內容

[FNN（Feedforward Neural Network） 2](#_Toc166514619)

[RNN（Recurrent Neural Network） 3](#_Toc166514620)

[CNN（Convolutional Neural Network） 3](#_Toc166514621)

[FNN RNN CNN三者之間的差別 3](#_Toc166514622)

[激活函數（Activation Function） 5](#_Toc166514623)

[Entropy 與 Cross Entropy 的差別 6](#_Toc166514624)

[Entropy（熵） 6](#_Toc166514625)

[Cross Entropy（交叉熵） 7](#_Toc166514626)

[DFS 8](#_Toc166514627)

[BFS 9](#_Toc166514628)

[DFS VS. BFS 9](#_Toc166514629)

[CHARACTERISTICS OF SEARCH ALGORITHMS搜尋演算法的特性 10](#_Toc166514630)

[Minimax 11](#_Toc166514631)

[🕰️ 最小最大演算法的時間複雜度 12](#_Toc166514632)

[⚖️ 最小最大演算法的局限性 12](#_Toc166514633)

[✂️ α-β剪枝法中 12](#_Toc166514634)

[Alpha是甚麼? 12](#_Toc166514635)

[beta 是甚麼? 13](#_Toc166514636)

[ALPHA-BETA PRUNING 13](#_Toc166514637)

[🔄 最佳移動策略 14](#_Toc166514638)

[➕ 以极大和极小的策略进行移動 15](#_Toc166514639)

[🌳 遊戲樹的分析 15](#_Toc166514640)

[決策樹 (Decision tree) 16](#_Toc166514641)

[今日學習目標 16](#_Toc166514642)

[決策樹 16](#_Toc166514643)

[決策樹如何生成？ 17](#_Toc166514644)

[決策樹的混亂評估指標 18](#_Toc166514645)

[評估分割資訊量 19](#_Toc166514646)

[熵 (Entropy) 20](#_Toc166514647)

[Gini 不純度 (Gini Impurity) 21](#_Toc166514648)

[迴歸樹 22](#_Toc166514649)

[樹越深模型越複雜 23](#_Toc166514650)

[迴歸樹該如何選擇切割點? 24](#_Toc166514651)

[CART 決策樹 25](#_Toc166514652)

[決策樹模型的優缺點 26](#_Toc166514653)

[決策樹總結 26](#_Toc166514654)

[Please describe the process of agglomerative clustering algorithm, including how the dendrogram is generated and how the clusters are determined.請描述聚集聚類演算法的過程，包括樹狀圖是如何生成的，以及聚類是如何確定的 27](#_Toc166514655)

## FNN（Feedforward Neural Network）

- 中文描述：FNN（前饋神經網路）是一種**最基本的神經網路模型**，其中**訊息只會向前流動，不會形成循環或回饋。**它由多個層組成，每一層包含多個神經元（節點），**相鄰層之間的神經元之間存在權重**。每個神經元的輸出是前一層神經元的加權和，**經過激活函數處理後得到**。

- 英文描述：A Feedforward Neural Network (FNN) is a basic neural network model where information flows only in one direction, without cycles or feedback loops. It consists of multiple layers, each containing multiple neurons (nodes), with weighted connections between adjacent layers. The output of each neuron is the weighted sum of inputs from the previous layer, followed by an activation function.

## RNN（Recurrent Neural Network）

- 中文描述：RNN（循環神經網路）是一種**具有循環結構的神經網路**，能夠處理序列數據或時間序列數據。**RNN的特點是在每個時間步都會將當前的輸入與上一個時間步的隱藏狀態進行計算**，**產生當前時間步的輸出**。RNN**具有記憶功能**，可以**捕捉序列數據中的時間依賴性**。

- 英文描述：A Recurrent Neural Network (RNN) is a type of neural network with a loop structure that allows it to handle sequential or time-series data. In RNN, the current input at each time step is processed along with the hidden state from the previous time step to produce the output at the current time step. RNNs have the ability to capture temporal dependencies in sequential data.

## CNN（Convolutional Neural Network）

- 中文描述：CNN（卷積神經網路）是一種**專門用於處理圖像數據的神經網路模型**。它主要**包含卷積層（Convolutional Layer）、池化層（Pooling Layer）和全連接層（Fully Connected Layer）**。CNN通過使用卷積操作來提取圖像的特徵，並通過池化操作進行特徵降維，最終通過全連接層進行分類或回歸。

- 英文描述：A Convolutional Neural Network (CNN) is a type of neural network designed specifically for processing image data. It typically consists of convolutional layers, pooling layers, and fully connected layers. CNNs use convolution operations to extract features from images and pooling operations to reduce the dimensionality of the features, ultimately performing classification or regression tasks using fully connected layers.

## FNN RNN CNN三者之間的差別

**結構差異：**

FNN：訊息只向前傳播，沒有循環結構。

RNN：具有循環結構，能夠處理序列或時間序列數據。

CNN：主要用於處理圖像數據，包含卷積、池化和全連接層。

**Structural Differences:**

**Structural Differences:**

* **FNN (Feedforward Neural Network):** Information flows only forward without any cyclic structure.
* **RNN (Recurrent Neural Network):** Contains cyclic structure and is capable of handling sequential or time-series data.
* **CNN (Convolutional Neural Network):** Primarily used for processing image data and includes convolutional layers, pooling layers, and fully connected layers.

**應用領域差異：**

FNN：適用於一般的分類或回歸任務。

RNN：適用於序列建模，如自然語言處理（NLP）、時間序列預測等。

CNN：主要用於計算機視覺任務，如圖像分類、物體檢測等。

**Application Domain Differences:**

* **FNN:** Suitable for general classification or regression tasks.
* **RNN:** Suitable for sequence modeling tasks such as natural language processing (NLP) and time-series prediction.
* **CNN:** Mainly used for computer vision tasks like image classification and object detection.

**特點差異：**

FNN：較簡單，無法處理序列數據的時間依賴性。

RNN：具有記憶能力，能夠捕捉序列數據中的時間依賴性。

CNN：專門用於處理圖像數據，具有平移不變性（Translation Invariance）。

**Characteristic Differences:**

* **FNN:** Relatively simple and unable to handle time dependencies in sequential data.
* **RNN:** Possesses memory capabilities and can capture temporal dependencies in sequential data.
* **CNN:** Specifically designed for processing image data with translation invariance properties.

## 激活函數（Activation Function）

神經網路中的一種重要組件，它通常**被應用於神經元的輸出，用於引入非線性特性**，並決定神經元是否應該被激活（fired）。

在神經網路中，**每個神經元接收來自前一層的加權輸入（包括偏差項），然後通過激活函數計算出神經元的輸出**。激活函數**將線性組合的輸入轉換成非線性輸出**，這對於神經網路的學習和表示能力至關重要。

An activation function is a critical component in neural networks applied to determine the output of a neuron based on its input. It introduces non-linearity into the network, allowing it to model complex relationships and make decisions about whether a neuron should be activated (fired).

In neural networks, each neuron receives weighted inputs from the previous layer, including a bias term, and computes its output using an activation function. The activation function transforms the linear combination of inputs into a non-linear output, which is essential for the learning and representational capabilities of neural networks.

