人工智慧

2020.0831 蔡宗諺

實驗繳交規定

實驗繳交格式：

- 檔案名稱：學號_ AILab.pdf
- 投影片內容包含：
  - 姓名＋學號
  - 程式碼檔案之雲端網址
- 頁數不拘，須依照各項實驗要求繳交影片或截圖
- 程式碼存至雲端並以網址方式貼至附件。
- 一人繳交一份即可(可討論，勿抄襲)
- (很重要！未依格式繳交者予以扣分，尤其是檔案名稱。)
- 收到作業會回信（請自行確認）

實驗繳交時間

- 實驗繳交時間：2020/1/12 23:59
- Gmail：jerrytsai860216@gmail.com
- 有任何問題可以寄信給我
- 實驗室：大仁樓#200208 MCLAB3

實驗檔案網址

- 實驗的所有檔案，均放在雲端網址上，其中兩個影片給同學參考如何操作**LAB-1**與**LAB-2**。
- 雲端網址：

https://drive.google.com/drive/folders/1Tr6c79oKw_jH3g-dfpX4AdHZ6YVfLGO_?usp=sharing

實驗列表

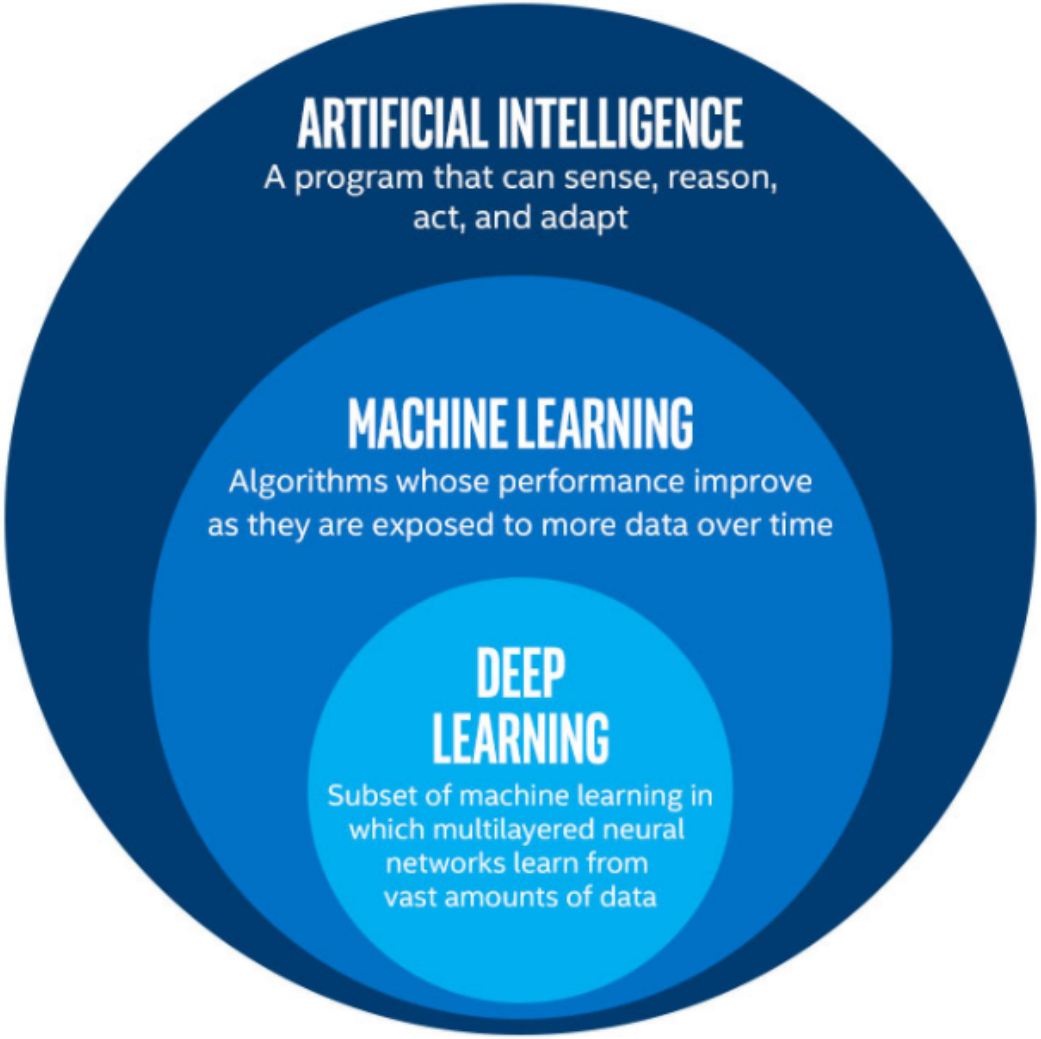
- Introduction
- Lab1：FizzBuzz
- Lab2：Titanic
- Lab3：A* 走迷宮

