台灣ETF價格預測競賽

設計文件

隊伍: **NCU\_newbie**

成員：[陳廷睿]、[劉亞昇]、[連丞宥]、[曾翊銘]

# 摘要

本次比賽使用特徵除了從主辦方提供的資料集中計算KDJ技術指標、行為偵測指標、也將原始價格資料進行標準化、同時抓取三大法人交易資料標準化後當成特徵。

使用的演算法有 Multi-Task Learning, Convolutional-LSTM, Residual Convolutional-LSTM, Attention-Based LSTM, Simple LSTM, CNN, Ensemble Learning(Weighted Majority Voting).

模型訓練的方式根據模型分為只使用目標18檔的資料，以及使用全部資料的訓練。

# 環境

Ubuntu 16.04.4 LTS (GNU/Linux 4.4.0-124-generic x86\_64)

Jupyter Nootbook

Python: PyTorch v0.3.1, Pandas, numpy, requests, TA-Lib, TensorFlow v1.4.0, keras

# 特徵

* Simple LSTM-1, CNN: 每一檔ETF的特徵皆為各自的成分股以及本身之每日收盤價、最高價、最低價、成交張數。
* Multi-Task Learning, Convolutional-LSTM, Residual Convolutional-LSTM: 轉化原始的每日收盤價映射到高維空間，透過計算與6, 11, 22, 43, 65, 130 MA的差距比例，每日成交張數也使用同樣方法映射到高維空間，同時計算傳統KDJ技術分析指標、訊號偵測指標，以及爬取三大法人的交易資訊做squashing後，連接上述所有作為特徵。
* Simple LSTM-2(劉亞昇 v1), Attention-Based LSTM(劉亞昇 v2): ETF-18中的每日開/高/低/收/成交量和三大法人處理過後，為六個欄位分別是：投信買進股數，投信賣出股數，外資買進股數，外資賣出股數，自營商買進股數，自營商賣出股數。

# 訓練模型

* Simple LSTM-1:   
  透過資料前處理將特徵分為各檔ETF之成分股的每日收盤價、最高價、最低價、成交張數進行basic LSTM model training，而 ground truth 則是將各檔ETF收盤價轉換為漲跌幅作為訓練的label。  
  訓練的模型分為兩種，每一種都有18檔ETF各自的models，而各models之間只有LSTM cell number以及特徵數量存在差異。第一種模型為透過18檔ETF各自的成分股之每日收盤價、最高價、最低價、成交張數進行自己的股價預測，time step為10天，一次預測後五天的漲跌幅。第二種模型則是將漲跌與股價分開來預測，股價部分同第一種模型，而漲跌部分則是透過第一種模型的特徵，label則是以每日為漲(1)或是跌(-1)進行預測，透過basic LSTM，activation function為relu。
* Simple LSTM-2(劉亞昇 v1):   
  使用架構為3層神經網路，第一層為普通的隱藏層神經元(來提取特徵)，第二層為RNNcell，第三層為一個全連接層(dense)。
* Attention-Based LSTM(劉亞昇 v2):   
  基於Simple LSTM-2的修改，主要修改部分是加入Attention-based的機制，網路架構從3層改為4層，第一層一樣為普通的隱藏神經元，第二層為RNNcell，第三層隱藏層神經元來處理RNNcell每個timeStep出來的output，第四層為一個全連接層(dense)。
* CNN: 透過18檔ETF各自的成分股之每日收盤價、最高價、最低價、成交張數進行自己的股價預測，input為 前10天的資料，output為後五天的預測漲跌幅。Label為與前一天相比之漲跌幅，我所採用的模型架構非常簡單，包含兩層卷積層，與一層全連接層並且使用softplus作為激勵函數。
* Multi Task Learning:   
  取九天的特徵預測未來五天的資訊，先使用所有的股票的資料集使用Adam優化器，LR=1e-3，loss function 用比賽的得分計算公式，train 3～5天，然後再用目標股票的資料集fine tune 半天，存下模型參數。
* Convolutional-LSTM, Residual Convolutional-LSTM：  
  取九天的特徵預測未來五天的資訊，先使用所有的股票的資料集使用Adam優化器，LR=1e-3，loss function 用比賽的得分計算公式，train 半天，存下模型參數。
* Ensemble learning:   
  取所有模型在過去test的得分能力，作為權重，合併出最後的預測結果。

# 訓練方式及原始碼

* Simple LSTM-1: <https://github.com/littlelienpeanut/ETF_prediction>   
  原始碼包含資料前處理部分(步驟為README.md)以etf\_pred\_value.py(預測股價)、etf\_ud\_pred.py(預測漲跌)兩部分。
* Simple LSTM-2(v1): <https://github.com/NepTuNew/TBrain-ETF>  
  這部分只使用了ETF-18的資料，將資料透過前處理，處理成差值，一個input data由時間序列所組成的5個維度(開/高/低/收/成交量)的資料，假如timeStep為5，那則需要6天的歷史資料來處理成各維度的差值，再丟到神經網路中訓練。
* Attention-Based LSTM(v2): <https://github.com/NepTuNew/TBrain-ETF>  
  這部分使用了ETF-18之外還有對應的三大法人資料，三大法人經過處理後為6個維度的資料，加上原本的5個維度，一樣處理成差值，所以這個版本的輸入會是一個11個維度的時間序列，再丟到神經網路中訓練。
* CNN:   
  原始碼包含資料前處理部分(data\_processing\_r)以及模型訓練與預測(model\_pre)，與用於模型合成的結果整理(merge\_result\_shift)。
* Multi-Task Learning, Convolutional-LSTM, Residual Convolutional-LSTM: [https://github.com/Ray941216/TBrainETF/](https://github.com/Ray941216/TBrainETF/tree/master/TBrain) （在TBrain folder下）  
  讀取pre-trained base 對每檔股票近一年的資料各自fine tune 幾個epoch 後，預測並輸出結果與local model參數作為次週預測的pre-trained base。
* Ensemble Learning: [https://github.com/Ray941216/TBrainETF/](https://github.com/Ray941216/TBrainETF/tree/master/Ensemble) （在Ensemble folder下）

# 結論

短期股價預測，若在新聞較少以及世界總體經濟穩定的情況下，是可以預測的，但若在世界總體經濟不穩定的區間，僅透過歷史價格是無法預測的。

本次比賽正式計分的區間正是後者，每一日的波動，都受到前日新聞影響，同時過去也沒有出現過多次的事件，對於模型來說是無法進行預測的。

最終比賽就變成了比運氣的比賽了。