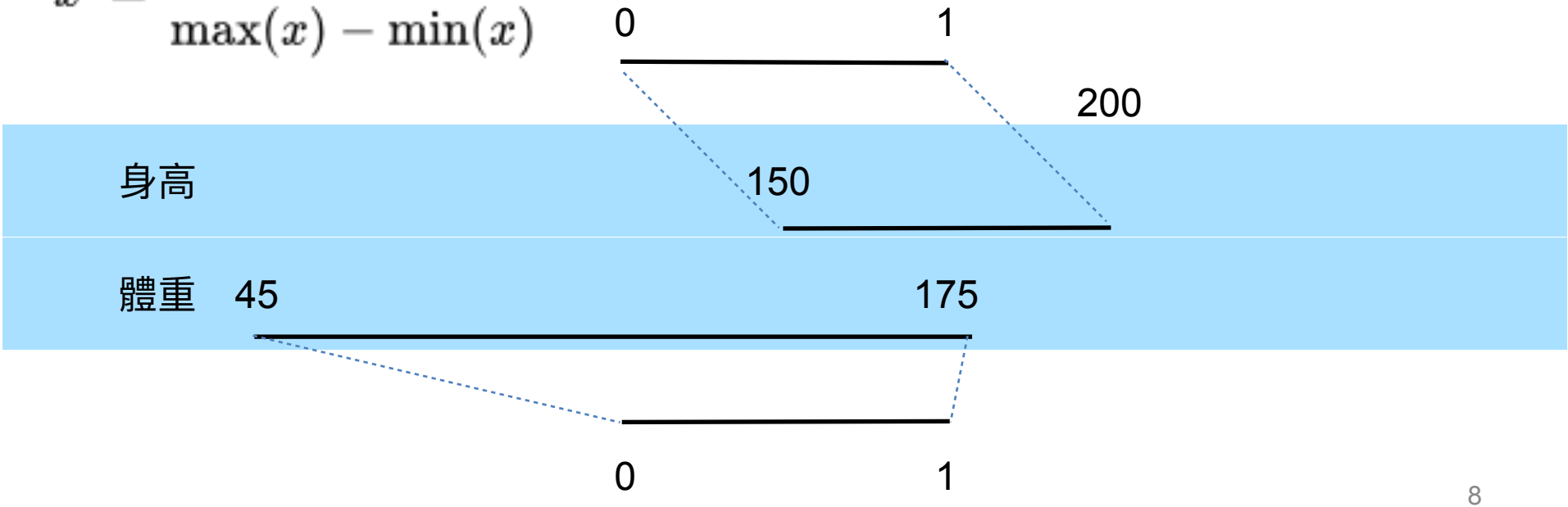
```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
data = scaler.fit_transform(data)
```



$$z = \frac{x - \mu}{\sigma}$$

最大最小縮放 Min-Max Scaling

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$



最大最小縮放 Min-Max Scaling

```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()
data = scaler.fit_transform(data)
```

特徵縮放

- 適用於基於距離的演算法
  - 例如kNN、k-Means clustering、SVM、PCA等
- 標準化
  - 適用於假設資料為常態分佈的演算法：例如線性回歸
  - 適用於假設資料中心為 0 的演算法：例如PCA
- 最大最小縮放
  - 適用於不限制常態分佈的演算法

特徵工程

- 特徵轉換 Feature Transformation
  - **特徵縮放 Feature Scaling**
- 特徵建構 Feature Construction
- **特徵選擇 Feature Selection**
- 特徵萃取 Feature Extraction
  - **降維 Dimensionality Reduction**

特徵建構

- 虛擬特徵
  - 多項式特徵 Polynomial Features
    - 以特徵 x 創建 x^2 或 x^3 等特徵
  - 交互特徵 Interaction Features
    - 以特徵 x和y 創建 x-y 或 x*y 等特徵
- ……
- **以專業領域知識建構特徵**
  - 例如從身高與體重創建BMI特徵

特徵工程

- 特徵轉換 Feature Transformation
  - **特徵縮放 Feature Scaling**
- 特徵建構 Feature Construction
- **特徵選擇 Feature Selection**
- 特徵萃取 Feature Extraction
  - **降維 Dimensionality Reduction**