Introduction

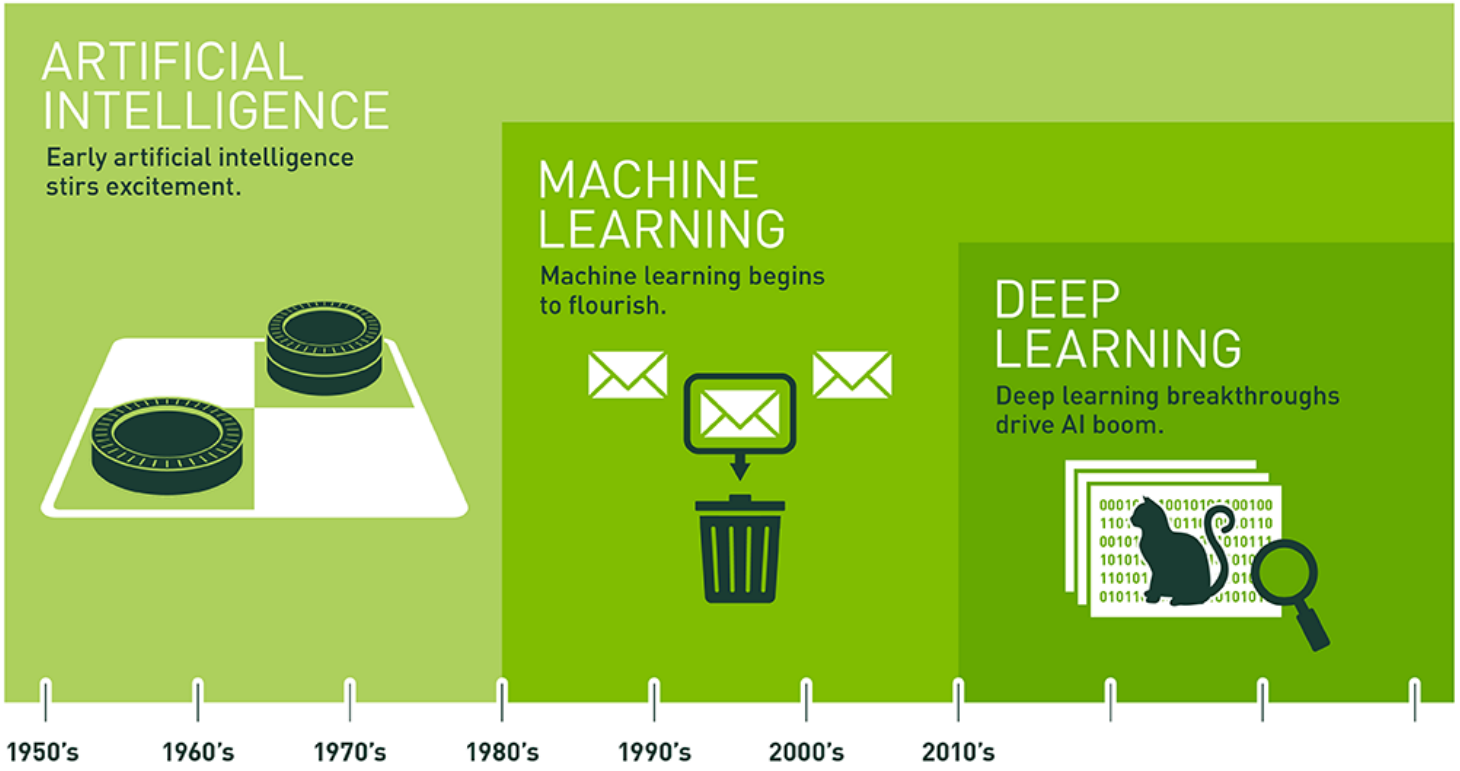
Introduction

- 人工智慧（Artificial Intelligence，AI）指由人製造出來的機器所表現出來的智慧。
- 通常人工智慧是指透過普通電腦程式來呈現人類智慧的技術。該詞也指出研究這樣的智慧系統是否能夠實現，以及如何實現。
- 通過醫學、神經科學、機器人學及統計學等的進步，預測則認為人類的很多職業也逐漸被其取代。

AI v.s. ML v.s. DL


ARTIFICIAL INTELLIGENCE
A program that can sense, reason, act, and adapt
MACHINE LEARNING
Algorithms whose performance improve as they are exposed to more data over time
DEEP LEARNING
Subset of machine learning in which multilayered neural networks learn from vast amounts of data

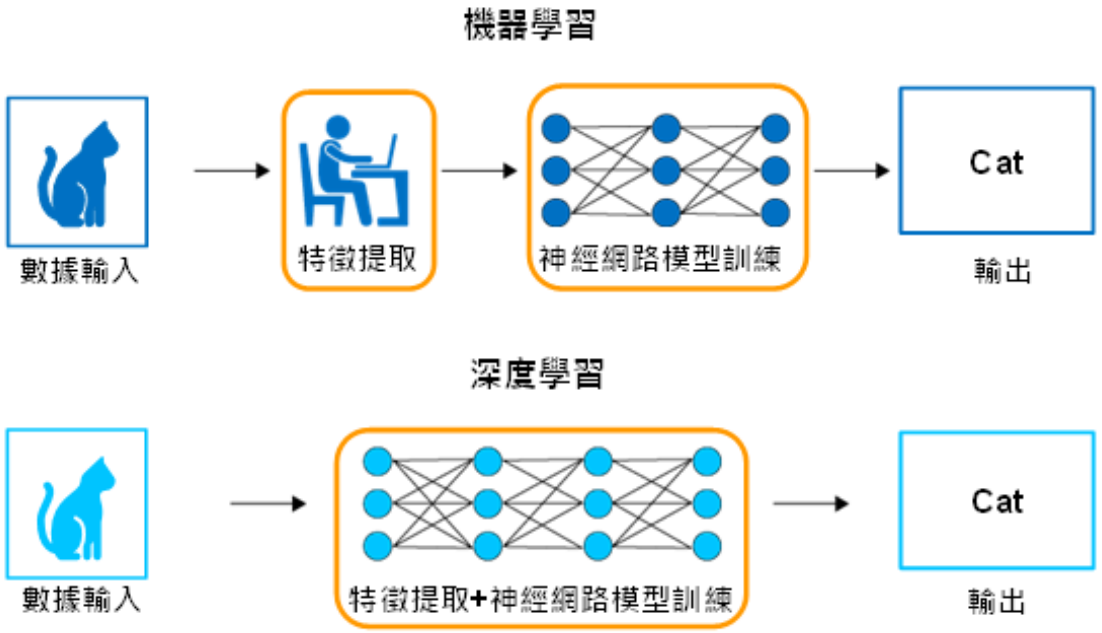
AI v.s. ML v.s. DL



Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

AI v.s. ML v.s. DL

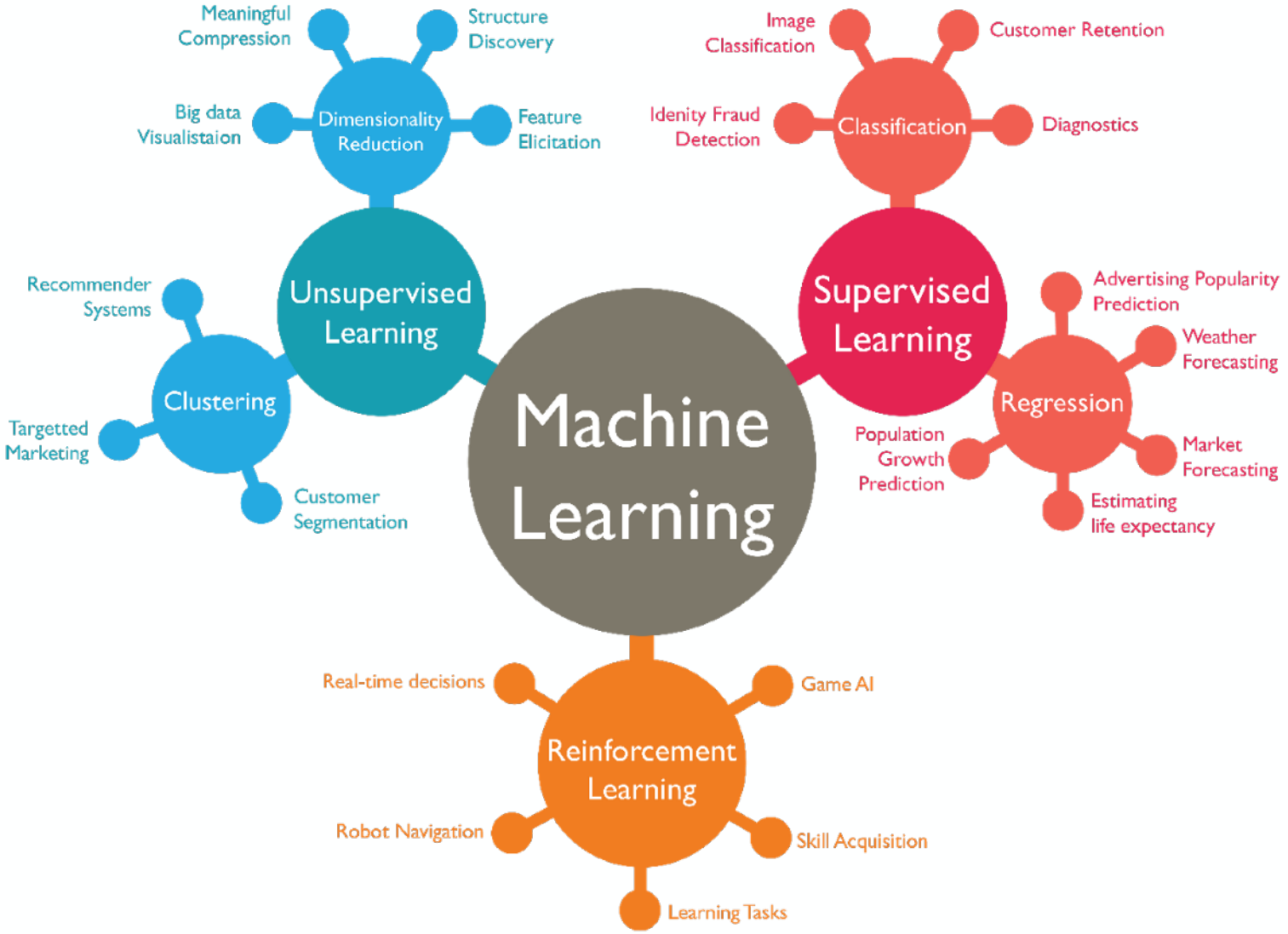
- Rule-based ：依據專家的設定的標準來判斷
- Machine Learning : 資料→特徵擷取→模型→答案
- Deep Learning :資料→模型(特徵擷取自學)→答案



