金融科技期末專題

第九組

個人投資風險管家

書面報告

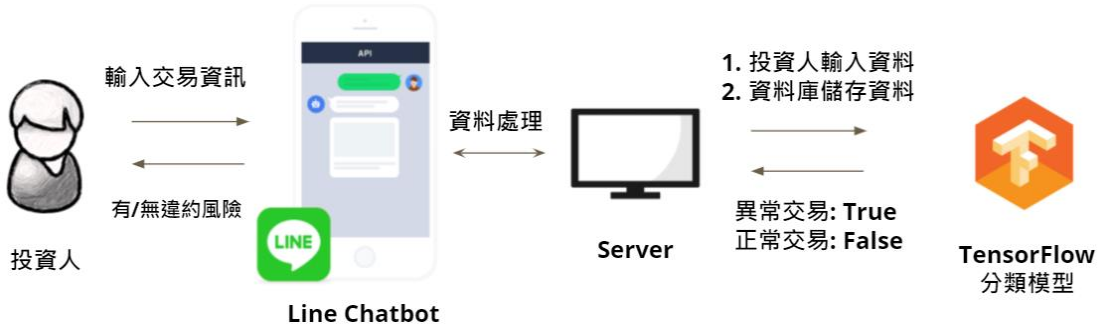
R09922129 詹鈞凱
R09631003 鄭子揚
R09631017 吳少云
R09631033 陳玠宏
R09631050 詹閎棋

● 問題定義

資訊透明化的現代，股票資訊在網路上非常容易地取得，而在 2019 年爆發新冠肺炎後越來越多人居家工作，再加上近年來股票市場的熱絡，因此越來越多人投入股票市場。投資新手們可能一開始交易股票確實會謹慎小心，然而在股票市場獲利之後野心開始變大，以至於有可能會因為看到股價波動性高的股票而直接投入，卻忘記帳戶的錢還有多少，進而造成違約交割的風險；或是有人是因為操作失誤，如想買零股卻下單下成整張的單位；因此，本專案希望能給予投資客在下單時可以檢查的一項工具，預期系統能給予投資客目前下單某項標的風險，當作是投資客一項評估投資風險的管家。

● 方法描述

使用深度學習進行建模，投資客想買新股票時可使用此系統先在 Line Chatbot 對話欄裡輸入投資人代碼、股票名稱、交易股數、交易類型，Chatbot 會將此資訊回傳至伺服器後並讓模型分析此次交易與過往所有投資人的交易行為相比，預測發生違約的機率高低，得到的結果再透過 line chatbot 回傳至使用者。圖一為我們研究方法的描述。

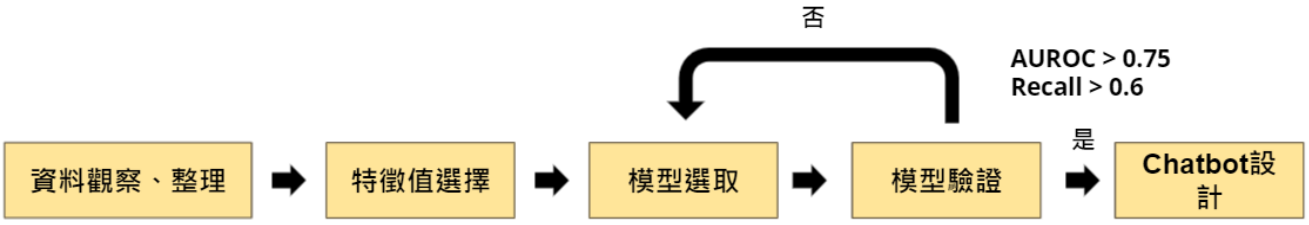


圖一、研究方法描述

● 資料集

資料集共分為四大筆，其來源皆自玉山證券提供。第一筆至第三筆資料為 2019 Q1 至 2021 年 Q1 內股票每日資訊，如開盤、收盤價、當日最高、最低價格.....等、所有投資人之交易紀錄和是否違約的標計、所有投資人的交易紀錄，包括：股票代碼、交易方式、交易價格及交易數量等，第四筆資料為台北股市股票產業種類。其中，投資人資料中的違約標計其有違約的人之紀錄有 97 筆，未違約之人數為 10 萬筆，其資料嚴重失衡。