特徵選擇

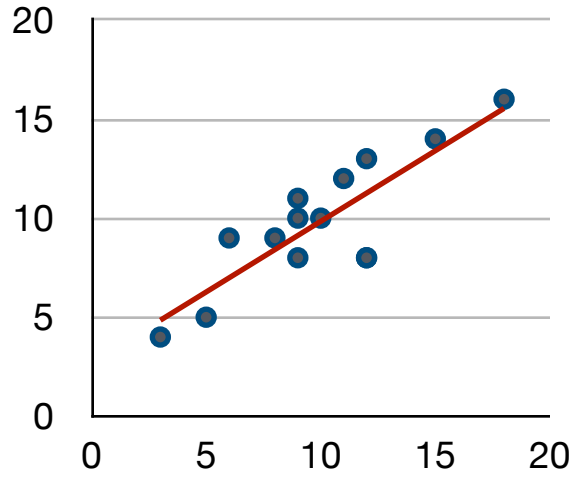
- 以專業知識選擇特徵
- 相關係數法
- 決策樹法
- …

Pearson積差相關係數

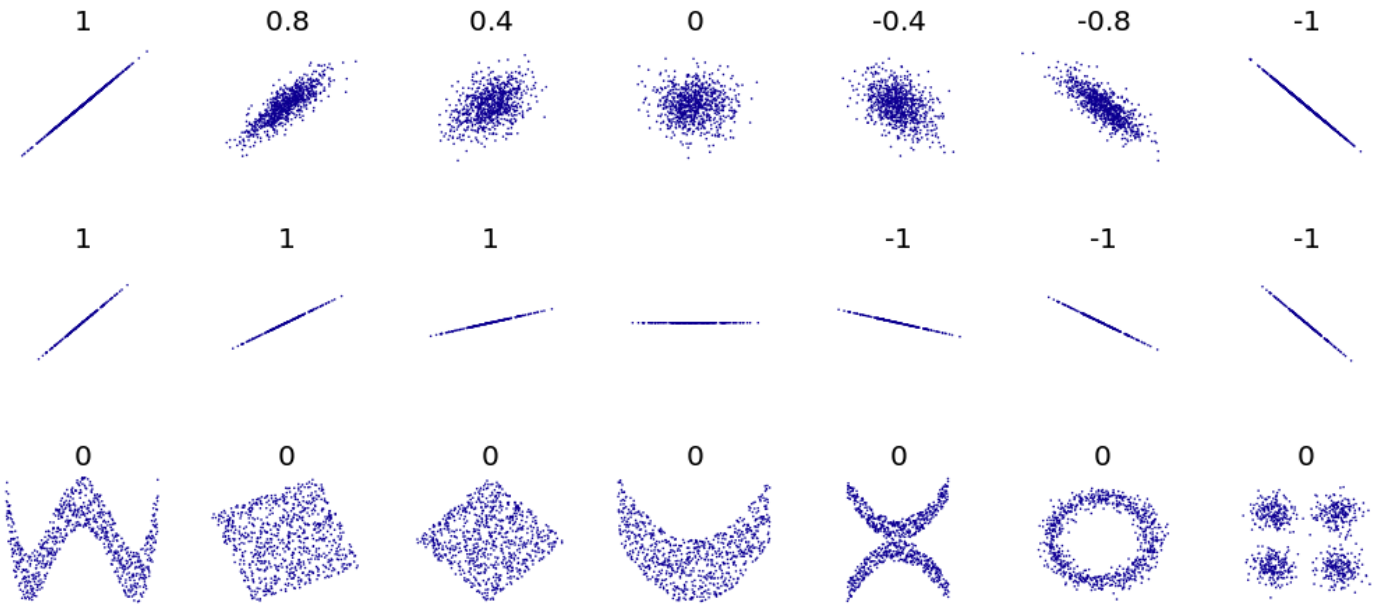
共變數

$$r = \frac{\sum_{i=1}^{n}(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \overline{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \overline{Y})^2}}$$

標準差　　標準差



Pearson積差相關係數



圖片來源：https://upload.wikimedia.org/wikipedia/commons/d/d4/Correlation_examples2.svg

相關係數的統計假設

- **兩個變量是連續變量**
- **兩個變量符合常態分佈**
  - 對極端值 outlier 較敏感
- 兩個變量間彼此獨立
- 兩個變量間有線性關係
- 兩個變量的變異數不為0

