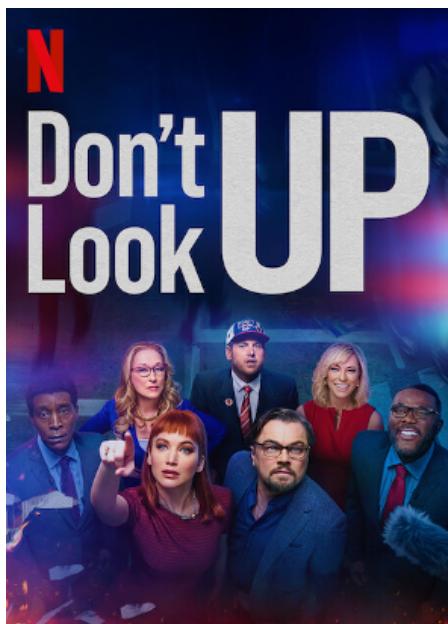


NASA - Nearest Earth Objects 利用機器學習模型判斷小行星特徵值與對地球危害程度預估

機械系E14091172蘇致宇 機械系E14086541李允評

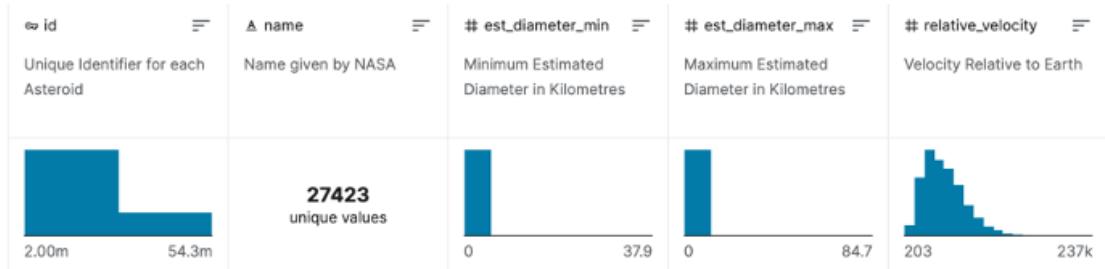
- 發想動機

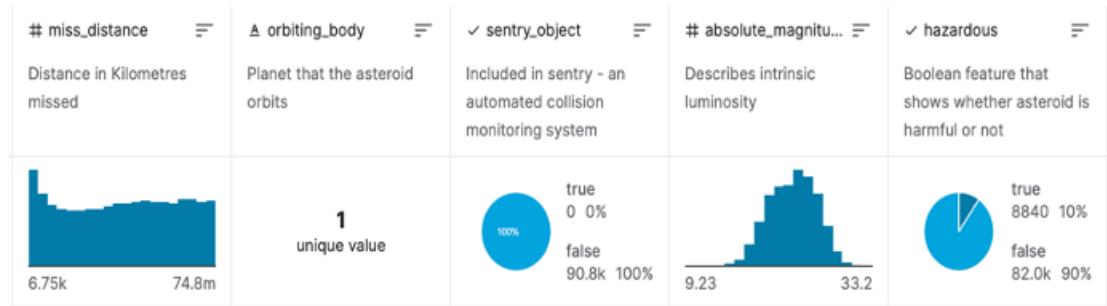
本報告主題發想自Netflix 2021年製作電影「千萬別抬頭 Don't Look Up」，劇中藉由彗星撞地球的危機影射現代社會中的政治經濟問題。其中彗星對地球是否會對地球造成危害產生了廣大的討論，由此我們發想出可以利用NASA Open API 進行機器學習的模型實作。



