結合元學習及 Transformer 之可調式股票預測

摘要

本研究旨在深入探討如何運用元學習結合 Transformer 模型，用於股票收盤價預測之應用。本研究以開源股票資料和台灣股票資料為驗證對象，以評估此方法的效能與實用性。

股票預測在金融領域扮演關鍵角色，但也面臨著極大挑戰。投資者依賴股票預測來做出買賣決策。有些方法採用基本分析，如財務報表和供需法則，來預測股價走勢；有些則運用技術分析，如移動平均線（MA）和移動平均收斂/發散指標（MACD），尋找 K 線圖的模式，並進行走勢預測；還有一些方法將兩者結合。在此背景下，Transformer 模型在時序資料中顯現出對於前後時序關聯性的優異捕捉能力。Transformer 模型成功透過熟練捕捉 K 線圖模式，並在股票收盤價預測上呈現相對準確的表現。

研究中引入元學習方法，以解決過度擬合的困擾，將 Adaboost 和 Transformer 兩種神經網路融合於一體。透過元學習，試著用常見的技術指標來提升神經網路對收盤價的預測準確性。

實驗部分以 Python 進行實作，運用開源股票資料，進行實證分析以探究不同領域之實驗結果。最終的實驗驗證結果顯示，元學習方法成功解決過度擬合的問題，提升股票收盤價預測的精確性。

關鍵詞: meta learning、Transformer、Adaboost、技術指標、股票分析

目錄

圖目錄

一、緒論

1.1 研究動機

當前，社會環境中的消費者物價指數（CPI）持續上升，使得僅依賴有薪收入的主動式收入變得愈發不堪負擔。辛苦賺來的薪資在不知不覺中被通貨膨脹所侵蝕。在資本主義體制下，通過投資實現資本的增值，並實現被動收入的增加已變得至關重要。這樣做不僅可以保護資本不受通膨的影響，還可以實現持續增長。當然，投資存在風險，例如美國次級房貸風暴和日本經濟泡沫爆破等事件都強調了風險管理的重要性。如果有一個工具可以幫助預測金融市場的走勢，那麼這將有助於降低投資風險。

假設年化平均 10%，是透過定期定額投資所得到。如果，我們今天能夠預測走勢，那麼就可以選擇甚麼時候進行投資，甚麼時候不投資，而不是使用定期定額的方式，那麼相信投資報酬率會大於 10%。

當前，人工智慧（AI）模型對於時序資料的前後關聯性已經有了相當高的理解。因此，我們考慮使用對時序性敏感性較高的 Transformer 模型來預測股票市場的動向。為了進一步增強 Transformer 模型的預測性能並減少過度擬合問題，我們引入了元學習（meta learning）的方法。採用 Adaboost 以及 Transformer 的結合，由 Transformer 作為 Adaboost 的 weak learner，每一輪被預測錯誤的資料樣本，在下一輪的訓練中該資料權重會增加，作為模型的重點訓練，由此改善過擬和問題。使用常見的技術指標，由 Adaboost 針對較困難的資料，進行反復訓練，來試著提升 Transformer 模型在股票預測上的效能。

1.2 研究背景

傳統的股市分析方法包括基本分析和技術分析。基本分析主要依賴財務報表和供需法則等經濟原則，以評估資產價值。以石油市場為例，基本分析可能會考慮到當石油價格達到 100 美元時，鑽油機啟動並增加供應，導致價格下跌。當價格下跌到 50 美元時，鑽油機可能停止生產，供應減少，導致價格回升。這樣的供需動態解釋了價格在 50 美元和 100 美元之間的波動。

另一方面，技術分析著重於過去價格和交易量數據，通常使用 K 線圖模板等技術指標來預測未來價格走勢。以石油市場為例，技術分析可能觀察到當價

格達到 100 美元時，市場趨向下跌，而當價格降至 50 美元時，市場趨向上漲。這樣的價格走勢模式基於過去的觀察和經驗，投資者嘗試根據這些模式來預測未來價格。

[1]在金融領域當中，傳統方法像是 auto-regressive integrated moving average (ARIMA) model，這個模型先將資料轉換成 stationary，將轉換後的資料輸入進模型，再透過 moving average model，將過去的誤差資料進行修正而得到未來的預測。

然而，隨著人工智慧（AI）的出現，股票市場分析取得了更大的進展，特別是在技術分析領域。傳統的技術分析可能受限於人為主觀因素，容易忽略或錯過一些關鍵模式。AI 在股票預測中的應用，從機器學習到深度學習，已經為投資者提供了更精確的工具。 [2, 3, 4]這些方法包括使用隨機森林和線性回歸等機器學習技術，以及使用卷積神經網絡（CNN）、循環神經網絡（RNN）和長短期記憶網絡（LSTM）等深度學習技術。

Transformer 模型，採用 self-attention 的方法，有別於 RNN，不用考慮到輸入以及輸出的距離，attention based 的 Transformer 模型沒有遺忘的問題，著名的應用有 ChatGPT，是一個引人注目的突破，因為它在處理時序性資料方面表現出色。 [5]將每日的開盤價、最高價、收盤價、最低價、成交量作為輸入，輸入進 Transformer-based model，得到收盤價的 moving average 的預測值。[6]Transformer 模型憑藉其優異的前後文理解能力，以及對時序性的敏感性，使用收盤價作為輸入，輸入進 Transformer model，得到未來一天的收盤價預測值，在股票價格預測中表現卓越。這些進步的 AI 技術確實提高了股票市場的預測準確性，有助於投資者更好地應對市場波動。

1.3 研究目的

參照 [6]的研究，了解到 Transformer 在股票市場預測中已經取得了一定的成功，特別是在獲利和收盤價預測方面。然而，目前關於 Transformer 在股票市場的應用的文獻相對較少，這表明仍有很大的發展和改進潛力。

建立在 [6]的研究基礎上，我們的研究旨在通過結合 Transformer 與元學習的方法，利用幾個常見的技術指標，試著提高股票市場預測的準確性。提高預測的準確性可以增加獲利並加強風險管理的能力。

在當今全球經濟體中，金融市場的變化迅速，市場環境複雜，而世界各國都在不斷投入人工智慧技術應用於金融領域。這種變化不僅包括金融市場的波動，還包括國際金融局勢的複雜性，如中美貿易戰。如果有一個工具可以高度準確地預測金融市場的動向，那麼它將對全球經濟產生深遠影響，甚至可能被用於全球金融政策操作。因此，為了因應金融市場的變化，各國必須具備相應的技術和能力，以確保經濟的穩定和安全。

1.4 研究方法簡介

圖 1-1 呈現了本論文用於訓練網路模型的流程圖。本研究旨在結合元學習和 Transformer 技術，以提升股票預測的準確性。技術分析中存在數百種不同的技術指標，這些指標可以獨立使用，也可以組合使用以提高預測的精確性。我們採用 Adaboost 和 Transformer 方法，試圖探索各種技術指標的不同組合，以識別對預測具有較高影響力的技術指標和 K 線組合。这些被識別出來的指標和組合將有望成為提高預測模型性能的關鍵因素。