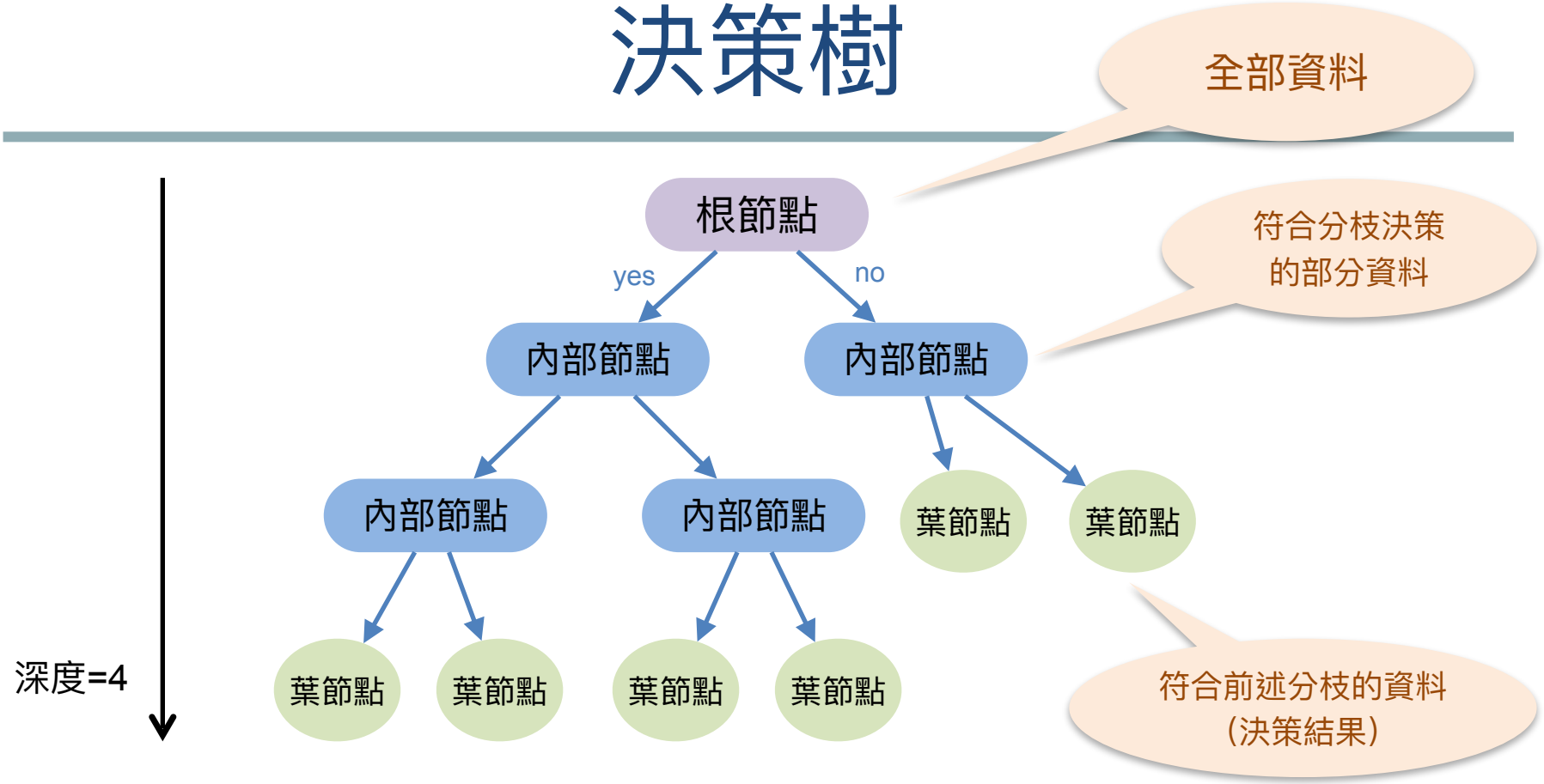
以相關係數選擇特徵

- 排除與標註低度相關的特徵
- 減少彼此高度相關的特徵

以相關係數選擇特徵

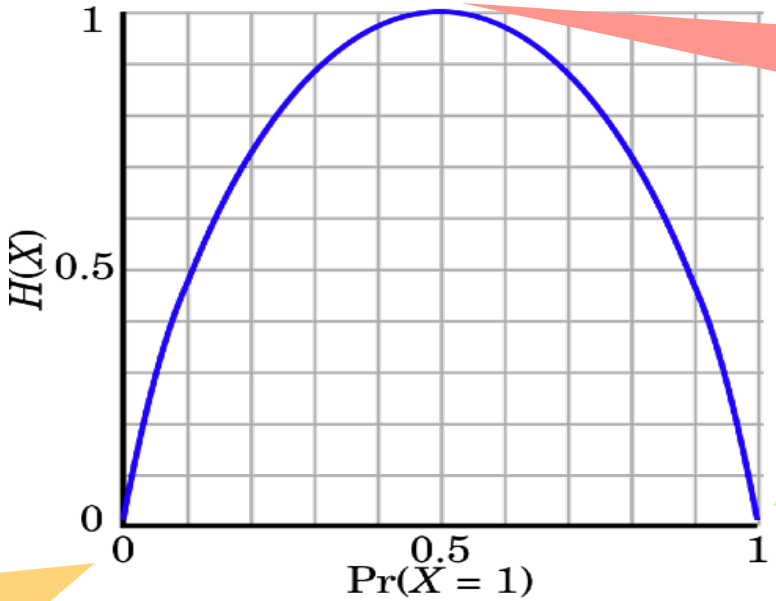
```
import pandas as pd

corr = data.corrwith(df['output'])

selected_features = corr[correlations.abs() > 0.2].index
data = df[selected_features]
```

決策樹


全部資料
根節點
yes
no
內部節點
內部節點
符合分枝決策
的部分資料
內部節點
內部節點
葉節點
葉節點
葉節點
葉節點
葉節點
深度=4
符合前述分枝的資料
(決策結果)

信息熵

information entropy



資訊很亂

資訊很純

0%正確

100%正確

基尼不純度指標

Gini impurity

一個隨機選中的樣本在子集中被分錯的可能性

$$\mathrm{I}_G(p) = \sum_{i=1}^{J} p_i \sum_{k \neq i} p_k = \sum_{i=1}^{J} p_i(1-p_i) = \sum_{i=1}^{J} (p_i - {p_i}^2) = \sum_{i=1}^{J} p_i - \sum_{i=1}^{J} {p_i}^2 = 1 - \sum_{i=1}^{J} {p_i}^2$$

決策樹的特徵重要性

- 彙整每個節點的重要性得分
  - 每個節點的不純度減少量

以決策樹的特徵重要性選擇特徵

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.feature_selection import SelectFromModel

tree = DecisionTreeClassifier()
tree.fit(data, df['output'])