- **實驗的設計**



圖二、實驗路線圖

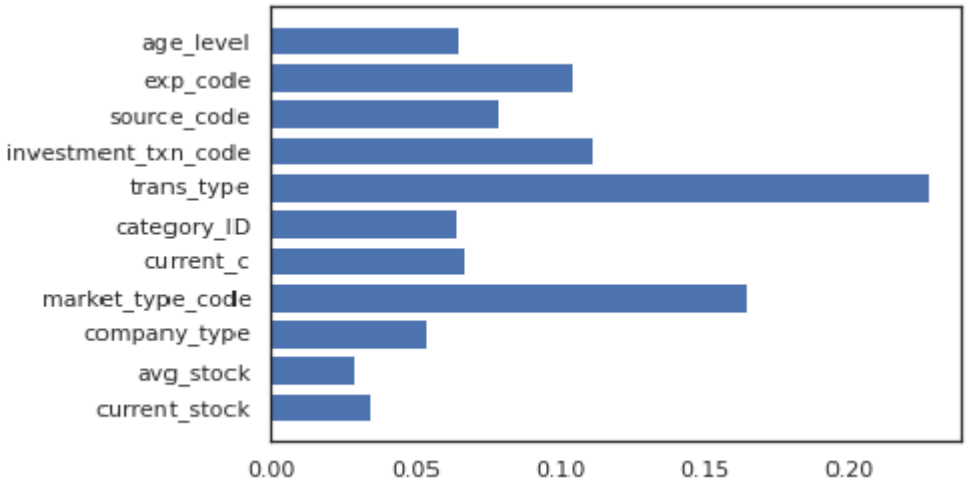
- 資料集的清洗和處理
  - 本專題中如何使用這個資料集
    - 由於玉證的資料集提供的有、無違約資料極度的不平衡，大約是 1:1000 的比例，因此在無違約的 100,000 名投資人中隨機挑選 2960 位，與有違約的共 97 位投資人，找到他們從 2019 年 Q1 到 2021 年 Q1 的所有交易資料做統整，記錄他們過去的交易模式，例如: 投資客本身的年齡區間、玉山的交易經驗年份、以及操作股票的手段(像是當沖或現股買賣為主等)、交易股票之公司的類別、每次交易的平均股數、平均交易金額等特徵，以及此投資客是否曾經有違約之紀錄。

  - 如何抓出 outlier
    - 在有違約的 97 筆資料裡發現有有四筆資料是找不到任何交易紀錄的，對於這四筆資料因沒有任何交易紀錄存在，但卻直接被標記違約資料這是無從考證的，因此將這四位投資者特別抓出並踢除；此外對於二次違約的紀錄，對於投資客來說第一次違約時已經造成信用的毀壞，而站在證券業者的角度，當投資客有第一次違約時就會對其採取限制帳戶等行動，因此第二次違約的意義性並不大，對此，我們將二次違約的紀錄從資料集中去除，因此只留下具有交易紀錄且第一次違約的資料當作違約資料集。

  - 如何處理缺值：如何進行補值
    - 客戶違約標記的資料中，前三位違約投資人因違約日不在玉證所提供的交易紀錄中，因此發生了缺值的問題。因應投資習慣的特性，我們採用違約投資最後一筆「買入」的交易紀錄進行補值，此包含購買股數、交易類型、公司類型與公司規模。

- 特徵的處理
  - 特徵正規化：
    - 對於資料集中 "平均交易股數"、"當日最後一筆交易股數" 等特徵，其分布相當廣泛，因此在這裡將其取 Log，轉換成常態分布的收斂形式。

  - 特徵選取或萃取：
    - 在原本資料集中具有投資客名稱、年齡層、玉證交易經驗、開戶別、交易類別(如現股、當沖、融資、融券、借券、現股當沖)、交易公司產業類別(電子、傳產、食品、金融股、航運、其他、No Data)、交易之公司種類(上市、上櫃、興櫃)、平均每次交易股數、當天之"最後一筆交易股數"、平均交易金額，"當日最後一筆交易金額"，共 27 項 Features，我們使用 XGBoost 演算法進行特徵篩選，使用其提供之套件輸出各特徵影響結果之權重，除了剔除掉幾個相關性過低的幾樣特徵，我們也觀察到某些特徵太過強烈，會導致模型在判斷時會有 overfitting 的情形，例如:電子類股的交易非常熱門，因此可能有違約交易的可能性也較高，選擇此特徵可能造成模型會把所有投資電子類股的投資人皆歸類到違約交易中。另外我們統計此資料集中所有投資人過去的所有消費習慣，對投資上市上櫃公司的次數和各產業別的次數進行加總當作模型依判斷特徵，目的是透過分析哪種偏好的投資習慣會造成違約的風險高低程度。圖三是我們最終採納的特徵對模型輸出的影響程度，表一則是各特徵輸入模型的資料型態以及代表之意義。

