姓名：黃展皇

學號：110621013

0. 各種模型能力確認(先確認各個模型的能力到哪裡，再找最好的進行超參數修改)(以下數字皆為 Kaggle public sub 的分數)

- 考慮題目是跟短期股市波動有關的收盤價預測，與之前幾天的價格/交易量/市場信心有較大的關聯，因此模型上除了使用助教範例的 linear 模型以外，還加上 multi_layer_linear、DNN、LSTM 做嘗試，結果：
  1. Linear：25.72812，壓在 baseline 上
  2. multi_layer_linear：10.24172{learning_rate=0.01, epochs=100, middle=600}(middle 為中間多的層數)
  3. DNN：169.69460{node=16-8-4-2-1}
  4. LSTM：200.28982{多對一，LSTM(8, activation='relu')+ Dense(1)}

  綜合可選用 multi_layer_linear 為接下來調整的模型：

```
## 對 multi_linear()做各種參數變化並觀察分數
    ### 改 middle
    ### 10.24172{learning_rate=0.01, epochs=100, middle=600}
    ### 10.45073{learning_rate=0.01, epochs=100, middle=6000} middle 60 or 6 都很差
    ### 11.22146{learning_rate=0.01, epochs=100, middle=1000}
    ### 10.68334{learning_rate=0.01, epochs=100, middle=200}

    ### 改 epochs
    ### 9.79706{learning_rate=0.01, epochs=1000, middle=600}
```

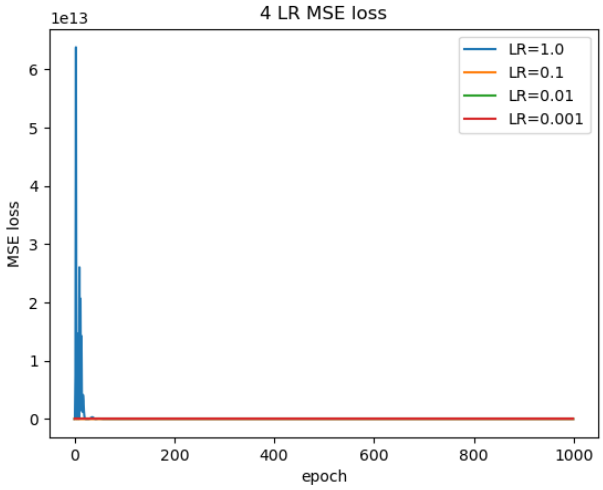
9.41729{learning_rate=0.01, epochs=10000, middle=600}
18.76628{learning_rate=0.01, epochs=1000, middle=2000}

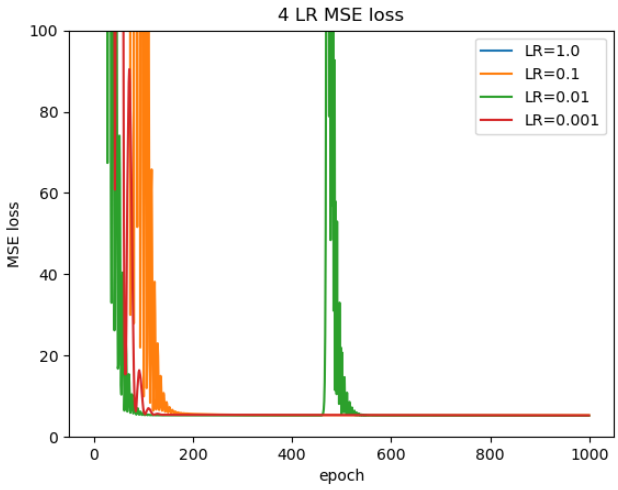
改 lr
9.88990{learning_rate=0.1, epochs=1000, middle=600}
9.40615{learning_rate=0.001, epochs=10000, middle=600}

改三層
9.74213{learning_rate=0.01, epochs=1000, middle=600,600}
9.34241{learning_rate=0.001, epochs=10000, middle=600,600}
9.32999{learning_rate=0.001, epochs=10000, middle=600,1200}
9.34474{learning_rate=0.001, epochs=10000, middle=1200,600}
16.16026{learning_rate=0.001, epochs=10000, middle=1200,1200}
加 reLU 效果很差