sfm = SelectFromModel(tree, threshold=0.05)
data = sfm.fit_transform(data, df['output'])
```

特徵工程

- 特徵轉換 Feature Transformation
  - **特徵縮放 Feature Scaling**
- 特徵建構 Feature Construction
- **特徵選擇 Feature Selection**
- 特徵萃取 Feature Extraction
  - **降維 Dimensionality Reduction**

降維

- 減少特徵，但盡可能不會喪失資料特性
- 常見方法
  - **主成分分析 Principal Component Analysis (PCA)**
  - 線性判別分析 Linear Discriminant Analysis (LDA)
  - 獨立成分分析 Independent Component Analysis (ICA)
  - t-分佈隨機鄰域嵌入 t-Distributed Stochastic Neighbor Embedding (t-SNE)
  - ……

主成分分析

Principal Components Analysis (PCA)

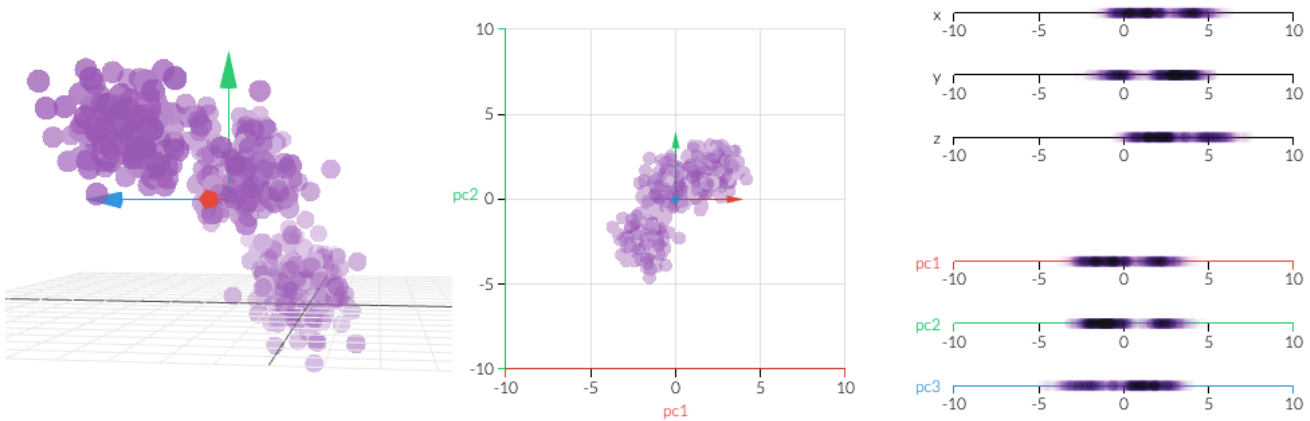
- 功能
  - 機器學習：降維
  - 科學研究：找出重要的主成分
- 方法
  - 將資料進行線性轉換，轉換為主成分
  - 保留重要的主成分，刪除對解釋資料幫助較小的主成分

降維示範：三維資料

3D example

https://setosa.io/ev/principal-component-analysis/

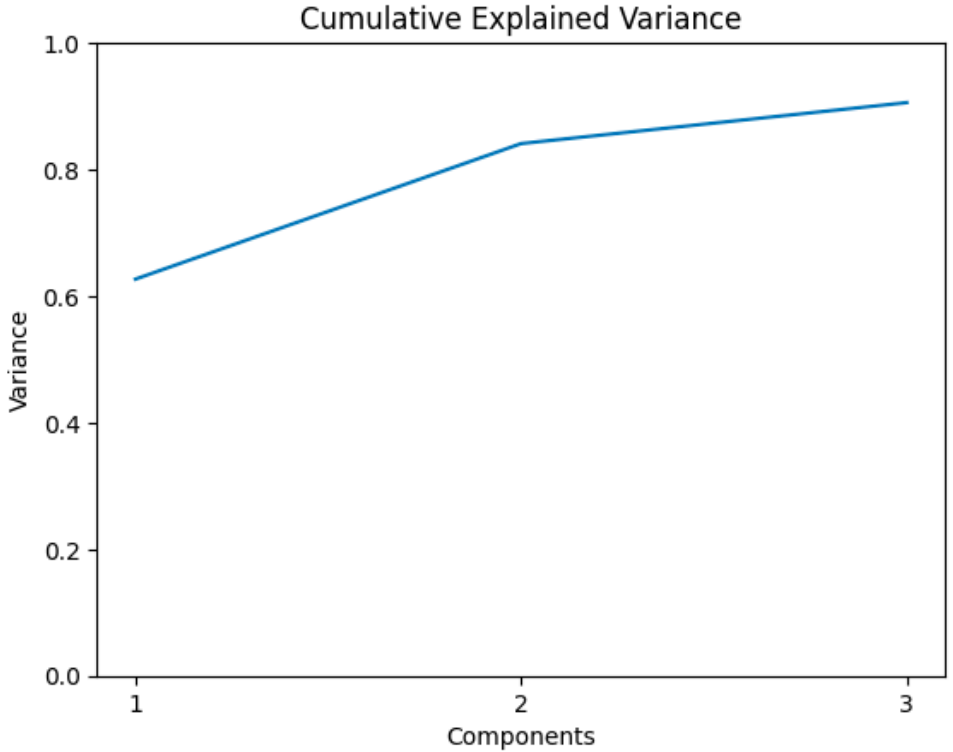
With three dimensions, PCA is more useful, because it's hard to see through a cloud of data. In the example below, the original data are plotted in 3D, but you can project the data into 2D through a transformation no different than finding a camera angle: rotate the axes to find the best angle. To see the "official" PCA transformation, click the "Show PCA" button. The PCA transformation ensures that the horizontal axis PC1 has the most variation, the vertical axis PC2 the second-most, and a third axis PC3 the least. Obviously, PC3 is the one we drop.



主成分分析