一張含有 文字, 螢幕擷取畫面, 字型, 數字 的圖片

自動產生的描述**Sigmoid Function** (Logistic Function): It compresses the input into the range of (0,1)(0,1) and is often used in the final layer of binary classification tasks.

**Tanh Function** (Hyperbolic Tangent Function): It compresses the input into the range of (−1,1)(−1,1) with a centering property.

**ReLU Function** (Rectified Linear Unit Function): It outputs the input if it's positive, otherwise outputs zero, commonly used in hidden layers.

**Softmax Function**:It converts a vector into a probability distribution, often used in the final layer of multi-class classification tasks

## Entropy 與 Cross Entropy 的差別

中文描述：

### Entropy（熵）

是訊息理論中衡量**隨機變量不確定度或混亂程度的度量**。在機器學習中，熵用於**衡量一個概率分佈的不確定性**，計算方式為將每個事件的概率乘以其自然對數，然後**求和取負值。熵越高，概率分佈的不確定性越大**。

1. Entropy:

- Definition: Entropy is a measure of uncertainty or disorder of a random variable in information theory. In machine learning, entropy is used to quantify the uncertainty of a probability distribution. It is calculated by multiplying each event's probability by its natural logarithm, then summing the results and taking the negative value. Higher entropy indicates greater uncertainty in the probability distribution.

### Cross Entropy（交叉熵）

是一種**衡量兩個概率分佈之間的差異或不匹配程度的指標**。在機器學習中，交叉熵**常用於衡量模型預測的概率分佈與真實概率分佈之間的距離**。交叉熵的計算方式是**將真實概率分佈的每個事件的概率乘以其自然對數，然後與模型預測的概率分佈相乘，最後求和取負值**。

英文描述：

2. Cross Entropy:

- Definition: Cross Entropy is a metric that measures the difference or mismatch between two probability distributions. In machine learning, cross entropy is commonly used to quantify the distance between the predicted probability distribution of a model and the true probability distribution. Cross entropy is calculated by multiplying each event's probability from the true distribution by its natural logarithm, then multiplying it with the corresponding probability from the model's predicted distribution, and finally summing the results and taking the negative value.

總之，熵衡量機率分佈內的不確定性，而交叉熵衡量兩個機率分佈之間的差異。交叉熵通常用作機器學習任務中的損失函數，例如分類，以根據預測分佈和真實分佈之間的差異來優化模型參數

In summary, entropy measures uncertainty within a probability distribution, while cross entropy measures the divergence between two probability distributions. Cross entropy is often used as a loss function in machine learning tasks such as classification to optimize model parameters based on the difference between predicted and true distributions.

Formula

## DFS

一張含有 文字, 螢幕擷取畫面, 圖表, 字型 的圖片

自動產生的描述

It follows each path to its greatest depth before

moving on to the next path

 When leaf node is reached, trace it back 到達葉子節點時回溯

A stack is maintained for controlling the search 控制堆疊來控制搜尋

 The first entry is popped 第一個入的已經彈出

 Spanned nodes are inserted at head of the list 新加入的節點家在list的前面

## BFS

一張含有 文字, 螢幕擷取畫面, 圖表, 行 的圖片

自動產生的描述

Examine all nodes one level down from the root node

More memory usage more than depth-first

 Depth increases gradually

 Goals in lower depth will be found first

A queue佇列is maintained for controlling the search, Newly spanned nodes新跨入的節點 are inserted at back of the queue插入到後面的佇列

## DFS VS. BFS

Ways of managing open list

DFS: inserted at front (as stack)

BFS: inserted at back (as queue)  
 It is often not easy to consider optimality, time complexity and space complexity simultaneously

DFS: lower memory requirement

BFS: minimum-length path  
 Which is better?

It depends on the problems

What is the shape of state space

Where are the goal states located

DFS VS. BFS

 DFS is more often used: Less memory usage

 Search tree may have very deep path, and goal node is in shallower part

Breadth-first good, but depth-first poor

8-Puzzle: very deep path, DFS not good

 All paths are of similar length

DFS good (do not try lower level)

 Search tree has high branching factor

DFS good (BFS consumes too many memories)

## CHARACTERISTICS OF SEARCH ALGORITHMS搜尋演算法的特性

Completeness完整性：如果存在目標狀態，則如果保證找到目標狀態，則搜尋方法是完整的。例如拼圖問題的 DFS 不完整。

A search method is complete if it is guaranteed to find a goal state provided there exists goal state.

e.g. DFS for puzzle problem is not complete.

Optimality 最適性 如果保證找到現有的最佳解，則搜尋方法是最優的。「最好」與「善良」的定義相關

A search method is optimal if it is guaranteed to find the best solution that exists“Best” is relevant to the definition of “Goodness” .

Irrevocability (no back tracing) 不使用回溯的方法是不可撤銷的。不可撤銷的方法，例如爬山，往往會被局部最優所愚弄

Methods that do not use backtracking are irrevocable. Irrevocable methods, such as hill-climbing, tend to be fooled by local optima.

Admissibility可接受性（Admissibility） 如果搜尋演算法能夠保證在任何時候存在一條最優路徑時都能找到解決方案，那麼該搜尋演算法就是可接受的。  任何演算法 A\* 都是可接受的。

A search algorithm is admissible if it is guaranteed to find a optimal path to a solution whenever such a path exists.

Any algorithm A\* is admissible.

## Minimax

一張含有 文字, 螢幕擷取畫面, 設計 的圖片

自動產生的描述

一張含有 行, 圖表, 設計, 三腳架 的圖片

自動產生的描述

## 🕰️ 最小最大演算法的時間複雜度

The time complexity of the minmax algorithm depends on the path factor and depth of the game tree. The degree is the depth power of the channel factor, which is O(B^D). The complexity increases with the branching factor and depth.

最小最大演算法的時間複雜度取決於遊戲樹的分支因子和深度。分支因子表示每個遊戲狀態下可能的移動數量，而深度表示遊戲樹中的層數或回合數。最小最大演算法的時間複雜度是分支因子的深度次方，即O(B^D)。複雜度的增長隨著分支因子和深度的增加而增加。

## ⚖️ 最小最大演算法的局限性

The main limitation of the minmax algorithm is that **it is not efficient when dealing with certain games**. **When the branching factor and depth of the game are very large, the game tree will become very large, causing the game tree search to be very time-consuming and computationally resource-consuming**.

