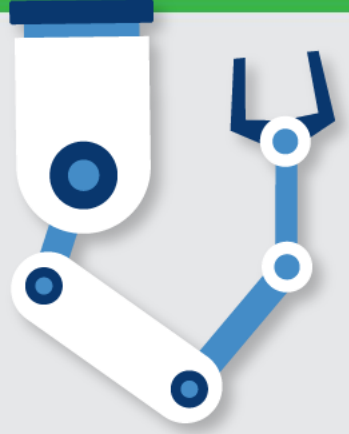
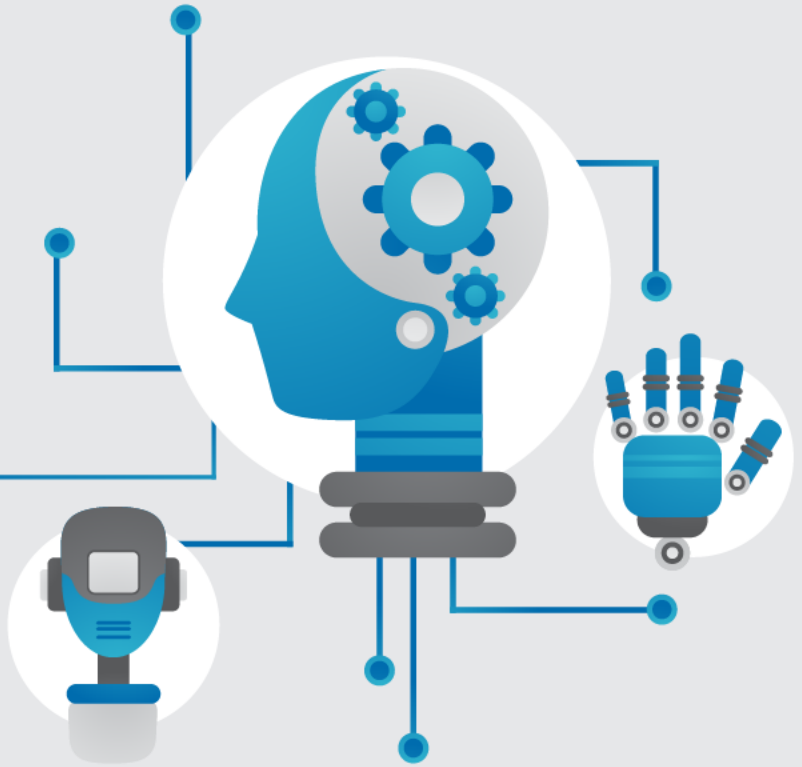


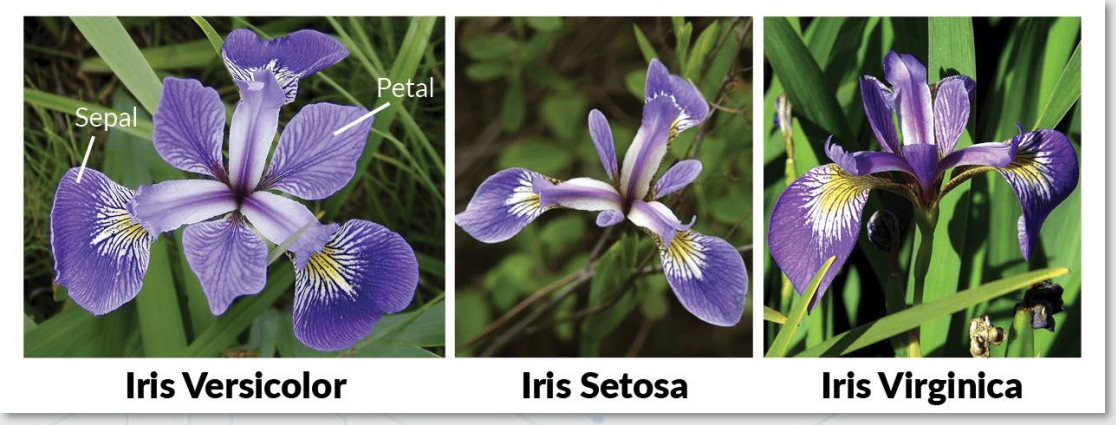
# 鳶尾花種類預測實作



# Iris資料集介紹



- Iris資料集是統計分析領域中常用的範例，數據由Edgar Anderson收集
  - 內容包含山鳶尾 (Iris Setosa)、變色鳶尾 (Iris Versicolor)和維吉尼亞鳶尾 (Iris Virginica) 3種
  - 特徵有花萼 (sepal)長、寬及花瓣 (petal)長、寬，4項特徵
  - 每種各50筆資料



