玉山人工智慧公開挑戰賽

隊伍：不做Feature Engineering 是一件很嘻哈的事情
成員：吳懿峰、李得瑋、譚丞佑、陳冠宇、徐嬿鎔、陳鵬宇、吳睿得、林偉捷

★摘要

本組完整建模流程與方式，大致分為以下五點 :

一、**EDA:** 發現test set大部分的卡號皆未在train set出現，因此後續將分開處理。發現刷卡金額的分布在train/test set極其相似，因此對金額著重處理。

二、**特徵工程 :** 使用了欄位之間統計數據(avg, std 等)、欄位之間倆倆交互關係、graph embedding，value count by time window等方式建立特徵欄位，將high cardinality categorical features進行encoding

三、**模型建立與挑選 :** 主要使用 LightGBM (Light Gradient Boosting Machine)、 XGB、DNN(pytorch)等模型，最後根據 validation f1 score 挑選表現較好的模型。

四、**模型細節處理:** 根據cross validation找尋threshold最佳值(選取平均分數高且變異數低之閥值)，並透過custom loss function加重懲罰 false positive。

五、**Ensemble :** 根據各模型的 Precision / Recall 等特性制定策略，將多種模型融合並得出最後的預測。

★環境

[請說明本次比賽所使用的系統平台、程式語言、函式庫]

系統平台：Jupyter Notebook, VScode
程式語言： Python
函式庫： numpy, pandas, lightgbm, scikit-learn, gensim, stellargraph, networkx

★特徵

[請說明本次比賽所使用的特徵]

1. 小時
2. Count by bank: 在相同銀行帳戶下，消費過不同'acquirer', 'coin', 'mcc', 'shop', 'city', 'nation', 'status', 'trade_cat', 'pay_type', 'trade_type', 'hour', 'fallback', '3ds', 'online', 'install', 'excess' 的次數。
3. Count by bank (two columns): 在相同的銀行帳戶以及以下任一欄位下'acquirer', 'card', 'coin', 'mcc', 'shop', 'city', 'nation'，消費過不同'acquirer', 'card', 'coin', 'mcc', 'shop', 'city', 'nation', 'status', 'trade_cat', 'pay_type', 'trade_type', 'fallback', '3ds', 'online', 'install', 'excess', 'hour' 的次數(不包含相同名稱的欄位)，eg. Count the number of times a card used under same bank-acquirer.
4. Count by two columns: 透過Group by 任兩個以下欄位['status', 'trade_cat', 'pay_type', 'trade_type', 'fallback', '3ds', 'online', 'install', 'excess', 'hour', 'acquirer', 'card', 'coin', 'mcc', 'shop', 'city', 'nation'] 計算其消費次數。
5. 每日消費次數: 不同的'acquirer', 'bank', 'card', 'coin', 'mcc', 'shop', 'city', 'nation'的每日消費次數。
6. 消費金額與不同的'acquirer', 'bank', 'card', 'coin', 'mcc', 'shop', 'city', 'nation'進行統計，如平均、中位數、最大值、變異數等
7. 不同的'acquirer', 'bank', 'card', 'coin', 'mcc', 'shop', 'city', 'nation'下累積的消費金額。
8. Embedding:
    A. Graph embed: 以每天的交易，創出 'bank' 與 'acquirer', 'bank', 'card', 'coin', 'mcc', 'shop', 'city', 'nation' 的 bipartite graph，再完成只有 'acquirer', 'bank', 'card', 'coin', 'mcc', 'shop', 'city', 'nation' 的 homogenous graph，並以此生成 walk，喂進 gensim Word2Vec model
    B. Embed other targets: 對預測力好的欄位 'trade-type' 'money' 'online' 進行預測，使用 Neural Network，並將 'bank' 'acquirer', 'bank', 'card', 'coin', 'mcc', 'shop', 'city', 'nation' 先喂進 embedding 層，最後取出 embedding 做特徵使用

★訓練模型

[請說明本次比賽所使用的訓練模型、參數]

本次比賽最終預測值參考了種不同模型的結果:

1. Same card Lightgbm:
   i. 模型: Lightgbm
   ii. 參數: 'objective': 'binary', 'learning_rate': 0.01, 'reg_alpha': 0.5, 'reg_lambda': 0.5, 'max_depth': -1, 'num_leaves': 200, 'seed': 6, num_boost_round=4000, verbose_eval=50, feval=lgb_f1_score
2. Different card Lightgbm:
   i. 模型: Lightgbm
   ii. 參數: 'objective': 'binary', 'early_stopping_rounds': 100, 'learning_rate': 0.01, 'reg_alpha': 0.5, 'reg_lambda': 0.5, 'max_depth': -1, 'num_leaves': 150, 'seed': 44, 'metric': ['auc', 'binary_logloss'], num_boost_round=3000, verbose_eval=50, feval=lgb_f1_score
3. Focal Lightgbm: 首先根據以下公式制定Custom Loss Function

$$\mathrm{FL}(p_t) = -\alpha_t(1-p_t)^{\gamma}\log(p_t)$$

   i. 模型: Lightgbm
   ii. 參數: 'objective': 'binary', 'early_stopping_rounds': 100, 'learning_rate': 0.01, 'reg_alpha': 0.5, 'reg_lambda': 0.5, 'max_depth': -1, 'num_leaves': 150, 'seed': 44, 'metric': ['auc', 'binary_logloss'], num_boost_round=3000, verbose_eval=50, feval=focal_loss_lgb_f1_score, fobj=focal_loss, alpha=0.5, gamma=1.

★訓練方式及原始碼

原始碼

連結: https://github.com/ugotsuyokunaru/FraudDetection

訓練方式:

1. 先找出在Training set 以及 Testing set 皆出現之Card number，並將其儲存於一個list，令其為same_list。
2. 製作3種不同的模型:
   i. 將 Training set 以每30天做切割，得到三個training subset，再將每一個subset分別進行feature engineering。使用Lightgbm進行模型訓練，並令其為lgbm_diff 資料表。
   ii. 不對 Training set 進行切割，用原始 Dataset 逕行做Feature engineering，再將此Dataset用Lightgbm進行模型訓練，並令其為lgbm_same 資料表。
   iii. 同i之處理，然在使用Lightgbm進行模型訓練時，使用自定義之Focal Loss以增加懲罰項權重，並令其為Focal 資料表。
3. 模型整合: 當Focal 判定為1時，預測結果為1；當lgbm_diff判定為1且其預測項目之txkey不在same_list，預測結果為1；當lgbm_diff判定為1且其預測項目之txkey在same_list，預測結果為1，其餘則皆預測為0。

★結論

從一開始的Baseline model到最後的預測結果，成長最多的有三個時間點，一是將high cardinality columns進行交互關係的encoding，二是將high cardinality columns進行graph embedding降維，三是將不同模型融合。

有趣的是，我們發現在testing set 裡的許多 card number, bank number 或是 shop 等，在training set 都是沒見過的，因此我們在做embedding時，是針對都有出現的類別欄位當作目標值，如 online, money等。而這樣的發現也促成我們訓練不同模型的想法，將完整的資料集拿來預測相同卡號的人，將每30天切割的資料集拿來預測不同卡號的人，得到了不錯的預測效果。

再來麻煩的一點是這個資料集極端不平衡，本來我們想透過

oversampling的方式平衡資料，然而當我們增加到400個以上的變數時，便因為資料量太大而放棄了這個想法，轉而從修改Loss Function以調控懲罰權重下手，並使用Focal Loss來實踐我們的假設，得到一個False Positive Rate很小的一個模型。

而最後，透過這三個模型的融合，幫助我們得到了第九名的成績。本次參賽過程真的很累，團隊成員每個人都有自己要忙的事，因此每個人都只能犧牲空閒時間抑或睡眠來推進我們比賽的進度。再加上資料量非常龐大，因此到了比賽尾端已經很難在有餘力繼續試驗我們的假說，實在是有些可惜。但仍然感謝玉山銀行以及趨勢科技合辦這樣一個有趣的比賽，也讓我們在一邊實做的過程中一邊成長。