圖 1-1、整體實驗流程

元學習模型的架構，我們使用 Adaboost.R2 和 [6]的 Transformer model，以開盤價、當天最高股價、收盤價、當天最低股價、當天成交量以及 5 天移動平均線作為特徵，總共有 6 個特徵用於訓練每一輪的 Transformer。使用訓練過的元學習模型來預測股票收盤價，每一輪訓練過的 Transformer 都將這 6 個特徵作為輸入，並生成預測結果。在每一輪的 Transformer 的預測結果中，我們根據各個 Transformer 的權重選擇權重中位數，作為整個模型的最終輸出結果。根據這一算法，我們訓練了多個 Transformer，每個 Transformer 根據常見的技術指標和 K 線圖進行訓練。最後，從這些 Transformer 的輸出結果中，選擇權重中位數，類似於嘗試各種 K 線圖和常見的技術指標之間的搭配，以獲得最佳的預測效果。这種方法旨在提高預測模型的性能，並同時確認加入元學習，是否能更進一步提升預測效果。

1.5 主要貢獻

本研究的主要貢獻可以總結如下：

1. 元學習和 Transformer 結合：本研究採用元學習結合 Transformer 模型，這種結合使得模型能夠有效地對技術指標間較為困難的關係反覆訓練，最終得到股票預測效果的提升。

2. 台灣股票市場驗證：本研究進行了對台灣股票市場的實證研究，驗證了元學習方法和純粹的 Transformer 模型在實際股票預測中的成效。這個實證結果不僅擴展了模型的應用範圍，還提供了實用性的參考，尤其對於致力於投資股票市場的投資者和金融機構。

綜合以上兩點，本研究的主要貢獻在於提供了一個有效的方法，讓模型針對較為困難的資料關係加強訓練，並在台灣股票市場上取得了實證成功，這對於提升股票預測的準確性和實際應用具有重要價值。

1.6 論文架構

本論文共分為六大章，章節安排如下:

第一章:敘述本篇研究最初的想法及概念。首先，在前三節裡依序介紹本篇研究的動機、背景、目的與方法簡介，以及主要貢獻，最後，介紹本篇研究的組織架構。

第二章:介紹與本論文相關的研究，其中有股票傳統預測方法、深度學習之方法、集成式學習，還有取得技術指標的函式庫。

第三章:介紹 Transformer 模型架構、訓練資料集的處理、該模型的訓練和測試。

第四章:介紹結合元學習與 Transformer 模型架構、訓練資料集的處理、該模型的訓練和測試。

第五章:介紹結合元學習與 Transformer 模型在股票預測上面的實驗結果。

第六章:提出未來展望，總結本研究內容以及討論將來研究方向。

二、相關研究

2.1 傳統股票預測方法

由 [7] 得知傳統方法有技術分析和基本分析， [6] 而技術分析相信任何在股票市場的資訊，像是開、高、收、低、成交量等資訊，皆會反應在股價移動趨勢上面。有 MA、KD 等好幾種技術指標，而依靠人類的智慧從眾多的技術指標汲取用得上的指標加以分析得到股價的走勢預測。 [6] 而基本分析包含供需法則和財務報表、新聞、輿論、政治等 unstructured textual information。

[8] 股票價格有時高有時低，不過時間拉長，市場最終會做調整，股票價格會逐步向其真正的價值靠近。價值投資者分析公司基本面得知其真正的價值後，和目前價格做比較，價格低於實質價值買入，價格高於實質價值則做空。效率市場假說 (Efficient Market Hypothesis, EMH)表示，由於市場的所有情況，皆已反映在價格上面，所以無論是基本分析，還是技術分析，依靠過去的資料去預測未來股價是不可行的。再加上隨機漫步理論(Ransom Walk Theory)，如果有所獲利，也只是和機率、運氣有關，丟硬幣次數多了，也有機會丟出連續 10 次正面。華倫．巴菲特(Warren Buffett)對此回應，如果仔細去查那些連續得到 10 次正面硬幣結果的人，可能會發現他們可能都學過證券分析、價值投資的方法。[9] 財務報表中的損益表 (P / L) 可以了解一間公司一年的獲利有多少。而獲利中，主要是由公司的本業所賺取的”營業淨利”，還是由本業以外的財務活動，像是買賣股票或是變賣公司事業部門的”稅前淨利”可以看出來一年中的收入是由哪種方式入帳，才不會有像是今年公司獲利很多，以為公司前景大好，但這額外多出來的獲利，其實是因為公司經營量能不足，將旗下事業部門變賣而得到的獲利。由財務報表來分析公司前景，進而決定是否繼續投資該公司。

[10] 技術分析對於 EMH 和隨機漫步理論的回應是，K 線圖的趨勢是確實存在的，而趨勢的存在就代表價格並不是像隨機漫步理論所認為的，價格變動是序列獨立的。 EMH 和技術分析的前提＂市場預先反映一切＂是貼近的，只是學術界認為市場過於快速地反映所有資訊，以致交易者沒有機會利用相關資訊創造獲利。可是技術預測的基礎是：重要的市場資訊遠在明朗化之前，就早已反映在市場價格中。技術分析是研究果，基本分析是研究因。技術分析者，不見得知道目前線圖趨勢的原因，但背後是有原因的，不過基本分析重視結果。收盤價線圖的上升與下降，背後有經濟的供需法則以及交易者的心理層面作為解釋，所以由此看出技術分析的確有某種程度的反映出收盤價，是有跡可循的，

背後不完全是毫無道理可循。在技術分析裡面，線圖的谷底或折返低點稱為支撐，相反地，折返高點則稱為壓力。當線圖從低點(點 1)上升至碰到了壓力(點 2)，下跌至前低點碰到支撐(點 3)後，又上升突破了前壓力點(點 4)，這樣的趨勢反轉，背後的心理解釋為，在點 1 的時，有一群交易者先做多，等到了點 2 時，當初做多的一些人，會覺得當初部位規模太小，如果降為點 1 附近的話，做多者可能加碼，當初做空的人，會懷疑自己是不是判斷錯誤，會希望價格可以返回點 1 附近，讓做空者可以順利解套。還有場外觀望者，還沒進場或在點 1 提早結束多頭部位的人，都會希望當降回點 1 附近時，可能會做多進場。如果有機會降回點 1 附近，前面所述的一開始進場者、一開始做空，以及場外觀望者，都有機會買進。當線圖來到點 3 時，如果有大量買進的情況，那麼線圖就有可能突破前壓力點，來到點 4。如果在點 3 時，買進的交易量不足以支撐下跌的趨勢，而跌破前面點 1 的低點時，上述的心理層面將相反過來。任何資訊和情況都會反映在股票趨勢上面，憑藉著重大指標，像是道瓊工業指數和道瓊運輸指數依序上漲，達到指標的互相確認，再加上成交量的擴增，來判斷新的多頭市場已經到來，憑藉著線圖趨勢，來判斷目前的市場健康狀態。也可以依靠線圖來預測股價趨勢，可以用 MA 來計算包寧傑帶狀(John Bollinger)，再搭配線圖來判斷趨勢。

2.2 深度學習股票預測模型

深度學習可以把高維度的特徵資料關係給解析出來，在眾多資料組合中有些資料有關聯，有些則無。有數百種技術指標，依照傳統的技術分析，可以知道有些技術指標能互相搭配增加預測準確度。各個指標間可能存在依靠傳統分析尚未發現的技術指標搭配，可以依靠深度學習將人類智慧還沒有發現的指標搭配，或是藏在單一指標資料間的關係給分析出來，增加預測的準確度。

