資料科學導論 Competition report 1

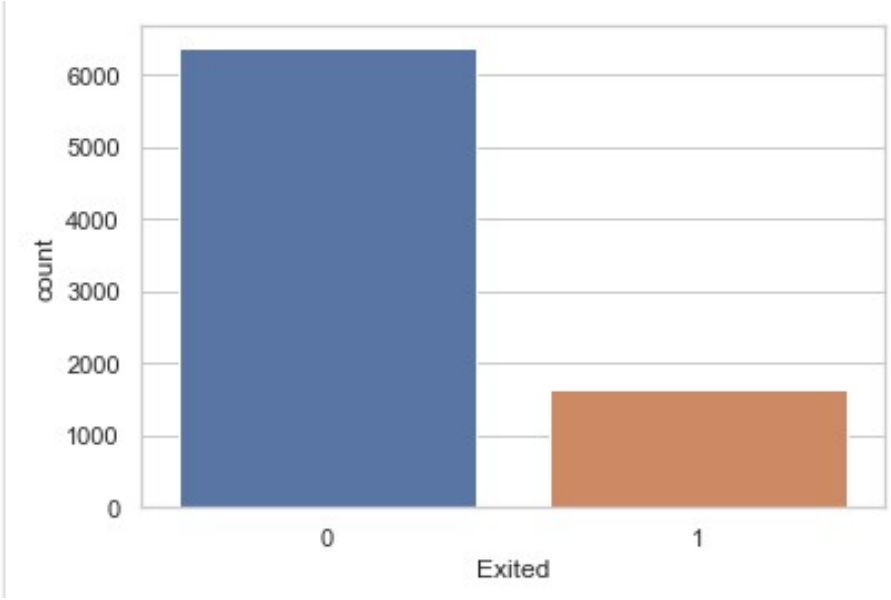
組別：學分戰士(team17)
組員：H24086024 施文千、H24081341 耿皓昀、D54061344 林郁富

壹、競賽敘述與目標

藉由一筆銀行客戶資料，包括年齡、性別、所得等變數，並運用 scikit learn 中的機器學習套件，分析並預測客戶未來是否還會與銀行有往來。

貳、資料前觀察、處理

資料觀察：



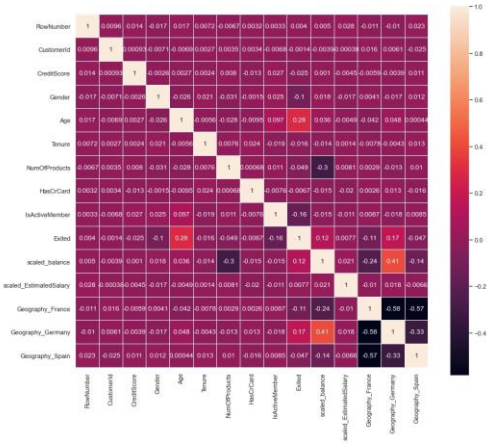
▲由上圖可知，此資料為不平衡資料

	CreditScore	Age	Tenure	Balance	NumOfProducts	EstimatedSalary
count	8000.000000	8000.000000	8000.000000	8000.000000	8000.000000	8000.000000
mean	630.804875	42.988125	4.567375	100569.286116	1.534250	100103.554906
std	59.076984	7.515870	1.272758	31431.771836	0.582554	24511.460933
min	446.000000	28.000000	2.000000	62397.410000	1.000000	57522.070000
25%	590.000000	38.000000	4.000000	62397.410000	1.000000	79216.925000
50%	631.000000	42.000000	5.000000	111118.600000	1.000000	100056.110000
75%	672.000000	47.000000	6.000000	126500.100000	2.000000	121104.542500
max	753.000000	81.000000	7.000000	188500.680000	4.000000	142481.990000

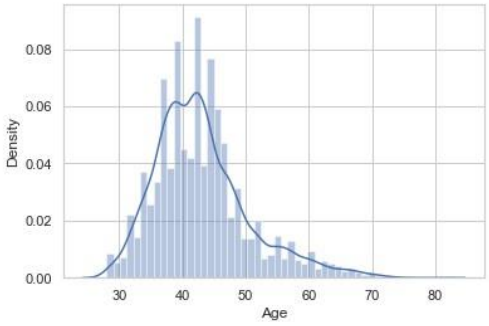
▲觀察連續型變數之敘述統計量

```
RowNumber          0
CustomerId         0
Surname            0
CreditScore        0
Geography          0
Gender             0
Age                0
Tenure             0
Balance            0
NumOfProducts      0
HasCrCard          0
IsActiveMember     0
EstimatedSalary    0
Exited             0
dtype: int64
```