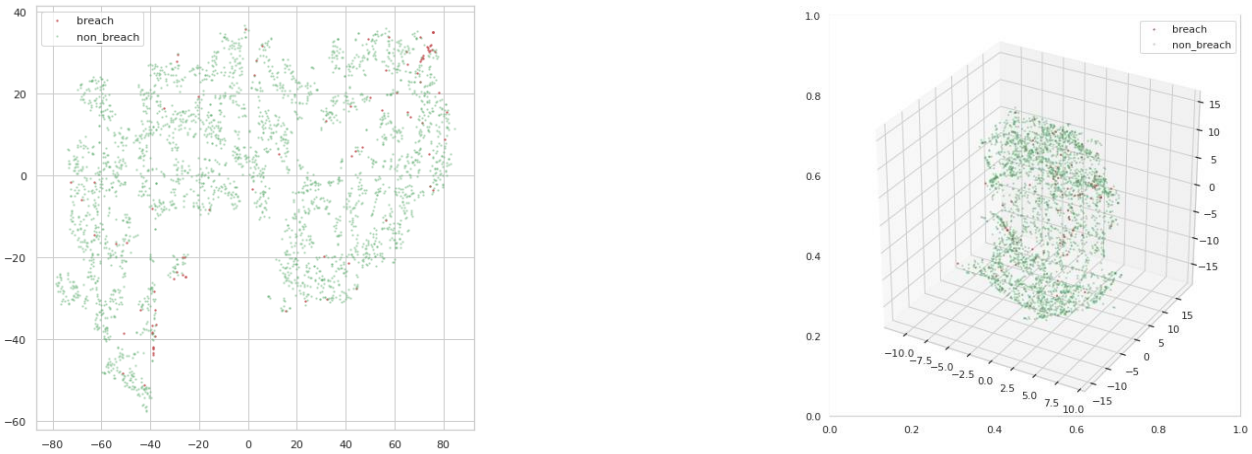


圖三、特徵權重

表一、篩選後特徵的資料型態及含意

	特徵名稱	資料型態	備註
投資人輸入	投資者姓名	string	轉換成顧客代碼
	交易股票名稱	string	轉換成7種類股(current_c) 0: 電子, 1: 傳產, 2: 金融, 3:食品生技, 4: 航運, 5: 其他, 6: No data
	交易股數(current_stock)	int	此交易的股票數量
	交易類型(trans_type)	int	0: 現股, 1: 融資, 2: 融券, 3: 現股當沖, 5: 借券, 9: 當沖
資料庫中資訊	年齡分布(age_level)	int	以年齡區間區分1~6之代碼（玉證未提供詳細說明）
	交易經驗代碼(exp_code)	int	0: 未滿一年, 1: 1~2年, 2: 2~3年, 3: 3~5年, 4: 5~10年, 5: 10年以上
	開戶別(source_code)	int	1: 富國戶, 2: 一般開戶
	累積交易別	int	該投資人過去累積各交易類型的次數
	累積市場別	int	該投資人過去累積投資的公司類型

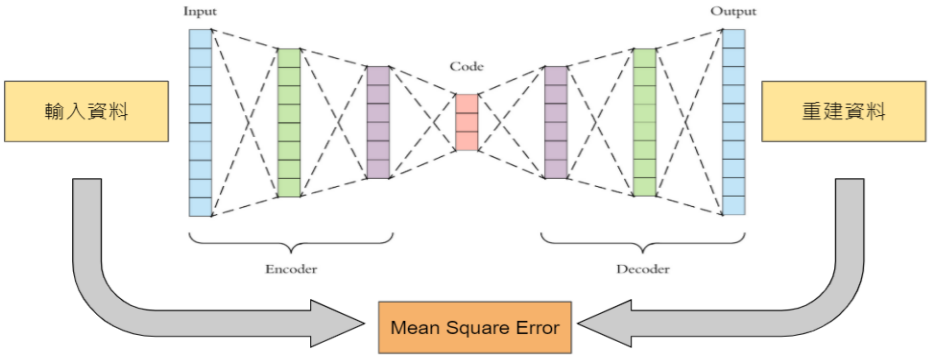
- 資料的可視化呈現（visualization）
  - 使用 t-SNE(t-Distributed Stochastic Neighbor Embedding)，降維方法來顯示有無違約的資料落點，其原理為在高維度的資料點與低維度的資料點的相似度以高斯分布與機率的方式來定義，並最小化高維度與低維度資料機率的 KLD 散度。我們使用此工具將資料投影至 2 維和 3 維的空間(如圖四所示)，達到資料視覺化的呈現，可以直觀地看出資料的分布。



