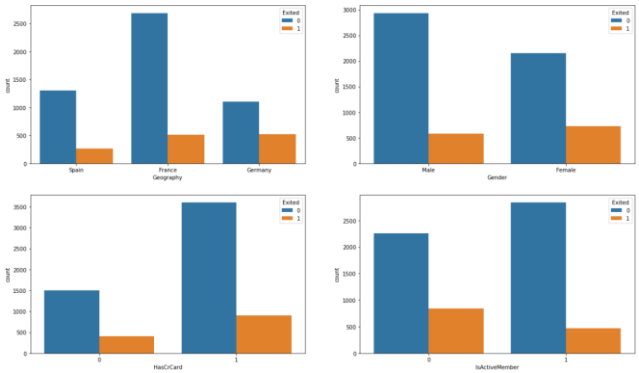
# Competition 1

1. 小組成員：統計113孫亞瑄H24091304、統計113陳亭瑄 H24096150、統計113龍以欣H24094051

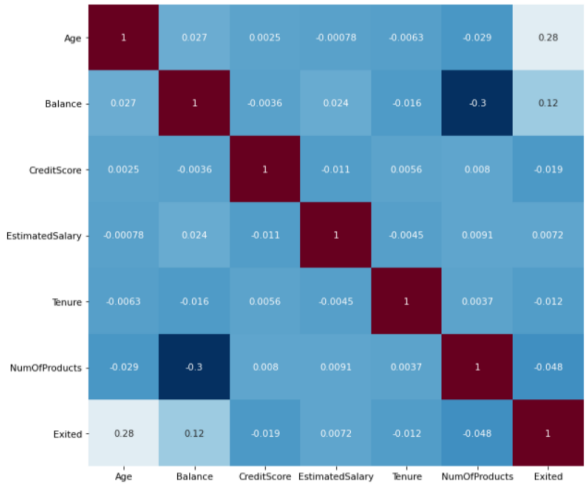
2. 競賽敘述與目標：鑑於數百金融銀行客戶的行為數據，目標是預測客戶最終是否會退出銀行，即將來不再在銀行進行交易。因此這次比賽是透過特徵資料的分析並進行訓練，目標是希望能成功預測出銀行客戶的流失。

3. 資料前處理：訓練數據共有13個特徵，其中一欄是預測目標，即退出欄（1：退出，0：不退出）。
(1)資料特徵縮放:我們使用sklearn.preprocessing的StandardScaler對資料進行標準化，針對CreditScore、Age、Tenure 、Balance 、NumOfProducts、EstimatedSalary的欄位進行標準化。
(2)類別資料的處理:使用sklearn.preprocessing的OneHotEncoder將類別型資料Geography、Gender、HasCrCard、IsActiveMember重新編碼成0和1。
(3)不採用的資料欄位:RowNumber、Customer Id、Surname，我們猜測這三個欄位與要訓練的資料並沒有太大的相關性，因此不採用這三個欄位。

4. 特徵處理與分析：
利用matplotlib和seaborn對特徵進行資料視覺化，觀察各特徵與y(Exited)之間的關係




並利用相關性矩陣觀察各特徵之間的關聯，不過沒有太大的發現。



5. 預測訓練模型：

以下為我們嘗試用過的訓練模型及修改的參數值

sklearn:

(1)RandomForestClassifier：對n_estimators(樹的數量)及max_depth(樹最大深度)進行修改

(2)DecisionTreeClassifier：調整max_depth、random_state

(3)KNeighborsClassifier：調整n_neighbors

(4)SVM 中的 SVC：修改gamma參數

(5)Logistic Regression

(6)MLPClassifier：對solver(‘lbfgs’, ‘sgd’, ‘adam’)、activation(‘identity’, ‘logistic’, ‘tanh’, ‘relu’)、hidden_layer_sizes(隱藏層數量)、random_state、max_iter(迭代次數)、warm_start(重用上一次調用的解決方案以適合初始化) 這些參數進行修改

(7)Gradient Tree Boosting：調整learning_rate、n_estimators、max_depth等參數

others:

(1)XGBoost：對max_depth(最深深度)、learning_rate(學習速率)、gamma(懲罰項係數) 、n_estimators(總迭代次數)這些參數進行修改，另外利用feature_importances_查看各特徵對模型的重要性

(2)CatBoost：調整iterations(總迭代次數)、depth(樹深度)、cat_features(類別特徵的column)、early_stopping_rounds(訓練一定次數若未改進則提前停止)，另外利用feature_importances_查看各特徵對模型的重要性

最終我們選擇針對XGBoost研究，觀察出在max_depth=3的時候對不同切分方式的訓練資料(透過設定random_state)，皆能有較好的表現。再來我們便討論該如何找到更好的參數(eta、gamma等)，最終我們使用gridsearch來尋找參數，也成功地提升了我們的final score。

6. 預測結果分析：

在這個階段，我們自行切割原本的訓練資料，並先套用模型進行預測

```
#Load data
data=pd.read_csv("train.csv")
X=data.iloc[:,0:-1]
y=data.iloc[:,-1]
train_X,test_X,train_y,test_y=train_test_split(X,y,train_size=0.8,stratify=y,random_state=3)
```

而預測的評分標準也是根據競賽網站上有的數值(Accuracy、Precision、Recall、fScore、Final)，分別計算出各個的數值，簡單進行判斷我們套用的模型會不會得到好的結果，再將我們認為可行的結果上傳，最終得到測試分數。

```
clf=XGBClassifier(eval_metric=['logloss','auc','error'],use_label_encoder=False,max_depth=2,eta=0.1,n_estimators=300)
clf.fit(adj_train_X,train_y)
pred_y=clf.predict(adj_test_X)

accuracy=accuracy_score(test_y,pred_y)
precision=precision_score(test_y,pred_y,zero_division=0)
recall=recall_score(test_y,pred_y)
f1 = f1_score(test_y,pred_y)
final=0.3*accuracy+0.3*precision+0.4*f1
print("%.5f %.5f %.5f %.5f %.5f" %(final,accuracy,precision,recall,f1))
print(clf.feature_importances_)
print(clf.n_estimators)
```

```
0.75024 0.87375 0.79808 0.50920 0.62172
```

(1)RandomForestClassifier：

一開始利用此模型進行預測，自己切割的資料的Accuracy可以達到0.86左右，而測試資料的Accuracy可以達到0.8755，若增加n_estimators的數值就會逐漸增加Accuracy，我們認為這是一個還不錯的訓練模型，因此不斷尋找參數想達到最好的成績，但大概就卡在這個數值範圍，因此我們繼續嘗試其他模型

(2)Knn：

不調整任何參數的Accuracy比RandomForestClassifier低很多，因此不再嘗試此模型

(3)SVM 中的 SVC：

測試資料的Accuracy可以達到0.87左右，但因為不太了解gamma參數對此模型的影響，因此我們先保留這個演算法的結果，並繼續嘗試其他的方法

(4)Logistic Regression：

不調整任何參數的Accuracy比RandomForestClassifier低很多，因此不再嘗試此模型

(5)Neural Network：

在sklearn.neural_network官方網站的例子中看到可以將max_iter設置1，並用for迴圈去跑演算法，就可以看到每一次計算結果的好壞，並從中改良演算法。在研究神經網路時，一開始對隱藏層層數進行改變給定層數為3層(50,50,50)，但在跑時，發現演算法一直不能收斂，就改變最大迭代次數，發現在迭代次數為1000時，演算法有收斂了，但是跑出來的結果不如預期的好。神經網路背後的原理和可調整的參數太多，在嘗試後一直找不到規則可循，就不再嘗試這個演算法了。

(6)XGBoost：

我們在比賽的過程中，發現網路上很多人建議機器學習的比賽可以透過XGBoost這個方法來進行訓練，而當我們採用XGBoost最初的預設模型之後，發現整體的分數都有提升，因此我們開始著手研究這個模型背後可以更改的參數。

