CH 5

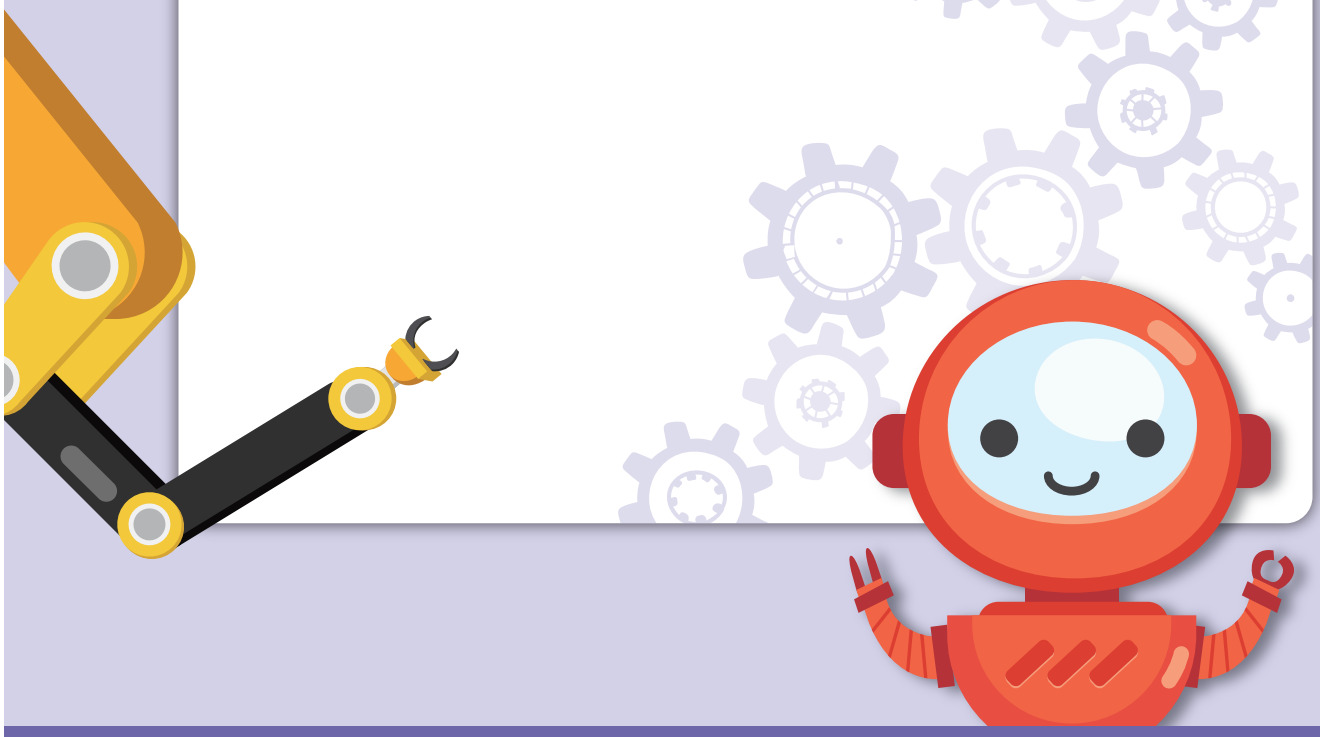
國際 AI 競賽走一遭

5-1 Kaggle 網站介紹

5-2 經典賽事介紹

5-3 Kaggle 資料上傳與排名

5-4 總結



5-1 Kaggle 網站介紹

前面的章節，讀者想必已掌握了機器學習的基本觀念，並且對實作具備了一些信心，接下來我們就來帶領你迎接更進階的挑戰！

目前，全球頂尖資料科學家與 AI 專家學者，有一個共同的競技場，那就是 Kaggle（https://www.kaggle.com/）。這個網站成立於 2010 年，最早是一個進行資料挖掘和預測競賽的網路平台，提供私人企業和研究者在上面發布數據資料，由資料科學家在上面取得資料，創建最佳的預測模型並進行競賽，形成一種眾包模式[註1]的問題解決平台。

Kaggle 為了號召全球頂尖高手參與競賽，會提供大大小小的獎金（最高可達數百萬美金）。目前它已經是世界上最大的資料科學家社群，擁有 300 萬個專家會員註冊，集結超過 19,000 個真實的公共資料集（包含 2020 席捲全球的武漢肺炎 COVID-19 相關資料），而且不僅有競賽，它所提供的 Notebooks、Discuss、Courses 等機制，還能讓所有人在上面學習、協作與交流。

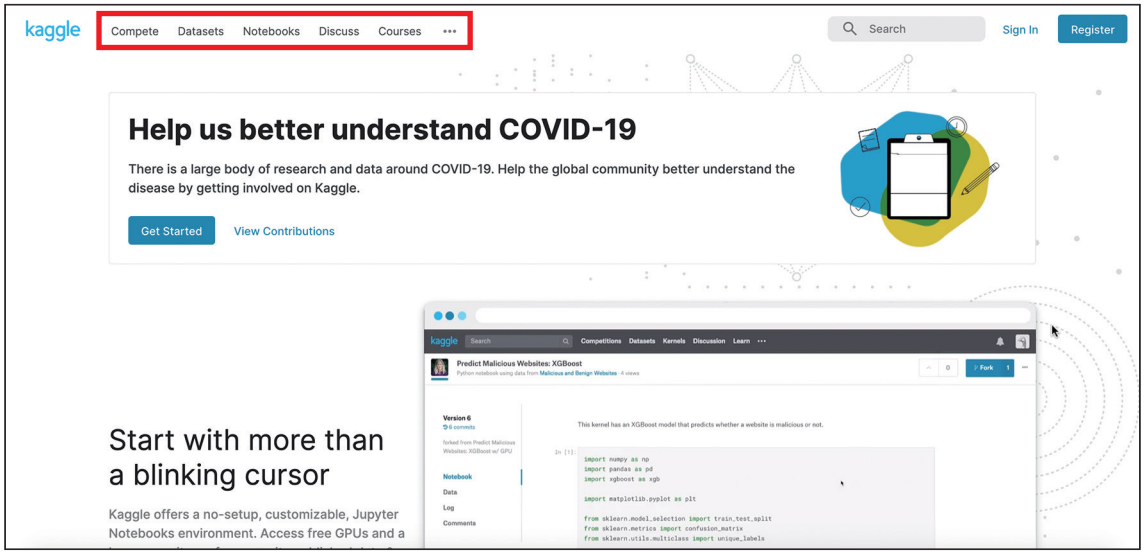
本書最後一章，就讓我們來實際遊歷一次 Kaggle 的競賽流程，並且手把手教讀者取得一個國際競賽的成績！

知識補充

Kaggler 等級

在 Kaggle 上的 300 萬名會員通稱為 Kaggler，依據每個會員的貢獻度與技術實力共分五個等級，從基礎到高階分別為：Novice、Contributor、Expert、Master、Grandmaster，每位新手加入時都是 Novice，Kaggle 會給每個新手一份清單去完成，以便升級成為 Contributor，然後逐步累積實作、分享、比賽、獎章等等，等級才能一直往上。

註 1　眾包，也稱「群眾外包（Crowdsourcing）」，指個人或組織利用大量的網路用戶取得需要的服務和想法，將工作先分配給很多參與者，再合併成最終結果，優點是可以提供超出組織的思考範圍的結果、帶來更寬廣且創新的想法。