圖四、有無違約資料之分布圖

- 模型選擇
  - 使用 auto-encoder 當作模型的架構(如圖五和圖六)，其優點是 output 的重建向量可以越接近 input 原始向量，而 input 資料使用的皆為正常交易的資料，所以若有一筆違約資料進入 pre-trained model，模型會較難以去重建出接近原始資料(原因是 weights 都是

由未違約資料所訓練而成），因此 loss 就會偏高，進而判斷出異常行為的發生。此模型裡我們使用的 optimizer 為 Adam， loss 為 mse， Batch-size 設為 128, Epoch = 650，但 Epoch 在訓練過程約至 300 左右會因 loss 無法下降而 early stop(如圖七之一和圖七之二）。

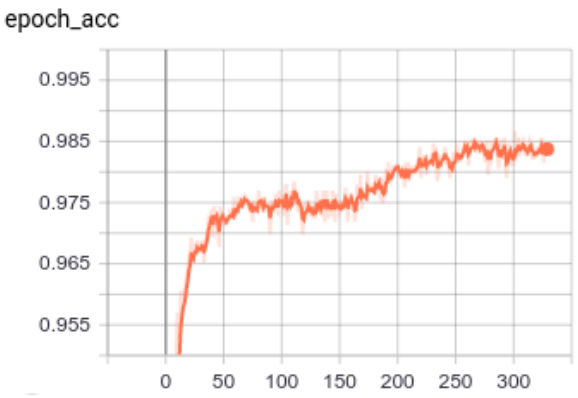


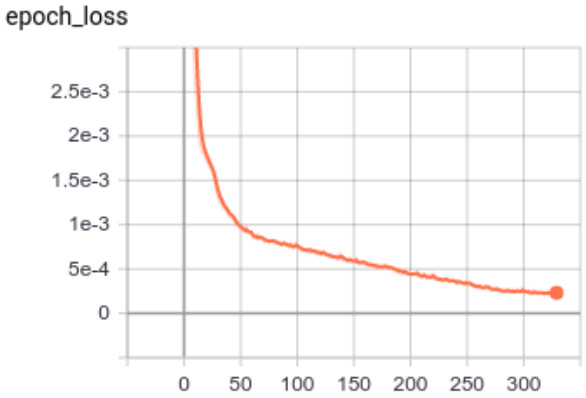
圖五、AutoEncoder 之架構

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 14)	210
dense_1 (Dense)	(None, 128)	1920
dense_2 (Dense)	(None, 64)	8256
dense_3 (Dense)	(None, 32)	2080
dense_4 (Dense)	(None, 16)	528
dense_5 (Dense)	(None, 8)	136
dense_6 (Dense)	(None, 16)	144
dense_7 (Dense)	(None, 32)	544
dense_8 (Dense)	(None, 64)	2112
dense_9 (Dense)	(None, 128)	8320
dense_10 (Dense)	(None, 14)	1806

Total params: 26,056
Trainable params: 26,056
Non-trainable params: 0

圖六、AutoEncoder 之參數





圖七之一、 模型訓練過程的準確度　　圖七之二、 模型訓練過程的 loss

- 訓練資料的劃分
  - 我們分開成三種資料集，各自對應到 Training、Validation、Testing。而在 Training、Validation 的部分因為我們採用 Auto-Encoder，所以我們先將無違約的資料以 8:2 的比例切出，其中 8 成的無違約資料再以 8:2 的比例區分成 Training 和 Validation；至於剩下的兩成未違約資料再加上有違約的 91 筆資料來當作 Testing(如圖八所示)，全部資料數量如圖九所示。

```
Our testing set is composed as follows:
0    520
1     91
Name: label, dtype: int64
```