除了直接修改模型中的參數外，我們也自行使用for迴圈希望能找到最佳的值，但因為效率不太好，最後找到了sklearn.model_selection的GridSearchCV這個方法，也就是網格搜索，這個方法是給定一些參數值的範圍，透過類似for迴圈的概念，不斷組合出最好的參數值。

```
clf=XGBClassifier(eval_metric=['logloss','auc','error'],use_label_encoder=False)

param_dist = {
        'max_depth':[2,3,4],
        'learning_rate':np.linspace(0.01,0.3,30)
        }
grid = GridSearchCV(clf,param_dist,cv = 3)
grid.fit(adj_train_X,train_y)
best_estimator = grid.best_estimator_
best_estimator
```

```
C:\anaconda3\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The use of label encoder in XGBClassifier is deprecated an
d will be removed in a future release. To remove this warning, do the following: 1) Pass option use_label_encoder=False when co
nstructing XGBClassifier object; and 2) Encode your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
```

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, enable_categorical=False,
              eval_metric=['logloss', 'auc', 'error'], gamma=0, gpu_id=-1,
              importance_type=None, interaction_constraints='',
              learning_rate=0.26999999999999996, max_delta_step=0, max_depth=3,
              min_child_weight=1, missing=nan, monotone_constraints='()',
              n_estimators=100, n_jobs=12, num_parallel_tree=1,
              predictor='auto', random_state=0, reg_alpha=0, reg_lambda=1,
              scale_pos_weight=1, subsample=1, tree_method='exact',
              validate_parameters=1, verbosity=None)
```