→最終得知 9.32999{learning_rate=0.001, epochs=10000, middle=600,1200}為最佳模型

1. (5%) 使用四種不同的 Learning Rate 進行 Training（方法參數需一致），作圖並討論其收斂過程（橫軸為 Iteration 次數，縱軸為 Loss 的大小，四種 Learning Rate 的收斂線請以不同顏色呈現在一張圖裡做比較）。



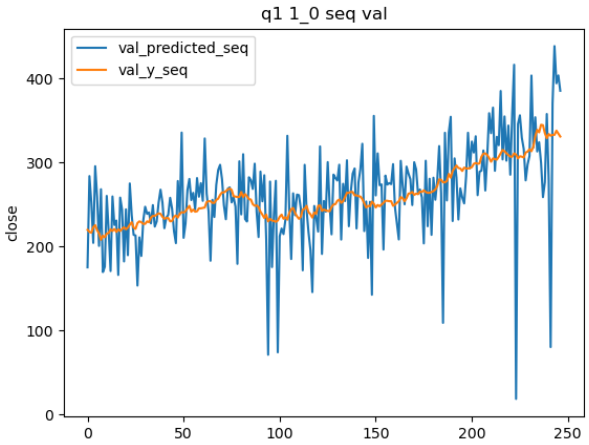


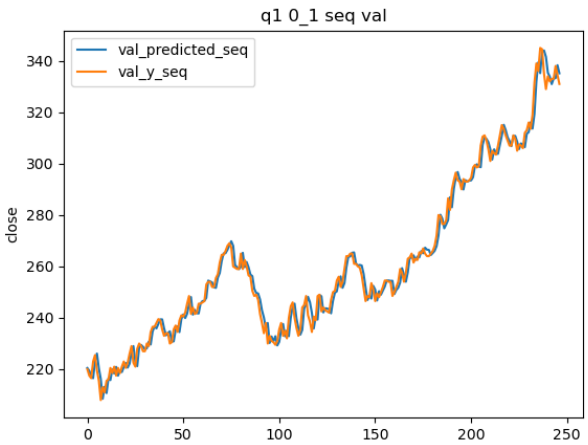
在 0 選定好的模型跟超參數後開始進行題目要求的測試，第一題是 LR，這邊選定[1.0, 0.1, 0.01, 0.001]四個 LR 進行測試得到兩圖，其中右圖是左圖在 loss 範圍 0~100 作圖。

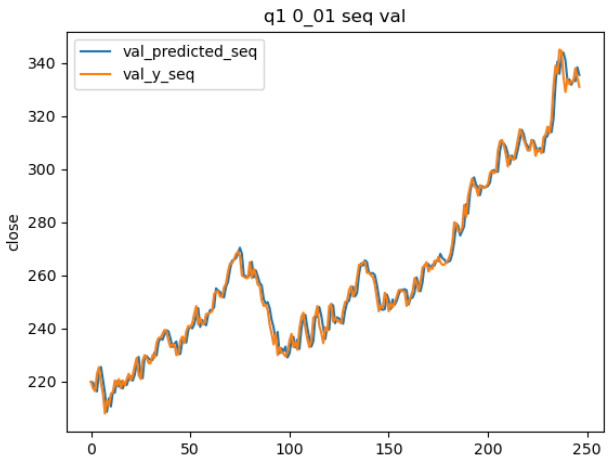
由圖可知在 LR=1.0 時在一開始 loss 非常大，數量級達到 10 的 13 次方，並且訓練 1000epochs 中也只降到 915.4912，無法在右圖中呈現；LR=0.1 時相較 1.0 有較快的在 200epochs 前就收斂到 5 左右並穩定無跳動，最終 5.3421；LR=0.01 時則是 4 個學習率中最快收斂的(100epochs 前)，但在第 470epoch 時有個 loss 的急遽升高(~5→116.7081)，比較 460epoch 的 5.2833 跟 1000epoch 的 5.2033，可見在跳動後有更快更好的 loss 表現；最後是 LR=0.001 時僅次於 0.01 的收斂速度，穩定無跳動，最終 5.3157。