◆ 圖 5-1 Kaggle 網站首頁

在 Kggle 網站首頁上的功能選單，你可以看到這些選項：

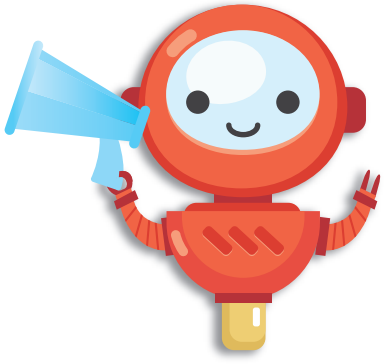
1.Compete：搜尋或參加大大小小的競賽。

2.Datasets：取得全球各種類別且高品質的數據資料集。

3.Notebooks：直接編寫程式並且直接使用資料來建造模型。

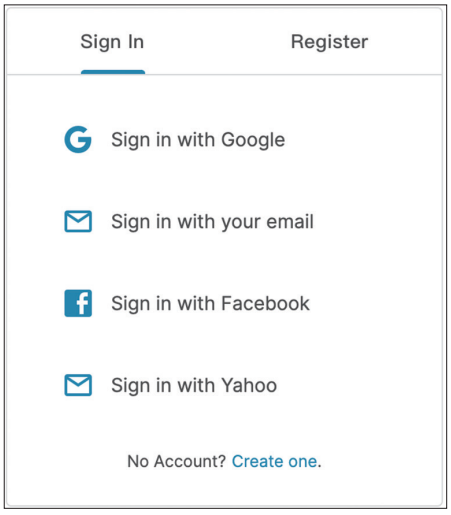
4.Discuss：在討論區詢問或查詢任何問題。

5.Courses：免費取得各種機器學習的線上課程。

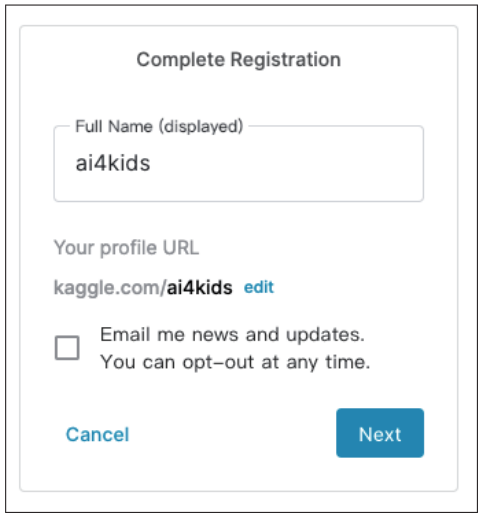


5-1-1 登入 Kaggle

Kaggle 的註冊登入，在首頁的右上方，流程非常簡單，有 Google 或 Facebook 帳戶，就可以在一分鐘內完成登入程序。

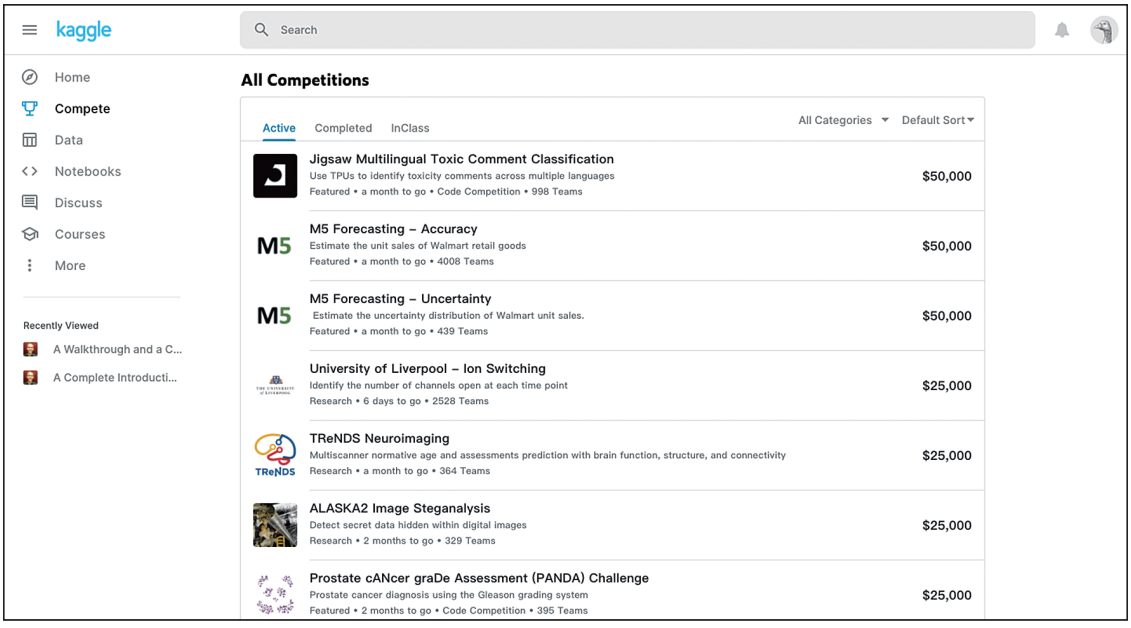


◆ 圖 5-2 Kaggle 提供單一登入機制（single-sign-on）



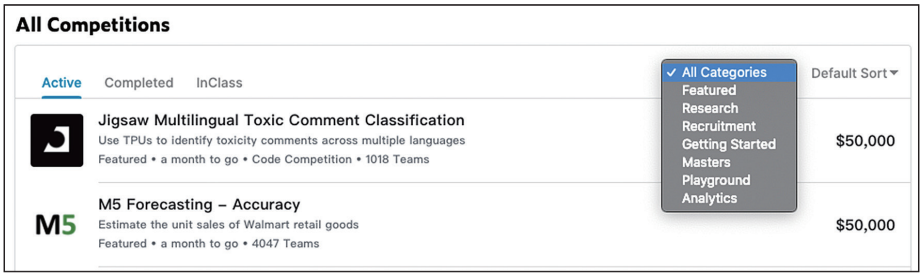
◆ 圖 5-3 輸入 Full Name 送出即可完成登入

競賽區 Compete



◆ 圖 5-4 Kaggle 競賽區

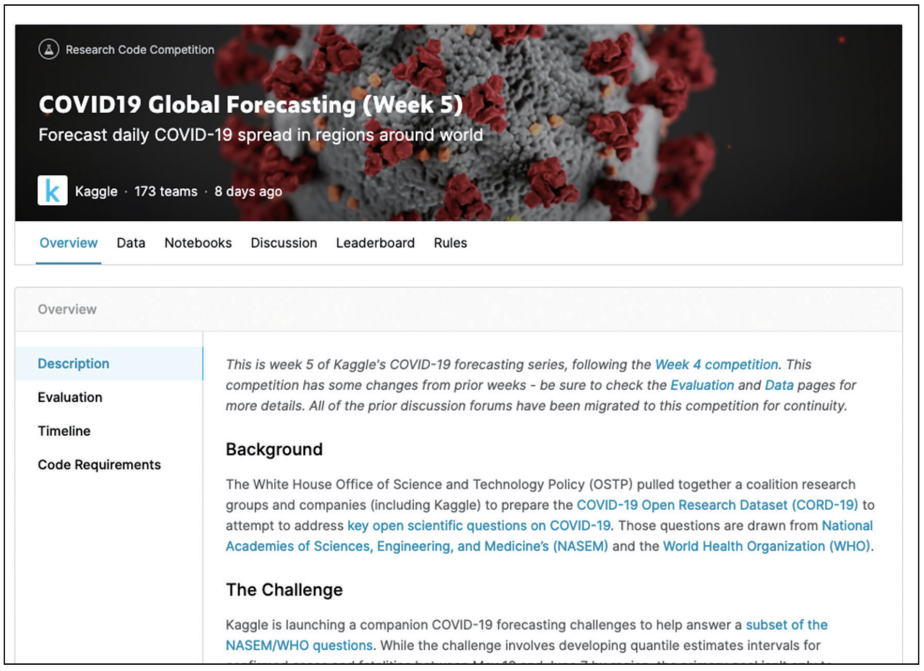
這裡集結了全世界的資料科學競賽，分成「Active」、「Completed」、「InClass」三個頁籤，分別為進行中、已結束、封閉式三種類型競賽。從列表中可以瀏覽各種競賽的名稱與簡介，左上角的篩選功能，還能透過「Category」選擇適合的競賽屬性等級，以及競賽的排序方式。



◆ 圖 5-5　Kaggle 競賽區的 Category 選單

對於新手參賽者來說，我們通常在 Category 會選擇「Getting Started（入門）」或「Playground（遊樂場）」兩種競賽，它們都是相對比較簡單、有趣的資料集，適合用來練習。