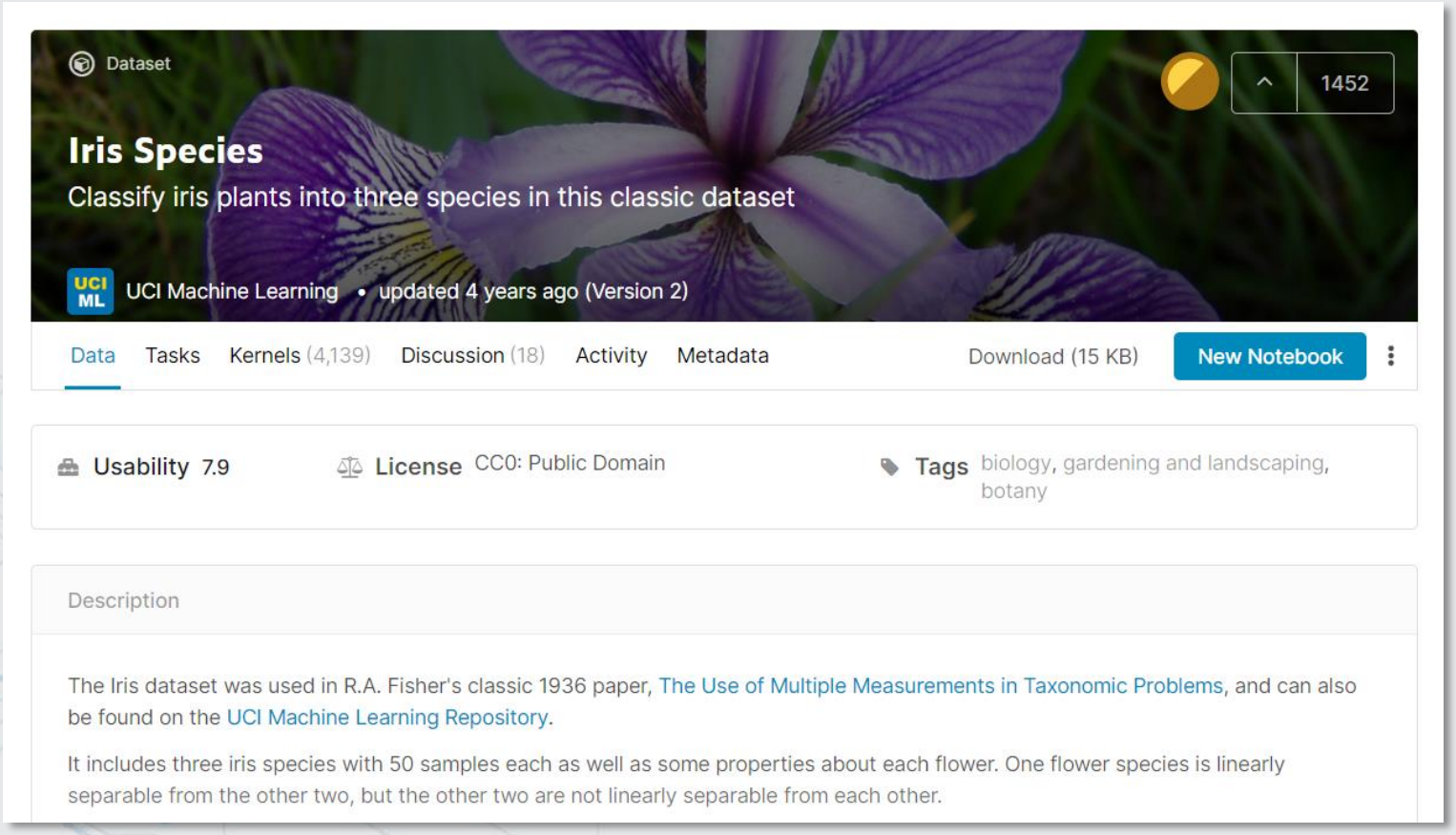



en.wikipedia.org

# Kaggle資料集 - Iris Species



https://www.kaggle.com/uciml/iris



# Kaggle資料集 - Iris Species



## 資料檔案：Iris.csv

- 共150筆資料（列，row）
- 每列有6 個欄位資料
- 第一個欄位為ID
- 最後一個欄位為label，說明該筆資料屬於哪個種類
- 第2~5欄位為該資料的屬性，花萼的寬高、花瓣的寬高