圖、千萬別抬頭宣傳海報

- 資料庫介紹





在資料庫中總共包含了三項類別型資料、五項連續型資料及兩項布林值

- 問題定義

在接下來的分析方法中我們計畫達成兩大目標，分別是：

1. 利用視覺化方法分類並找到個變數之間的關聯
2. 利用機器學習模型實踐對地球危害小行星的辨別

- 資料前處理

- 資料分類

使用 `pandas.isnull().sum` 函式統計資料庫中資料缺失的狀況，在這裡並無資料缺失因此可以直接進入下一階段的分析。

```

id                      0
name                     0
est_diameter_min        0
est_diameter_max        0
relative_velocity       0
miss_distance           0
orbiting_body           0
sentry_object            0
absolute_magnitude      0
hazardous                0
dtype: int64

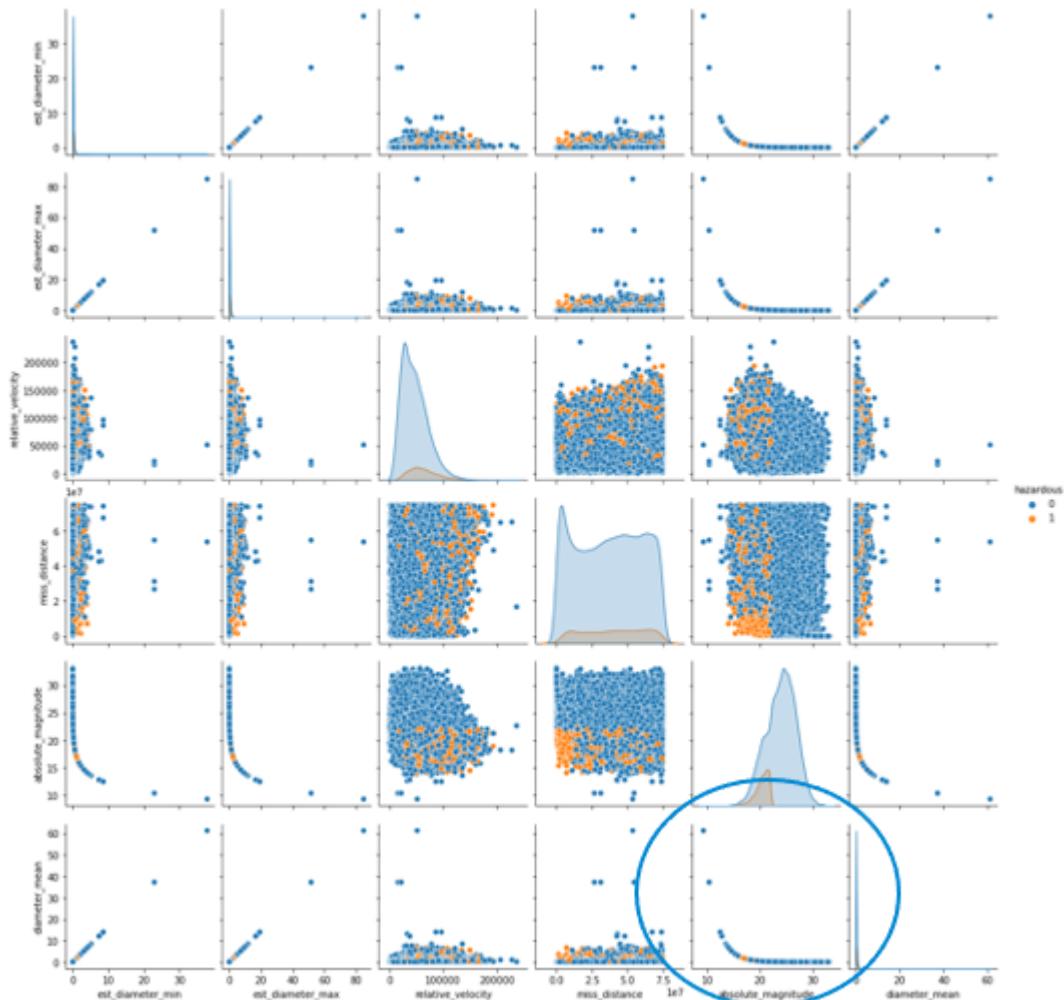
```

`id`, `name`, `orbiting_body`, `sentry_object`等四項類別型變數在特徵工程上沒有明顯的價值，因此在後續的分析忽略不採計。

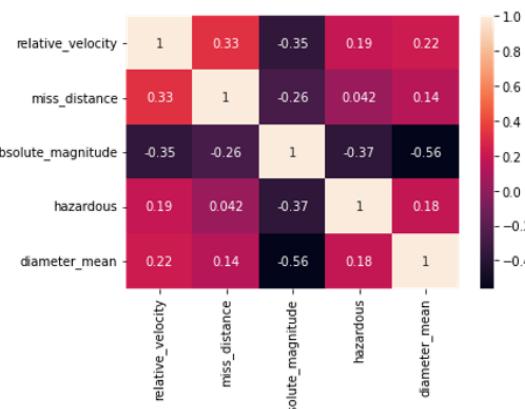
同時在分類時將 `hazardous` 的布林值 `True`, `False` 轉為數值型態 `1`, `0` 方便操作。並將變數裡的 `est_diameter_min` (`Minimum Estimated Diameter in Kilometres`), `est_diameter_max` (`Maximum Estimated Diameter in Kilometres`) 平均，創造一全新變數欄位 `diameter_mean`。