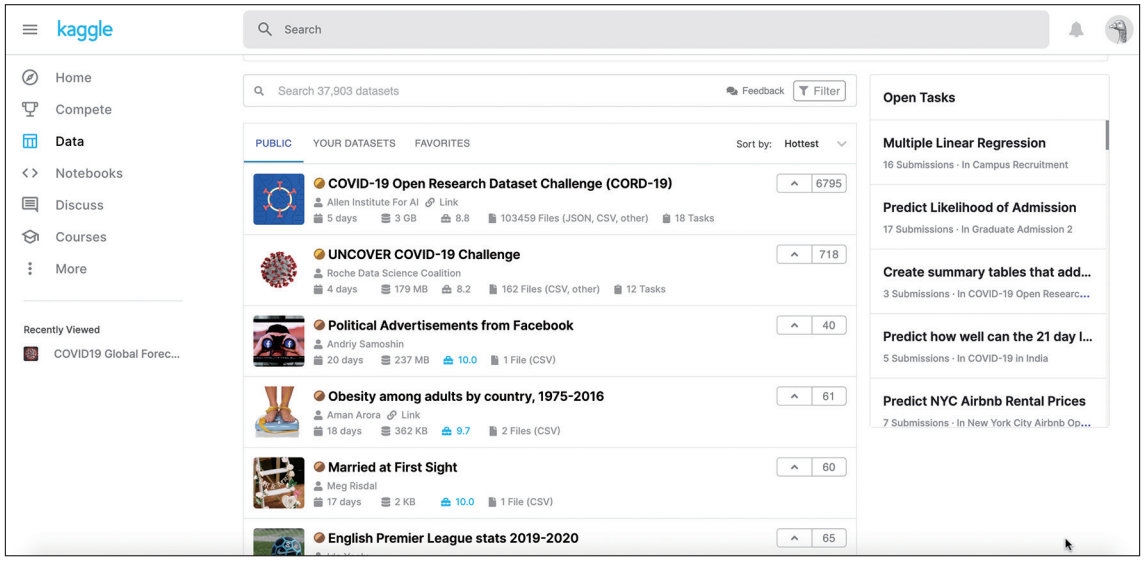
點擊任何一項競賽，即可進入單一競賽首頁，裡面擁有關於此競賽的所有詳細資訊：包含競賽的描述、評估標準、時程表、程式碼的要求等等。



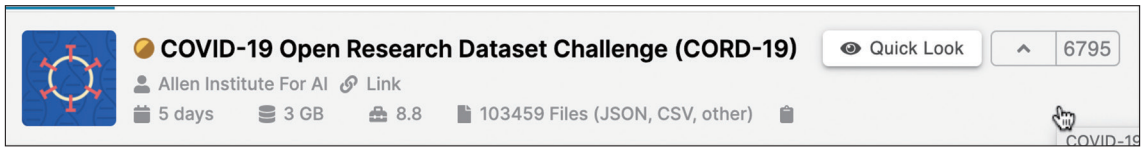
◆ 圖 5-6　武漢肺炎 COVID19 預測競賽 預測全球各地 COVID-19 每天傳播的數據

資料集 Dataset

Kaggle 最為人所稱道的便是匯聚全世界上萬筆高品質且公開的資料集，資料類型也非常豐富，包含商業、網路社群、電腦科學、醫療、教育、時尚、娛樂、藝術等多元性，而且不斷地更新中，還提供友善的工具讓會員篩選資料集大小、檔案型態等等，以便於實作應用。讀者若想有朝一日成為資料科學家，Kaggle 絕對是一個讓你精進實力的寶地。



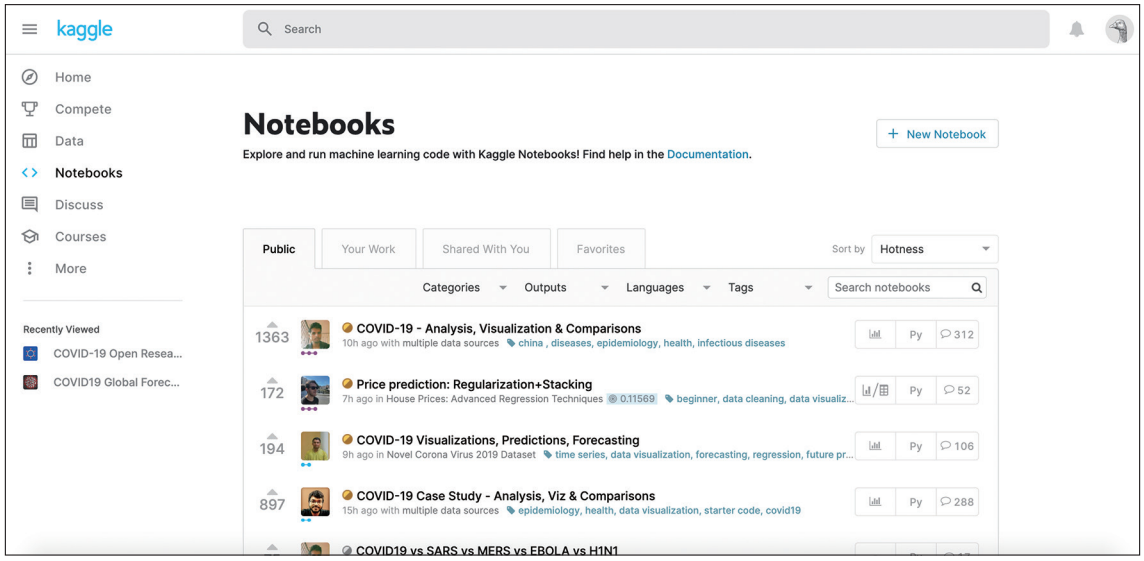
◆ 圖 5-7　Kaggle 資料集區



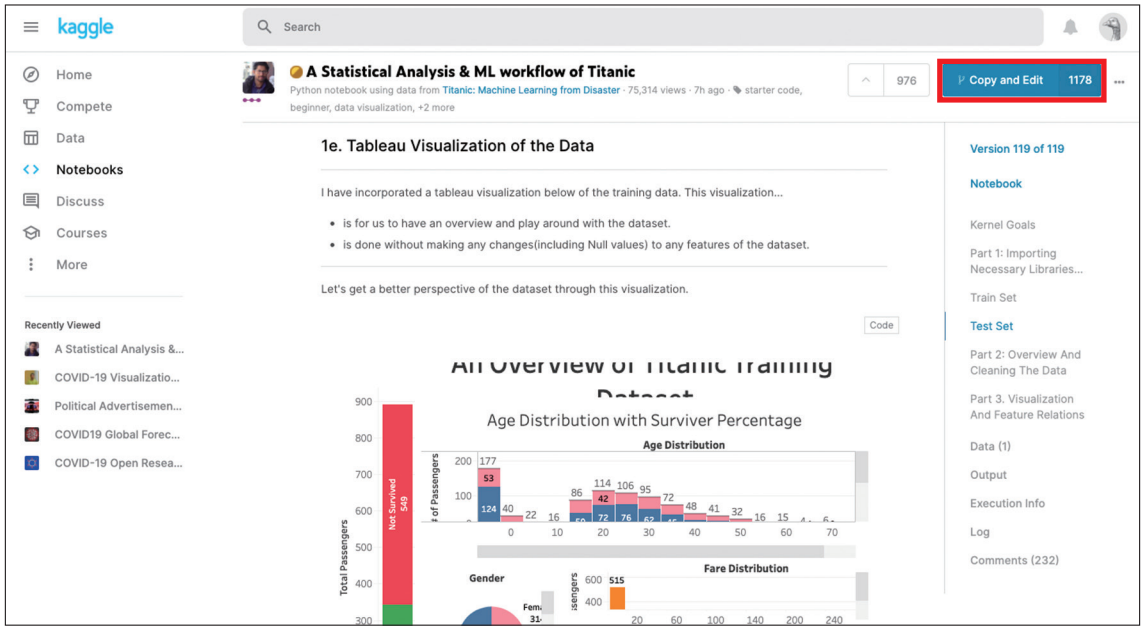
◆ 圖 5-8　Kaggle 每項資料集均有清楚的標示

筆記本 Notebooks

Kaggle 是全世界最大的資料科學家社群，原因是因為他們鼓勵所有會員分享他們的成果，Kaggle Notebook 就是眾多高手們無私貢獻文章與程式碼的園地，它同時也是瀏覽器版的 Jupyter Notebooks，能直接執行資料科學和機器學習的程式，而且運用的是雲端運算資源，不會消耗本機電腦的計算量。而 Kaggle 上的資料集也都預先就載入在 Notebooks 雲端，會員不用重新下載。