| | Id | SepalLengthCm | SepalWidthCm | PetalLengthCm | PetalWidthCm | Species |
|---|---|---|---|---|---|---|
| 2 | 1 | 5.1 | 3.5 | 1.4 | 0.2 | Iris-setosa |
| 3 | 2 | 4.9 | 3 | 1.4 | 0.2 | Iris-setosa |
| 4 | 3 | 4.7 | 3.2 | 1.3 | 0.2 | Iris-setosa |
| 5 | 4 | 4.6 | 3.1 | 1.5 | 0.2 | Iris-setosa |
| 6 | 5 | 5 | 3.6 | 1.4 | 0.2 | Iris-setosa |
| 7 | 6 | 5.4 | 3.9 | 1.7 | 0.4 | Iris-setosa |
| 8 | 7 | 4.6 | 3.4 | 1.4 | 0.3 | Iris-setosa |
| 9 | 8 | 5 | 3.4 | 1.5 | 0.2 | Iris-setosa |
| 10 | 9 | 4.4 | 2.9 | 1.4 | 0.2 | Iris-setosa |

# MLP模型建置流程



1. 資料前處理
2. 決定模型架構
3. 編譯與訓練模型
4. 模型評估

# 資料前處理



› 從檔案Iris.csv讀進資料，並去掉不需要的欄位

```
1 import pandas as pd
2 data=pd.read_csv('Iris.csv')
3
4 data=data.drop('Id', axis=1)
```

# 資料前處理



## 資料正規化

- 根據每個欄位中的最大值，將欄位中的所有值轉換至0～1。

```
6 num_data=['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']
7 for col in num_data:
8     # normalization
9     data[col]=data[col]/data[col].max()
```

# 資料前處理



› 轉成np.array

```
11 x=data.drop('Species', axis=1).values
12 y=pd.get_dummies(data.pop('Species')).values
```

# 資料前處理



## 切出訓練和測試資料

```
14 from sklearn.model_selection import train_test_split
15 x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2)
16
17 print(x_train.shape)
```

```
In [3]: print(x_train.shape)
(120, 4)
```

# 決定模型架構



## 模型建置

```
19 from keras.models import Sequential
20 from keras.layers import Dense
21
22 model=Sequential()
23 model.add(Dense(1024, activation='relu', input_shape=(4,)))
24 model.add(Dense(1024, activation='relu'))
25 model.add(Dense(1024, activation='relu'))
26 model.add(Dense(3, activation='softmax'))
```

# 編譯與訓練模型



› 設定參數，訓練模型

```
28 model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['acc'])
29 history=model.fit(
30         x_train, y_train,
31         validation_data=(x_test, y_test), # 資料量不夠，直接test
32         batch_size=32,
33         epochs=30,
34         shuffle=True)
```

# 模型評估



## 顯示訓練歷程

```
36 import matplotlib.pyplot as plt
37 def show_train_history(train_history):
38     plt.figure(figsize=(10,5))
39     plt.plot(train_history.history['acc'])
40     plt.plot(train_history.history['val_acc'])
41     plt.xticks([i for i in range(len(train_history.history['acc']))])
42     plt.title('Train History')
43     plt.ylabel('acc')
44     plt.xlabel('epoch')
45     plt.legend(['train', 'validation'], loc='upper left')
46     plt.show()
47 show_train_history(history)
```

# 模型評估



- 顯示訓練歷程


Train History
train
validation
acc
epoch
1.0
0.9
0.8
0.7
0.6
0.5
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29

# 模型評估



## › 顯示Confusion matrix

```
from sklearn.metrics import confusion_matrix
def show_confusion_matrix(cnf_matrix, classes_num):
    cnf_matrix=cnf_matrix.astype('float')/cnf_matrix.sum(axis=1)
    plt.figure(figsize=(5,5))
    plt.imshow(cnf_matrix, cmap='Blues')
    plt.colorbar()
    plt.xticks([i for i in range(classes_num)])
    plt.yticks([i for i in range(classes_num)])
    thresh = cnf_matrix.max() / 2.
    for i in range(classes_num):
        for j in range(classes_num):
            plt.text(
                    j, i,
                    format(cnf_matrix[i, j]*100, '.1f')+'%',
                    horizontalalignment="center",
                    color="white" if cnf_matrix[i, j] > thresh else "black")
    plt.ylabel('True label')
    plt.xlabel('Predicted label')
    plt.tight_layout()
    plt.show()
import numpy as np
y_true=np.argmax(y_test, axis=1)
y_pred=np.argmax(model.predict(x_test), axis=1)
cnf_matrix = confusion_matrix(y_true, y_pred)
show_confusion_matrix(cnf_matrix, 3)
```

# 模型評估



- 顯示各類別預測狀況