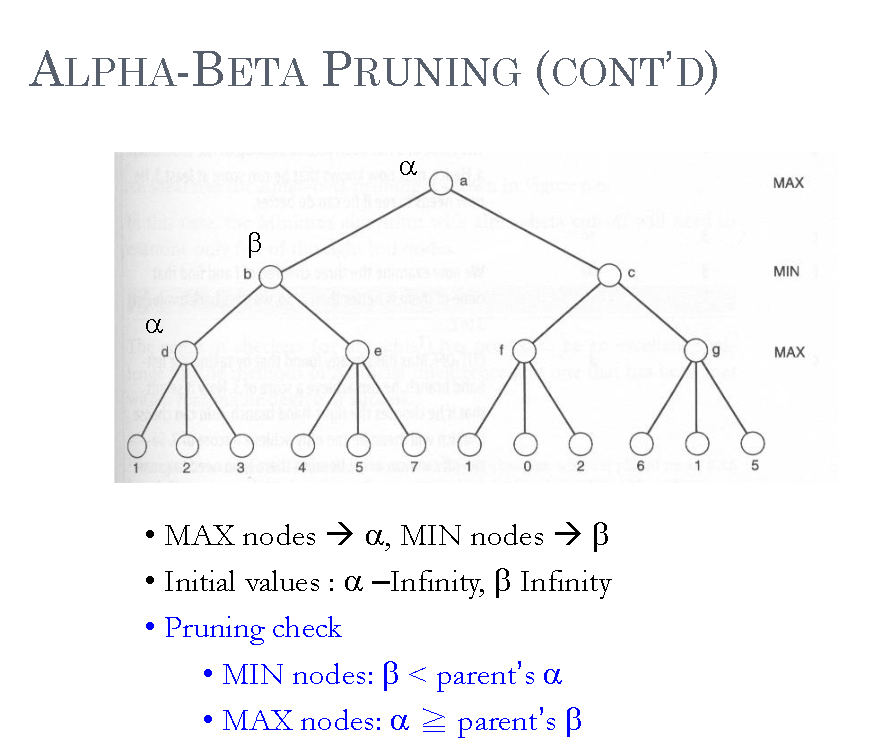
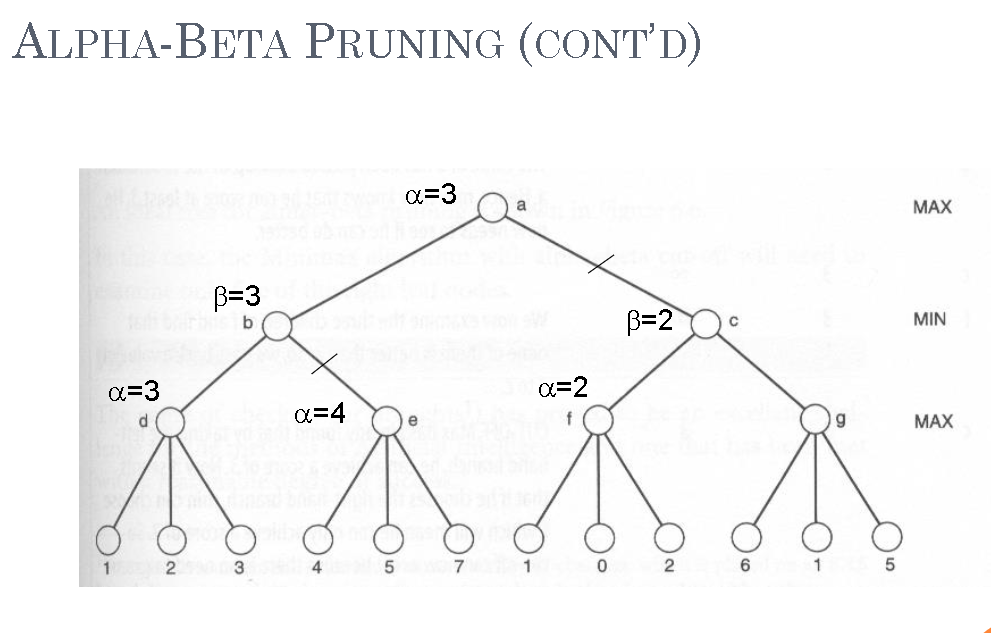
For example, in chess, the branching factor is on average 35 and the depth is on average 100, which results in a very large number of nodes in the game tree, making it difficult to perform a complete search. In order to solve this problem, we need to use other algorithms, such as α-β pruning.

最小最大演算法的**主要局限性在於它在處理某些遊戲時的效率不高**。當遊戲的**分支因子和深度非常大時，遊戲樹會變得非常龐大，導致遊戲樹搜索的時間和計算資源非常消耗**。

例如在國際象棋中，分支因子平均為35，深度平均為100，這導致遊戲樹的節點數量非常龐大，難以進行完整的搜索。為了解決這個問題，我们需要使用其他的演算法，如α-β剪枝法。

## ✂️ α-β剪枝法中

### Alpha是甚麼?

alpha is the temporary maximum during the execution.

alpha 是執行期間的暫時最大值。

alpha can increase only (could be replaced by a larger entry) during the the maximization process.

**alpha 只能在最大化過程中增加**（可以用更大的條目取代）

alpha is the maximum finally after all elements are visited

**alpha 是所有元素被存取後最終的最大值**

### beta 是甚麼?

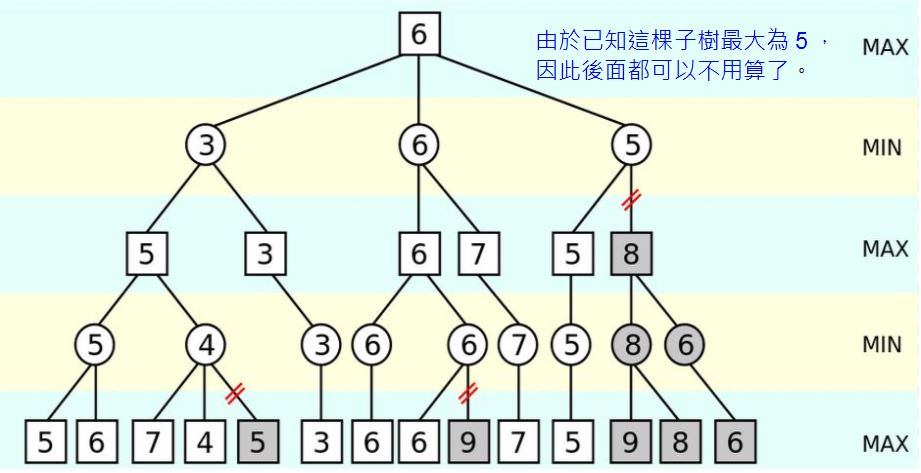
**beta is the temporary minimum during the execution.**

**beta 是執行期間的暫時最小值。**

**beta can decrease only beta 只能減少**

**beta is the maximum finally after all elements are visited** **beta 是所有元素訪問後最終的最大值**

### ALPHA-BETA PRUNING



A method that can often cut off a large part of the game tree. Based on the idea that if a move is clearly bad, there is no need to waste time to search the consequences of it completely

ALPHA-BETA 剪枝 

一種通常可以剪掉賽局樹大部分的方法。基於這樣的想法：如果一個舉動明顯是不好的，那麼就沒有必要浪費時間去徹底尋找它的後果

**The α-β pruning method is an optimized min-max algorithm used to reduce the amount of calculations in game tree search**. **It improves search efficiency by limiting the search scope and avoiding unnecessary parts of the game tree**.

**When a node is searched, the α-β pruning method will determine the nodes that do not need to be searched based on the known best movement strategy, and then prune.**

This method can greatly reduce the time and computing resources of game tree search and improve the efficiency of the game algorithm.