◆ 圖 5-9　Kaggle 筆記區



◆ 圖 5-10　在別人分享的 Notebooks 點「Copy and Edit」就可以開始實作

5-2 經典賽事介紹

5-2-1 新手賽事介紹

在 Kaggle 社群，常有新手詢問「有專門推薦給新手入門的賽事嗎？」答案是有的！

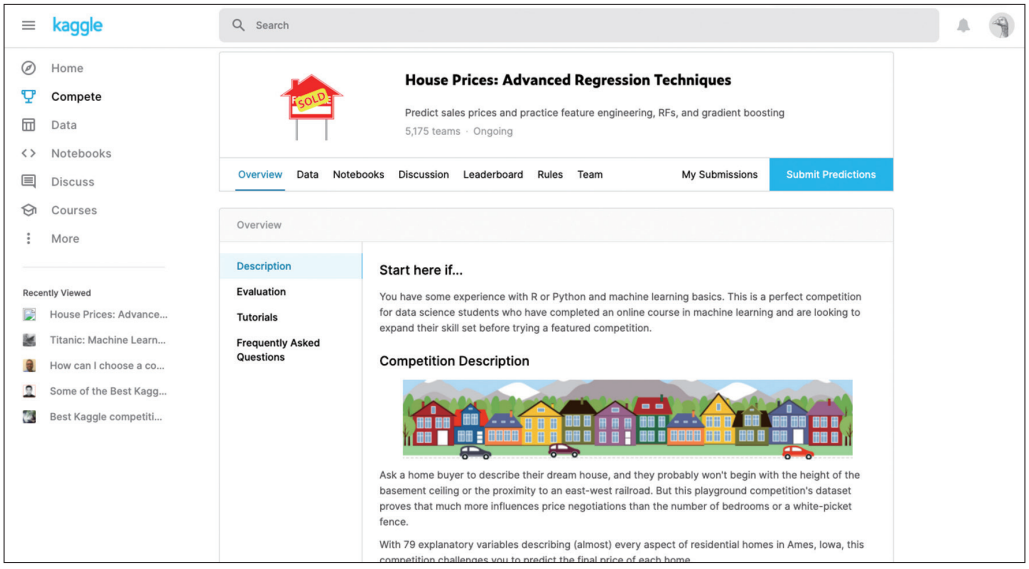
一般來說適合新手操練實力的競賽，推薦以下三項：

迴歸問題：房價預測

Regression Problem：**https://www.kaggle.com/c/house-prices-advanced-regression-techniques**

具備一點機器學習迴歸分析能力的新手，房價預測資料集會是最好的入門選擇，它針對每筆房屋提供了洋洋灑灑 79 種欄位資訊，幾乎涵蓋了住宅的方方面面，例如：建物大小、屋齡、建材、建物風格、樓層、臨路寬度、車庫 .... 甚至是有沒有游泳池等等，本競賽的最終目標，是透過這些房屋資訊去預測每個房屋的最終價格。

由於是初學者的比賽，所以房價預測沒有競賽截止日，成績也不列入 Kaggle 的積分，不過此資料集卻有非常多的 tutorial（教學）可以參考學習。

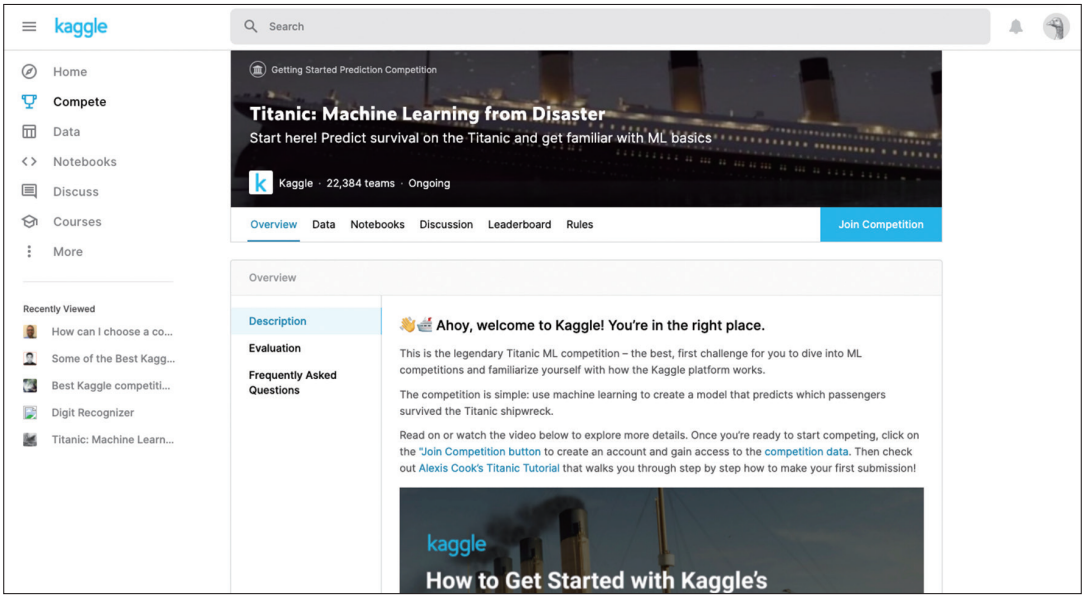


◆ 圖 5-11　房價預測資料集

分類問題：鐵達尼號生存預測

Classification Problem：**https://www.kaggle.com/c/titanic**

鐵達尼號的沉沒是歷史上最著名的船難之一。1912 年 4 月 15 日，她在處女航與冰山相撞後沉沒，不幸的是，船上沒有足夠的救生艇供所有人使用，導致 2,224 名乘客和機組人員中的 1,502 人死亡。這樁慘烈的船難，只有少數人倖存下來，他們獲救或許有一些運氣，但從數據中似乎可以挖掘出，有些人比其他人更有生還的可能性。在這一個預測競賽中，新手要建立一個預測模型來回答分類問題：乘客誰會生？誰會歿？可以用來訓練預測模型的乘客數據有：姓名、年齡、性別、社會經濟階層、船艙別等等。