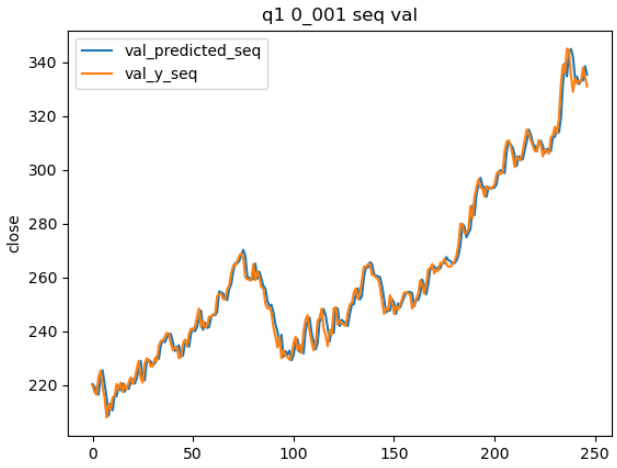
統整以上最終 loss 曲線可知 LR=0.01 應該為最佳解，可以在保有高速收斂的同時能夠跳脫局部最小值達到最低的 loss。

同時附上在 val data 上的預測表現，以及 Kaggle 的成績：





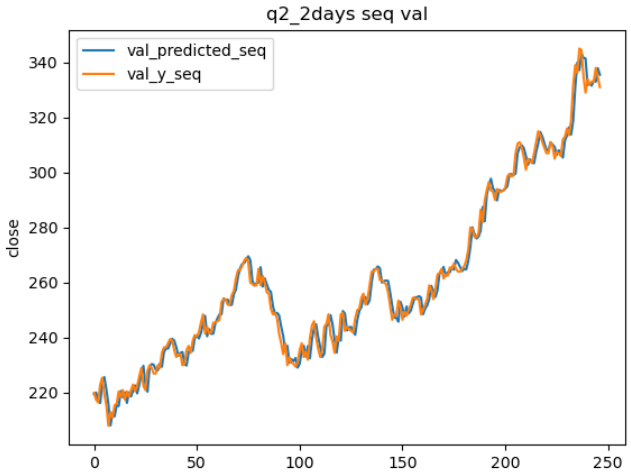


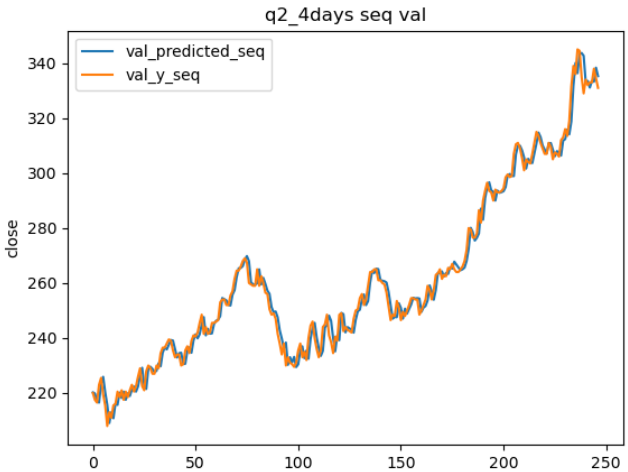


9.79152{learning_rate=0.1, epochs=1000, middle=600,1200}
9.76572{learning_rate=0.01, epochs=1000, middle=600,1200}
10.11540{learning_rate=0.001, epochs=1000, middle=600,1200}
116.05683{learning_rate=0.1, epochs=100, middle=600,1200}
11.83569{learning_rate=0.01, epochs=100, middle=600,1200}
10.06587{learning_rate=0.001, epochs=100, middle=600,1200}