[11] 用收盤價和成交量計算出技術指標，透過情緒字典，將新聞文章轉換成新聞情緒，技術指標和新聞情緒兩者都是 time series，透過日期將兩者做 concatenate，作為兩層的 LSTM 的輸入。得到的效果，比只用單一技術指標或是單一新聞情緒還要好。透過技術指標和新聞情緒之間的關係來提高預測準確度。

[12] 作者提出 muli-filters neural networks(MFNN)，由兩層 multi-filters 所組成，一個 multi-filter 是由 convolutional unit 和 recurrent unit 所組成，最後再透過 fully connected layer 和 softmax 做分類預測，預測接下來是上升趨勢還是下降。主要是透過 MFNN 對做了標準化的輸入資料，開、高、收、低、成交量以及一

段時間內交易金額總量，做 feature engineering 得到 feature map，再將 feature map 再做一次 feature engineering，更進一步地將特徵萃取出來，最後再用 fully connected layer 和 softmax 從萃取的特徵得到預測。作者做了市場模擬，基於該模型的預測，並運用交易策略，當模型預測上升趨勢時做多，下降趨勢時做空，最後成功得到獲利。此論文使用深度學習將高維度資料間的關係萃取出來，模型從資料間得到的訊息得到預測。

[13] 閱讀了年代從 2000 年到 2019 年的 138 個期刊文章。此論文中提到的深度學習模型在股票預測上，有使用 DNN、CNN 和 LSTM。在 EMH，因為所有情況皆已反映在價格上面，所以股價都是最公平的價格，不會有被低估或被高估的股票價格。隨著時間，EMH 逐漸受到挑戰，一些經濟泡沫的案例，像是，1986 到 1991 年日本泡沫經濟，以及 2007 年美國的次級房屋借貸危機，都顯示出，價格遠遠超出了實際價值的現象。像這樣的市場亂象，有些學者試著提出各種解釋，而這些解釋還是讓人存疑。後來解釋的模型開始往人類心理學的方向研究，也就誕生了行為財務學（Behavioral Finance, BF），羅聞全(Andrew W. Lo)學者將 EMH 結合 BF 提出了適應性市場假說(Adaptive Markets Hypothesis, AMH)來解釋這樣的市場亂象。也說明了已經有些現象，EMH 是不足以說明，EMH 是不足夠的，或不完全正確。

深度學習方法，背後參考了傳統方法裡面的技術分析，像是把技術指標作為輸入，由深度學習模型來分析。又或是技術指標結合新聞情緒作為輸入，讓模型分析技術指標和人類心理間的關係。若傳統技術分析是有效的，甚或是能成功得到獲利，那麼使用深度學習也同樣能得到獲利。

2.3 集成式學習

[14] 集成式學習是由好幾個 classifier 組成 meta classifier，集成式學習的方法有，majority voting 和 plurality voting、bagging 以及 boosting。本論文中使用的 Adaboost，使用集成式學習中的 boosting。Boosting 的方法為，隨機從 training data 採用不重覆抽樣，抽出 training sample d_1 來訓練 weak learner C_1，再來一樣隨機不重覆抽樣，抽出 training sample d_2，然後加上 50%的第一次訓練完的 classifier 所做出的錯誤預測，也就是錯誤分類的資料。依此類推，訓練了 n 個 classifier 後，將 n 個 classifier 得到的預測值透過 majority voting 結合，得到最後的預測值。Boosting 的方法可以有效得到良好的效果，但也要小心過度擬合的問題。

Adaboost 稍加對 boosting 做了些改變，用全部的 training data 來訓練每一輪的 classifier。一開始全部的 training data 被賦予同樣的權重，訓練第一輪的 classifier，訓練完後進行預測。第二輪的時候，將前一輪(這裡是第一輪)預測錯誤的資料，權重增加，預測正確的資料，權重減少。這樣的話，在第二輪的訓練中，會加強在上一輪中預測錯誤的資料、難以分辨的資料，訓練完後進行預測。假設有 n 個 classifier，經過 n 輪後，用 n 個訓練好的 classifier 進行預測，並用 majority voting 得到最後的預測。

上述是 Adaboost 用在分類的問題上面，而 [15] Adaboost.R2 是基於 [16] Adaboost.R 的修改，Adaboost.R2 用來解決回歸(Regression)問題。Adaboost.R2 對整個 training data 做 bootstrap 且重複抽樣，每一輪會根據 weak regressor 的錯誤率來更新下一輪該 training sample 會被抽中的機率，和 ground truth 差異越多越容易被抽中，同時也根據錯誤率來賦予 weak regressor 的權重，錯誤率越小，和 ground truth 的差異越小，權重越高。最後從各個 weak regressor 中，取權重中位數所預測的值，作為最後得到的預測值。

2.4 技術指標的取得，TA-Lib

[17] 可以幫忙計算出技術指標的函式庫，底層核心是使用 C/C++，以 Cython 製造而成的 python wrapper，python 可以透過 python wrapper 呼叫 C 函式庫。能夠計算出 200 個指標，其中有 MACD、RSI、包寧傑帶狀。也能夠判斷 K 線圖的型態，其中一個型態有早晨之星(morning star)。Ta-lib 廣泛地被用在，需要金融市場資料的技術分析，交易軟體的開發。採用 BSD License，可以自由地整合進自己的開源軟體，並且應用在商業用途上。TA-lib 於 2001 年發布，並被使用了 20 年以上，是個相當穩定，且經得起時間考驗的函式庫。

三、Transformer 模型之股票預測

3.1 神經網路架構

圖 3-1 為 Transformer 股票預測的架構，也是作為本論文 Adaboost.R2 的 weak learner，此架構是參考 [6]。 [13] 表示深度學習股票預測的應用是近來才開始的，那時候對於前後文掌握度較佳的是 LSTM 模型。 [18] Transformer 於 2017 年發表，比 LSTM 更優秀的時序掌握模型的問世後， [19, 20, 21] 許多應用 Transformer 於時序資料的論文如雨後般的春筍一一出現， [22] 也有基於 Transformer 在時序資料處理上的強化 Informer，但 Transformer 在股票預測上的文章相對較少。於是試圖在既有的成果上 [6]，看能否做出貢獻。