▲從上圖中此資料無遺失值



▲從上圖中可得知此 age 與 exited 兩變數 correlation 最大



▲由上圖可得知，年齡之分布右偏

資料處理－最終使用部分：

```
df['Gender'] = df['Gender'].map( {'Female': 0, 'Male': 1} ).astype(int)
df1['Gender'] = df1['Gender'].map( {'Female': 0, 'Male': 1} ).astype(int)
```

```
df['Geography_France']=df['Geography_France'].astype('int')
df['Geography_Germany']=df['Geography_Germany'].astype('int')
df['Geography_Spain']=df['Geography_Spain'].astype('int')
```

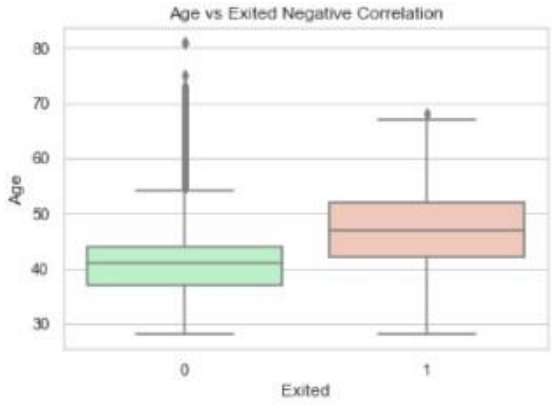
▲設性別與國家為 dummy variable

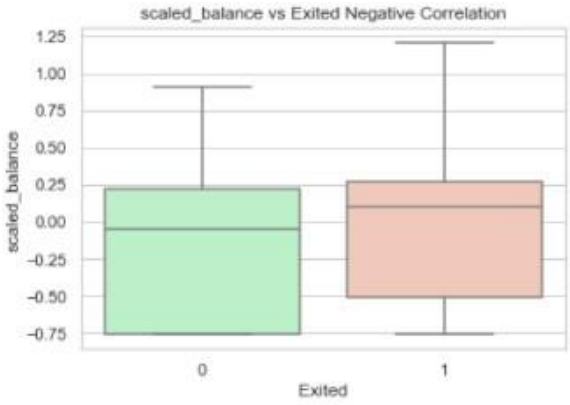
```
df['scaled_balance'] = rob_scaler.fit_transform(df['Balance'].values.reshape(-1,1))
df['scaled_EstimatedSalary'] = rob_scaler.fit_transform(df['EstimatedSalary'].values.reshape(-1,1))
```

▲將 balance、estimatedsalary 做標準化

```
df=df.drop('Surname',axis=1)
df=df.drop('RowNumber',axis=1)
df=df.drop('CustomerId',axis=1)
```

▲將與會影響預測且與預測目標無關之變數去除



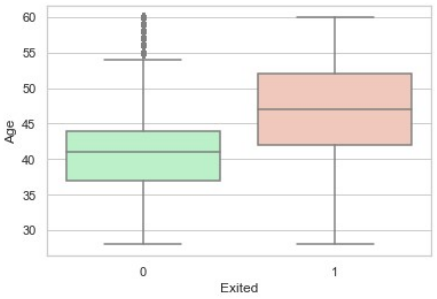


▲將連續變數做標準化及化盒型圖，發現在 exited=0 時 age 的 outlier 特別多

```
def outlier_treatment(datacolumn):
 sorted(datacolumn)
 Q1,Q3 = np.percentile(datacolumn , [25,75])
 IQR = Q3 - Q1
 lower_range = Q1 - (1.5 * IQR)
 upper_range = Q3 + (1.5 * IQR)
 return lower_range,upper_range

lowerbound,upperbound = outlier_treatment(df.Age)
df.drop(train[ (df.Age > upperbound) | (df.Age < lowerbound) ].index , inplace=True)
```