```
outcome={"RowNumber":test_X["RowNumber"],
         "Exited":pred_y}
df_outcome=pd.DataFrame(outcome)
df_outcome.to_csv("predict_125.csv")
```

而我們這個模型預測出來的accuracy和f-score也是最好的一次成績

| | | | | | |
|---|---|---|---|---|---|
| 125 | 2021-12-17 23:53:57 | predict_125.csv | 0.8925 | 0.765625 | 0.6950354609929078 |

7. 感想與心得：

**龍以欣：**

在這次比賽中，我學會使用sklearn的套件，並將課堂上所教的資料標準化、類別型資料處理運用在比賽中，也在網路中找到關於機器學習比賽的流程建議，這給我一個很大的方向，知道該如何進行一次的比賽和在比賽中需要做甚麼事。我針對Neural Network和XGBoost進行研究，我覺得最花時間和困難的地方是學習演算法背後的原理，在研究Neural Network時，看不太懂演算法的計算原理，使我在用神經網路套件時不知道該從哪個參數下手，所以花了非常多時間，但是最終還是沒有找到其中的規律，最後放棄了這個演算法。在研究XGBoost時，其背後的原理接近random forest，所以在研究原理的過程中，比神經網路順利多了，也比較知道該從哪個參數進行改變。

**陳亭瑄：**

在這次的比賽過程中，我了解到資料前處理是一件非常重要的事情，必須先對資料的內容十分熟悉，才能決定要用那些特徵去訓練，而我和組員也利用不同的方法做前處理(標準化的資料特徵不同、類別型資料轉成數值型採用get_dummies的方法)，經由彼此的互相討論後採取了最好的方法。

我也嘗試了許多不同的sklearn的模型，包括RandomForest、SVC、Knn、XGBoost等不同的模型，我認為套模型並不是一件困難的事，最困難的是要了解其背後的原理，並研究他們各別有哪些參數可以調整，而參數又該如何調整才是最好的，因此在這個部分讓我花了最多的時間，除了手動更改參數值外，我也在查資料的過程中，碰巧學習到了GridSearchCV(網格搜索)的方法，並重新嘗試以上的模型，雖然我對一些參數的定義及該如何調整還不是很熟悉，但至少知道哪些參數是可能會造成影響的，依然有很大的收穫。

另外，我也自行在網路資源中找到將多個模型合併的套件--Combo，可惜是在競賽即將結束的前幾個小時，所以只初步的使用了一下、嘗試多種組合，但並未有足夠的時間深入了解其背後原理，而且成績也沒有先前來的高，因此最終我們還是採取之前XGBoost預測目標的方式。雖然這次的比賽對我來說是很陌生的一件事，因為這個領域也是第一次接觸，但透過競賽的方式，也讓我願意花更多的時間嘗試許多不同的方法，進而提升自己的能力，而且當我們獲得好的成績時，我也得到了滿滿的成就感。

**孫亞瑄：**

在這次的競賽中，我能將從課程學到的方法練習，並自學了許多內容，對我來說是一次寶貴的經驗。

剛開始從拿到資料後，我依照課程教的步驟，先進行資料前處理，將連續型的特徵標準化，類別型的特徵使用one-hot encoding轉換，接著套入RandomForest模型進行預測，就已經獲得了不錯的accuracy(0.87)。

然而想提升各項評估指標並沒有那麼容易，於是我開始上網查許多方法並自行嘗試。首先有嘗試進行資料的EDA，例如利用長條圖觀查各特徵的分布狀況，用相關性矩陣觀察各特徵間的關聯，不過可能因為實力有限沒有觀察出甚麼結果。接著嘗試對特徵進行特徵工程，檢查資料有無缺失值或異常值，也有試著對特徵進行加工，合併兩個特徵組成新特徵，不過也沒發現特別的改變。我們也有利用train_test_split及設定Stratify=y，將訓練資料切分，用sklearn.metric自行計算各項指標，甚至透過Cross Validation的方式，讓我們可以不用經過上傳就能自己對答案，還可以防止overfitting的問題。另外也有利用grid search尋找並調整模型的參數，和使用xgboost中的feature_importances_觀察各特徵在模型學習中的權重，進而發現有些特徵不放入模型學習，反而可以獲得更高的score。

而在模型選擇的方面，我們先後嘗試了非常多模型，其中比較最印象的是神經網路及xgboost。因為我覺得神經網路很酷，所以我嘗試使用MLPClassifier，然而其中的超參數實在太多，包括layer及node的數量，solver的選擇，learning_rate及max_iter等等，太過複雜且找不出規律，已經花了太多的時間，因此之後我便想嘗試其他方法，而很幸運的我嘗試了xgboost，並得到了不錯的成果。

在xgboost中，我們調整了許多的超參數，首先發現max_depth在2和3的時候出來的accuracy會較好，可能是太多層就會出現overfitting，接著調整eta(learning_rate)，發現在不同eta下模型學習效果差很多，而我也嘗試網路上有寫到調低eta且調高n_estimators的方式讓模型學習，以及試著調整gamma的值，最終我們也使用xgboost做出成績最好的一筆預測。

從開始的randomforest，花我超級多時間但沒甚麼進展的neural network，及自行嘗試獲得好成績的xgboost，還有可以自行轉換類別型變數的catboost，其他許許多多如decision tree、LogisticRegression、KNN…等等，許多的模型都讓我印象深刻。而我也很湊巧的查到Ensemble Learning的相關知識，了解到原來有bagging、boosting和stacking這三種把多個模型合併的學習方式，甚至自己找來了combo的套件嘗試進行stacking，將多個模型合起來，雖然沒有獲得比之前更好的成績，不過蠻好玩的，而這些都使我印象深刻。

在這次的競賽中，我花了許多時間自學了許多知識，原本只是想在競賽得名，雖然現在名次還沒公佈，但我認為我獲得的已遠遠超過了名次，能夠和組員一起討論及嘗試，認真得投入競賽，學習並獲得成長，是非常開心的一件事。而我也在自學的過程中了解還有許多可以學習的知識以及自己的不足，希望再之後能繼續朝資料科學前進!

# Git & Github

陳亭瑄：
Repository：https://github.com/thchen0116/DataScience-HW
Pages：https://thchen0116.github.io/DataScience-HW/HW3/html/HW3.html

龍以欣：
Repository：https://github.com/LungIHsin/Data-Sicience-HW
Pages：https://lungihsin.github.io/Data-Sicience-HW/HW3_Website/first_website.html

孫亞瑄：
Repository：https://github.com/arthur21508/DataScience-HW
Pages：https://arthur21508.github.io/DataScience-HW/H24091304_hw3/hw3_HTML.html