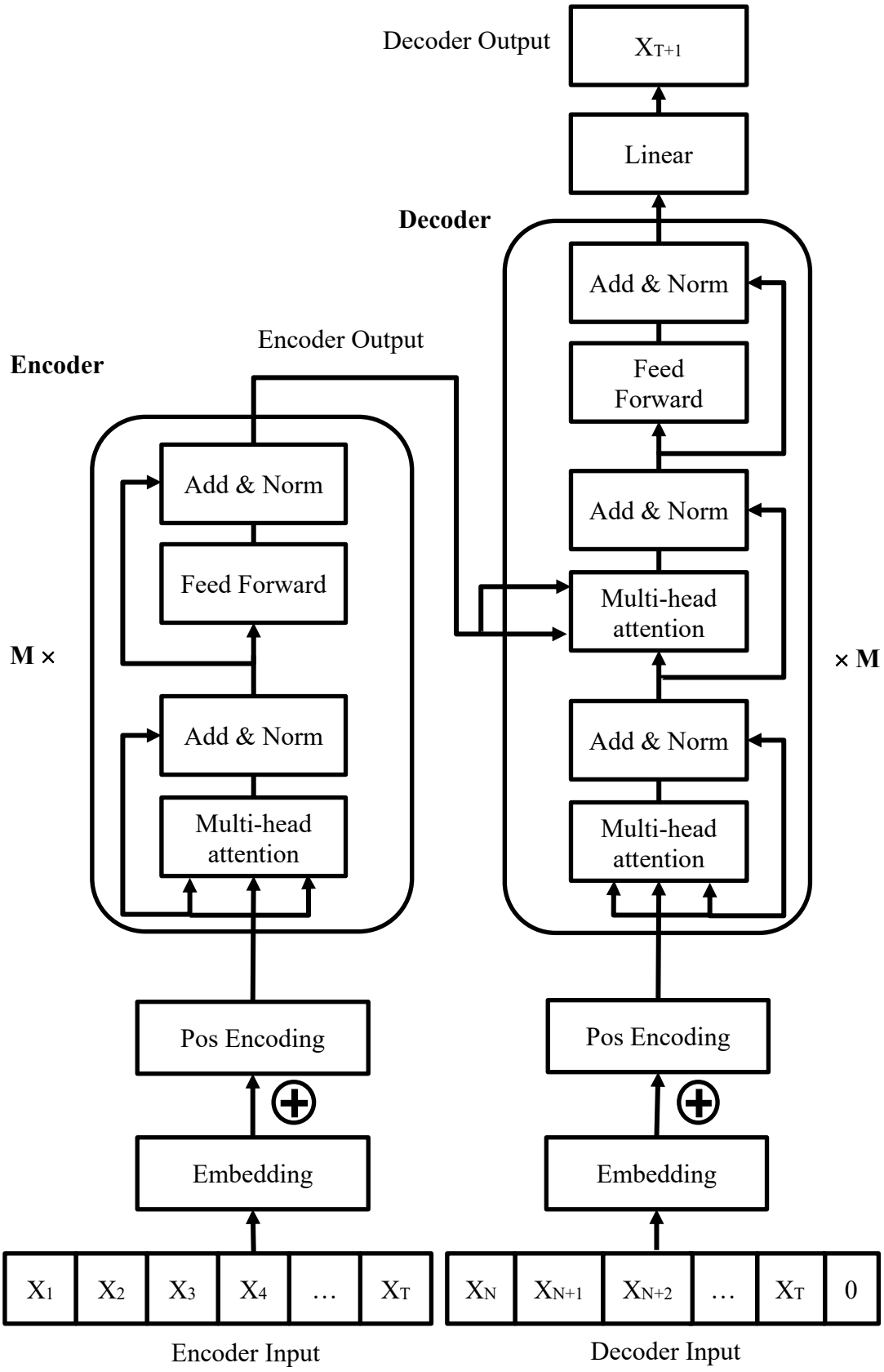


圖 3-1、Transformer 股票預測架構

使用 Transformer 來預測未來一天的股票收盤價。像 [18] 對自然語言做 embedding 那樣，在此也對 encoder input 做 embedding。與自然語言不同的是，這邊是用 fully connected network。令 encoder input 為 $X = \{x_t : 1, \dots, T\} \in R^T$，X 經過 d 維度的 embedding 後，產生出矩陣 $A \in R^{T\times d}$

因為 Transformer 本身沒有時序地概念，使用 positional encoding 讓模型能得到時序的特徵，利用 sine 與 cosine 的不同頻率來製造具有獨一無二性且具規律相似性的資料特性，這些特性賦予了資料具有順序性，如公式(1)。t 代表著 1 到 T 裡面的 t_{th} 天的收盤價，在 1 到 T 之間給予相對距離。產生出 PE 矩陣，$PE \in R^{T\times d}$。t_{th} 天收盤價用 d 維度來表示，d 維度中的偶數用 sine，奇數用 cosine，將運 算結果存至 PE 的對用位置。像是 $PE_{(1,2)} = PE[1, 2] = \sin(1/10000^{2*2/d})$，$PE_{(1,3)} = PE[1, 3] = \cos(1/10000^{2*3/d})$。將得到的 A 和 PE 做 concatenate 得到矩陣 Concat_M，$Concat\_M \in R^{T\times 2d}$。

$$PE_{(t,2s)} = \sin(t/10000^{2s/d}),$$
$$PE_{(t,2s+1)} = \cos(t/10000^{2s/d}) \quad (1)$$

[23] Transformer encoder 和 decoder 都有使用 residual connection 和 layer normalization，兩者皆可以改善梯度消失的問題，幫助模型更好地學習。兩邊也同時使用了 [18] attention 機制，如公式(2)。用 Q 和 K 算出關係權重除以根號 d 達到 normalize 的效果，用這個權重乘上 V 後經 activation function softmax 得到 attention score，$Q \in R^{T\times d}$，$K \in R^{T\times d}$，$V \in R^{T\times d}$，Q 為 query、K 為 key、V 為 value 矩陣。

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d}})V \quad (2)$$

圖 3-2 Q、K、V 矩陣是由 PE 和 A 的 concat 所得到的 Concat_M，Concat_M 透過三個 linear layer 權重分別為 W^q、W^k、W^v 而得到。有了 Q、K、V，就可以帶入公式(2) 也就是圖 3-3 的運算得到收盤價與收盤價之間的 attention score。Self-attention 讓 1 ~ T 間的日收盤價關係可以互相比較，查看彼此關係，第 1 天收盤價和 1 ~ T 天收盤價做比較，第二天收盤價和 1 ~ T 天做比較，以此類推，所有天數的收盤價都能彼此比較關係。