```
from sklearn.decomposition import PCA

pca = PCA(n_components=3)

pca_result = pca.fit_transform(data)

explained_variance = pca.explained_variance_ratio_
cumulative_variance = np.cumsum(explained_variance)
```

PCA結果：解釋變異量

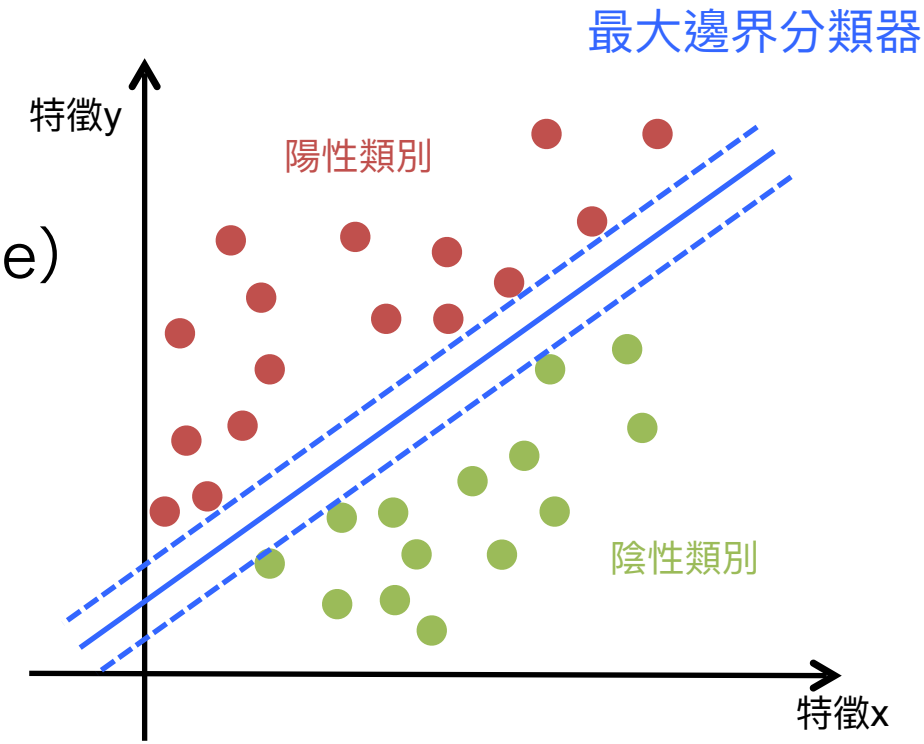

Cumulative Explained Variance
1.0
0.8
0.6
0.4
0.2
0.0
Variance
1
2
3
Components

支持向量機

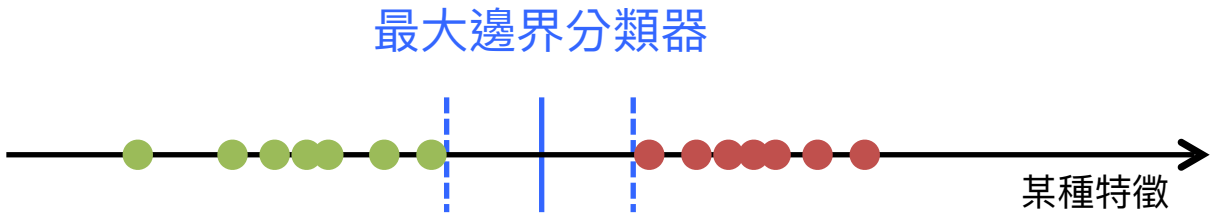
臺北醫學大學
TAIPEI MEDICAL UNIVERSITY

支持向量機

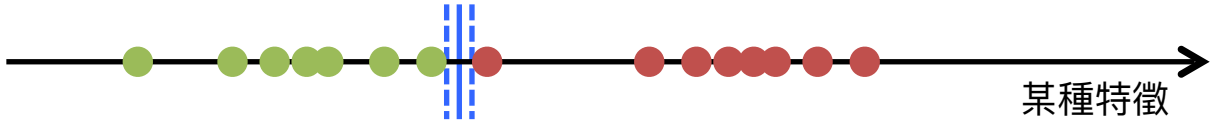
- 找到兩個線性超平面，
  使其間的距離最大化
  - 線性超平面（hyperplane）
    - 一維：一個點
    - 二維：一條線
    - 三維：一個平面
    - 四維以上：超平面



支持向量機（一維）

