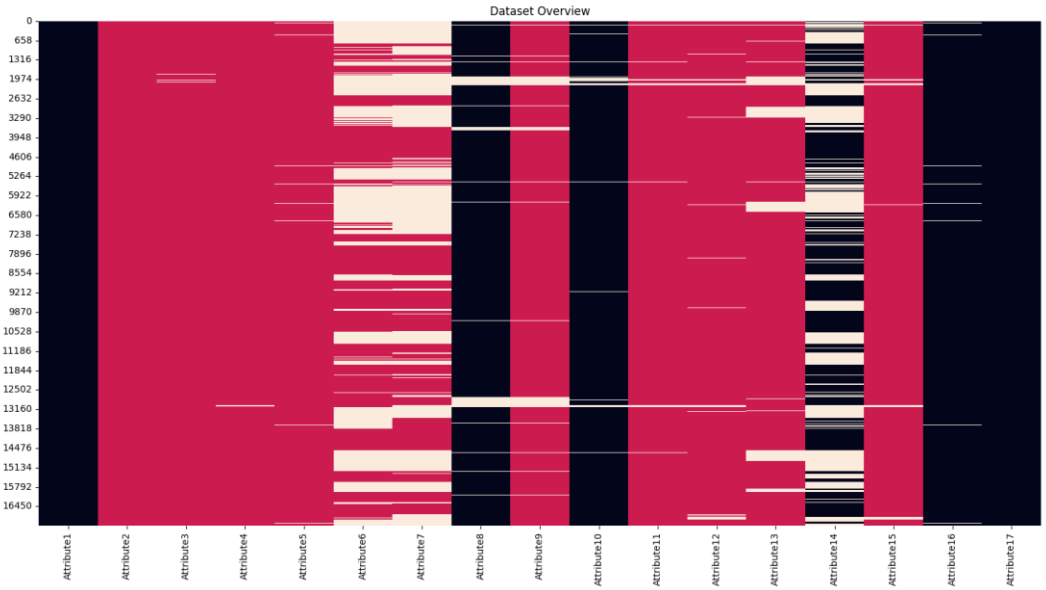
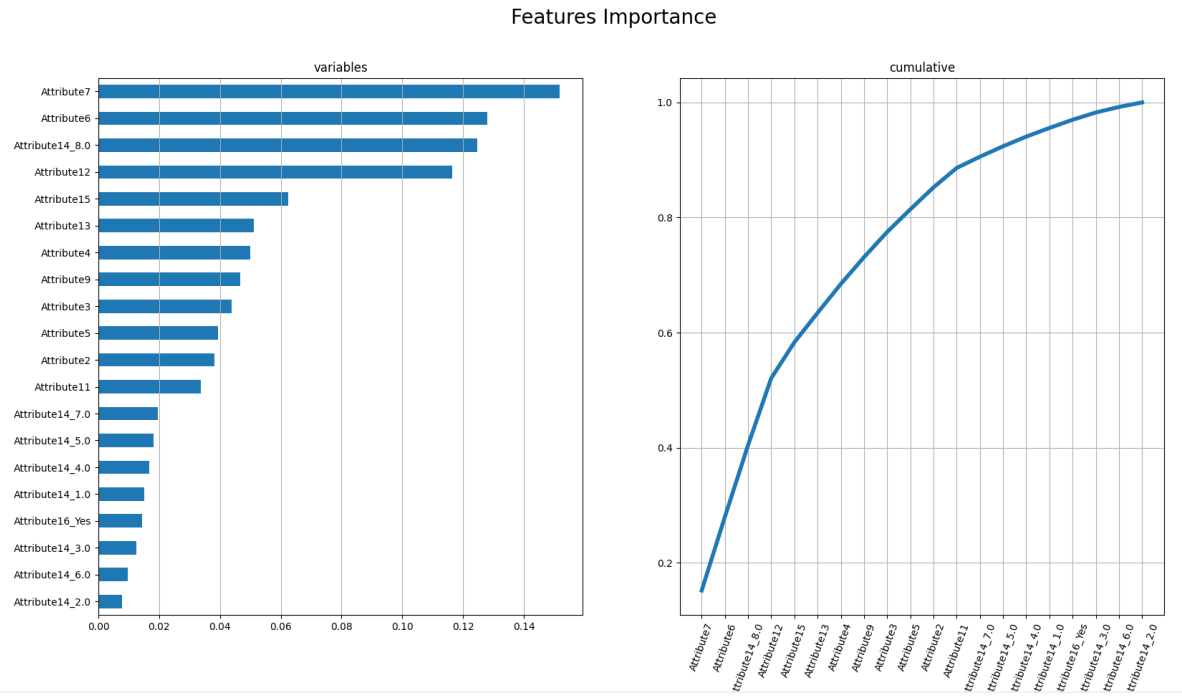
Data Science HW1 B10815044 謝鈞曜