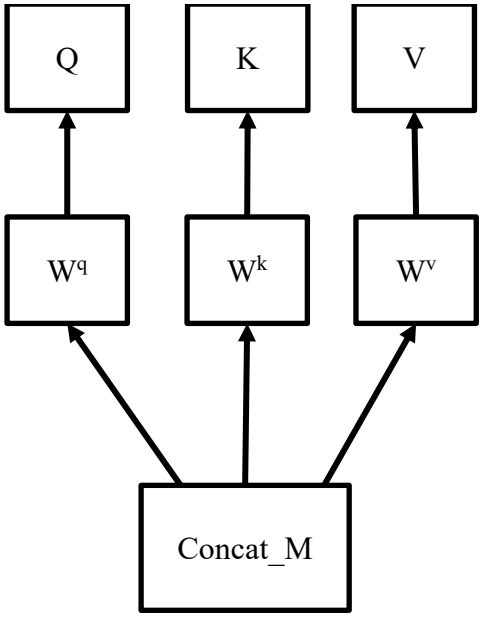


圖 3-2、self-attention 架構_1

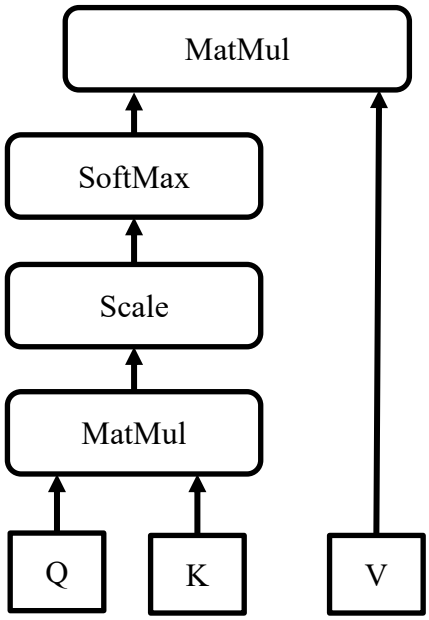


圖 3-3、self-attention 架構_2

由多個 attention 組成 multi-head attention 公式(3)，多個 attention score 做 concatenate 後，經由權重 W^O 的 linear layer 得到輸出，效果比單一 attention 來的更好。圖 3-4 是 multi-head attention 的架構圖，以本論文來說，圖 3-4 的 Q、K、V 在圖 3-1 在做 self-attention 的時候都是 Concat_M，linear layer 則是如圖 3-2，分別為 W^q、W^k、W^v 權重的 linear layer，h 是 multi-head attention 的 head 數量。

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1,\text{head}_{2,\ldots},\text{head}_h)W^O \quad (3)$$
$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

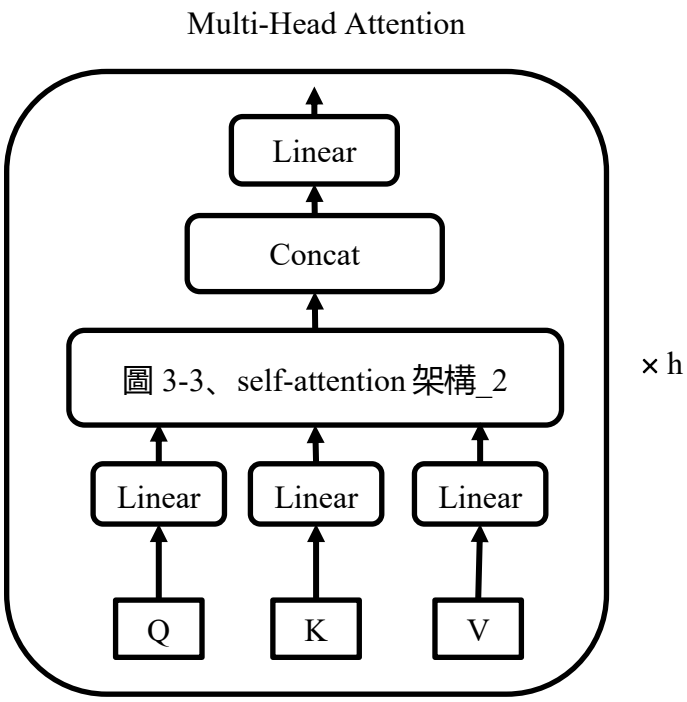


圖 3-4、Multi-Head Attention with several attention layers

Encoder 將輸入的資料關係萃取出來，輸出給 decoder，讓 decoder 使用提取到的資料關係來做預測。Encoder output 的資料作為 attention 機制的 key 和 value 輸入進 decoder 的 multi-head attention，decoder 的部分則是使用從 decoder 的 self-attention 的 output 做為 query，實現 cross attention。圖 3-1 的架構和 [18]的差別是，decoder 的 self-attention 沒有使用 look ahead mask，因為 decoder input 全部都是歷史資料，不會有看到未來答案的問題。收盤價在 self-attention 時，得到了 local information，藉由 cross attention 比較 1 ~ T 和 2 ~ T 的關係，1 和 2 ~ T，2 和 2 ~ T 比較，依序比較。憑藉著 self-attention 和 cross attention 的資訊來預測 T + 1 的收盤價。最後，圖 3-1 的 decoder input 最右邊的 0 是 zero padding。

3.2 訓練資料集處理

如果碰到有資料缺失的情況，直接選擇不用該日的資料。將欲訓練的收盤價按照 8:2 切分成 training data 以及 validation data，避免讓資料有大幅度的震盪，進而影響到模型的學習效果，按照公式(4)將所有資料(training data 加 validation data)做 normalize。$\hat{x}_t$是在時間 t 已經 normalize 的收盤價，x_t 是時間 t 尚未 normalize 的原始收盤價。μ和σ分別為整個 training data 的平均和標準差。

$$\hat{x}_t = \frac{x_t - \mu}{\sigma} \tag{4}$$

將 normalize 過的資料依照圖 3-5 準備，圖 3-5 顏色塗滿的是各組的 label data，前面 1 ~ T 天收盤價為第一組的 feature data，encoder input data shape 為(batch size,

T)，target data shape 為(batch size)。使用 moving window 前面 T 天的收盤價為 features，第 T+1 天的收盤價為 label data。由前面 T 天的收盤價來預測 T + 1 天的收盤價。若 T 為 9 的話，由 $x_1, x_2, \ldots, x_9$ 來預測 x_{10}，第 10 天的收盤價。下一組，由 $x_2, x_3, \ldots, x_{10}$，預測第 11 天的收盤價 x_{11}，依此規則逐一預測未來一天的收盤價。

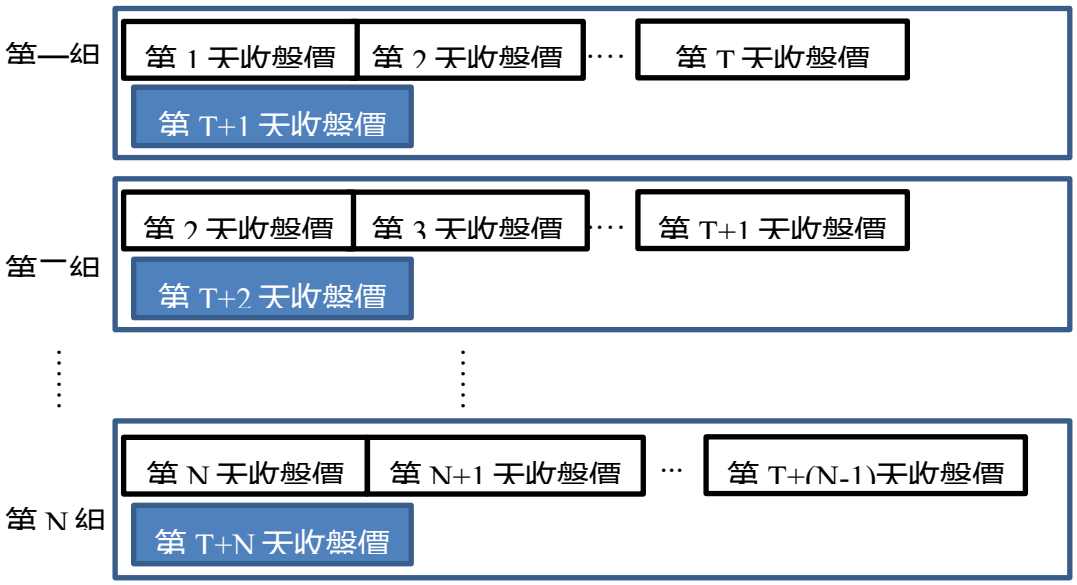


圖 3-5、Encoder input data 範例和 label data

3.3 模型訓練及測試

[6]這邊介紹 hyperparameter 的設置，batch size 為 16，使用 mean square error(MSE)(5)為 loss function，使用 MSE 是假設資料符合常態分布。$\hat{y}_i$是預測值，y_i是 ground truth，N 是 sample size。