**α-β剪枝法是一種優化的最小最大演算法，用於减少遊戲樹搜索中的計算量。它通過限制搜索範圍，避免遊戲樹搜索不必要的部分，從而提高搜索效率。**

**當搜索到某個節點時，α-β剪枝法會根據已知的最佳移動策略來確定不必搜索的節點，從而剪枝。**

這種方法可以大大減少遊戲樹搜索的時間和計算資源，提高遊戲演算法的效率。

## 🔄 最佳移動策略

In game theory, **the optimal movement strategy refers to the strategy developed by players in the game that can not only maximize their own benefits, but also inhibit their opponents' maximum benefits.** The player's goal is **to gain maximum benefit through optimal movement strategies, while opponents work hard to prevent you from winning**. The optimal movement strategy can be formulated based on game rules and game status, and can be adjusted accordingly in each turn based on the opponent's movements.

遊戲理論中，**最佳移動策略表示玩家在遊戲中制定的既能使自己獲得最大利益，又能抑制對手獲得最大利益的策略**。玩家的目標**是通過最佳移動策略來獲得最大利益，同時對手也在努力阻止你獲勝。**最佳移動策略可以根據遊戲規則和遊戲狀態來制定，並且可以在每一個回合中根據對手的移動作出相應的調整。

## ➕ 以极大和极小的策略进行移動

**The "maximum" and "minimum" in the name of the minmax algorithm represent the two players in the game, namely the maximum player and the minimum player. The largest player tries to maximize his own interests, while the smallest player tries to minimize the largest player's interests.**

When it is the turn of the largest player to move, **he will choose the move that maximizes his own interests**; and when it is the turn of the smallest player to move, **he will choose the move that maximizes the interests of the largest player.** Such strategies are carried out alternately until the conditions for the end of the game are reached.

**最小最大演算法的名稱中的"最大"和"最小"表示的是遊戲中的兩個玩家，即最大玩家和最小玩家。最大玩家試圖最大化自己的利益，而最小玩家則試圖最小化最大玩家的利益。**

當輪到最大玩家移動時，他會**選擇使自己利益最大化的移動**；而當輪到最小玩家移動時，**他會選擇使最大玩家得到最小利益的移動**。這樣的策略交替進行，直到達到遊戲結束的條件。

## 🌳 遊戲樹的分析

The minmax algorithm uses game trees to analyze various movement strategies of the game. **A game tree is a tree-like structure consisting of all possible moves and game states of the game.**

By recursively computing each node of the game tree, **the minmax algorithm can determine the benefit generated by each move and select the next move based on the best benefit.**

The depth of the game tree depends on the complexity of the game and the length of the game.

最小最大演算法使用遊戲樹來分析遊戲的各種移動策略。**遊戲樹是由遊戲的所有可能移動和遊戲狀態構成的樹狀結構**。

通過遞歸運算遊戲樹的每個節點，**最小最大演算法可以確定每個移動所產生的利益，並根據最佳的利益來選擇下一步的移動**。

遊戲樹的深度取決於遊戲的複雜性和棋局的長度。

# 決策樹 (Decision tree)

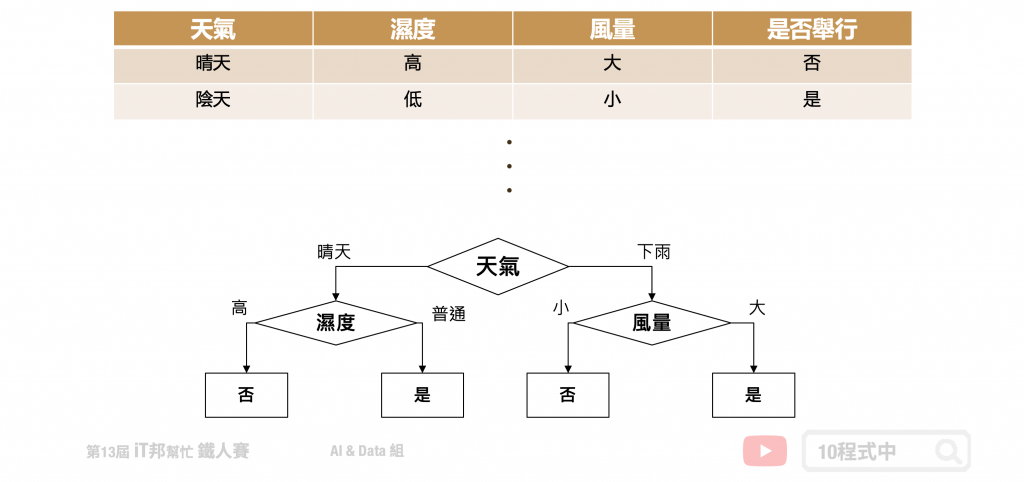
## 今日學習目標

* 決策樹演算法介紹
  + 決策樹如何生成？
  + 如何處理分類問題？
  + 如何處理迴歸問題？
* 實作決策樹分類器
  + 觀察決策樹是如何生成的。
* 實作決策樹迴歸器
  + 查看決策樹方法在簡單線性迴歸和非線性迴歸表現。

## 決策樹

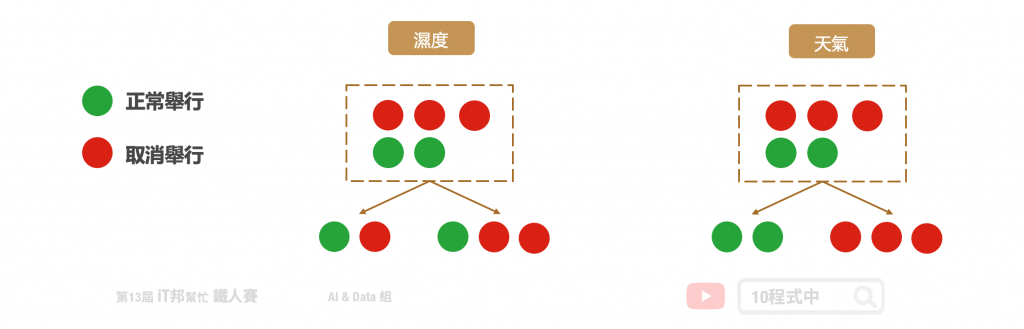
決策樹會根據訓練資料產生一棵樹，依據訓練出來的規則來對新樣本進行預測。決策樹演算法可以使用不同的方式來評估分枝的好壞(亂度)，例如像是 Information gain、Gain ratio、Gini index。依據訓練資料找出合適的規則，最終生成一個規則樹來決策所有事情，其目的使每一個決策能夠使訊息增益最大化。就好比我們評估今天比賽是否舉行，天氣因子可能站比較大的因素，而 Co2 的濃度高低可能佔的因子程度較低。因此在第一層的決策中以天氣的特徵先進行第一次的決策判斷。接著第二層再從所有特徵中尋找最適合的決策因子，直到設定的最大樹的深度即停止樹的生長。

The decision tree generates a tree based on the training data, and the new sample is predicted according to the trained rules. Decision tree algorithms can be used in different ways to evaluate the quality (disorder) of branches, such as information gain, gain ratio, and gini index. Based on the training data, the appropriate rules are found, and finally a rule tree is generated to make decisions about everything, with the aim of maximizing the message gain for each decision. For example, if we evaluate whether or not a race will be held today, the weather factor may be a relatively large factor, and the concentration of CO2 may account for a lower factor. Therefore, in the first level of decision-making, the first decision-making judgment is made based on the characteristics of the weather. The second layer then looks for the most suitable decision-making factor from all the characteristics until the maximum depth of the tree is set, which stops the tree growth.