2. (5%) 比較取前 2 天和前 4 天的資料的情況下，於 Validation data 上預測的結果，並說明造成的可能原因。

在 val data 上的預測表現，以及 Kaggle 的成績：



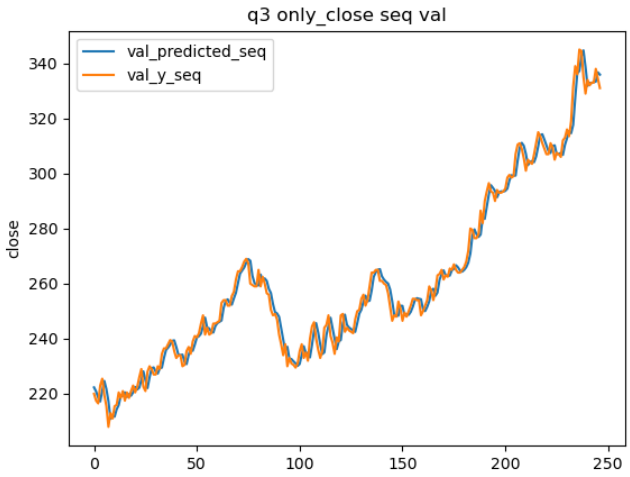


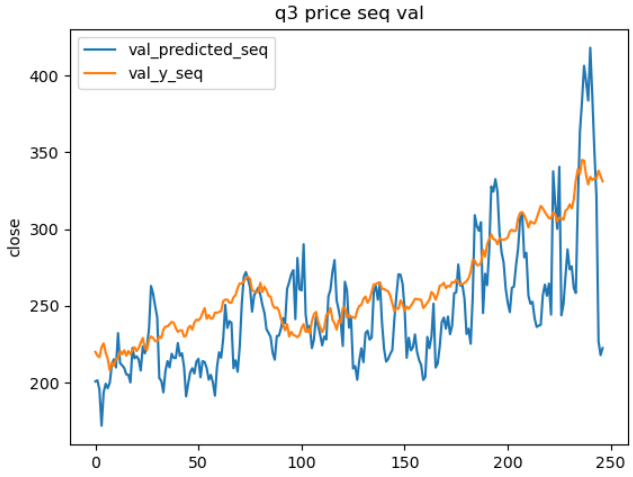
143.49388{learning_rate=0.001, epochs=1000, middle=600,1200, 2dayData}
10.11540{learning_rate=0.001, epochs=1000, middle=600,1200, 4dayData}

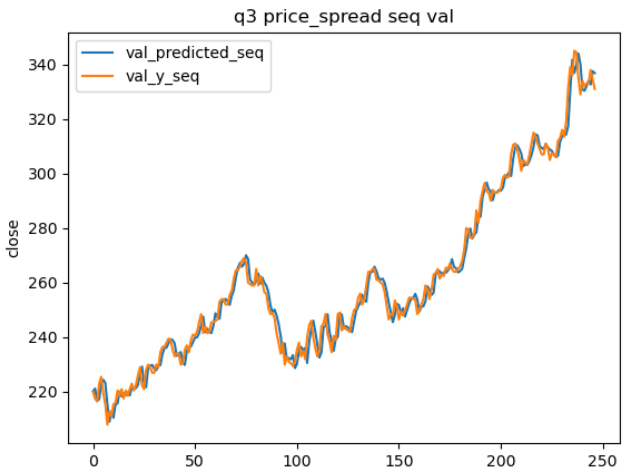
從 val data 的圖上基本上只有非常些微的差別，大概是在部分情況(如第 80 天的下跌)4 天資料預測會比 2 天資料預測更符合 val_y，但在 Kaggle 成績尚有相當明顯的落差，4 天資料預測顯著要比 2 天資料預測好。

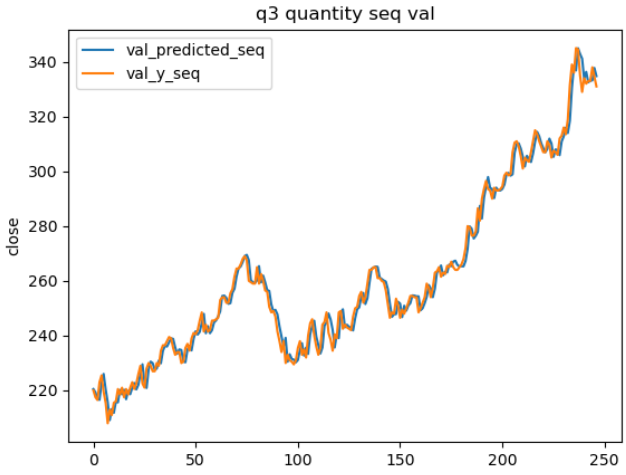
3. (5%) 比較只取部分特徵和取所有特徵的情況下，於 Validation data 上預測的結果，並說明造成的可能原因。

附上在 val data 上的預測表現，以及 Kaggle 的成績：









9.82741{learning_rate=0.001, epochs=1000, middle=600,1200, no quantity Data(shares amount turnover)}
163.29943{learning_rate=0.001, epochs=1000, middle=600,1200, no price Data(open high low close)}