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 \tag{5}$$

Adam optimizer 和 learning rate 0.0001，1000 epoch。Dropout rate 0.1，input length T = 9，decoder input 的 N = 2。Embedding dimension，d = 32，所有的訓練權重初始值 wrights 和 biases，都各別從 uniform distribution[-1, 1]之間抽樣出來。

將前處理好的資料輸入進模型中訓練，訓練中會將資料切分成一個 batch 為 16，剩下資料不足 16 的話，直接將剩下的歸為一組 batch。每一個 epoch 訓練完後，會用當下 epoch 訓練好的模型對 validation data 進行預測，算出其 MAE。算出的 MSE 為其 val_loss 值，若 val_loss 值比之前的都還要低，代表當下的模型權重是截至目前為止效果最好的模型，將其模型權重儲存起來，整個訓練結束後，我們可以得到 val_loss 最小的模型權重，得到最好的預測結果。

將最佳權重載入至模型中，前處理好的資料輸入進該模型做預測，用得到的預測值來計算 MSE 和 MAE。比較 MSE 和 mean absolute error(MAE)(6)來判斷

模型效果。$\hat{y}_i$是預測值，y_i是 ground truth，N 是 sample size。MAE 比較出預測值和 ground truth 的絕對誤差，讓誤差不會被一正一負抵銷掉。

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|\hat{y}_i - y_i| \tag{6}$$

MSE 和 MAE 越小，效果越好。把預測收盤價曲線圖畫出來，觀察趨勢吻合程度。

四、結合元學習與 Transformer 之股票預測

4.1 神經網路架構

這邊可以把新的資料結構和 3-1 的差距描述出來。

圖 4-1 中的 Transformer 為圖 3-1Transformer 架構，在 Adaboost.R2 的 framework 之下，由數個 Transformer 做為 weak learner。

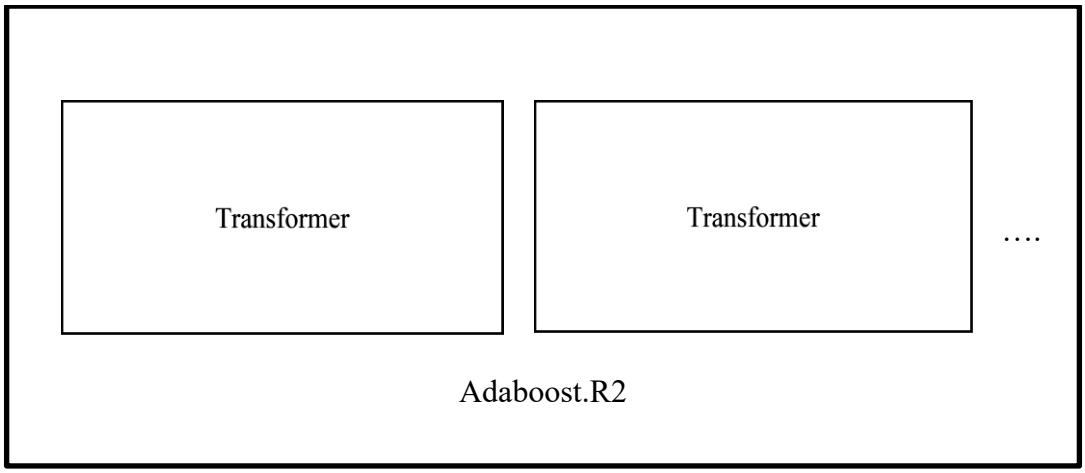


圖 4-1、元學習模型架構

[15] Adaboost.R2 演算法初始給每個 training pattern 賦與權重 $w_i = 1$, $i = 1,\ldots,N_1$。接下來重複以下步驟，直到下面步驟中的 average loss $\bar{L} \geq 0.5$ 或 weak learner 全數訓練完畢即停止。

1. 每個 training sample i 存在於該 training set 的機率 $p_i = w_i / \sum w_i$，$\sum w_i$為整個 training set 裡面的 member 權重總合。採重複抽樣，從數值範圍$[0, \sum w_i]$抽樣 N_1 個 training samples 來組成 training set。抽樣到的 N_1 個 training samples 作為 index，對應到 training pattern，完成一次 bootstrap，得到一組 feature 和 label data。

2. 建構 weak learner，這裡的 weak learner 為 Transformer。假設每個 weak learner 會將 feature data 映射到 label data，$h_t: x \rightarrow y$。

3. 將 training set 中的每個 member 輸入進步驟 2 中建構的 weak learner 得到預測值$y_i^{(p)}(x_i)$ $i = 1,\ldots,N_1$。

4. 本論文選擇的 loss function 為 linear，計算每個 training sample 的 loss 值，$D = \sup \left| y_i^{(p)}(x_i) - y_i \right|$ $i = 1,\ldots,N_1$，$L_i = \frac{\left| y_i^{(p)}(x_i) - y_i \right|}{D}$，$L_i$ 式子中除以 D 確保了 $L_i \in [0,1]$。

5. 計算 average loss: $\bar{L} = \sum_{i=1}^{N_1} L_i p_i$。

6. $\beta = \frac{\bar{L}}{1-\bar{L}}$，β是預測器的信心量測指標，β越低代表預測的信心程度越高。

7. 更新權重: $w_i \rightarrow w_i \beta^{1-L_i}$，Li 越小權重減的越多，在下一輪該 pattern 被選為 training set 的成員的機率越低。

8. 完成重複步驟 1 ~ 7 的訓練步驟後，來到 adaboost.R2 的預測階段。今天一個 input x_i，假設總共 T 個 weak regressor，每個 weak regressor 對 x_i 做預測 h_t, t=1,…,T，得到了累積預測h_f，這個就是模型權重中位數。

$$h_f = inf\left\{y \in Y:\ \sum_{t:h_t \le y} \log\left(\frac{1}{\beta_t}\right) \ge \frac{1}{2}\sum_t \log\left(\frac{1}{\beta_t}\right)\right\} \quad (7)$$

每個 weak regressor h_t 有其預測值$y_i^{(t)}$，第 i 個 pattern 和相對應的β_t。所有的 weak regressor 針對 pattern i 做預測，得到的預測值由小到大排列，weak regressor 及其權重對應著預測值的排列。也就是$y_i^{(1)} < y_i^{(2)} < \ldots < y_i^{(T)}$，$\beta_t$對應 $y_i^{(t)}$。這時候將 $\log(1/\beta_t)$累積相加，一直加到可以剛好滿足(7)的不等式，所得到的最小剛好滿足不等式的 t，weak regressor t 的預測值即為 Adaboost.R2 的預測值。