## 決策樹如何生成？

**決策樹是以一個貪婪法則來決定每一層要問什麼問題，目標是分類過後每一群能夠很明顯的知道是屬於哪一種類別。**延續上面的例子，以分類問題來說假設要評估明天比賽是否舉行。在樹的第一層節點中我們要從已知的兩個特徵分別是溫度與特徵選一個作為該層的決策因子。假設目前訓練集有五筆資料，其中正常舉行的有兩筆資料，取消舉行的有三筆資料。在樹的結構中左子樹為決策正常取行，而右子樹是決策取消舉行。我們可以發現當特徵為天氣的時候可以一很清楚的將這兩類別完整分開，因此我們會將天氣作為這一層判斷的因子。這就是決策樹在生成中的貪婪機制**。然而要如何去判斷每次決策的好壞，就必須依靠亂度的評估指標。The decision tree uses a law of greed to determine what questions to ask at each level, with the goal of making it clear to each group which category it belongs to. Continuing with the above example, let's assume that the game will be held tomorrow in the case of a classification problem. In the first layer of the tree, we want to choose one of the two known features, temperature and features, as the decision factor of the layer. Suppose that there are five data in the current training set, of which two data are held normally and three data are cancelled. In the structure of the tree, the left subtree is the normal row for the decision, while the right subtree is the decision cancellation held. We can find that when the feature is weather, the two categories can be clearly separated, so we will use weather as a factor in this layer of judgment. This is the mechanism of greed in the generation of decision trees. However, how to judge the quality of each decision, we must rely on the evaluation index of chaos.**



## 決策樹的混亂評估指標

我們需要客觀的標準來決定決策樹的每個分支，因此我們需要有一個評斷的指標來協助我們決策。決**策樹演算法可以使用不同的指標來評估分枝的好壞，常見的決策亂度評估指標有 Information gain、Gain ratio、Gini index。**我們目標是從訓練資料中找出一套決策規則，讓每一個決策能夠使訊息增益最大化。**以上的指標都是在衡量一個序列中的混亂程度，其數值越高代表越混亂。然而在 Sklearn 套件中預設使用 Gini。**

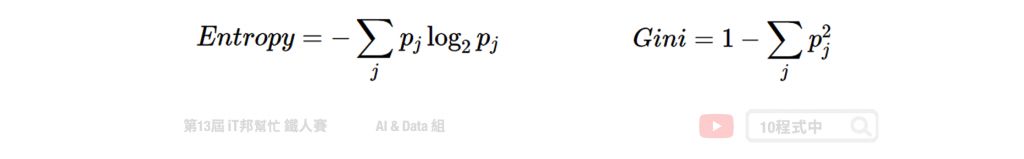
**We need objective criteria to decide on each branch of the decision tree, so we need to have a judgmental metric to assist us in our decision-making. The decision tree algorithm can use different indicators to evaluate the quality of branches, and the common decision chaos evaluation indicators are Information gain, Gain ratio, and Gini index. Our goal is to find a set of decision rules from the training data so that each decision maximizes the message gain. All of the above metrics measure the degree of chaos in a series, with higher values representing more confusion. However, Gini is used by default in the Sklearn suite.**

* Information gain (資訊獲利)
* Gain ratio (吉尼獲利)
* Gini index (吉尼係數) = Gini Impurity (吉尼不純度)

## 評估分割資訊量

Information Gain 透過從訓練資料找出規則，讓每一個決策能夠使訊息增益最大化。其算法主要是計算熵，因此經由決策樹分割後的資訊量要越小越好。**而 Gini 的數值越大代表序列中的資料亂，數值皆為 0~1 之間，其中 0 代表該特徵在序列中是完美的分類。**常見的資訊量評估方法有兩種：資訊獲利 (Information Gain) 以及 Gini 不純度 (Gini Impurity)。

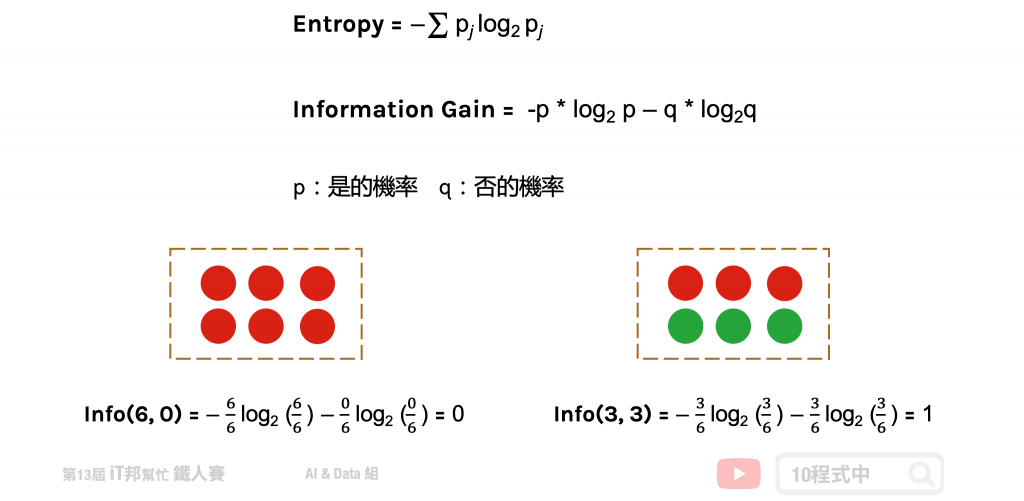
Information Gain maximizes message gain by identifying rules from training data to make every decision. The algorithm mainly calculates entropy, so the amount of information segmented by the decision tree should be as small as possible. The larger the value of Gini, the larger the data in the sequence, the value is between 0~1, where 0 means that the feature is perfectly classified in the sequence. There are two common methods of information assessment: Information Gain and Gini Impurity.



## 熵 (Entropy)

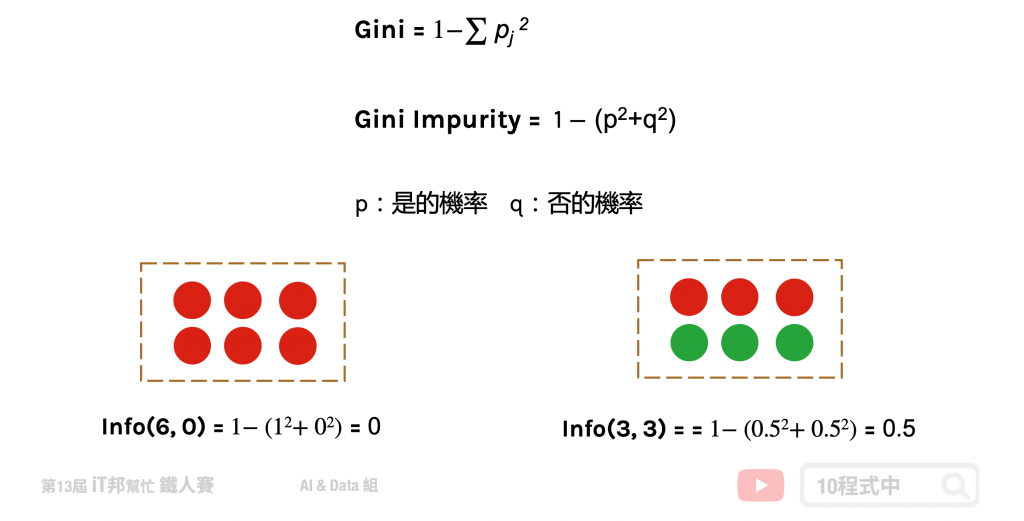
熵 (Entropy) 是計算 Information Gain 的一種方法。在了解 Information Gain 之前要先了解熵是如何被計算出來的。**其中在下圖公式中 p 代表是的機率、q 代表否的機率。我們可以從圖中範例很清楚地知道當所有的資料都被分類一致的時候 Entropy 即為 0，當資料各有一半不同時 Entropy 即為 1。**