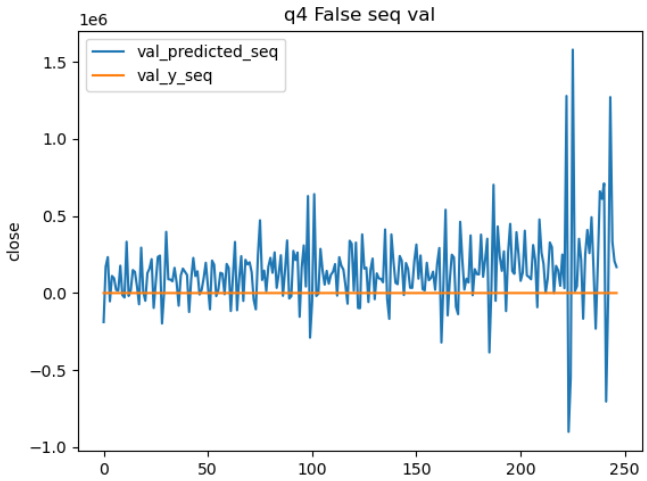
9.48439{learning_rate=0.001, epochs=1000, middle=600,1200, no price spread Data(change)}
9.90882{learning_rate=0.001, epochs=1000, middle=600,1200, only close Data}

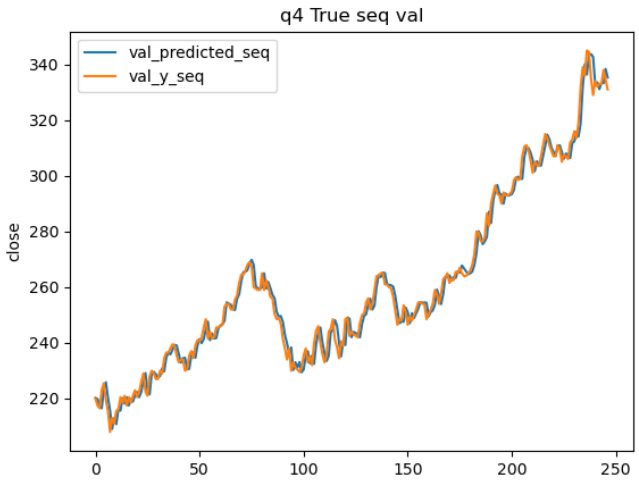
在只取部分特徵和取所有特徵的情況下，我做了四種部分特徵的資料，分別是只有前一天收盤價的 only close、沒有價格資訊的 price(去除'open', 'high', 'low', 'close')、沒有交易量資訊的 quantity(去除'shares', 'amount', 'turnover')以及沒有漲跌價差的 price_spread(去除'change')。

從 val data 的圖上基本上只有 price 跟 val_y 有較大的落差跟波動，這也不難理解，少了前幾天的價格資訊而要預測收盤價格本就是不切實際的想法，並且也不確定隨著預測的高低有什麼資訊可以得知；而在 only close 中模型只知道前幾天收盤價，因此直接進行了最後收盤價的外延；而 quantity 以及 price_spread 則在圖形上有跟 only close 在局部有些差異，但在 Kaggle 成績上 only close 是也應該是最差的，而相較 only close 多點資訊的 quantity 以及 price_spread 有再好一點。由以上證明價格對預測收盤價最重要，並且交易量資訊以及漲跌價差皆能提供些許幫助。

