# Predicting realtime stock price with deep-learing model in spark

組員:葉津源, 陳冠甫

## Abstract:

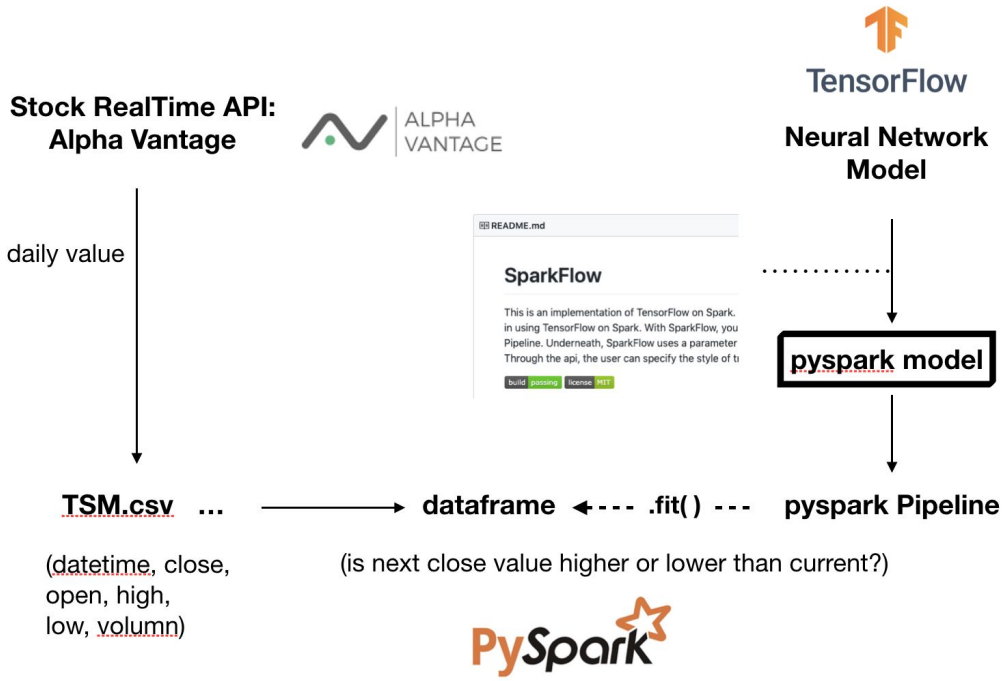
這次的期末專題，我們這組想做一個跟股票預測有關的題目，並且搭配這學期上課所學到的一些知識來製作。我們目標是利用前10天的15支股票的收盤價格，來預測第11天美股TSMC(TSM)的收盤價。

我們找了三個類別(每個類別各有14檔+TSM本身)的股票當Training Data，分別個訓練一個模型。

```
['GOOGL','FB','MSFT','AAPL','INTC','ORCL','IBM','NVDA','ADBE','TXN','AVGO','ACN','CRM','QCOM','TSM']
['MS','VFH','IYF','IXG','RYF','UYG','DFNL','PSCF','IAK','KBWP','CHIX','BDCS','KBWR','PFI','TSM']
['CY','KHC','AMAT','EBAY','URBN','ROST','ADI','LRCX' ,'RRGB', 'MCD', 'TER', 'ACGL', 'TSCO', 'TIVO','TSM']
```

## Approach:

### 1. Training




### Step 1

我們利用Alpha Vantage的 Daily Price API，抓取股票大約4年的歷史資料，其中包含了科技,ETF(指數型基金)等...類別，並將資料存成.csv放在VM中。每筆資料包含了Date, Open, Close, High, Low, Volumn。

我們分成三類 Tech, ETF, Random

## Step 2

接著用PySpark對資料做一些前處理，並將資料轉換成Training時需要的Dataframe。

## Step 3

創造一個tensorflow的graph，將feature組合成spark vector,將label轉換成one hot encode，接著造Adam optimizer，然後將以上結果放進 SparkAsyncDL (spark flow的套件)，組合成spark pipeline並開始訓練，最後儲存訓練的結果。

```
mg = build_graph(small_model)
#Assemble and one hot encode
va = VectorAssembler(inputCols=final_df.columns[1:151], outputCol='features')
encoded = OneHotEncoder(inputCol='result', outputCol='labels', dropLast=False)
adam_config = build_adam_config(learning_rate=0.001, beta1=0.9, beta2=0.999)

spark_model = SparkAsyncDL(
    inputCol='features',
    tensorflowGraph=mg,
    tfInput='x:0',
    tfLabel='y:0',
    tfOutput='out:0',
    tfLearningRate=.001,
    iters=200,
    predictionCol='predicted',
    labelCol='labels',
    verbose=1,
    optimizerOptions=adam_config
)

print('save model in ckptpath')
p = Pipeline(stages=[va, encoded, spark_model]).fit(final_df)
p.write().overwrite().save(ckptdir)

print('task all done')
```