- 資料視覺化分析

- 將所有連續型變數利用pairplot的形式同步繪製觀察其相關性。大致上皆為隨機散佈，可以觀察到diameter(小行星直徑)和absolute magnitude(絕對星等)呈現負相關。



- 皮爾森積動差相關係數：
將各連續性變數的相關係數以熱圖方式呈現

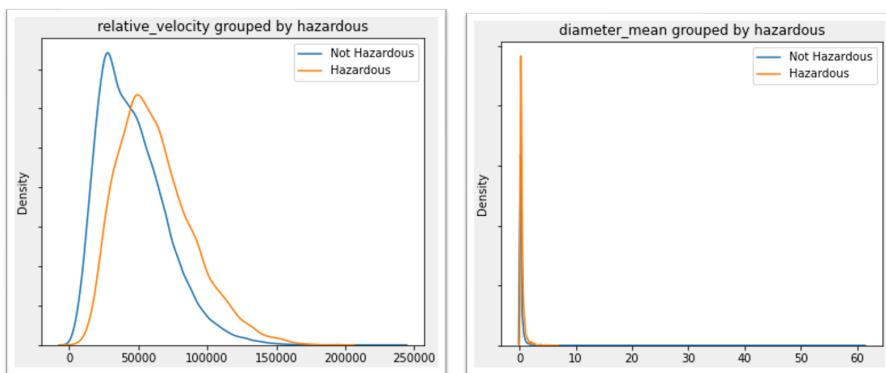


3. 核密度估計(Kernel Density Estimation):
 核密度估計可以看出隨機變數落在特定值的可能性，
 公式如下：

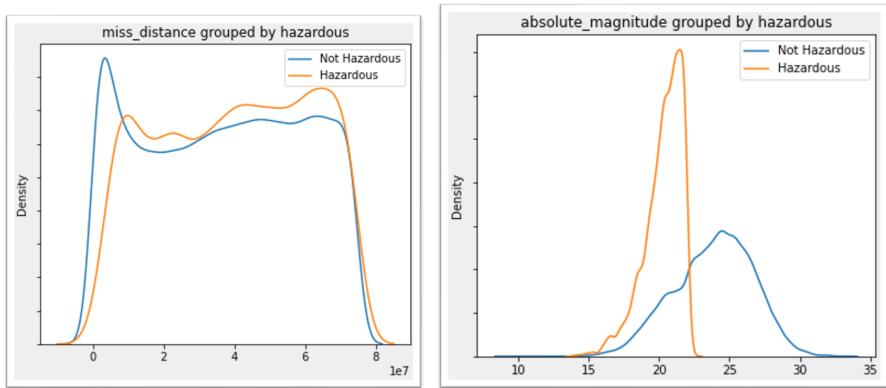
$$K(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right)$$

$$\hat{f}_n(x) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x - x_i}{h}\right)$$

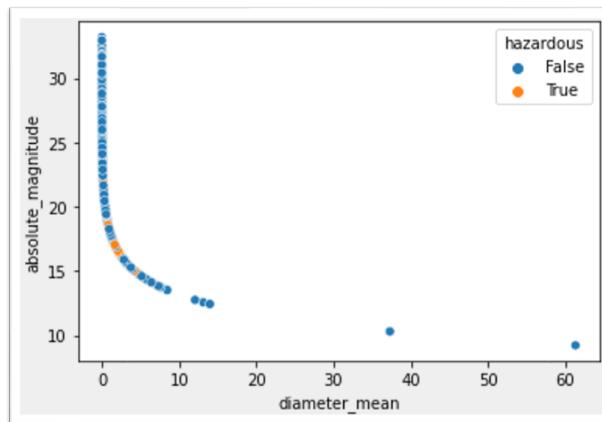
套用seaborn中kde.plot()的方法實踐核密度作圖，這裡討論relative velocity, diameter, miss distance, absolute magnitude等四個變數，並以hazardous作為hue分類



左圖、具威脅小行星移動速度略快於不具威脅性類別
 右圖、資料集中小行星大小分佈集中



左圖、距離與威脅性較無明顯關聯
 右圖、具威脅性小行星絕對星等集中於20左右
 又，絕對星等和小行星直徑大小成負相關關係（如下圖）
 ，因此我們可以知道具威脅性小行星集中在特定大小類型