4. (5%) 比較資料在有無 Normalization 的情況下，於 Validation data 上預測的結果，並說明造成的可能原因。

   附上在 val data 上的預測表現，以及 Kaggle 的成績：



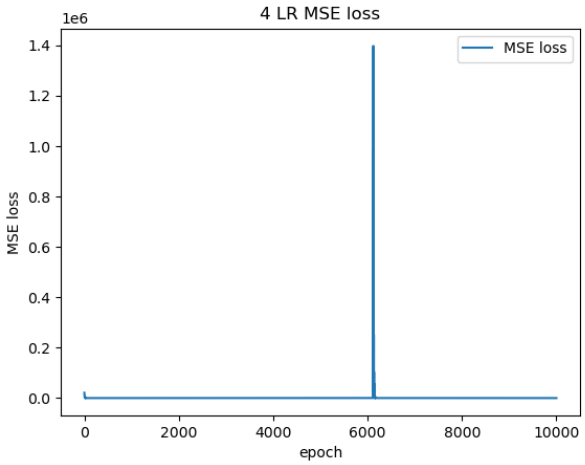


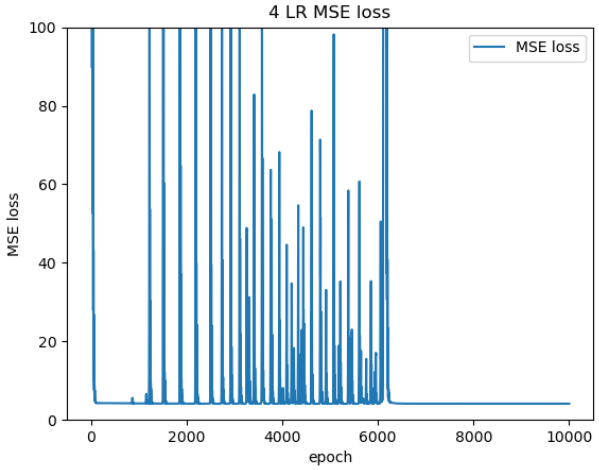
10.11540{learning_rate=0.001, epochs=1000, middle=600,1200, Normalization}
1006381.86377{learning_rate=0.001, epochs=1000, middle=600,1200, no Normalization}

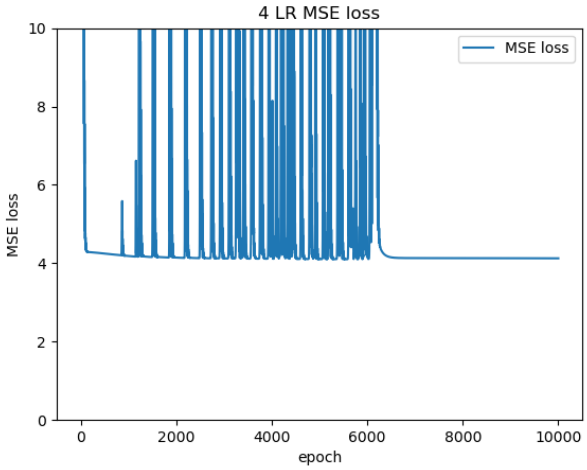
這部分的結論就很明顯，對於交易量/交易金額動不動數量級就達 7~9 的數字而言，其他價格的數量級~3 會顯得沒有作用，並且訓練上也會有非常差的 loss 曲線，同時 Kaggle 分數 1006381.86377。有 Normalization 的日子，才是好日子。

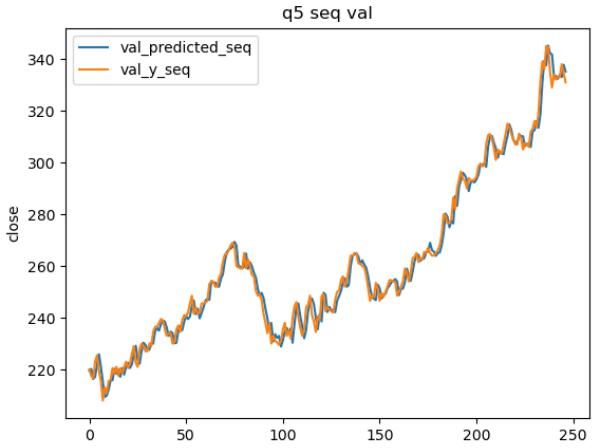
5. (10%) 請說明你超越 Baseline 的 Model（最後選擇在 Kaggle 上提交的）是如何實作的（若你有額外實作其他 Model，也請分享是如何實作的）。

在自由嘗試的部分根據我對於股市短期波動的了解，加入了 K 棒相關特徵(漲勢上下影線、跌勢上下影線、十字線上下影線)以及五日平均線等特徵，訓練過程 loss、val data 的預測表現以及 Kaggle 的成績：









12.07070{learning_rate=0.01, epochs=10000, middle=600,1200, add K 棒系列特徵 and 5MA}

很可惜的是雖然加了實際操作股票時真的會用到的短中期特徵，但在 Kaggle 上的表現並不算太好(沒有超越原來最高的 9.32999)，可以觀察到 loss 有多次程度不一的跳動直到約 6200epoch 後趨於穩定，而 val data 表現得有些起伏，並沒有緊貼著 val_y 的走勢，也導致了較高的誤差。

→最高分數 model：9.32999{learning_rate=0.001, epochs=10000, middle=600,1200}