**Entropy is a method of calculating Information Gain. Before we get to know Information Gain, it's important to understand how entropy is calculated. In the formula below, p represents the probability of being yes and q represents the probability of no. As we can clearly see from the example in the diagram, Entropy is 0 when all data are classified consistently, and 1 when half of the data is different.**



## Gini 不純度 (Gini Impurity)

Gini 不純度是另一種亂度的衡量方式，**它的數字越大代表序列中的資料越混亂**。公式如下所示，其中 p 代表是的機率、q 為代表否的機率。我們可以從圖中範例很清楚地知道當所有的資料都被分類一致的時候混亂程度即為 0，當資料各有一半不同時混亂程度即為 0.5。



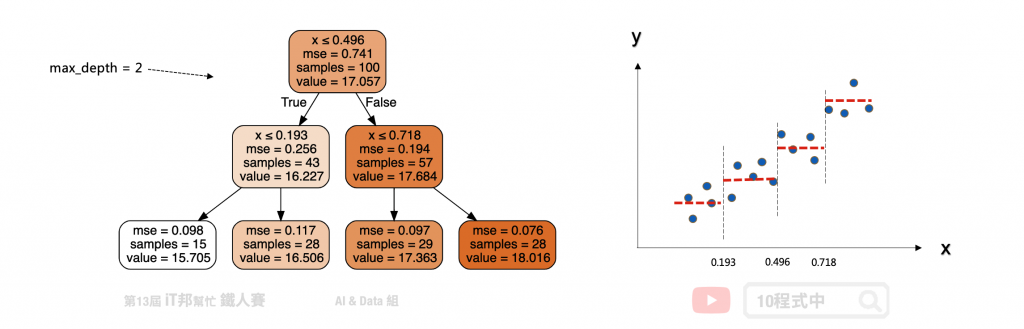
## 迴歸樹

**決策樹迴歸方法與分類有點類似差別僅在於評估分枝好壞的方式不同，我們又可以稱作迴歸樹**。當數據集的輸出爲連續性數值時，該樹算法就是一個迴歸樹。透過樹的展開，並用葉節點的均值作爲預測值。從根節點開始，對樣本的某一特徵進行測試。經過評估後，將樣本分配到其子結點。此時每一個子節點對應著該特徵的一個值。依照這樣方式進行，直至到達葉結點。此時誤差值要最小化，並且越接近零越好。

The decision tree regression method is somewhat similar to classification, except that the way of evaluating the quality of the branches is different, which we can also call a regression tree. When the output of the dataset is a continuous value, the tree algorithm is a regression tree. Through the unfolding of the tree, the mean of the leaf nodes is used as the prediction value. Starting from the root node, a feature of the sample is tested. After evaluation, the samples are assigned to their child nodes. Each child node corresponds to a value for the feature. This is done until the leaf node is reached. In this case, the error value should be minimized, and the closer to zero the better.

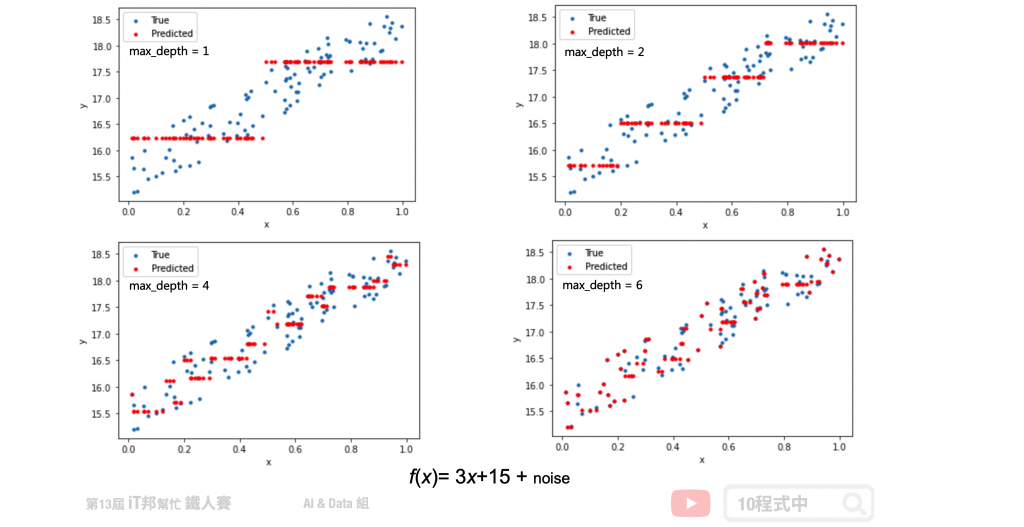
迴歸樹的生長過程很推薦看[這篇](https://zhuanlan.zhihu.com/p/82054400)文章

以下舉一個例子假設 x 是輸入 y 是輸出，我們可以在一個平面上繪製出資料與正確答案間的分佈。假設迴歸樹的最大深度設定兩層。首先在第一層中會將所有的資料從中間切一刀此斷點為 x=0.496 當大於設定的值的數據點會繼續往右子樹下去延伸，反之小於 0.496 的資料點會往左子樹走。此時將會切出一個分支出來並往下擴展並形成第二層的決策分支。一直不斷持續拓展直到設定的最大深度終止，此時的節點即為葉節點也就是最終的模型輸出值。As an example, assuming x is the input and y is the output, we can plot the distribution between the data and the correct answer on a plane. Suppose the maximum depth of the regression tree is set to two levels. First of all, in the first layer, all the data will be cut from the middle, this breakpoint is x=0.496, when the data point greater than the set value will continue to extend to the right subtree, otherwise, the data point less than 0.496 will go to the left subtree. A branch will be cut out and expanded down to form a second level of decision branch. It continues to expand until the set maximum depth is terminated, at which point the node is the leaf node, which is the final model output.