絕對星等與小行星平均直徑散佈圖

- 機器學習模型實作

總共選用七種不同的機器學習模型，分別為：

1. Decision Tree
2. XG Boost
3. KNN
4. Random Forest
5. Gaussian Naive Bayes
6. SVC
7. Logistic Regression

以Decision Tree為例，示範程式碼的建立方法

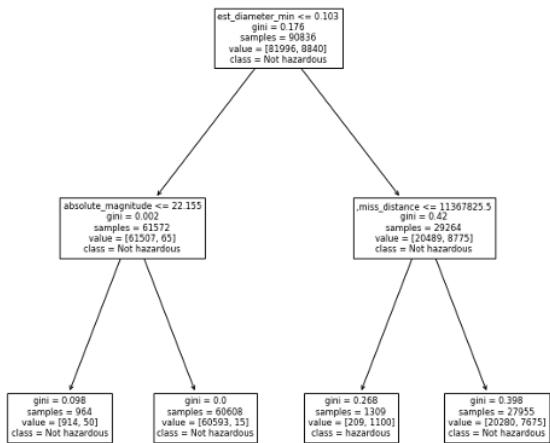
```
# 匯入決策樹模型
from sklearn.tree import DecisionTreeClassifier

# 創造決策樹模型
model = DecisionTreeClassifier(
    criterion='gini',           # 設定最佳化方法為 Gini Index
    max_depth=2,                # 設定最大深度為 2
    max_leaf_nodes=2 ** 2       # 設定最多葉子個數為 4
)
# 訓練決策樹模型
model.fit(x, y)

# 確認模型是否訓練成功
pred_y = model.predict(x)
# 計算準確度、f1 score
acc = accuracy_score(y, pred_y)
f1 = f1_score(y, pred_y)

# 輸出準確度
print('accuracy: {}'.format(acc))

accuracy: 0.9124906424765511
```



深度為2 最大葉子數為4 的Decision Tree視覺化模型圖形

- 模型結果比較

	DTC	XG Boost	KNN	RF	GNB	SVC	LR
Train Accuracy	0.9125	0.9133	0.9179	0.9129	0.897	0.903	0.9027
Valid Accuracy	0.9124	0.9128	0.8812	0.9125	0.897	0.902	0.9026
error(%)	0.011	0.0547465	3.99826	0.0438	0.011	0.022	0.0111

DTC: Decision Tree Classifier

KNN: K Neighbors Classifier

RF : Random Forest Classifier

GNB: Gaussian Naive Bayes

SVC: Support Vector Machine(Regression)

LR : Logistic Regression

Average Valid Accuracy為我們使用K次交叉驗證工具(K-Fold Cross-Validation)，在 K-Fold 的方法中我們會將資料切分為 K 等份，以K-1份作為訓練資料，剩下1份作為驗證資料並在訓練完畢後驗證預測計算其誤差值。

在上述七種模型的測試當中，Valid Accuracy與Train Accuracy並無太大區別，可以確定在模型試驗中並沒有過度擬合的問題。

我們也觀察到在訓練準確度(Train Accuracy)中支援向量機(SVC)和邏輯回歸(Logistice Regression)模型的準確度非常相近，推測是因為兩者皆是採取回歸分析的方法進行訓練，與其他分類學習法不同，因此結果較為接近。

- **問題與討論**

除了使用Scikit Learn中的accuracy_score()函式進行模型準確度的評分，我們還引入f1_score評估機器學習模型成效。

F1 Score的定義如下：

$$F1\ Score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

$$Precision = \frac{TP}{TP+FP} \quad Recall = \frac{TP}{TP+FN}$$

是基於混淆矩陣所定義出的分數，為召回率(Recall)和準確率(Precision)的，跑分結果如下表：

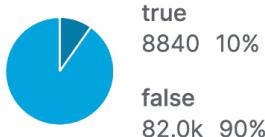
Model	f1_Score
KNeighborsClassifier	0.402892
XG Boost	0.243659
Random Forest	0.226931
DecisionTreeClassifier	0.216770
Gaussian Naive Bayes	0.064219

然而在先前accuracy_score獲得高分的模型在f1_score評斷下的表現卻不甚理想，重新檢視資料庫本身我們發現了一項問題：作為輸出Y值的變數hazardous除了本身為布林值僅有真假兩項，資料集本身更有數據不平衡的問題。對地球有威脅性的資料僅佔整體數據10%。