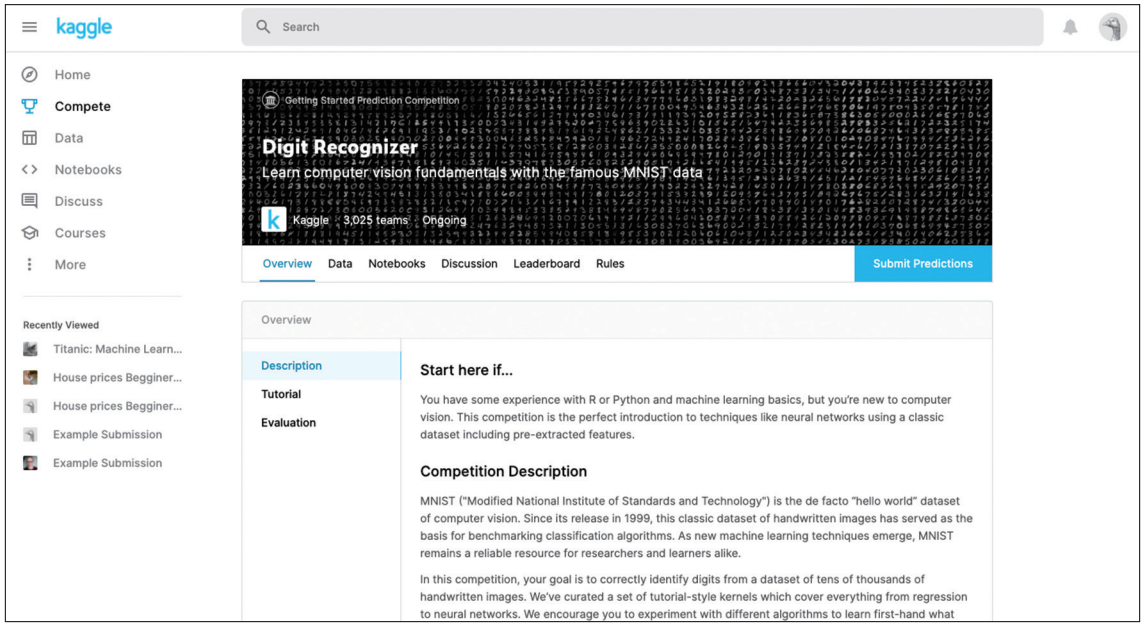


◆ 圖 5-12　鐵達尼資料集

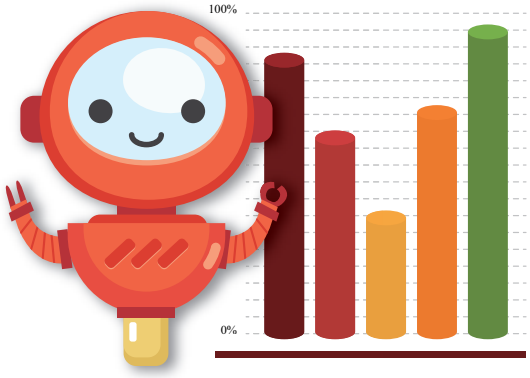
電腦視覺：手寫數字辨識

Computer Vision：**https://www.kaggle.com/c/digit-recognizer**

手寫數字資料集 MNIST 算得上是電腦視覺領域的「hello world」數據集（也就是入門必練的經典範例）。自從 1999 年發布以來，這個經典的手寫圖像就成為各種演算法試煉的基本資料集，目標是從數萬個手寫圖像中正確識別數字。除了第四章用 K-means 實作分群外，也常有 Kaggler 運用其他機器學習方法或神經網路演算法來預測。



◆ 圖 5-13　手寫數字資料集



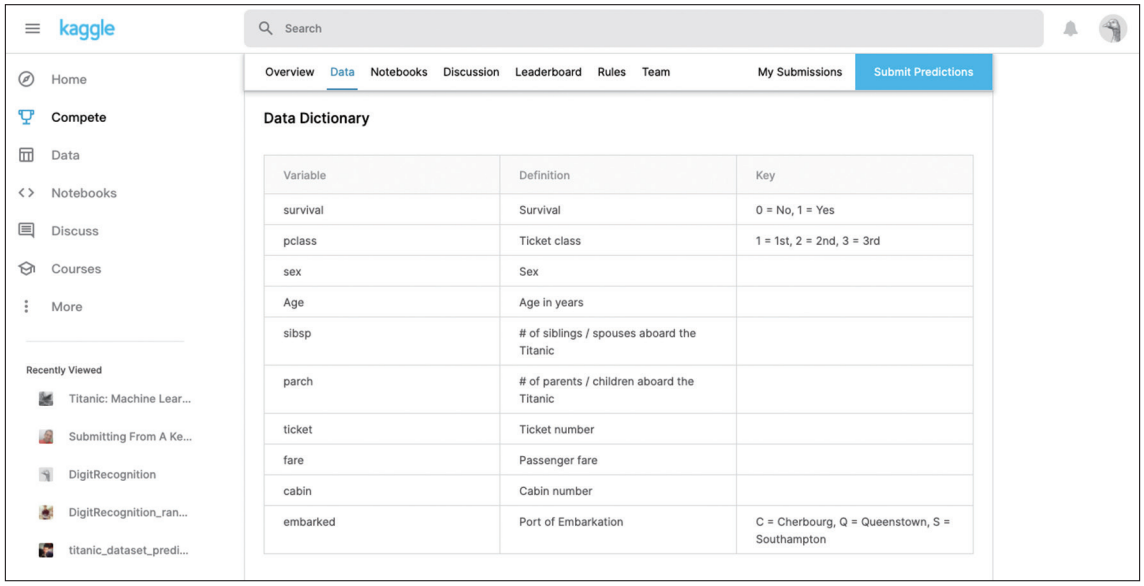
5-3 Kaggle 資料上傳與排名

現在就讓我們正式進入 Kaggle 的競賽步驟，以新手身分入門競賽「鐵達尼號生存預測」。

5-3-1 觀察資料集

首先登入 Kaggle 網站，搜尋「titanic」來到 Titanic: Machine Learning from Disaster

競賽首頁（或者也可以直接輸入網址：https://www.kaggle.com/c/titanic），接著點擊「Data」頁籤，來到下載資料集的頁面（如圖 5-14）。



◆ 圖 5-14　鐵達尼號資料集的概況

我們可以透過這個資料集介紹的頁面來總覽鐵達尼號資料的概況：資料已經切分為訓練集（train.csv）、測試集（test.csv）兩個 csv 檔案，訓練集共有 891 筆資料（PassengerId 1 ～ 891），包含 12 個欄位資訊，測試集則有 148 筆資料（PassengerId 892~1309），去掉「Survived」生存與否的欄位，只留下剩餘的 11 個欄位資料。我們的任務是用訓練集來建構出一個機器學習模型，然後匯入測試集到模型中來預測乘客的生存與否，當我們將測試集的預測數據上傳到

Kaggle，Kaggle 便會將我們的預測與正確答案做比對，輸出一個預測準確率的排名結果。

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171	7.25		S
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	female	38	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2. 3101282	7.925		S
4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0	113803	53.1	C123	S
5	0	3	Allen, Mr. William Henry	male	35	0	0	373450	8.05		S
6	0	3	Moran, Mr. James	male		0	0	330877	8.4583		Q
7	0	1	McCarthy, Mr. Timothy J	male	54	0	0	17463	51.8625	E46	S
8	0	3	Palsson, Master. Gosta Leonard	male	2	3	1	349909	21.075		S
9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27	0	2	347742	11.1333		S
10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14	1	0	237736	30.0708		C

◆ 圖 5-15　訓練資料集的欄位資料

如圖 5-15 我們可以獲得的乘客資訊包含：

PassengerId：乘客的編號

Survived：生存與否，0=No、1=Yes

pclass：艙等，也代表乘客的社會經濟地位，1= 上等、2= 中等、3= 下等

Name：姓名

Sex：性別

Age：年齡，不足一歲以小數點表示

SibSp：乘客的家庭關係，兄弟姐妹或夫妻

Parch：乘客的家庭關係，家長或孩子

Ticket：票號

Fare：票價

Cabin：船艙號碼

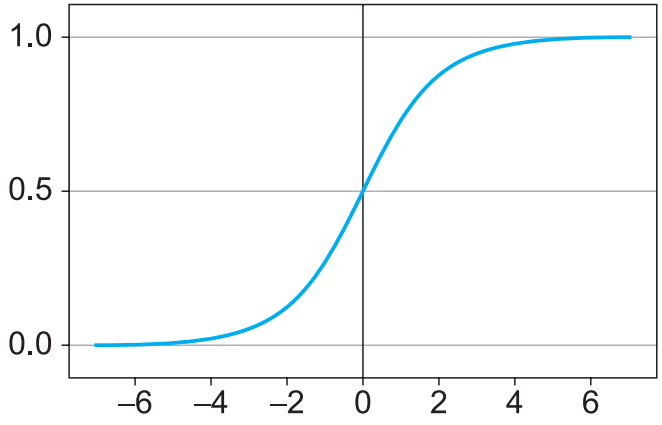
Embarked：登船港口

除了訓練集（train.csv）、測試集（test.csv）兩個 csv 檔案，Kaggle 還另外提供一個檔案是「gender_submission.csv」，這是一個提交給 Kaggle 的檔案範例，只要保留 PassengerId 與 Survived 兩個欄位資料即可上傳讓 Kaggle 評比。

5-3-2 模型簡介 - 羅吉斯迴歸分類器

在第三章我們是用「決策樹」來建構鐵達尼號的預測模型，現在我們嘗試用另一個演算法「**羅吉斯迴歸（Logistic regression）**」來實作看看結果有什麼不同。