Machine Learning

- Supervised Learning
  - Regression
  - Classification
- Semi-Supervised Learning
- Unsupervised Learning
  - GAN
  - Cluster
- Reinforcement Learning

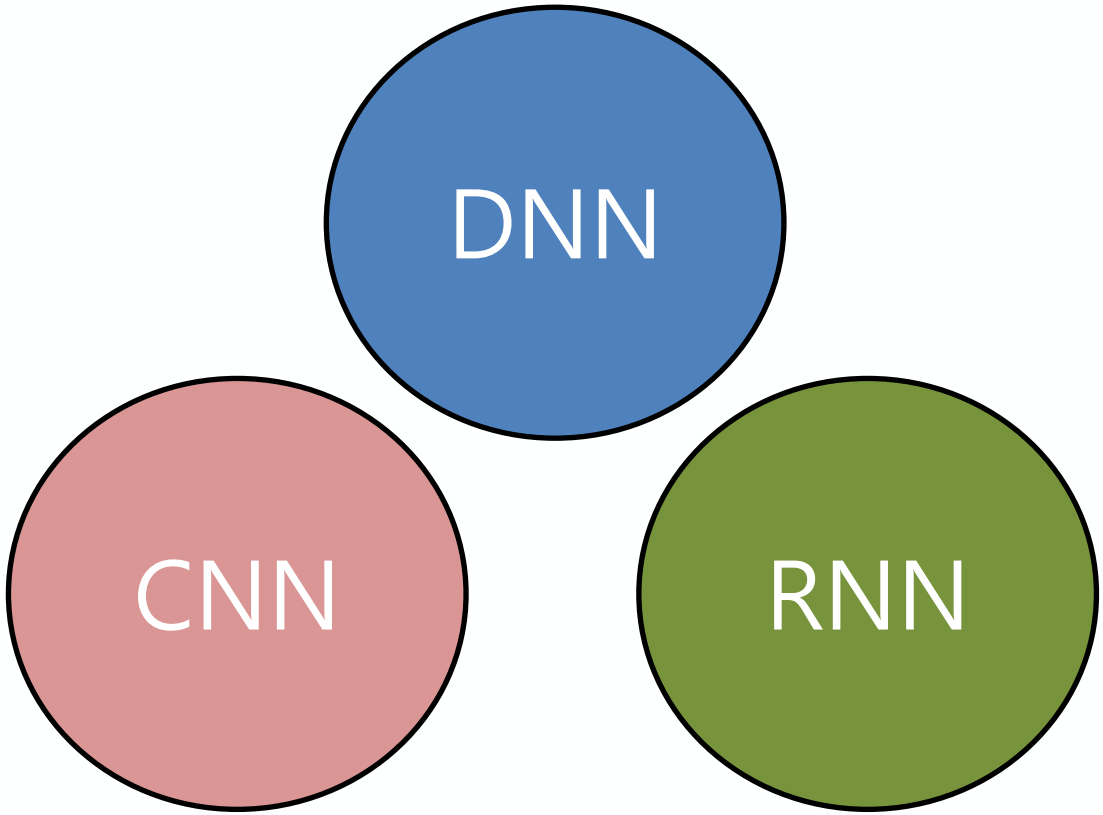
Machine Learning



Machine Learning

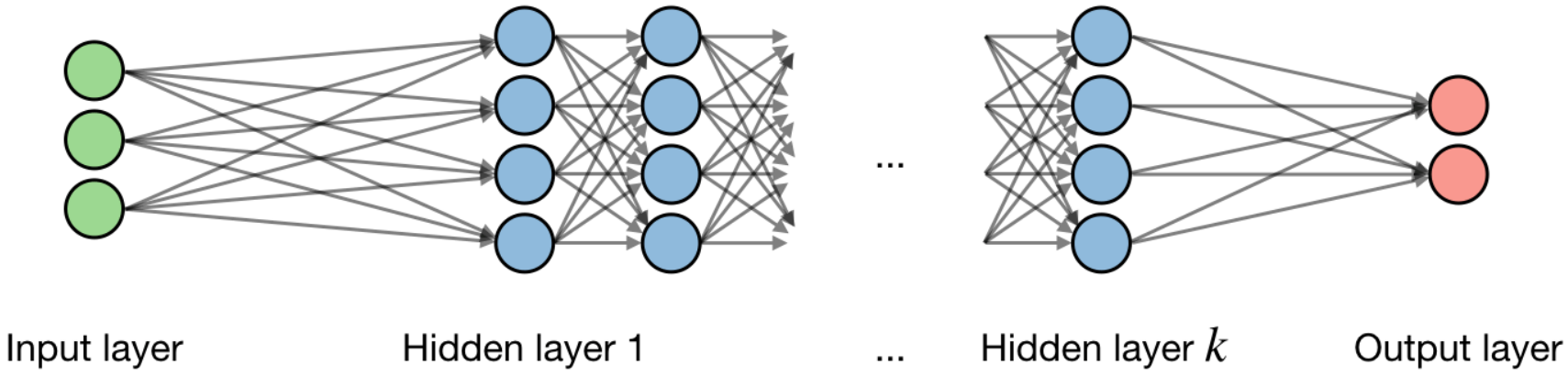
- Machine Learning Algorithms
  - Deep Learning
    - Deep Boltzmann Machine (DBM)
    - Deep Belief Networks (DBN)
    - Convolutional Neural Network (CNN)
    - Stacked Auto-Encoders
  - Ensemble
    - Random Forest
    - Gradient Boosting Machines (GBM)
    - Boosting
    - Bootstrapped Aggregation (Bagging)
    - AdaBoost
    - Stacked Generalization (Blending)
    - Gradient Boosted Regression Trees (GBRT)
  - Neural Networks
    - Radial Basis Function Network (RBFN)
    - Perceptron
    - Back-Propagation
    - Hopfield Network
  - Regularization
    - Ridge Regression
    - Least Absolute Shrinkage and Selection Operator (LASSO)
    - Elastic Net
    - Least Angle Regression (LARS)
  - Rule System
    - Cubist
    - One Rule (OneR)
    - Zero Rule (ZeroR)
    - Repeated Incremental Pruning to Produce Error Reduction (RIPPER)
  - Regression
    - Linear Regression
    - Ordinary Least Squares Regression (OLSR)
    - Stepwise Regression
    - Multivariate Adaptive Regression Splines (MARS)
    - Locally Estimated Scatterplot Smoothing (LOESS)
    - Logistic Regression
  - Bayesian
    - Naive Bayes
    - Averaged One-Dependence Estimators (AODE)
    - Bayesian Belief Network (BBN)
    - Gaussian Naive Bayes
    - Multinomial Naive Bayes
    - Bayesian Network (BN)
  - Decision Tree
    - Classification and Regression Tree (CART)
    - Iterative Dichotomiser 3 (ID3)
    - C4.5
    - C5.0
    - Chi-squared Automatic Interaction Detection (CHAID)
    - Decision Stump
    - Conditional Decision Trees
    - M5
  - Dimensionality Reduction
    - Principal Component Analysis (PCA)
    - Partial Least Squares Regression (PLSR
    - Sammon Mapping
    - Multidimensional Scaling (MDS)
    - Projection Pursuit
    - Principal Component Regression (PCR)
    - Partial Least Squares Discriminant Analysis
    - Mixture Discriminant Analysis (MDA)
    - Quadratic Discriminant Analysis (QDA)
    - Regularized Discriminant Analysis (RDA)
    - Flexible Discriminant Analysis (FDA)
    - Linear Discriminant Analysis (LDA)
  - Instance Based
    - k-Nearest Neighbour (kNN)
    - Learning Vector Quantization (LVQ)
    - Self-Organizing Map (SOM)
    - Locally Weighted Learning (LWL)
  - Clustering
    - k-Means
    - k-Medians
    - Expectation Maximization
    - Hierarchical Clustering