觀察

觀察 Attribute1 的資料可以發現它就是代表資料的時間，基本上跟預測的結果不太有關係，因此我直接就剃除這個 feature 了，再來分析資料的情況(如圖表 1)，發現有些資料是遺失的所以需要補值，後續透過 sklearn 分析 feature importance(如圖 2)發現 Attribute8 跟 Attribute10 基本上也是沒什麼用的資訊，因此也相繼剃除了，開始分析之後發現預測結果很極端 TP 跟 FN 都很高，推測是資料分布太不平均所導致。



圖表 1 黑色代表 category data 紅色 numerical data 白色代表 missing data



圖表 2 分析 Feature importance

資料預處理

資料分布

首先我先把 Yes 跟 No 的資料分成兩堆，然後我希望 Yes 跟 No 的資料數量可以大概各一半，因為 No 的資料比較多，所以如果資料(observation)不是完整的就直接 drop 掉，但是 Yes 的資料相對少很多，因此如果有缺漏的話用 K-NN 來補值(用一堆 Yes 的資料來做 K-NN 也可以得到相對精準的值)，處理過後 No 堆的資料有 5696 筆，而 Yes 則是 3139，此時資料還是不平均的狀況，所以我在訓練的模型的時候會從 No 堆的 5696 隨機抓 3139 筆資料，這樣就可以達到各半了。

訓練集與測試集比例

7:3, 8:2, 9:1 都嘗試過，結論是沒有很顯著的影響，可能是資料集還算夠。

模型訓練

模型參數調整

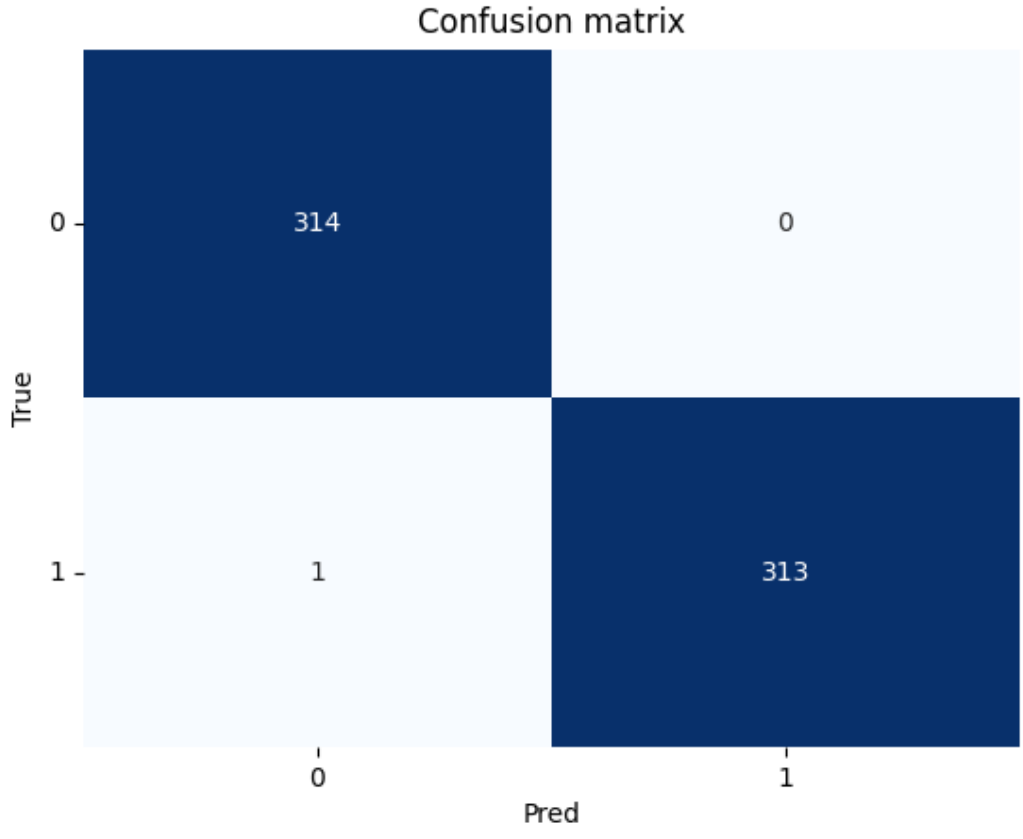
利用 sklearn.model_selection.GridSearchCV 找出最佳的參數組合，一開始跑很久 CPU 跟 GPU 都沒吃滿，後來發現把 n_job 設成-1 就可以滿載跑，節省了不少時間。