羅吉斯迴歸與第三章所介紹的線性迴歸是不同的，最大的差異在於一般迴歸預測的是連續型的數值，例如用降雨量去預測氣溫、用消費去預測 GDP、或是用工作年資 + 性別去預測薪水。而羅吉斯迴歸則是預測一件事會不會發生（常用於二元分類問題），例如：客人會不會回購？借款的人會不會倒帳？房屋會不會成交？疾病會不會發生？把線性函數變形轉化為羅吉斯函數，將低於 0 與超過 1 的部分去除平滑化，帶入 **sigmoid function**，就可以合理產出介於 0~1 的預測機率。所以羅吉斯迴歸是屬於分類問題的機器學習模型。



◆ 圖 5-16　通過羅吉斯迴歸函數只會產出介於 0~1 的預測機率

5-3-3 程式實作

鐵達尼號生存預測模型的建構，與第三章的實作步驟相同，分別為：

1.載入所需的套件

2.載入並觀察資料集

3.資料前處理

4.準備訓練集與測試集

5.訓練模型

我們就一步一步用 colab 編寫 Python 程式實作羅吉斯迴歸模型。

Step1 載入所需套件

Kaggle 鐵達尼資料集可以從網站中下載 https://www.kaggle.com/c/titanic/data，但由於我們已經將 Kaggle 的鐵達尼資料集準備好，放在 google 雲端硬碟上，所以可以使用以下程式碼將資料下載。

```
!wget --no-check-certificate "https://drive.google.com/uc?export=dow
nload&id=13bGRvk1Vq9tFRzMWsXZwOg7TzZLQj83O" -O 'train.csv'
!wget --no-check-certificate "https://drive.google.com/uc?export=dow
nload&id=1kAiZkVAb5DxBS467VlIQKqCfDLOVRaI7" -O 'test.csv'
```

接著我們載入所需的套件

```
# 載入會使用到的套件
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder
```

Pandas 與 NumPy 是我們在 3-4 節程式實作中就介紹過的函式庫，用於分析資料與大量的矩陣運算。

LabelEncoder 是標籤編碼器，用於將文字型的資料轉換為模型可理解的數值型資料。

Step2 載入並觀察資料集

我們在上一個步驟所載入的 Pandas，是 Python 的資料分析函式庫，用 pd.read_csv 即可讀取檔案進來。

```
利用 pd.read_csv 讀入我們所需的檔案訓練集 train.csv 與測試集 test.csv
titanic_train = pd.read_csv(kaggle_dir + 'train.csv')
titanic_test = pd.read_csv(kaggle_dir + 'test.csv')

接著用 print 來輸出訓練集與測試集的維度
print(f'訓練資料的筆數與特徵值數目：{titanic_train.shape}')
print(f'測試資料的筆數與特徵值數目：{titanic_test.shape}')
```

輸出結果會得到：

訓練資料的筆數與特徵值數目：(891, 12)

測試資料的筆數與特徵值數目：(418, 11)

這跟我們在 Kaggle 網站上的資料描述一致，訓練資料共有 891 筆，12 個欄位資訊（也可以說是 12 個維度的資料），測試資料則有 418 筆，11 個欄位資訊（少了存、歿的答案）。我們可以直接用 head()，來輸出前五筆資料進行觀察：

```
# 檢視訓練集的前五筆資料
titanic_train.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

◆ 圖 5-17　鐵達尼號訓練集前五筆資料

```
# 檢視測試集的前五筆資料
titanic_test.head()
```

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S

◆ 圖 5-18　鐵達尼號測試集前五筆資料

從資料中我們可以觀察到，乘客資訊裡有數值型資料如：Pclass 艙等，也有非數值型資料如：Sex 性別，同時從 12 個變量維度中，我們也可以大致判斷，有些資料對於生存預測可能幾乎不存在任何影響性，例如 Name 姓名或是 Ticket 票號，所以這些資訊就是我們在下一個步驟「資料前處理」要捨去的。

Step3 資料前處理

在建構模型前，資料前處理是必要步驟，又稱特徵工程，在這個階段主要有兩個任務要完成：

1. 補缺失值：鐵達尼號總共有 1,309 筆乘客資訊，但有很多資料是缺漏的，例如：年齡、船艙號碼，這些並不是每位乘客都有清楚的記載，稱為「缺失值」，在電腦中無法進行四則運算，所以我們必須透過「補值」的方式，將缺的資料補上才能建模型，而補值的方式有很多種，以下會介紹。
2. 資料轉換：電腦無法將字串（例如性別 male）進行數學的四則運算，所以要將字串轉換成數字。

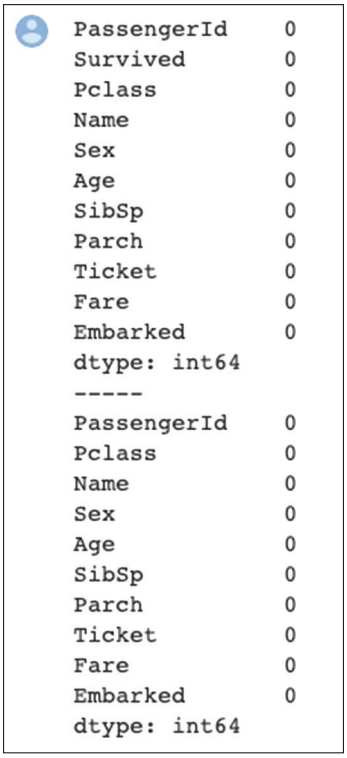
```
# 我們先處理缺失值的部分
# 可以利用 isna() 指令把缺失值統計出來
print(titanic_train.isna().sum())
print("----")
print(titanic_test.isna().sum())
```

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
----
PassengerId      0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin          327
Embarked         0
dtype: int64
```

◆ 圖 5-19　訓練集（上）與測試集（下）有缺失值的欄位數量

輸出結果如圖 5-19，無論是訓練集或測試集，年齡、船倉號碼都有缺失值，訓練集也缺失了兩筆登船港口的資料。因此這些就是我們要補值的欄位。

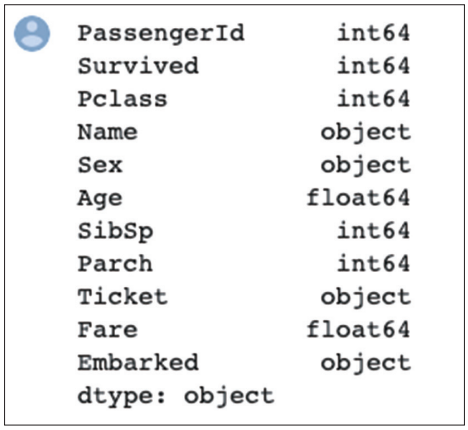
```
# 首先，針對資料中我們熟悉的部分做補值
# Age 欄位是指乘客的年齡，如果沒有比較好的補法，可以使用捨去的方式，或是補上中位數，平均值等等的方法
# 用 fillna() 可以補缺失值，以 titanic_train.Age.mean() 的方法，補上平均值
titanic_train.Age.fillna( titanic_train.Age.mean(), inplace=True)

# 記得訓練集做了什麼前處理，測試集也要跟著做一樣的動作
# 所以測試集也要補上平均值
titanic_test.Age.fillna( titanic_test.Age.mean(), inplace=True)

# Cabin 的缺失值太多，將近佔了整體資料的八成，我們可以大膽的用 titanic_train.drop() 捨去這個欄位
titanic_train.drop(columns='Cabin', inplace=True)

# 訓練集做了什麼，測試集也要跟著做
titanic_test.drop(columns='Cabin', inplace=True)

# 接著是 Embarked，從上方可以看到只有 2 筆缺失，可使用衆數來補值
titanic_train.Embarked.fillna( titanic_train.Embarked.mode()[0], inplace=True)

# 訓練集做了什麼 , 測試集也要跟著做
titanic_test.Embarked.fillna( titanic_test.Embarked.mode()[0], inplace=True)

# 測試集的 Fare 有缺失值，但是訓練集沒有，所以我們這邊針對測試集做補值
# 使用平均值補值
titanic_test.Fare.fillna( titanic_test.Fare.mean() , inplace=True)

# 接著再次檢查是不是還有缺失值
print(titanic_train.isna().sum())
print("-----")
print(titanic_test.isna().sum())
```



```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Embarked       0
dtype: int64
-----
PassengerId    0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Embarked       0
dtype: int64
```