Deep Learning



Tools

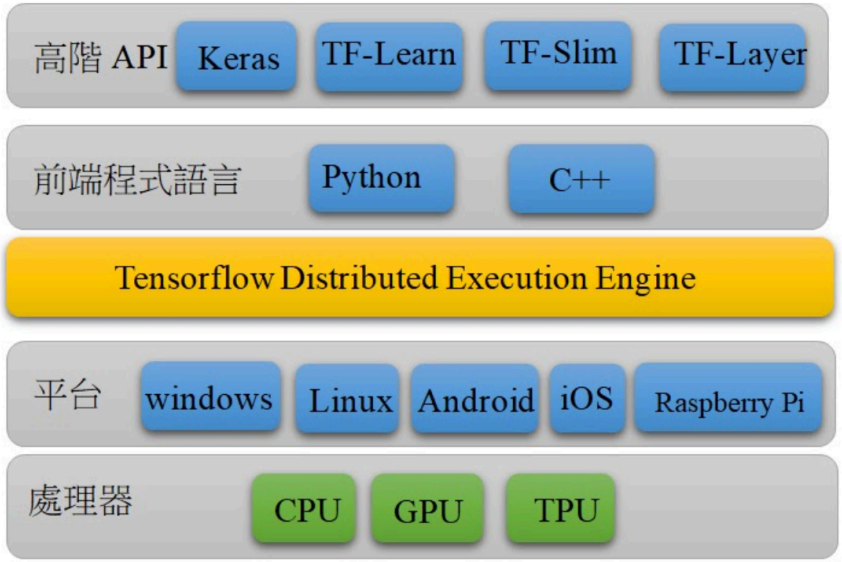
Keras簡介



Fully Connected Multilayer Perceptron

Tensorflow/Keras簡介

- 處理器：**Tensorflow**可以在**CPU**、**TPU**、**GPU**上執行
- 平台，具有跨平台能力
- 前端程式語言：**Tensorflow**可以支援多種前端程式語言，目前有相當多的程式語言可以做運用。
- 高階**API**：**Tensorflow**可以開發許多種高階的**API**，例如：**Keras**、**TF-Learn**、**TF-Slim**等。



Keras簡介

- **Keras是一個開放原始碼，基於Python高階深度學習的程式庫**
- **Keras是以Tensorflow或Theano為Backend來建立Model的框架**
- **Keras 強調極簡主義—你只需幾行程式就能構建一個神經網絡。**
- **本次主要以Tensorflow為Backend為主。**

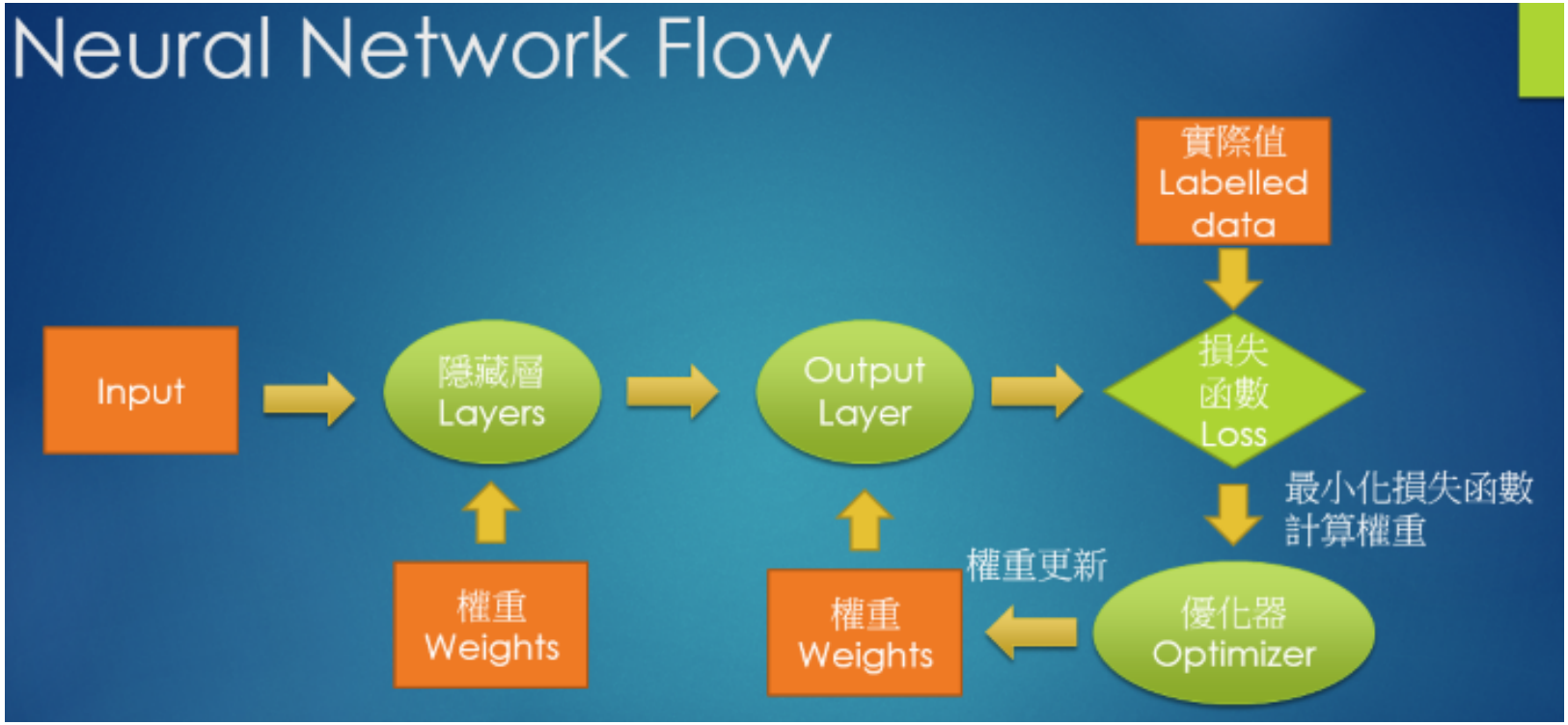
	Keras	Tensorflow
使用語言	Python	Python\C++\...等
學習難度	簡單易上手	相對困難
使用彈性	中	高
適合使用者	初學者	進階使用者