▲將 outlier 去除



▲將 outlier 去除後的 age 盒型圖

資料處理－較無效果部分：

```
train.loc[train['Age'] <=38, 'Age_cat'] = 0
train.loc[(35<train['Age']) & (train['Age']<=42), 'Age_cat'] = 1
train.loc[(43<train['Age']) &(train['Age']<=47), 'Age_cat'] = 2
train.loc[train['Age'] >47, 'Age_cat'] = 3
```

▲將 age 依據 quantile 做分類，
但效果不顯著，未使用

```
train.drop(['Tenure'],axis=1,inplace=True)
```

▲將與 exited cor 接近 0 的變數刪除，
但效果不顯著，未使用

▲此外，我們也嘗試將連續變數依據 quantile 做分類，但效果依舊不顯著

參、預測訓練模型

Step 1 分割資料

由於此資料集無法獲得測試資料之目標參數結果，因此我們考慮將訓練資料以 7:3 拆開，作為訓練及測試資料。

Step 2 處理不平衡資料

因為此資料為不平衡資料，我們主要是想要預測 exited=1 的部分，然而此部分為少數資料，容易因為過少導致預測所有值為 0 的準確度很高，使模型都預測為 0，為避免此情況發生，我們採取 oversampling 的方法，使少數樣本透過隨機抽樣增加，使其比重提高，讓模型不會忽略掉它

此外，我們也試過使用 downsampling 的方法，但效果不顯著

Step 3 預測模型選擇

(1)Logistic Reggresion

使用 LogisticRegression 並運用資料切割出來的 train data 訓練模型，並拿此訓練模型預測 test data

(2)Random Forest Classifier

使用 Random Forest 模型，並運用資料切割出來的 train data 訓練模型，並拿此訓練模型預測 test data

(3)XGBoost Classifier

使用 XGBoost 模型，並運用資料切割出來的 train data 訓練模型，並拿此訓練模型預測 test data

(4)CatBoost Classifer

創造 CatBoost 模型，並運用資料切割出來的 train data 訓練模型，並拿此訓練模型預測 test data

(5)AdaBoost Classifier

創造 AdaBoost 模型，並運用資料切割出來的 train data 訓練模型，並拿此訓練模型預測 test data

(6)LightGBM Classifier

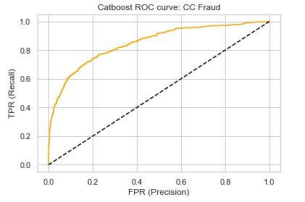
創造 LightGBM 模型，並運用資料切割出來的 train data 訓練模型，並拿此訓練模型預測 test data

▲以上 6 種模型，我們皆嘗試將參數調整以得到更佳的預測結果，但後來發現調參數對於預測效果無顯著增加，因此都使用預設參數值

肆、預測結果分析

結果評估：

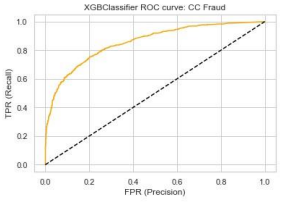
	precision	recall	f1-score	support
0	0.88	0.96	0.92	1916
1	0.75	0.50	0.60	484
accuracy			0.86	2400
macro avg	0.82	0.73	0.76	2400
weighted avg	0.86	0.86	0.85	2400



Area under curve (AUC): 0.8487325091875291

▲catboost

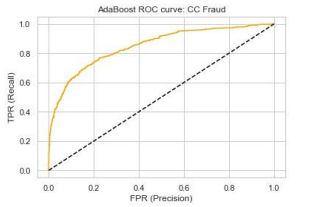
	precision	recall	f1-score	support
0	0.88	0.96	0.92	1916
1	0.76	0.46	0.57	484
accuracy			0.86	2400
macro avg	0.82	0.71	0.74	2400
weighted avg	0.85	0.86	0.85	2400



Area under curve (AUC): 0.8550343777497887

▲XGboost

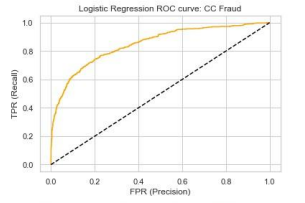
	precision	recall	f1-score	support
0	0.88	0.95	0.91	1916
1	0.70	0.49	0.58	484
accuracy			0.86	2400
macro avg	0.79	0.72	0.75	2400
weighted avg	0.84	0.86	0.85	2400