最佳參數組合如下圖

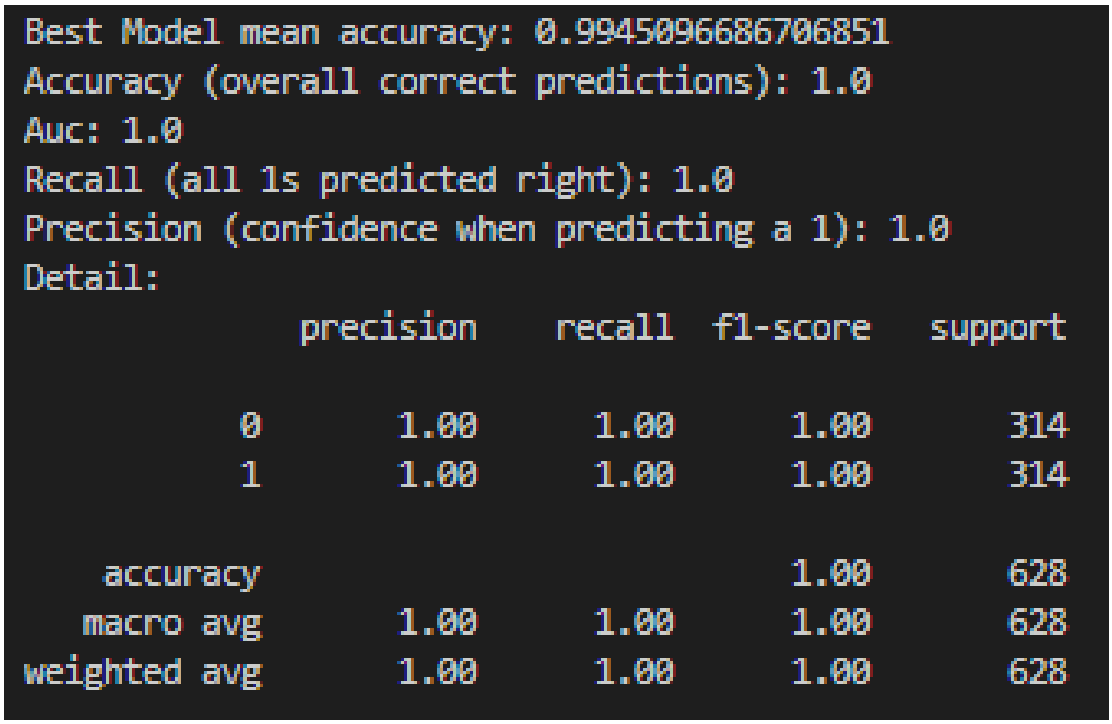
```
Best Model parameters: {'subsample': 0.8, 'n_estimators': 1750, 'min_samples_split': 8, 'min_samples_leaf': 1, 'max_features': 7, 'max_depth': 3, 'learning_rate': 0.15}
```

結果

基本上 FP 跟 FN 都在 3 以內，最好的情況 FP 跟 FN 都可以到 0 (如圖 3)，平均的準確值大約在 99.4%~99.57%左右(如圖 4)。



圖表 3 Confusion Matrix



圖表 4 Accuracy

結論

以上內容是最後弄出最佳解的結論，當然沒有包含剛開始走過的坑，所以沒有像上述內容那麼順利，也沒有辦法一開始就能發現這麼多細節，全部都是自己慢慢爬文研究得出的結果，嘗試各種能讓預測變的更準的可能性，剛開始有點太糾結於模型參數的調整，但後來我發現比較重要的是 Data Mining 跟 Data Preprocessing，主要把資料分布平均之後整個準度就上升了，因此我的測試結果直接從大概 77%直接跳到 99%，透過此作業也發現其實自己蠻喜歡做資料分析的，特別是資料視覺化的部分，我覺得很療育，但有時候也是蠻煩的，要一直等模型跑完，然後準確度又沒有提升，結論還是不要太糾結，有空想一下想一下，想到東西再回來做做看就可以了

參考資料

https://towardsdatascience.com/machine-learning-with-python-classification-complete-tutorial-d2c99dc524ec