Keras簡介

- Keras可以快速又方便運算的主要原因是:
  - 已經將訓練模型的輸入層、隱藏層、輸出層，做好架構。
  - 使用者只需要加入並且填寫正確的參數。

  ex.神經元個數、activation function的函式...等。
- Keras主要模型分成兩種：
  - Sequential（循序性）：通常都使用這個
  - Functional（多功能型）
- Keras建構神經網路流程如下：
  1. 創建Model
  2. 建立與添加Layer
  3. Compile Model
  4. 訓練 Model
  5. 訓練完成後可用於prediction (預測) 或 evaluation (評估)

Keras簡介


Neural Network Flow
實際值
Labelled
data
Input
隱藏層
Layers
Output
Layer
損失
函數
Loss
最小化損失函數
計算權重
權重
Weights
權重更新
權重
Weights
優化器
Optimizer

Keras簡介

Define a set of Function(Model)

- 決定神經網路的結構
- 層數
- Activation Function

Define Loss Function

- 評估Function/參數的好壞
- 找到一組Function使Loss越小越好

(Gradient Descent)

Pick the best Function

- 找到最好的Function
- Optimizer

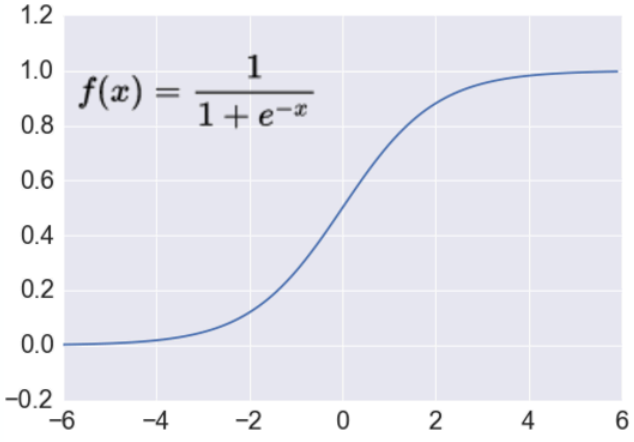
類似Gradient Descent (SGD, Adam等)

Keras簡介

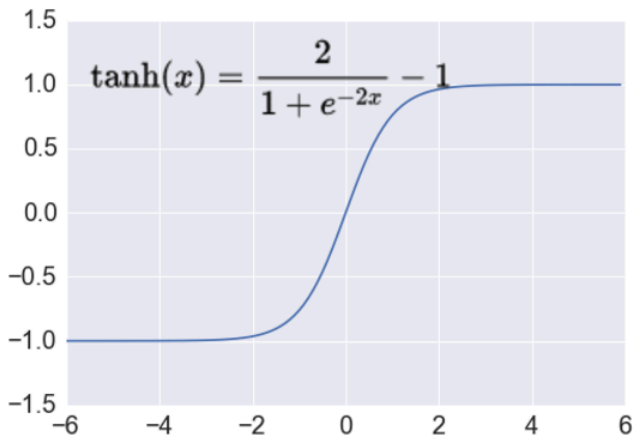
	Classification	Regression
Output Layer	Softmax	Linear/ Non-Linear
Loss Function	Cross-Entropy	MSE/MAE