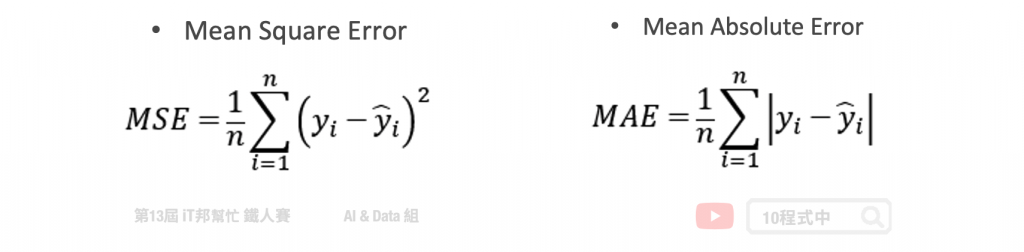
## 樹越深模型越複雜

假設我們生成一個 f(x) = 3x+15 + noise 的資料，其中 noise 為一個 0~1 之間的隨機數。從以下的測試可以看出隨著決策樹深度的增加，決策樹的擬合能力不斷上升。決策樹已經不僅僅擬合了我們的線性函式 3x+15，同時也擬合了我們添加的噪音(noise)。The deeper the tree, the more complex the model Let's say we generate a data with f(x) = 3x+15 + noise, where noise is a random number between 0~1. From the following tests, it can be seen that the fitting ability of the decision tree increases as the depth of the decision tree increases. The decision tree already fits not only our linear function 3x+15, but also the noise we add.



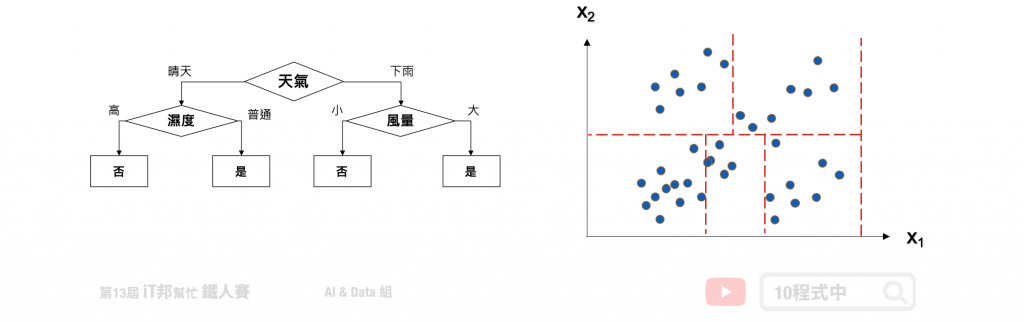
## 迴歸樹該如何選擇切割點?

在分類模型中決策樹是以亂度作為決策樹生成時候的評估指標。但是迴歸樹透過是 MSE 或 MAE 來評估模型，並找出誤差最小的值作為樹的特徵選擇與切割點。其中前者是均方差，後者是和均值之差的絕對值之和。How to choose a cutting point for a regression tree? In the classification model, the decision tree uses chaos as the evaluation index when the decision tree is generated. However, the regression tree evaluates the model by being an MSE or MAE, and finds the value with the least error as the feature selection and cutting point of the tree. where the former is the mean square deviation and the latter is the sum of the absolute values of the difference from the mean.



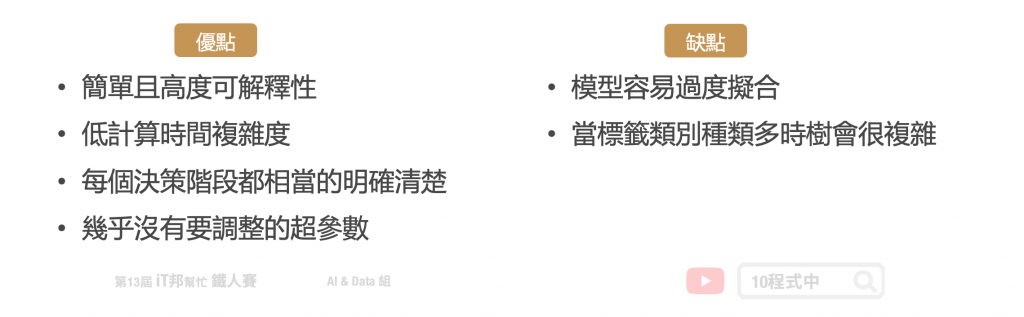
## CART 決策樹

在 Sklearn 套件中決策樹演算法是採用 CART (Classification and Regression Tree) 演算法，並且可以被拿來做分類和迴歸的預測。在決策樹的每一個節點上都是採用二分法，也就是每一個決策節點只分枝出兩個子節點。並且不斷地往下拓展，直到設定的最大深度為止，此時最大深度的節點稱為葉節點即為模型的預測輸出。In the Sklearn suite, the decision tree algorithm uses the CART (Classification and Regression Tree) algorithm and can be used to make predictions for classification and regression. The dichotomy is used at each node of the decision tree, that is, each decision node branches out only two subnodes. And continue to expand down until the maximum depth is set, at this time the node with the maximum depth is called the leaf node, which is the predicted output of the model.



## 決策樹模型的優缺點

建立決策樹的過程就是不斷的尋找特徵進行決策，透過這些決策盡量的使這些資料被分為同一個類別，且試著讓混亂程度越小越好。切記樹的深度越深不一定越好，他可能會造成過度擬合的問題。訓練好的模型我們能夠視覺化決策樹的結構，相對的可解釋性就變高。此外與其它的ML模型比較起來，決策樹執行速度是它的一大優勢。因為是樹狀結構，因此在進行機器學習的時候每個決策階段都相當的明確清楚，不是 0 就是 1。Advantages and disadvantages of the decision tree model The process of building a decision tree is to constantly look for features to make decisions, through which the data can be divided into the same category as much as possible, and try to make the confusion as small as possible. Keep in mind that the deeper the tree is, the better, and it can cause problems with overfitting. After the trained model is trained, we can visualize the structure of the decision tree, and the relative interpretability will be higher. In addition, the speed of decision tree execution is a major advantage compared to other ML models. Because it is a tree-like structure, each decision stage is quite clear when machine learning is carried out, and it is either 0 or 1.



## 決策樹總結

決策樹透過所有特徵與對應的值將資料切分，來找出最適合的分枝並繼續往下拓展。若決策樹深度越深則決策的規則將越複雜，模型預測也會越接近真實答案。但若訓練集中含有過多的雜訊，太深的樹就有可能產生過擬合的情形。因此單一的決策樹肯定是不夠用的，我們可以利用集成學習中的 Boosting 架構，對迴歸樹進行改良升級。

Decision Tree Summary The decision tree slices the data through all the features and corresponding values to find the most suitable branch and continue to expand downward. If the depth of the decision tree is deeper, the more complex the rules of the decision will be, and the closer the model prediction will be to the real answer. However, if the training set contains too much noise, trees that are too deep may be overfitted. Therefore, a single decision tree is definitely not enough, and we can use the boosting architecture in ensemble learning to improve and upgrade the regression tree.

## Please describe the process of agglomerative clustering algorithm, including how the dendrogram is generated and how the clusters are determined.請描述聚集聚類演算法的過程，包括樹狀圖是如何生成的，以及聚類是如何確定的

The agglomerative clustering algorithm starts by treating each data point as its own cluster. It then iteratively merges the closest clusters based on calculated distances. The merging process continues until all data points belong to a single cluster.

Throughout this process, a dendrogram is constructed to visualize how clusters merge over iterations. This dendrogram represents the hierarchy of cluster mergers, from individual data points at the bottom to the complete dataset at the top.