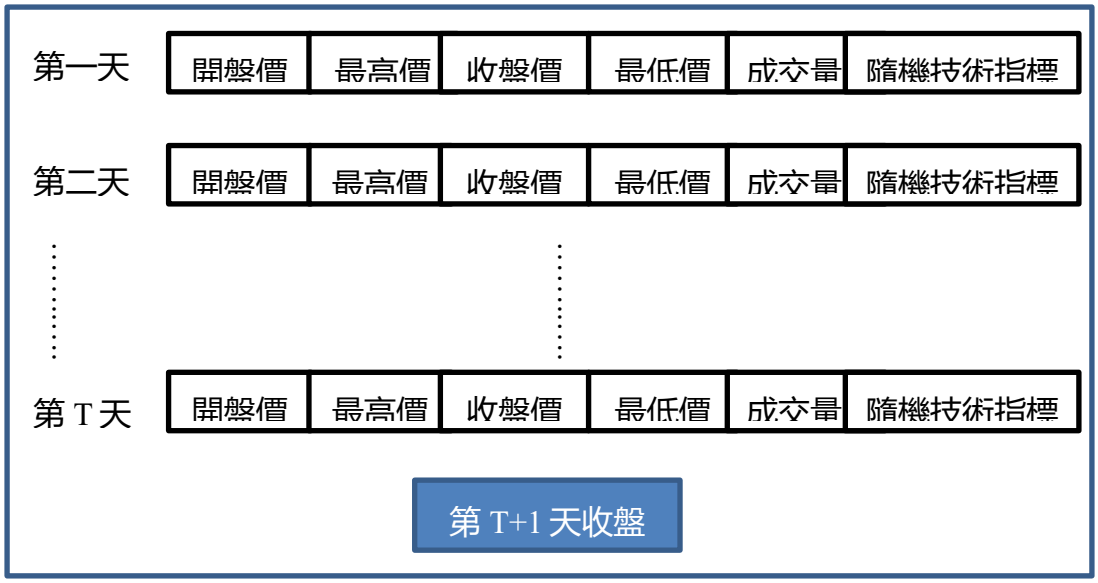
4.2 訓練資料集處理

要把技術指標間的比較，是透過 attention 的 QKV 矩陣乘法給描述出來。

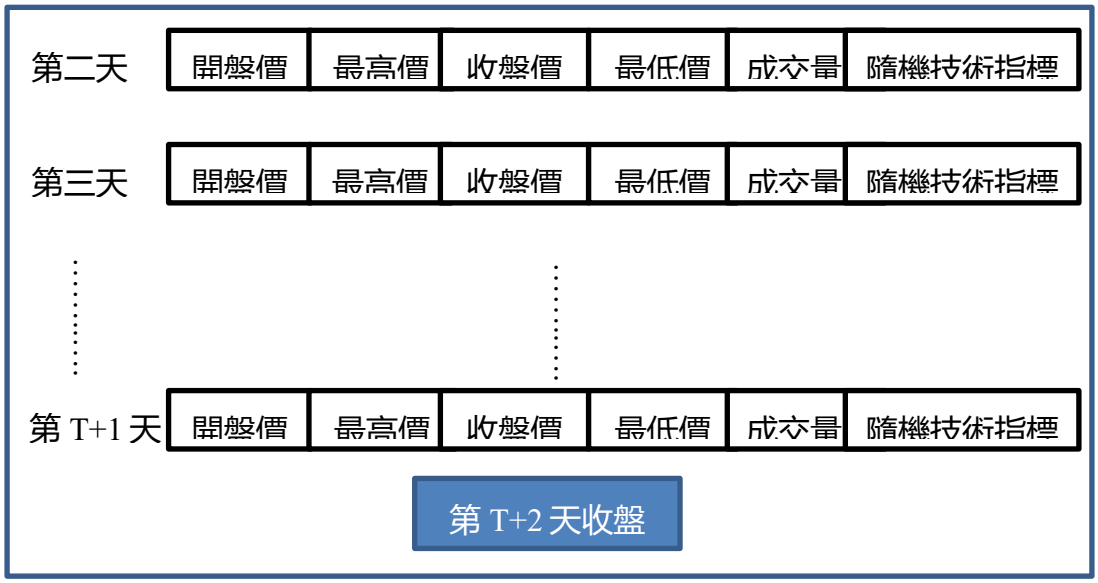
如果碰到有資料缺失的情況，直接選擇不用該日的資料。將欲訓練的收盤價按照 8:2 切分成 training data 以及 validation data，避免讓資料有大幅度的震盪，進而影響到模型的學習效果，按照公式(4)將所有資料(training data 加 validation data)做 normalize。和 3.2 的差別是，這次除了收盤價，還增加了開盤價、最高價位、最低價位。開、高、收、低、成交量和隨機技術指標個別求算各自的平均值和標準差。像是用開盤價的 training data 的部分算出開盤價的平均值和標準差，然後對整個開盤價資料進行 normalize。由於在計算隨機技術指標的標準差時，數值會過大超過 numpy.float64 的範圍，導致標準差變成 inf，按照(4)除以無限大的標準差，會讓所有 normalized 的隨機技術指標全部變成 0.0。為了解決這問題，將大於等於 10^{10} 的隨機技術指標數值設定成 NaN，然後先把整個 column 為 NaN 的 column 捨棄掉，接著再把有任一 NaN 的 row 整個捨棄掉。明明 numpy.float64 的範圍很大，為什麼選 10^{10} 是因為這樣能保留住較多的 row data。以台積電股票案例，會有 88 個 row，也就是 88 個天數，以及 5 個 column，也就是 5 種技術指標的資料被捨棄掉。

資料中的隨機技術指標是採用蒙地卡羅方法，在台積電股票中，捨棄掉過大的數值後，從剩下的 169 種技術指標中，隨機抽樣一個出來和相對應的開、高、收、低、成交量做結合。在大數法則情況下，抽樣的量越多，抽樣空間會越趨進於母體。中央極限定理表示樣本數越多，樣本平均會趨近於常態分佈，其平均數會等於母體平均數。以台積電股票案例來說，先將 2014/1/1 ~ 2023/12/31 視為一組資料，將一組資料按照 8:2 切分成 training data 和 validation data 後，將 training data 的部分(0.8 的部分)，裡面每一日相對應的開、高、收、低、成交量加上 隨機額外的一個技術指標；將 validation data 的部分(0.2 的部分)裡面每一日相對應的開、高、收、低、成交量加上 隨機額外的一個技術指標，然後重複這動作，直到湊滿 10 組。有考慮過先就一組來算標準差，然後用這標準差來對所有資料做 normalized，這樣或許也能避免掉資料計算過大導致的 inf。但是，因為樣本標準差和母體標準差的自由度有差，一個為 N-1，一個為 N，終究是不一樣的，最後還是選擇將過大的數值給捨棄掉的方法。透過大量的隨機抽樣，讓所有的開、高、收、低、成交量都能搭配到所有的隨機技術組合，藉此達到所有技術組合的任一搭配，得到所有任意搭配的股票預測效果。

將 normalize 過的資料依照圖 4-2 準備，將同一天的開、高、收、低、成交量和隨機技術指標放在一起，每 T 天的資料視為一組 feature data，相對應這組的 label data 為 T+1 天的收盤價。1 ~ T 天為 feature data，T+1 為 label data，為第一組；2 ~ T+1 天為 feature data，(T+1)+1=T+2 天為 label data，為第二組，依此類推。Encoder input data shape 為(batch size, T, 6)。