◆ 圖 5-20　資料處理後，訓練集（上）與測試集（下）已都沒有缺失值

```
# 觀察資料的型態
#  將 Object（物件）型態的資料都轉換成 int（整數）或 float（浮點數）， 讓電腦能夠進行四則運算
print(titanic_train.dtypes)
```



```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
Sex             object
Age            float64
SibSp            int64
Parch            int64
Ticket          object
Fare           float64
Embarked        object
dtype: object
```

◆ 圖 5-21　每個欄位的資料型態 int 整數、object 物件、float 浮點數

```
# 若資料沒有直觀轉換的方式，我們可以選擇直接捨去，例如 Name 及 Ticket，幾乎都是不重複的字串，所以我們直接捨去這兩個欄位
titanic_train.drop(columns=['Name','Ticket'], inplace=True)
titanic_test.drop(columns=['Name','Ticket'], inplace=True)
# 接著還剩下 Sex 及 Embarked 要做轉換
# 我們可以使用 sklearn 的 LabelEncoder 工具來做 label encoding（標籤編碼）

# 首先先取出套件，存到變數 le 中
le = LabelEncoder()
titanic_train.Sex = le.fit_transform(titanic_train.Sex)
# 測試集也要跟著做，但要注意這邊是 transform 指令，因為在訓練集已經讓工具知道對應關係了，這邊直接做轉換即可
titanic_test.Sex = le.transform(titanic_test.Sex)

# 接著做 Embarked 的轉換
titanic_train.Embarked = le.fit_transform(titanic_train.Embarked)
titanic_test.Embarked = le.transform(titanic_test.Embarked)
# 最後再次檢查是不是都是 int 或 float 型態
print(titanic_train.dtypes)
print("----")
print(titanic_test.dtypes)
```

```
PassengerId      int64
Survived         int64
Pclass           int64
Sex              int64
Age            float64
SibSp            int64
Parch            int64
Fare           float64
Embarked         int64
dtype: object
----
PassengerId      int64
Pclass           int64
Sex              int64
Age            float64
SibSp            int64
Parch            int64
Fare           float64
Embarked         int64
dtype: object
```

◆ 圖 5-22　經過資料轉換後，所有欄位都是數值型資料

Step4 準備資料集

```
# 我們再次把訓練資料給印出來，以便選擇欄位進入模型訓練
# 可以看到 PassengerId 是每一列獨一無二的編號，可以捨去不進行訓練
# 而 Survived 欄位其實就是我們要去訓練的 y（預測分類：生或歿）
titanic_train.head()
```

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	0	3	1	22.0	1	0	7.2500	2
1	2	1	1	0	38.0	1	0	71.2833	0
2	3	1	3	0	26.0	0	0	7.9250	2
3	4	1	1	0	35.0	1	0	53.1000	2
4	5	0	3	1	35.0	0	0	8.0500	2

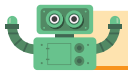
◆ 圖 5-23　資料前處理後，前五筆欄位資訊的數值

```
# 準備訓練集與測試集

# train_x 不需要這兩個欄位
train_x = titanic_train.drop(columns=['PassengerId', 'Survived'])

# train_y 就是 Survived 欄位
train_y = titanic_train['Survived']

# test_x 的規則與 train_x 相同，但是測試集本身就沒有 Survived 的欄位，所以不需要捨棄
test_x = titanic_test.drop(columns=['PassengerId'])
```

Step5 訓練模型

在 5-3-2 節簡單介紹了羅吉斯迴歸模型的演算法原理，現在我們使用 scikit-learn 函式庫 LogisticRegression() 就可以調用這個模型。

```
# 選擇羅吉斯迴歸模型（LogisticRegression）
from sklearn.linear_model import LogisticRegression
estimator = LogisticRegression()

# 進行模型訓練
estimator.fit(train_x, train_y)

# 進行模型預測
pred = estimator.predict(test_x)
# 提交檔生成
# 提交檔需要兩個欄位：測試集的 PassengerId, 以及每個 Id 對應到的 Survived,
即我們的預測值
submit = pd.DataFrame({'PassengerId': titanic_test.PassengerId,
'Survived': pred})

# 最後就可以存成 csv 檔案，提交至 Kaggle !
submit.to_csv(kaggle_dir +'titanic_baseline.csv', index=False)
```



知識補充

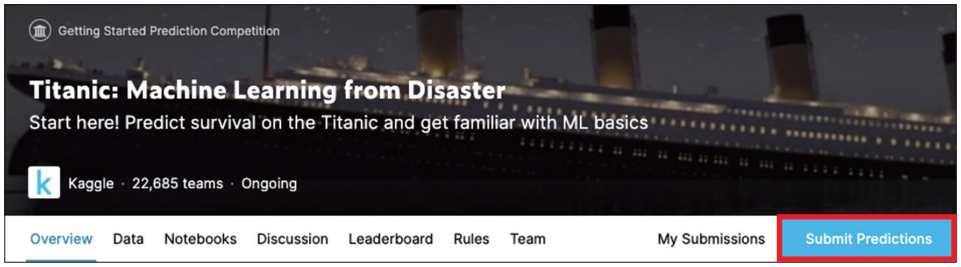
跟第三章比較，少了一組資料！

從上面的程式碼可以看到我們有 train_x、train_y、test_x，卻沒有第三章的 test_y 資料，這是因為我們從 train_x、train_y 來訓練模型，然後針對 test_x 來做出預測值，只是我們沒有 test_y（答案），來判斷模型所做的預測值準確率有多高，因此最後只要將模型輸出的預測值提交給 Kaggle，Kaggle 網站內部有 test_y 標準答案，就會自動比對，給出一個準確率的評分。

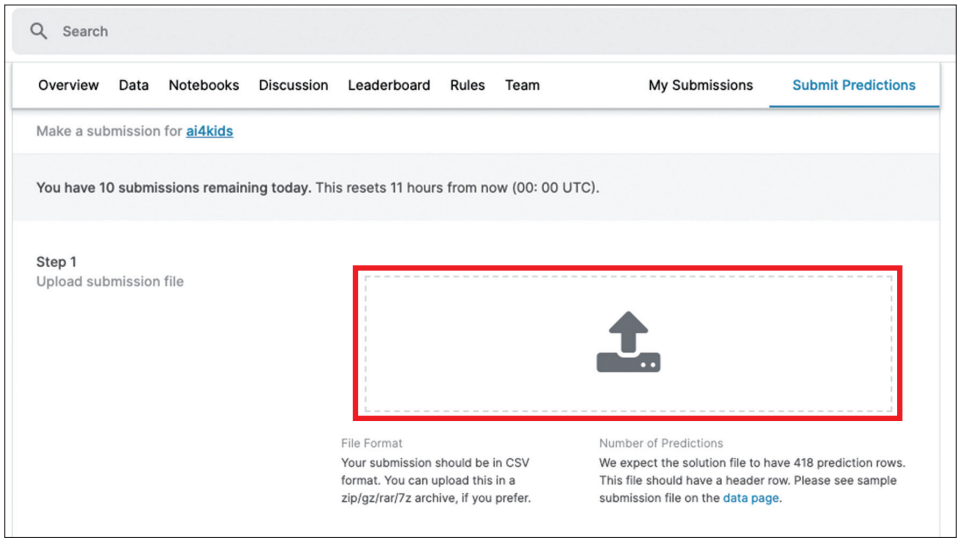
5-3-4 將資料上傳到 Kaggle

將預測資料上傳到 Kaggle 步驟也是很簡單的，主要在競賽頁面點擊 Submit Predictions 按鈕，接著選擇上傳的 csv 檔，按 Make Submission 提交，就完成了！