Activation Function

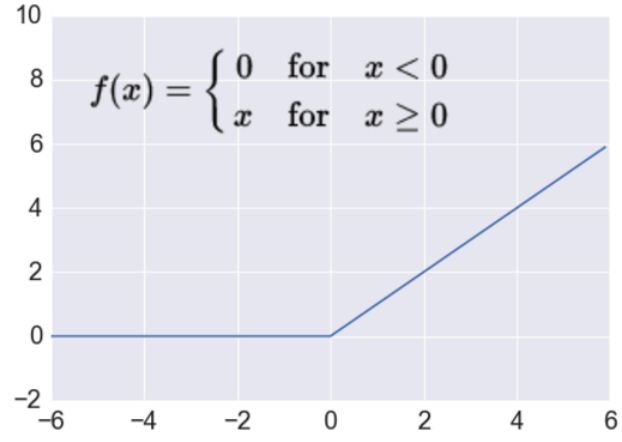
Sigmoid



TanH

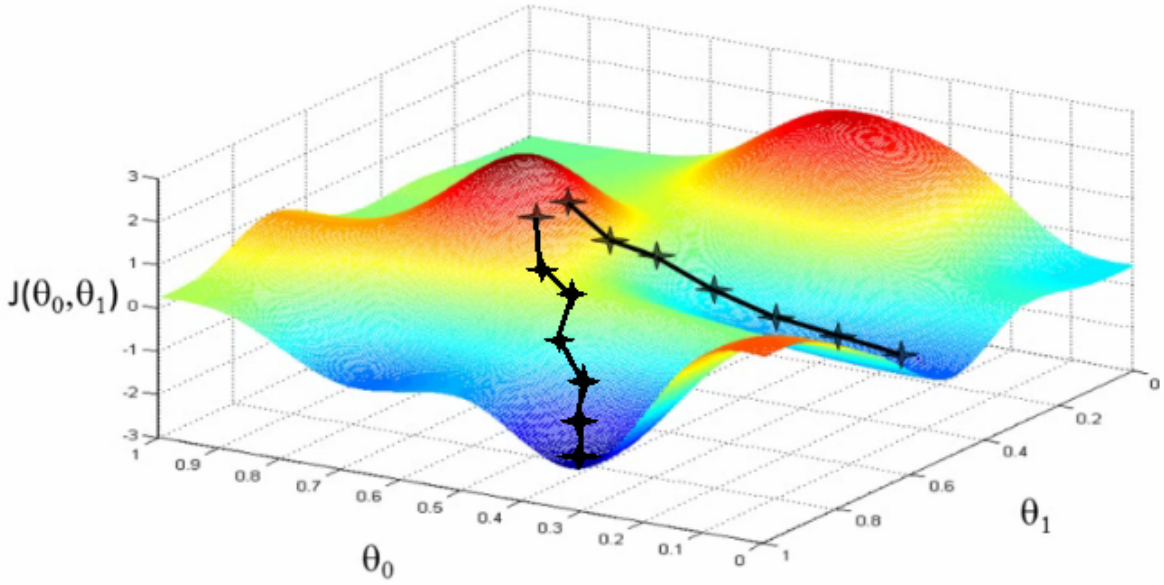


ReLU



去除小於 0 的結果

Loss Function – Gradient Descent



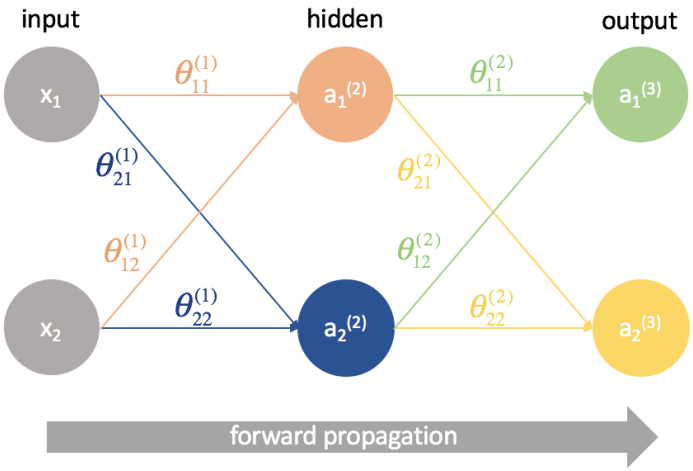
- 計算目前位置的梯度，往梯度最小的方向走，直到梯度為0為止
- 微分出來的梯度可以幫助找到更小的損失函數（Loss function）

Learning rate

- $x_{n+1} = x_n - \gamma \nabla L(x_n)$

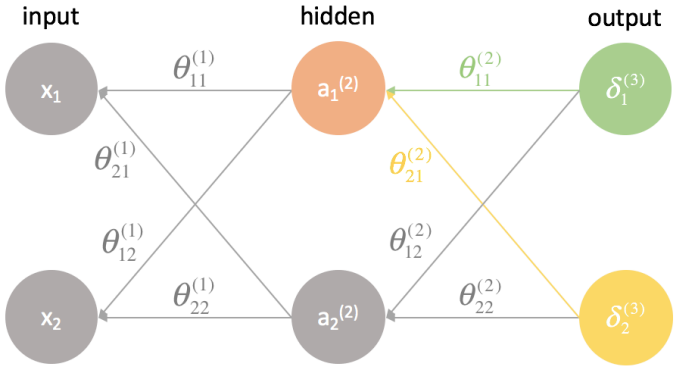
下一步位置 目前位置 Loss Function

Loss Function – Backpropagation



Forward

$$\delta_1^{(2)} = \left(\delta_1^{(3)}\theta_{11}^{(2)} + \delta_2^{(3)}\theta_{21}^{(2)}\right) f'\left(a^{(2)}\right)$$



Backward

Keras語法 – Model架構

```
model.add(Dense(units=40,input_dim=9,
                kernel_initializer='uniform',
                activation='relu'))
```

如下：

- **Dense**: Fully Connected Network
- **Units**: 這一層要設置幾個Neuron
- **Input_dim**: 輸入的維度為多少
- **Kernel_initializer**: Weight要初始化用什麼方法(RandomNormal, RandomUniform...
- **Activation**: 要使用什麼激活函數(softmax, softplus, softsign, relu, tanh, hard_sigmoid, linear...等等)

Keras語法 – Model Compile

```
model.compile(loss = 'binary_crossentropy',
              optimizer='adam',metrics= ['accuracy'])
```

- Loss: 要使用哪種Loss Function
- Optimizer: 要使用哪種優化器
- Metrics=accuracy: 評估方式用accuracy
- Model.summary: 可以查看模型摘要

```
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 40)                400
_________________________________________________________________
dense_2 (Dense)              (None, 30)                1230
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 31
=================================================================
Total params: 1,661
Trainable params: 1,661
Non-trainable params: 0
_________________________________________________________________
```

Keras語法 – Model Training

```
train_history = model.fit(
    x=train_Features,
    y=train_Label,
    validation_split = 0.1,
    epochs = 30,
    batch_size = 30,
    verbose=2)
```

- x為training data 。
- Y為training data對應的label 。
- epochs: 訓練週期次數。
- validation_split = 0.1 : 表示從training data中切出10%當作validation set(驗證集)，可以藉此觀察資料是否Overfitting。
- batch_size : 每次丟多少筆資料訓練
- Verbose : 可以顯示Model每次訓練的Loss及準確率。

Keras語法 – 延伸

- Keras官方文檔: https://keras.io/
- 如有對用法有疑慮的同學，可以至官方文檔查詢

Lab1

Lab1: FizzBuzz

- Write a program that outputs the string representa tion of numbers from 1 to *n*.
- But for multiples of three it should output “Fizz” in stead of the number and for the multiples of five output “Buzz”. For numbers which are multiples of both three and five output “FizzBuzz”.

https://leetcode.com/problems/fizz-buzz/

Lab1: FizzBuzz

規則如下：

- 遇到數字
  - 為3的倍數Print “Fizz” ，替代數字
  - 為5的倍數Print “Buzz” ，替代數字
  - 既是3的倍數又是5的倍數，Print “FizzBuzz”

Example:

n = 15

["1", "2", "Fizz", "4", "Buzz", "Fizz", "7", "8",

"Fizz", "Buzz", "11", "Fizz", "13", "14", "FizzBuzz"]

Lab1: FizzBuzz

https://joelgrus.com/2016/05/23/fizz-buzz-in-tensorflow/

Lab1:

實驗一：

- 將網址中的ipynb檔放到 jupyter notebook上，執行程式碼
- 參考神經網路架構圖，建立神經網路並訓練模型後，將訓練過程記錄至少十秒影片並附上準確率