圖八、 測試資料含有兩成未違約資料及 91 筆違約資料

```
Shape of the datasets:
    training (rows, cols) = (1953, 14)
    validate (rows, cols) = (489, 14)
    testing  (rows, cols) = (611, 14)
```

圖九、 全數資料之劃分情況

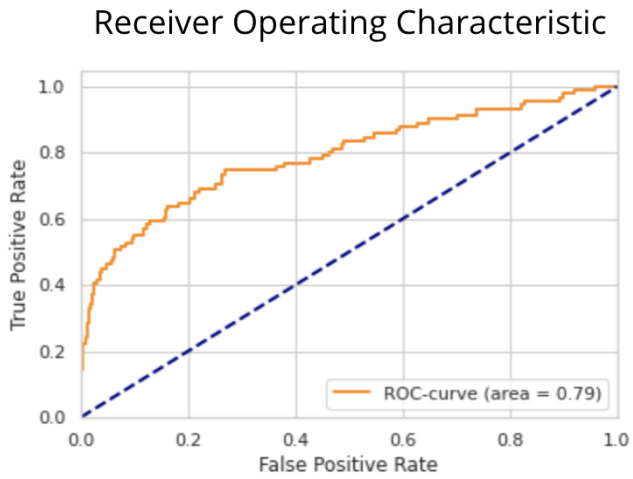
- Line Chatbot 研究
  - 使用 Line deveiop 建立 Line Messaging API，將專屬的 Channel access token 與 Chanel sercet 貼到新建立的 congig.ini
  - 使用 socket，與模型建立 TCP 雙向通訊
  - 使用 Python 製作客製化的對話，針對使用者傳入的訊息 decode 成模型期望的輸入，藉由 socket 傳送到模型 server；當模型 server 回傳結果時，在將此結果 encode 為使用者期望收到的形式，傳回給使用者端。將此 py 檔與 config.ini 放在同一層位址。
  - 為了使內網伺服器與外界網域溝通，需要讓本地端對外開放 server，這裡使用的方式是藉由 ngrok。ngrok 是一個反向代理，通過在公共的端點和本地運行的 Web 服務器之間建立一個安全的通道。ngrok 可捕獲和分析所有通道上的流量，便於後期分析和重放。反向代理在計算機網絡中是代理服務器的一種。
  - 接上 Webhook, Webhook 是一種通過通常的 callback，去增加或者改變 web page 或者 web app 行為的方法。這些 callback 可以由第三方用戶和開發者維持當前修改和管理，而這些使用者與網站或者應用的原始開發沒有關聯。

- 設計 Line Chatbot 的使用介面，製作視覺化選單

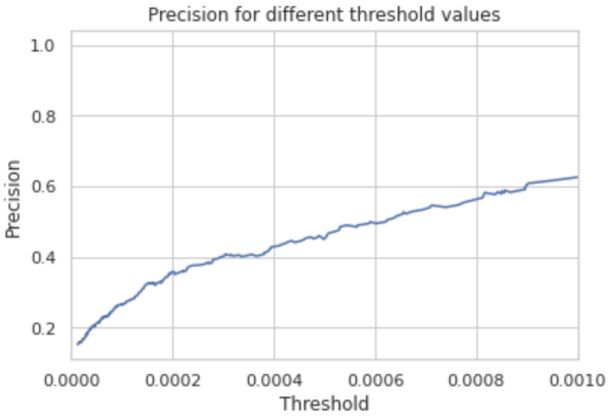
● 實驗結果及討論：

○ 模型結果

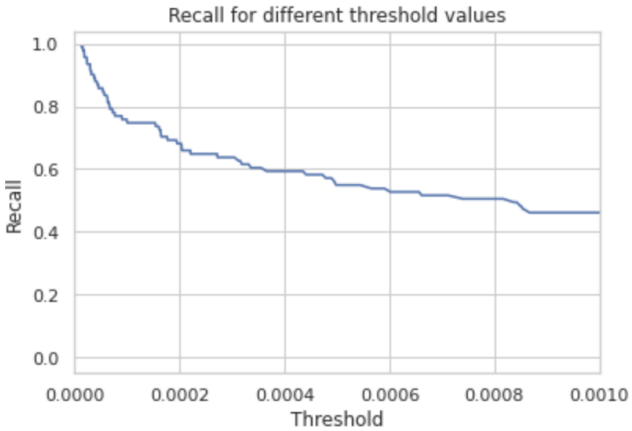
- 將 Testing data 經過訓練過的 Auto-encoder 後，以 ROC Curve 來評估模型的可靠度(如圖十)。再藉由 Precision 與 Recall 對 Threshold 的作圖（如圖十一、圖十二）決定 Threshold 的設置以得到表二。由圖可得到當模型的 Threshold 為 0.0003 時，Recall 為 63.74%而 Precision 為 40.00%。