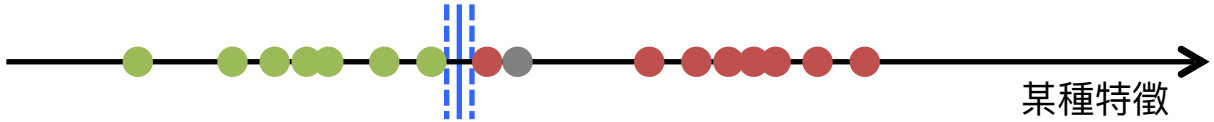


支持向量機（一維）



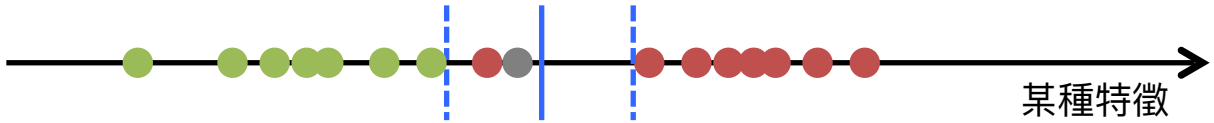
即使新資料距離陰性較近，仍會被歸類於陽性類別（過度配適問題）

支持向量機（一維）



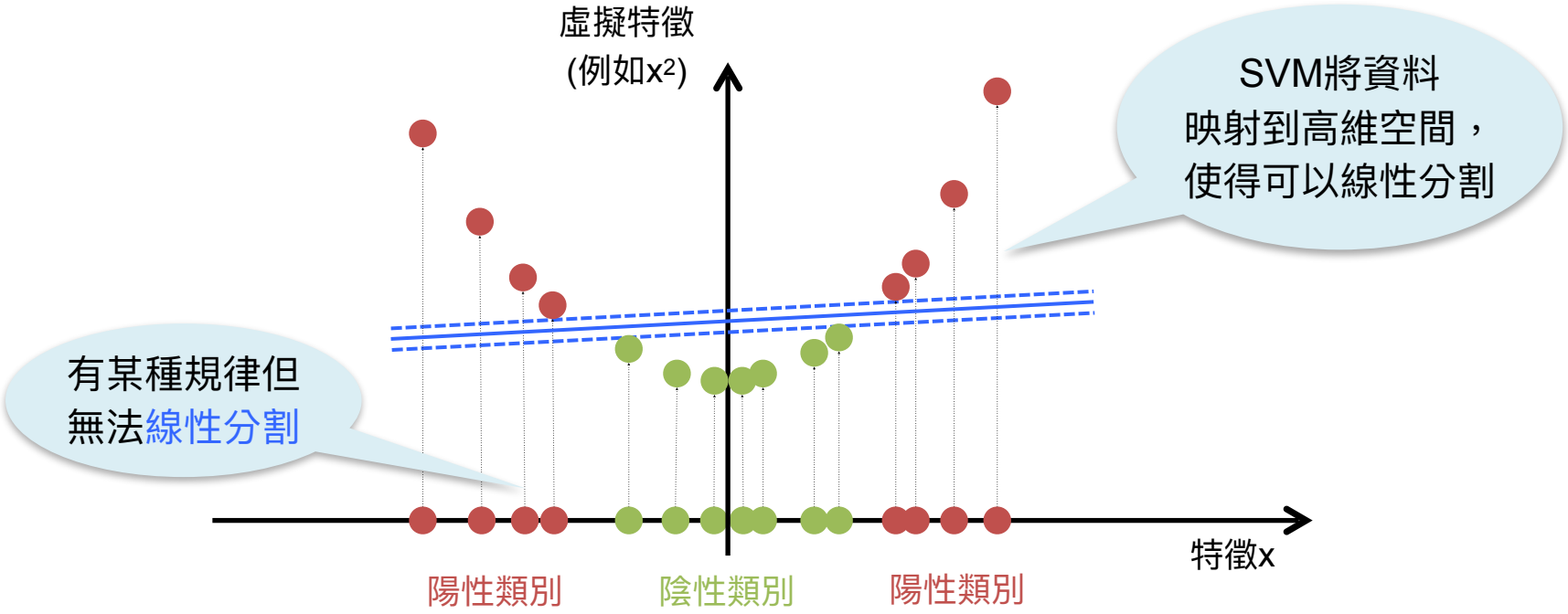
即使新資料距離陰性較近，仍會被歸類於陽性類別（過度配適問題）

支持向量機（一維）



訓練模型時容許少部分錯誤，使測試模型時有更好的正確性

核技巧 kernel trick



支持向量機

```
from sklearn import svm
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

x_train, x_test, y_train, y_test =
   train_test_split(data, df['output'], test_size=0.2)

model = svm.SVC(kernel='rbf')
model.fit(x_train, y_train)
y_pred = model.predict(x_test)
accuracy = accuracy_score(y_test, y_pred)
```