請敘述在實驗中遇到的問題與觀察到的東西不超過200字

（可包含你如何思考解決這個問題）

Lab2

Lab2: Titanic

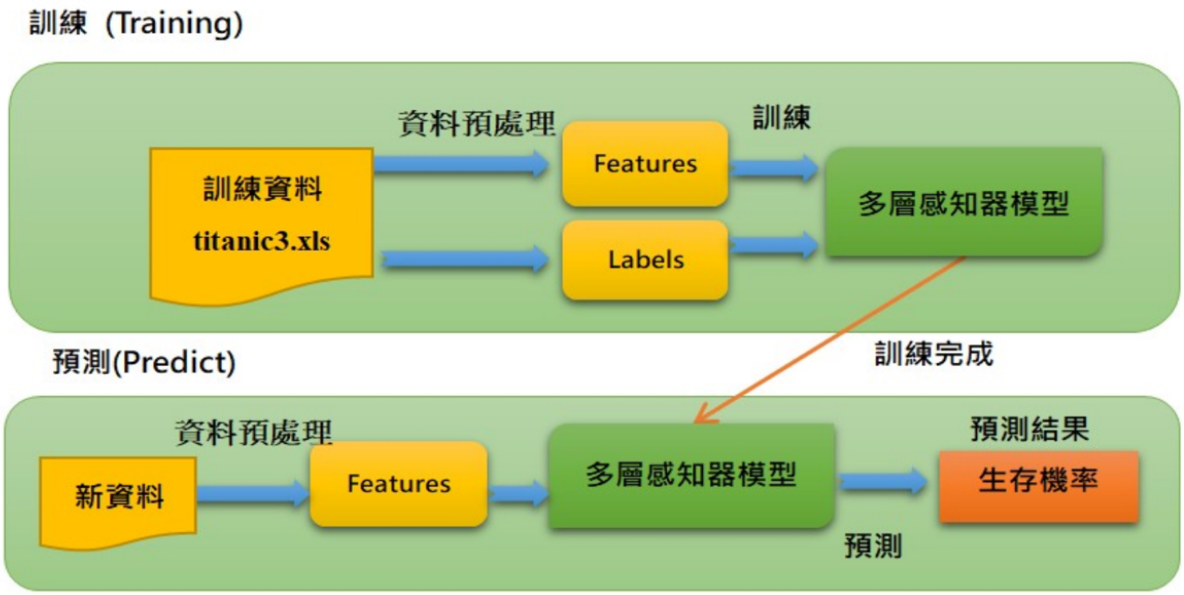
使用Keras建立多層感知器 MLP(Multilayer perceptron)模型，

預測鐵達尼號每一位旅客生存機率。

實驗包含三個步驟：

- 套件安裝與資料前處理
- 模型建立與神經網路設計
- 使用預測模型

大致流程如下：



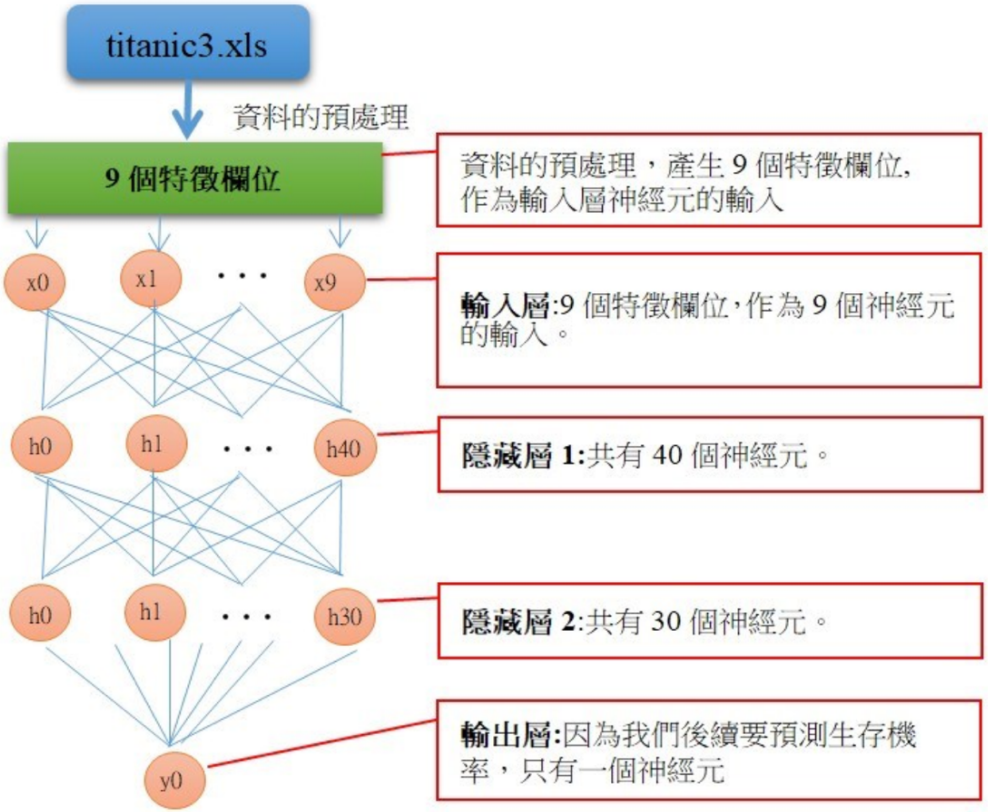
Lab2: Titanic

- 資料欄位包含：

欄位	欄位說明	資料說明
survival	是否生存?	0 = 否, 1 = 是
pclass	艙等	1 = 頭等艙, 2 = 二等艙, 3 =三等艙
name	姓名	
sex	性別	female:女性　male:男性
age	年齡	
sibsp	手足或配偶也在船上數量	
parch	雙親或子女也在船上數量	
ticket	車票號碼	
fare	旅客費用	
cabin	艙位號碼	
embarked	登船港口	C = Cherbourg, Q = Queenstown, S = Southampton

Lab2: Titanic

- **神經網路架構：**



Lab2:

實驗二：

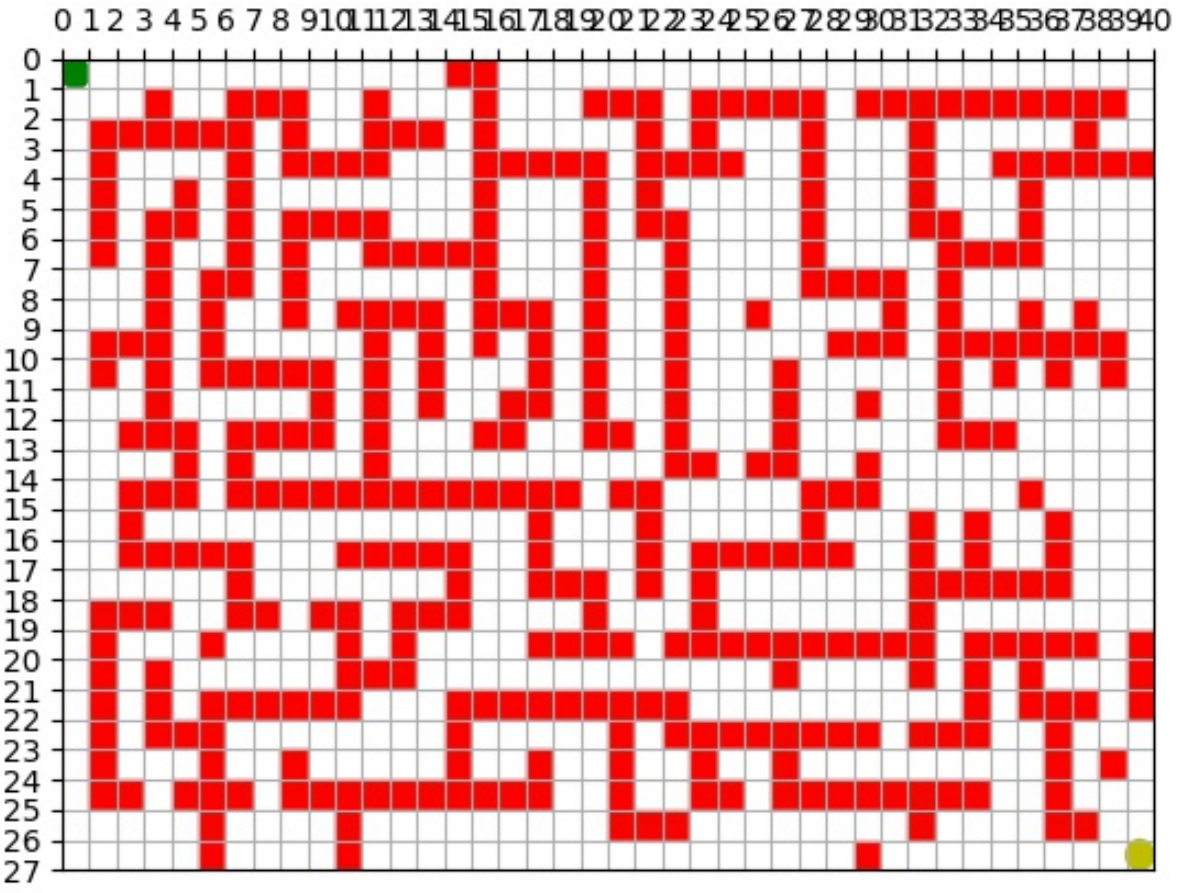
- 將網址中的ipynb檔放到 jupyter notebook上，執行程式碼
- 參考神經網路架構圖，建立神經網路並訓練模型後，將訓練過程記錄至少十秒影片並附上準確率及網路架構圖
- 請觀察改變模型參數準確率是否會有改變（如神經元個數/層數）