圖十、模型的 ROC curve



圖十一、 Precision v.s.Threshold



圖十二、Recall v.s Threshold

表二 :預測結果的混淆矩陣

	Predicted Non-Breach	Predicted Breach
True Non-Breach	443	87
True Breach	33	58

- 錯誤案例分析
  - 我們統計有 33 筆違約資料未被模型正確判斷為異常，我們將它分為以下兩種情形:
    1. 人為觀察交易細節亦難以觀測是否異常，可能需要對該投資人過去的交易資訊做客製化的分類模型才能找出異常資料
    2. 交易情形明確有異常，但是模型沒有發現。這個部分可能要透過調整模型參數、演算法才能提升判斷精準度(如圖十三)。

avg_stock	current_stock
27715.78947	47000

avg_stock	current_stock
794.6666667	1000

圖十三、最後一次交易股數明顯比平均交易股數還高，模型卻未發現

○ Chatbot 實際畫面

■ 圖十四為 line chatbot 的實際操作畫面，以下將詳述棋實際操作流程。首先加入個人投資風險管家的 lin ID。加入後可看到下方有輸入說明和輸入範例的按鈕。點選輸入說明會自動跳出一訊息提示輸入格式，點選「投資人」、「公司」和「交易別」按鈕有多個輸入選項可供替換。對個人投資風險管家輸入投資人姓名、投資股票標的、預成交股數和交易方式之後，管家會依據模型的回傳值顯示此次交易違約與否，如果違約的話，會顯示「！！！此交易偵測為異常，請再次確認！！！」；若無違 約則會顯示「此交易未偵測到異常，祝您交易愉快！」。

圖十四、Chatbot 實際操作介面→

無違約結果

有違約結果

輸入說明及輸入範例

● 結論及展望

○ 結論

透過此專案，投資人可以使用此個人管家系統進行風險評估，每當要下單時若想提升交易金額、或是想嘗試不同的交易型態而擔心有風險時都可以先在 Line 上輸入自己想下單的股票名稱、股數、交易型別，此系統將自動回傳有無可能違約的預測，達到避免因交易金額上的疏忽或不了解交易型態而有違約交割的風險，雖然此系統之 Precision 尚有進步空間，但對於投資者來說可以多加留意一番，達到下單前再次確認的目的。

○ 未來展望與改進方向

1. 將訓練後的模型直接與玉山證券 APP 串接，讓投資人下單時可直得到預測結果

2. 根據該投資人當前的交易與其過往在玉山證券所有交易經驗做比較

，透過 oneclass-svm 做異常值偵測，並與我們所做的模型做交叉比對，

可提高判斷違約的準確性

3. 投資人輸入新的一筆資料之後，將此交易資料當作新的訓練資料建檔在資料庫中，讓模型可以做滾動式的修正，讓模型學習可以更加準確。
4. 做當日股票熱度分析，給予每筆交易多一項指標
5. 使用 RNN，將交易的時間序納入考慮，將順序影響違約的可能性納入模型評估指標
6. 若過去從未於玉山證券有過交易經驗，系統接受投資人的下單時搜尋不到此人過去的信用紀錄，因此要高機率回傳異常偵測，以避免違約發生的可能

- 附錄
  - 參考文獻：
    - https://www.kaggle.com/robinteuwens/anomaly-detection-with-auto-encoders
    - Credit Card Fraud Detection using Autoencoders in Keras — TensorFlow for Hackers (Part VII)
    - https://jason-chen-1992.weebly.com/home/-autoencoder
    - https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html
    - https://developers.line.biz/zh-hant/services/bot-designer/
    - https://ithelp.ithome.com.tw/articles/10192928?sc=hot
    - XGBoost Documentation — xgboost 1.5.0-dev documentation
  - 原始碼：
    - https://github.com/tabowsy/fintech_project
  - 小組開會記錄：
    - HackMD：https://hackmd.io/hWgYN_TfRheM-0Mlf5AUQQ?view