若在未進行前處理的狀況下直接對其進行訓練，就會發生上述以預測Y值和實際Y值進行評斷的accuracy_score準確度獲得高分；模型本身的f1_score分數卻不理想的狀況。

✓ hazardous \equiv

Boolean feature that
shows whether asteroid is
harmful or not



解決這項問題我們預計在未來有機會操作時先以Random Under/Over Sampling進行資料前處理，避免相同情形產生。同時也藉此機會了解到資料集本身資料分佈的優劣也在機器學習領域佔有舉足輕重的地位。

- 完整程式碼

https://colab.research.google.com/drive/1kVkJloF3_xIRCgs_stndyRz-XIAgZXw?usp=sharing

- 修課心得

蘇致宇：我在上這門課之前沒有接觸過機器學習模型的相關知識，充其量只能算個略懂Python程式語法的新手。還記得在第一堂課一開始老師說到：「從真正讓你們去嘗試一份資料集的分析，就會知道機器學習和這些數學模型背後所代表的意思。」從這堂課我扎扎实實地體驗到什麼叫做實作的重要，常常在課堂上聽得一知半解的專業名詞或是公式，在程式跑完後看到運行結果的數據或是覺化的圖表呈現，會有一種「喔原來這邊是這樣子！」恍然大悟的體驗。也在努力讀懂Kaggle上不同大神針對同一份資料集做出的不同見解，和助教不厭其煩的幫忙debug找到我們實作出的程式中邏輯不夠通順有問題的地方，了解到原來一個問題可以從不同方向切入，也會看到不一樣的風景。「要對資料存有好奇心。」這是我最後報告完Term Project老師所提出的評語，也是我在這門課所體會到分析資料的有趣之處。

李允評：這是我第一次選修關於機器學習的課程，在此之前幾乎沒有該領域的先備知識。雖然有Python程式設計與實作課程的程式基礎，但面對本課程有深度的內容還是難以在短短一周之內理解並消化。可惜太晚才發現moodle上有公布過往的課程影片連結，若能提前預習課程內容，在報告與作業的撰寫上一定能更加得心應手。經

過這一週非常密集的機器學習課程，我有了很大的收穫，了解到模型究竟是如何具有學習能力，認識了很多python與其他套件的使用方法，也將過去曾在應用統計課程中學過的觀念實際應用，透過Term project 與HW將所學進行實作。由於我programming的經驗很少，對於機器學習的工具上手速度很緩慢，再加上短時間內要消化大量的新知識，並將其實際應用，對我而言頗有壓力，此課程無疑是我大學修過困難程度最高的一門課。正因如此，也成為了難能可貴的經驗，在討論Term project的過程中向我的隊友請教了無數次，釐清了許多觀念，透過此次合作從他的身上學到很多，非常慶幸能有一個很好的partner。此外，在Term project的報告中，也從其他組別學到了許多不同的觀點，以及認識到課堂上沒有教授的方法。此次課程實在獲益良多，若未來有機會，希望能選修學期間的機器學習相關課程，在時間充裕的情況下，更進一步地認識這個充滿無限可能性的領域。

建議：可以先向修課學生預告明確的課程難度，並建議沒有背景知識與程式經驗不足的同學，先自行預習課程教學影片。

- 參考資料

- Jet Propulsion Lab：
<https://cneos.jpl.nasa.gov/ca/>
- NASA Open API
- NASA - Nearest Earth Objects –
<https://www.kaggle.com/code/elnahas/nasa-nearest-earth-objects/notebook>
- N.E.O. Classification with Decision Tree
<https://www.kaggle.com/code/johnyoungsorense/n-e-o-classification-with-decision-tree/data>
- 全民瘋AI系列2.0
<https://ithelp.ithome.com.tw/users/20107247/ironman/4723>
- Precision, Recall, F1-scorer簡單介紹
<https://medium.com/nlp-tsupei/precision-recall-f1-score%E7%89%A9%E7%95%9C-4a2f3a2a2a1>

%B0%A1%E5%96%AE%E4%BB%8B%E7%B4%B9-f87baa
82a47

- 白話解釋核密度估計(Kernel Density Estimation)
<https://medium.com/qiubingcheng/%E7%99%BD%E8%A9%B1%E8%A7%A3%E9%87%8B%E6%A0%B8%E5%AF%86%E5%BA%A6%E4%BC%B0%E8%A8%88-kernel-density-estimation-18c4913f0b6a>