- 請敘述在實驗中遇到的問題與觀察到的東西不超過200字（可包含你如何思考解決這個問題）

Lab3

Lab3: 走迷宮

maze.txt



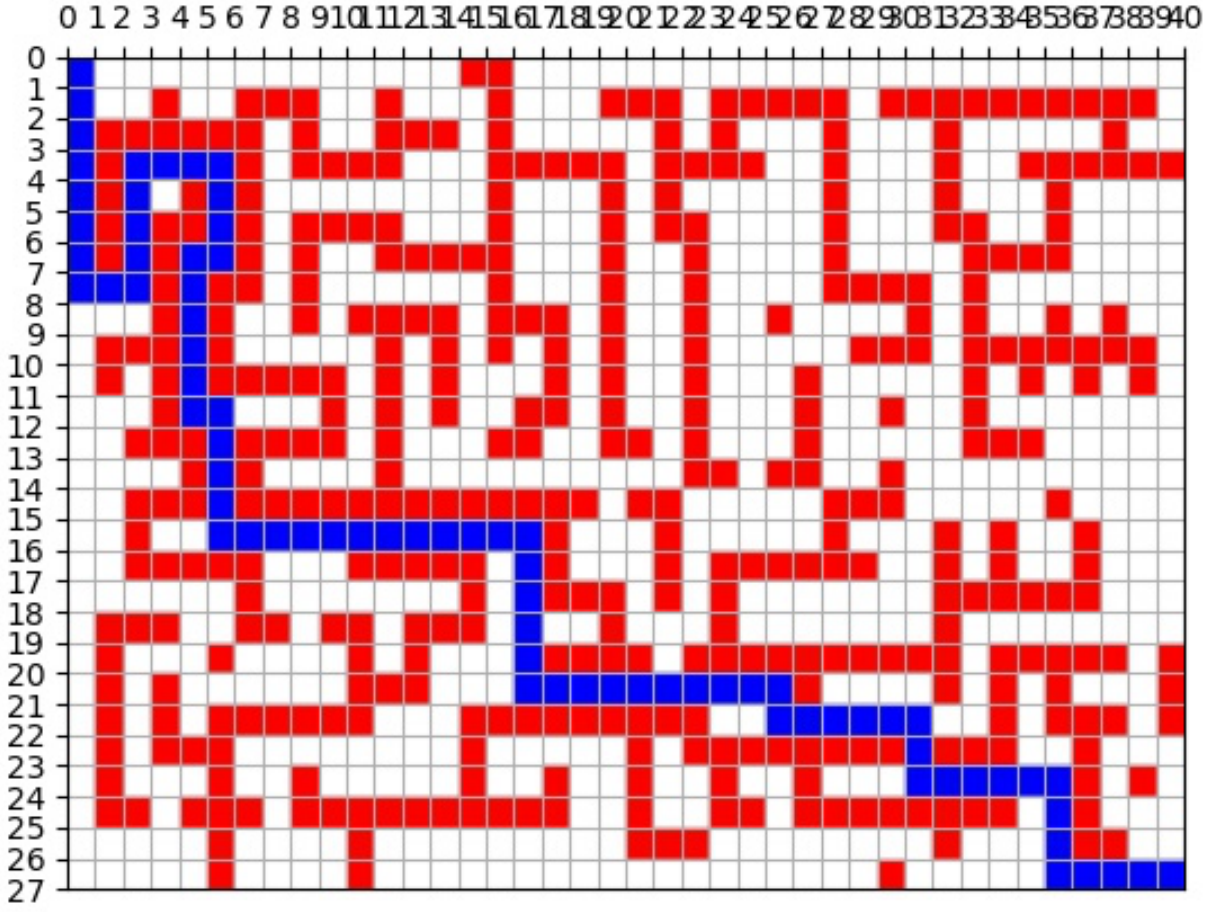
Lab3: 走迷宮

maze.txt

- 0：通道
- 1：障礙物
- 2：起點
- 3：終點

```
20000000000001100000000000000000000000
00010011100100010001110111110111111111110
01111110100111010000010100010001000001000
01000010111100011111011110010001001111111
01001010000000010001010000010001000100000
01011010111100010001011000010001100100000
01010010100111110001001000010000111100000
00010110100000010001001000011110100000000
00010100101111011101001001000010100101000
01110100000101010101001000001110111111110
01010111110101000101001000100000101010100
00010000010101001101001000100100100000000
00111011110100011001101000100000111000000
00001010000100000000011011001000000000000
00111011111111111111011000001110000010000
00100000000000001000100000100010100100000
01111100011111001000101111110010100100000
00000010000000100111010100000011111100000
01110011011011100001000100000001000000000
01000100001010000111101111111111011111010
01010000001110000000000001000010101000100
01010111111000111111111000000000010111010
01011100000000100000101111111101110010000
01000100100000100100100100100000000010100
01101110111111111110010011011111111001000
00000100001000000000111000000001000011000
00000100001000000000000000000010000000003
```

Lab3: 走迷宮



Lab3:

實驗三：

- 將網址中的astar.py檔利用cmd終端機執行程式碼
- 敘述整個實驗運作的流程與架構
- 並截圖最後的路徑圖，如檔案A_star_maze.jpg

請敘述在實驗中遇到的問題與觀察到的東西不超過200字

（可包含你如何思考解決這個問題）