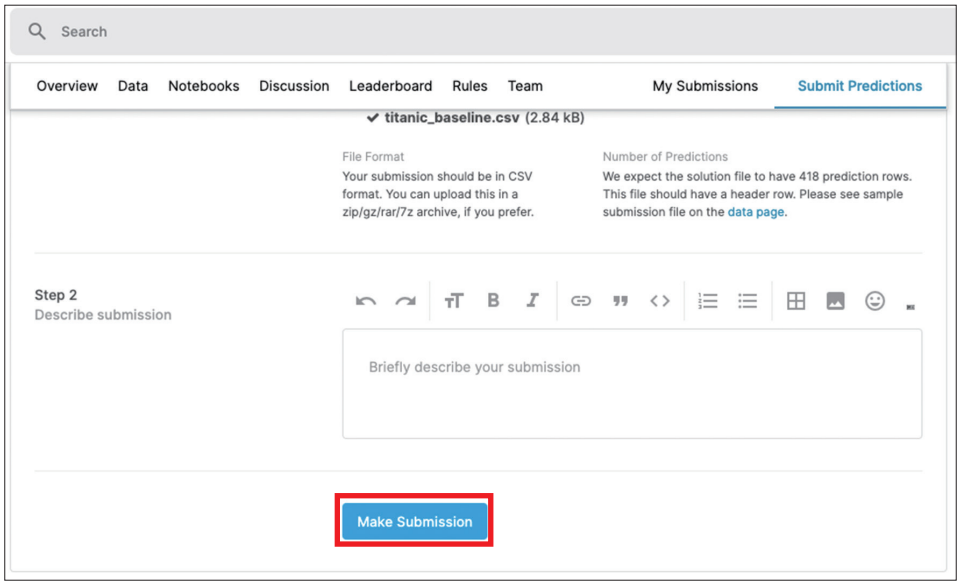
上傳後，Kaggle 幾秒內就會算出你的模型準確度有多高，並給你一個排名。



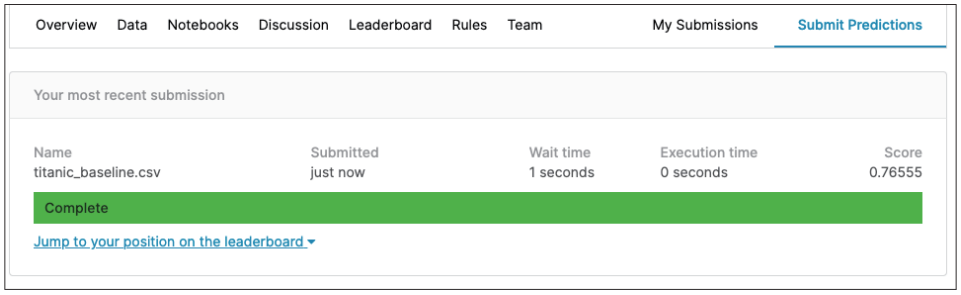
◆ 圖 5-24　點擊藍色按鍵 Submit Predictions 即可上傳預測檔案



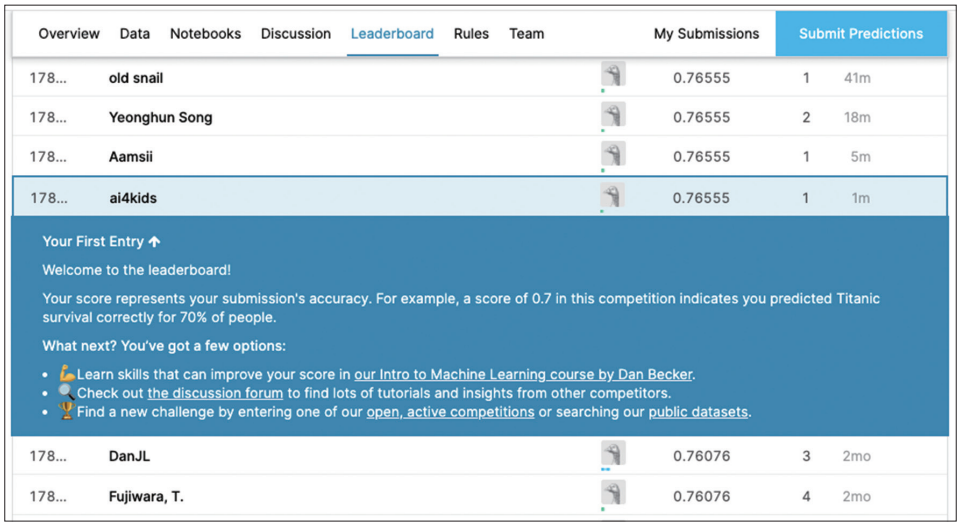
◆ 圖 5-25　點上傳圖示選擇檔案上傳



◆ 圖 5-26　按下 Make Submission 送出預測檔



◆ 圖 5-27　Kaggle 顯示預測的準確率為 0.76555



◆ 圖 5-28　競賽排名

我們在總參賽 22,685 隊伍中，排名第 1 萬 7 千多名，還有努力的空間，例如可以在資料前處理階段進行更細緻的特徵工程，或是用其他模型來訓練看看，可能會有更好的表現。

5-4 總結

本章我們從 Kaggle 網站開始介紹，帶領讀者了解競賽區、資料集、筆記本等相關功能，並一步一步指導你實際完成了 Kaggle 競賽的練習，取得了人生第一個 Kaggle 成績！這代表你向機器學習又邁出了一大步，是非常值得鼓勵的一件事！如果想知道有沒有更好的模型可以做出更精準的預測，歡迎讀者觀看本書所提供的影音教學，我們將提供更進階的幾種方法給大家參考。

如果讀者因此對機器學習產生了興趣，建議持續留在 Kaggle 這個平台，與來自世界各地的好手專家們互動，同時也能獲得最新的資訊與國際接軌。

人工智慧日新月異，本書的目的在打破 AI 的高、大、上，讓人人都可以參與這門新興科技的學習，並且從中獲得信心與成就感。無論讀者後續選擇如何進階到下一個階段，打造人工智慧的基本流程仍是第二章所提到的五個步驟：

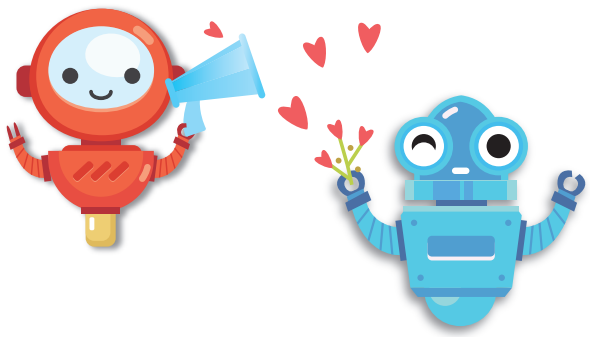
順著這個流程，將可以運用機器學習解決許多的問題，並在每次的解題專案中更近一步增進自己的實作能力。

總結本章的內容：

- Kaggle 網站的介紹。
- Kaggle 推薦給新手的資料集介紹。
- Kaggle 賽事參與的步驟。

◎表 5-1 機器學習方法比一比

步驟	第 3 章 決策樹	第 5 章 羅吉斯迴歸
讀取資料	只用 train.csv	train.csv、test.csv
資料分析	分析 Sex, Age 這兩個屬性與 Survived 的關係，並畫出柱狀圖	1. 利用 shape() 觀察 train.csv 與 test.csv 的資料筆數 2. 利用 head() 觀察前五筆資料欄位資訊
特徵工程 - 資料編碼	用 for 迴圈將 Sex 和 Embarked 兩個欄位轉換成 0~n 的數字	用 LabelEncoder() 將字串資料轉成數值資料
特徵工程 - 缺失值補值	用 fillna(999) 補缺失值	用平均數與眾數補缺失值
分割資料	把 train.csv 的前 750 筆當作訓練資料，750 之後當作測試資料	沒有分割資料
訓練與測試資料	訓練資料集 train_x train_y 測試資料集 test_x test_y	訓練資料集 train_x train_y 測試資料集 test_x
分類器模型選擇	決策樹	羅吉斯迴歸
驗證預測結果	用 Accuracy 和 Recall 兩個 metric（指標）來評估模型預測值與標準答案 test_y 比較後的準確程度	產生一個 titanic_baseline.csv 檔，上傳給 Kaggle 做排名評估




NOTE