Area under curve (AUC): 0.8487325091875291

▲adaboost

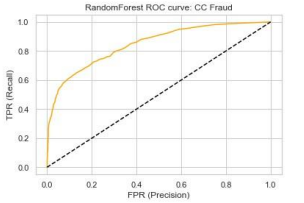
	precision	recall	f1-score	support
0	0.83	0.95	0.89	1916
1	0.57	0.25	0.34	484
accuracy			0.81	2400
macro avg	0.70	0.60	0.62	2400
weighted avg	0.78	0.81	0.78	2400



Area under curve (AUC): 0.8487325091875291

▲logistic regression

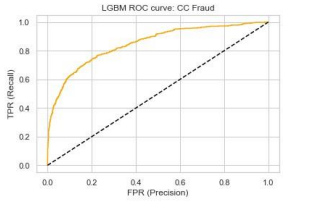
	precision	recall	f1-score	support
0	0.87	0.96	0.92	1916
1	0.76	0.45	0.57	484
accuracy			0.86	2400
macro avg	0.82	0.71	0.74	2400
weighted avg	0.85	0.86	0.85	2400



Area under curve (AUC): 0.846532678272572

▲ramdom forest

	precision	recall	f1-score	support
0	0.88	0.95	0.91	1916
1	0.71	0.50	0.58	484
accuracy			0.86	2400
macro avg	0.79	0.72	0.75	2400
weighted avg	0.85	0.86	0.85	2400



Area under curve (AUC): 0.8487325091875291

▲LGBM

- 以 ROC 曲線&AUC 選擇

  經比較後，得 XGboost 的 AUC 值最高，為 0.855

- 以 f1 score 選擇

  經比較後，得 catboost 的 f1 score 值最高，為 0.6

結論：

一開始訓練模型時，我們查了許多資料，發現網路上很多人會使用 random search 來尋找最佳參數，但根據我們嘗試，效果並無顯著成長，並且會花費許多時間，因此我們皆使用預設參數值；此外，剛開始我們模型跑出來的結果都不盡理想，無法有效抓出 exited=1 的資料，但在使用 smote、去除 outlier 後，測試結果在網站上的結果 accuracy 約為 0.82、precision 與 f1 score 皆約為 0.5，雖有成長，但還不是最好的模型，因此我們有對整個預測方法再做

調整，詳見 report 2

伍、感想與心得

耿皓昀(H24081341)：

這次競賽也是個蠻有趣的經驗，因為在這之前從來沒有參加過任何程式的競賽，但在過程中遇到了許多難題。像是在打程式碼的時候，因為在這個領域還沒有很了解，因此很多模型還有機器學習的方法都要自己上網查，而模型裡有很多參數也不懂，因此要不斷地爬文才能做出一個模型。但最讓我頭痛的還是資料前處理的部分，在我不知道有 oversampling 之前，我就一直不斷的嘗試把變數去掉或是把變數手動分類，但效果一直都不理想，直到發現 SMOTE，之後的成果才漸漸變好。在這個過程中有很多煎熬的時候，多次不斷的嘗試不同的方法但一直都沒有成功，真的很磨練我的毅力和心智。但我也把這當作一個磨練，要走上資科的路，這段陣痛期是必經的過程。最後也謝謝老師給予我們這次的機會讓我們自己摸透資料科學的領域。

施文千(H24086024)：

這個競賽與期末報告是第一次自己做預測，也有更理解之前上課用的各種演算法，光是處理不平衡資料的方法就學到了很多種，還有 outlier 的問題，平常其他統計課程中明明很常跟我們介紹到 outlier，但當開始操作時，卻遲遲沒有想起來，我想這應該是一種不錯的練習，以後不管在任何課程，我應該都不會再輕忽 outlier 的存在了；此外也發現到同一種方式，但對不同資料效果可能會差距甚大，像在期末報告與這次競賽中，我們都有嘗試將年齡做分組，這個動作在報告中效果不錯，但在競賽中效果卻是欠佳，算是一個發現吧！

因為剛開始搞錯組別，網站上 team19、team17 的資料皆為我們上傳的，最後最佳預測結果為 team17 的結果