To determine the number of clusters, you interpret the dendrogram by selecting a threshold height. Lower thresholds result in more, smaller clusters, while higher thresholds lead to fewer, larger clusters. This approach allows for flexible cluster determination based on the desired level of similarity between data points.

凝聚聚類演算法首先將每個資料點視為自己的聚類。然後，它根據計算的距離迭代地合併最近的簇。合併過程持續進行，直到所有資料點都屬於一個叢集。 在整個過程中，建立樹狀圖來視覺化群集如何在迭代中合併。此樹狀圖表示簇合併的層次結構，從底部的單一資料點到頂部的完整資料集。 要確定簇的數量，您可以透過選擇閾值高度來解釋樹狀圖。較低的閾值會產生更多、更小的簇，而更高的閾值會產生更少、更大的簇。這種方法允許根據數據點之間所需的相似性水平進行靈活的聚類確定。

The agglomerative clustering algorithm is a bottom-up approach used to group similar data points into clusters. Here's how the process works, including the generation of a dendrogram and determination of clusters:

1. \*\*Initialization\*\*:

- Begin with each data point as its own cluster.

2. \*\*Calculate Pairwise Distances\*\*:

- Compute the distance (or dissimilarity) between each pair of clusters. This could be based on various distance metrics such as Euclidean distance, Manhattan distance, or correlation distance.

3. \*\*Merge Closest Clusters\*\*:

- Identify the two closest clusters based on the calculated distances and merge them into a single cluster. The choice of distance and linkage criterion (e.g., single-linkage, complete-linkage, average-linkage) determines how clusters are merged.

4. \*\*Update Distance Matrix\*\*:

- Recalculate the pairwise distances between the new cluster and the remaining clusters. This involves updating the distance matrix to reflect the new clustering configuration.

5. \*\*Repeat\*\*:

- Continue the process of merging the closest clusters iteratively until all data points are merged into a single cluster.

6. \*\*Dendrogram Generation\*\*:

- Throughout the clustering process, track the merging of clusters and distances at each step. This information is used to construct a dendrogram—a tree-like structure that illustrates the hierarchy of cluster mergers.

- The dendrogram starts with individual data points at the bottom and shows how clusters merge progressively until reaching a single root representing the entire dataset.

7. \*\*Cluster Determination\*\*:

- Decide on the number of clusters by interpreting the dendrogram. Clusters are identified by cutting the dendrogram at a specified height, which corresponds to a threshold distance.

- Lower thresholds result in more and smaller clusters, while higher thresholds lead to fewer and larger clusters.

\*\*Example\*\*:

Imagine clustering points in a 2D space. Initially, each point is its own cluster. The closest pair of points/clusters are merged, and distances are recalculated. This process continues until all points belong to a single cluster. The dendrogram visually represents these steps, and clusters are determined by choosing a suitable height cutoff.

In summary, agglomerative clustering progressively merges similar clusters based on proximity, generating a dendrogram that helps interpret the hierarchy of clustering and determine the optimal number of clusters based on specified criteria.

\*\*示例\*\*：

想像在二維空間中對點進行聚類。最初，每個點都是一個單獨的聚類。然後，找到最接近的一對點或聚類並將它們合併，然後重新計算距離。這個過程一直持續下去，直到所有點屬於一個單一的聚類。階層圖以圖形方式展示了這些步驟，並且可以通過在適當的高度上切斷階層圖來確定聚類的數量。

凝聚式聚類算法是一種自下而上的方法，用於將相似的數據點分組成聚類。以下是該過程的工作方式，包括如何生成階層圖（dendrogram）和確定聚類的過程：

1. \*\*初始化\*\*：

- 將每個數據點視為自己的一個聚類。

2. \*\*計算成對距離\*\*：

- 計算每對聚類之間的距離（或不相似度），這可以基於不同的距離度量標準，如歐式距離、曼哈頓距離或相關距離等。

3. \*\*合併最接近的聚類\*\*：

- 根據計算出的距離找到兩個最接近的聚類，並將它們合併為一個新的聚類。聚類的合併方式取決於所選擇的距離和聯接標準（例如單鏈接、完全鏈接或平均鏈接）。

4. \*\*更新距離矩陣\*\*：

- 重新計算新聚類與其餘聚類之間的成對距離，並更新距離矩陣以反映新的聚類配置。

5. \*\*重複執行\*\*：

- 通過迭代地合併最接近的聚類來持續進行此過程，直到所有數據點都合併為一個單一聚類。

6. \*\*生成階層圖\*\*：

- 在聚類過程中，跟蹤聚類的合併和距離，並使用這些信息構建階層圖（dendrogram）。階層圖是一種樹狀結構，用於顯示聚類合併的層次結構。

- 階層圖從底部開始，每個數據點作為葉子節點，逐步顯示聚類如何合併，直到達到一個根節點代表整個數據集。

7. \*\*聚類確定\*\*：

- 通過解釋階層圖來確定聚類的數量。通過在特定高度切斷階層圖，即對應到一個閾值距離，來確定聚類。

- 較低的閾值導致更多且較小的聚類，而較高的閾值則導致較少且較大的聚類。

## Simple Matching Coefficient (SMC) / Jaccard Coefficient / Rao’s Coefficient

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述 一張含有 文字, 字型, 螢幕擷取畫面, 數字 的圖片

自動產生的描述

# GENETIC ALGORITHM

## GENETIC ALGORITHMS 敘述

 A method based on biological evolution.

 Create chromosomes which represent possible solutions to a problem.

 A fitness value for each chromosome is determined.

 The best chromosomes in each generation are bred with each other to produce a new generation.

 Reproduction is done by applying crossover over two or more chromosomes

 Mutation is applied so as to make random changes to particular genes.

GA is a form of local search

 Based on evolution

 Borrow terms from genetics

 Chromosome, crossover, mutation, fitness

遺傳演算法 

一種基於生物演化的方法。 

創造代表問題可能解決方案的染色體。 

確定每條染色體的適應度值。 

每代中最好的染色體相互繁殖以產生新一代。 

繁殖是透過在兩個或更多染色體上應用交叉來完成的。

GA 是局部搜尋的一種形式

基於進化 借用遺傳學術語 染色體、交叉、突變、適應度

## GA方法

1. Create a random population of chromosomes (the first generation)

2. If the termination criteria are satisfied, stop.

3. Determine the fitness of each chromosome.

4. Apply crossover and mutation to selected chromosomes from current generation to generate a new population of chromosomes – the next generation

5. Return to step 2

遺傳演算法 1. 創建隨機染色體群體（第一代） 2. 如果滿足終止條件，則停止。 3. 確定每條染色體的適合度。 4. 對現世代中選定的染色體應用交叉和變異，以產生新的染色體群體－下一代 5. 返回步驟 2

## GA TERMINATION 終止條件

 GA is terminated if...

 A limit on the number of generations

 Particular solution is found

 The highest fitness level has reached a particular value

 Culling

 All individuals below a given threshold are discarded

 Converge faster than the random version

終止 

遺傳演算法終止，如果...

代數限制  找到特定解 最高適應度已達到特定值 

剔除  丟棄所有低於給定閾值的個體  比隨機版本收斂得更快