column in dataframe

variable in tf model

150 -> 256

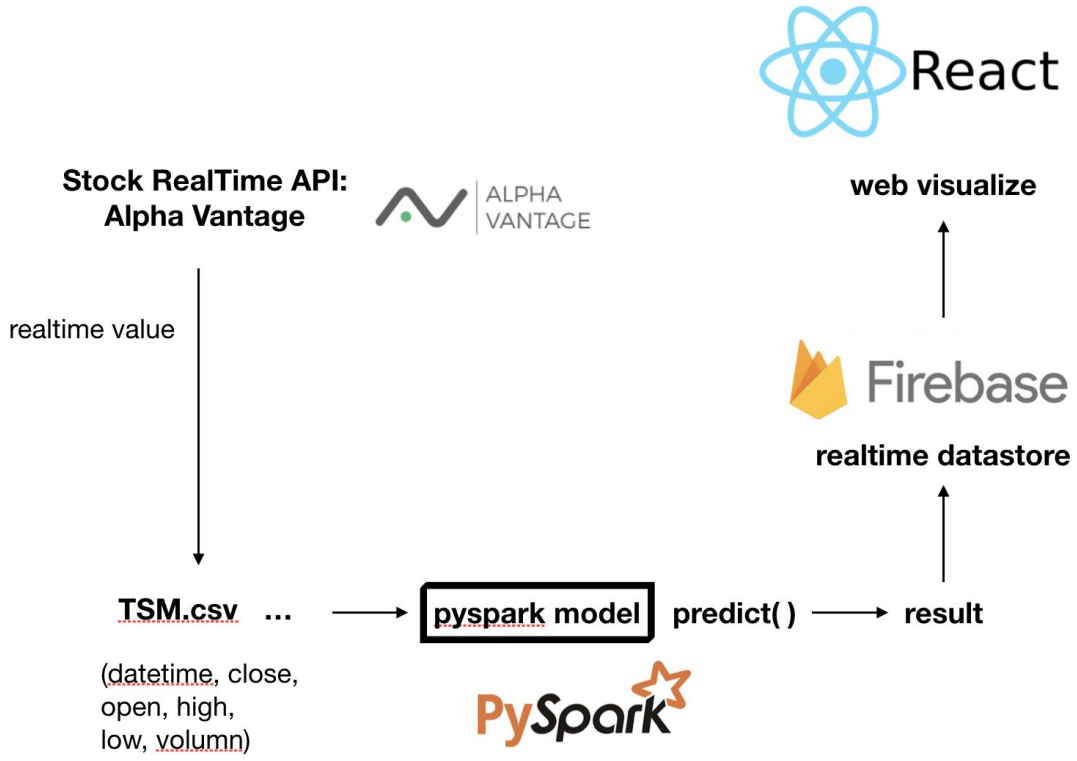
relu activation

256 -> 256

relu activation

256 -> 2

## 2. Predicting + Visualization:



### Step 1

從Alpha Vantage抓取資料，搭配前面訓練出來的Neural Network模型，對資料進行預測，並將資料傳送到Firebase Realtime Datastore。

### Step 2

將Realtime Datastore中的資料，透過Listener來監聽Datastore，當資料改變時，會改變前端React Component的State，讓網頁進行Realtime的更新。最後將做好的網頁Deploy到Firebase Hosting的服務上。

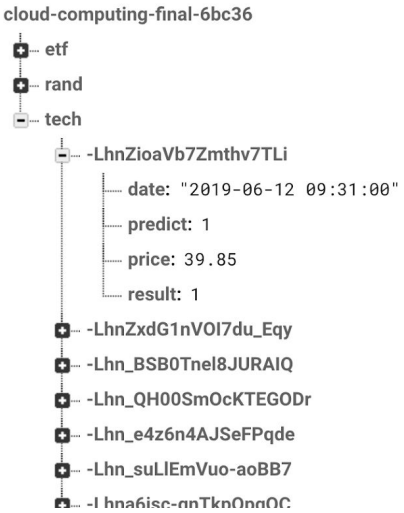
## Results:

Training的過程

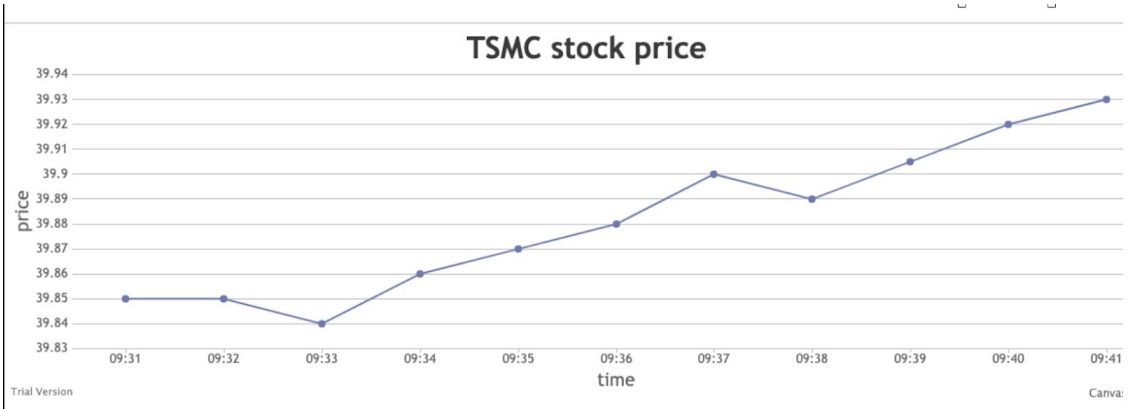
```
Partition Id: 123eb3e360fb4259ace35ea89d9b5e03, Iteration: 0, Loss: 371.149323
Partition Id: 2af7c849245d4e9f91a359631be571ea, Iteration: 0, Loss: 46.221939
Partition Id: b3596d22a9524a05af2813691e6c1ec7, Iteration: 0, Loss: 143.617676
Partition Id: ca2be052ace14d22986a85600bb271d2, Iteration: 0, Loss: 35.406776
Partition Id: 2af7c849245d4e9f91a359631be571ea, Iteration: 1, Loss: 139.247696
Partition Id: 123eb3e360fb4259ace35ea89d9b5e03, Iteration: 1, Loss: 197.460190
Partition Id: b3596d22a9524a05af2813691e6c1ec7, Iteration: 1, Loss: 96.722649
Partition Id: ca2be052ace14d22986a85600bb271d2, Iteration: 1, Loss: 36.886974
Partition Id: 2af7c849245d4e9f91a359631be571ea, Iteration: 2, Loss: 96.562553
Partition Id: 123eb3e360fb4259ace35ea89d9b5e03, Iteration: 2, Loss: 121.998642
Partition Id: b3596d22a9524a05af2813691e6c1ec7, Iteration: 2, Loss: 70.031273
Partition Id: ca2be052ace14d22986a85600bb271d2, Iteration: 2, Loss: 106.950455
Partition Id: 2af7c849245d4e9f91a359631be571ea, Iteration: 3, Loss: 82.900932
```

```
Partition Id: 123eb3e360fb4259ace35ea89d9b5e03, Iteration: 90, Loss: 0.694443
[Stage 16:>                                                         (0 + 4) /
ration: 91, Loss: 0.894650
Partition Id: b3596d22a9524a05af2813691e6c1ec7, Iteration: 92, Loss: 1.335120
Partition Id: ca2be052ace14d22986a85600bb271d2, Iteration: 91, Loss: 0.692730
Partition Id: 123eb3e360fb4259ace35ea89d9b5e03, Iteration: 91, Loss: 1.139917
Partition Id: 2af7c849245d4e9f91a359631be571ea, Iteration: 92, Loss: 1.000871
Partition Id: b3596d22a9524a05af2813691e6c1ec7, Iteration: 93, Loss: 1.236592
Partition Id: ca2be052ace14d22986a85600bb271d2, Iteration: 92, Loss: 0.685727
Partition Id: 123eb3e360fb4259ace35ea89d9b5e03, Iteration: 92, Loss: 0.841245
Partition Id: 2af7c849245d4e9f91a359631be571ea, Iteration: 93, Loss: 0.952797
Partition Id: b3596d22a9524a05af2813691e6c1ec7, Iteration: 94, Loss: 1.342754
Partition Id: ca2be052ace14d22986a85600bb271d2, Iteration: 93, Loss: 0.686798
Partition Id: 123eb3e360fb4259ace35ea89d9b5e03, Iteration: 93, Loss: 1.385706
Partition Id: 2af7c849245d4e9f91a359631be571ea, Iteration: 94, Loss: 1.059915
Partition Id: b3596d22a9524a05af2813691e6c1ec7, Iteration: 95, Loss: 1.179318
```

放在Firebase Realtime Datastore的資料




前端的網頁頁面




**tech**

| 時間 | 09:35 | 09:36 | 09:37 | 09:38 | 09:39 | 09:40 | 09:41 |
|---|---|---|---|---|---|---|---|
| 實際 | 漲 | 漲 | 漲 | 跌 | 漲 | 漲 | -- |
| 預測 | 漲 | 漲 | 漲 | 漲 | 漲 | 漲 | 漲 |

## Discussions:

這次的專題，在結合Spark和Tensorflow上花了一些時間。也因為Demo希望可以每分鐘都更新所以改用Minute資料來預測(但因為API限制，最後我們還是先把資料抓下來，一筆一筆的送)，我們發現用Daily資料來進行Training的模型來預測Minute的資料是不太可行的。

在資料的顯示上感覺可以做更好的表達，讓看的人可以更清楚介面顯示的內容。

## Github Link:

https://github.com/5loaves-2fish-12basckets/Stock-Spark?fbclid=IwAR0nrLCoy52vro3GGb5Int0Z2YIEIBRbPIlqLkM6NGoTeTN7IsGcAsysJ0c