第一組



第二組

圖 4-2、Encoder input data 範例和 label data

4.3 模型訓練及測試

Weak learner Transformer 的 hyperparameter 的設置，batch size 為 16，使用 mean square error(MSE)(5)為 loss function。Adam optimizer 和 learning rate 0.0001，1000 epoch。Dropout rate 0.1，input length T = 9，decoder input 的 N = 2。Embedding dimension，d = 32，所有的訓練權重初始值 wrights 和 biases，都各別從 uniform distribution[-1, 1]之間抽樣出來。Adaboost 的 hyperparameter 為 5 個 weak learner。

將前處理好的資料輸入進 adaboost.R2 框架中，詳細訓練過程在 4.1。從輸入進 adaboost.R2 框架中的資料中，採用 bootstrap 可重複抽樣，抽樣至原始資料的量後，輸入進第一個 weak learner 訓練，訓練中會將資料切分成一個 batch 為 16，剩下資料不足 16 的話，直接將剩下的歸為一組 batch。每一個 epoch 訓練完後，會用當下 epoch 訓練好的模型對 validation data 進行預測，算出其 MAE。算出的 MSE 為其 val_loss 值，若 val_loss 值比之前的都還要低，代表當下的模型權重是截至目前為止效果最好的模型，將其模型權重儲存起來，整個訓練結束後，我們可以得到 val_loss 最小的模型權重。將最佳權重載入至第一個 weak learner 中，對 training data 做預測，得到的預測值和 ground truth 算 loss 值，根據 loss 值更新資料權重，錯誤率較高的資料，下一輪抽樣時多增加該資料被抽重的機率，達到針對錯誤率高的資料做重點訓練，也根據 loss 值來計算模型權重，這邊的模型權重不是訓練模型時的權重，而是 adaboost.R2 框架下該框架會賦予所有 weak learner 的權重，供後續 adaboost.R2 預測時使用。第二個 weak learner 針對上一輪的失誤率較高的資料做重點抽樣，訓練、載入第二個 weak learner 的最佳權重、預測然後計算 loss 值、得到 adaboost.R2 賦予的權重，重複至所有 weak learner 皆訓練完畢。

前處理好的資料輸入進 5 個訓練好的 weak learner，5 個 weak learner 皆對輸入進來的 validation data 做預測，分別得到 5 個預測值，按照先前 adaboost.R2 賦予的權重做排序，最後取權重中位數的模型所給予的預測值，當作整個 adaboost.R2 最後的預測值。由 adaboost.R2 的預測值來計算 MSE 和 MAE，比較 MSE 和 MAE(6)來判斷 adaboost.R2 框架的效果。

五、實驗結果

5.1 實驗資料準備/實驗環境

本論文使用的資料總量為 10 年的資料量，想了解本論文提出的方法是否能用作股票收盤價預測，所以使用各種股票來做實驗。有使用台灣的台積電從 2013/1/1 ~ 2023/12/31，孟加拉國的 AGRANINS 從 2013/1/1 ~ 2022/7/13。實驗的緩硬體配置，可以從表 5-1 得知。

表 5-1、實驗軟硬體配置

CPU	Intel® Core™ i5-12400
GPU	NVIDIA GeForce RTX 3060
Anaconda version	23.11.0
Python version	3.11.5
Tensorflow version	2.15.0

5.2 Transformer 股票預測效率分析

5.3 元學習股票預測效率分析

5.4 元學習和 Transformer 測試資料集的預測準確度比較

5.5 問題與討論

本實驗僅使用了常見的技術指標來實驗，旨在確認元學習結合 Transformer 的方法，能否讓機器在常見的技術指標中自行搭配，進而和只使用單一收盤價技術指標相比，得到效能提升。確認這個方法可以找到技術指標間的關係後，指標間，是。

六、結論、未來展望

本論文發現將原本就可以預測股價的 Transformer，用集成式學習，將 Transformer 結合在一起，得到更好的預測結果。

既然將本來可以的方法

參考文獻

[1] J. Sha, Deep learning models in financial technology, 2022, p. 50.

[2] 林逸青, 以深度學習建構股價預測模型:以台灣股票市場為例, 臺中市: 國立臺中科技大學, 2019.

[3] 洪御仁, 應用機器學習於台灣 50 股價預測分析之研究, 新北市: 華梵大學, 2022.

[4] Wenjie Lu, Jiazheng Li, Yifan Li, Aijun Sun, and Jingyang Wang, "A CNN-LSTM-Based Model to Forecast Stock Prices," *Complexity,* p. 10 pages, 24 11 2020.

[5] Tashreef Muhammad, Md. Mainul Ahsan, Muhammad Ibrahim, Anika Bintee Aftab, Maishameem Meherin Muhu, Shahidul Islam Khan, and Mohammad Shafiul Alam, Transformer-based deep learning model for stock price prediction: a case study on Bangladesh stock market, Dhaka, Bangladesh: Ahsanullah University of Science and Technology, 2022.

[6] Chaojie Wang, Yuanyuan Chen, Shuqi Zhang, Qiuhui Zhang, "Stock market index prediction using deep Transformer model," *Expert Systems With Applications,* 2022.

[7] 郭兆倫, 基於強化式深度學習網路增強外匯自動交易程式效能, 基隆市: 國立台灣海洋大學, 2019, p. 36.

[8] Benjamin Graham, David L. Dodd, 證券分析 上冊(顏詩敏、李羨愉), 2021.

[9] 佐伯良隆, 憑直覺看懂會賺錢的財務報表, 2019.

[10] J. J. Murphy, 金融市場技術分析 上冊(暢銷經典版，黃嘉斌譯), 寰宇出版股份有限公司, 2021.

[11] Xiaodong Li, Pangjing Wu, Wenpeng Wang, "Incorporating stock prices and news sentiments for stock market prediction: A case of Hong Kong," *Information Processing and Management,* 21 1 2020.

[12] Wen Long, Zhichen Lu, Lingxiao Cui, "Deep learning-based feature engineering for stock price movement prediction," *Knowledge-Based Systems,* pp. 163-173, 15 1 2019.

[13] Mahinda Mailagaha Kumbure, Christoph Lohrmann, Pasi Luukka, Jari Porras, "Machine Learning techniques and data for stock market forecasting: A literature review," *Expert Systems With Applications,* 19 2 2022.

[14] Sebastian Raschka, Vahid Mirjalili, Python Machine Learning, Packt Publishing Ltd., 2017.

[15] Drucker, "Improving Regressors using Boosting Techniques," 1997.

[16] Y. Freund, R. Schapire, ″A Decision-Theoretic Generalization of on-Line Learning and an Application to Boosting,″ *Journal of Computer and System Sciences,* 19 12 1996.

[17] ″TA-Lib - Technical Analysis Library,″ [線上]. Available: https://ta-lib.org/.

[18] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin, ″Attention is all you need,″ 於 *31st Conference on Neural Information Processing Systems (NIPS 2017)*, California, 2017.

[19] R. Mohammdi Farsani, E. Pazouki, ″A Transformer Self-Attention Model for Time Series Forecasting,″ *Journal of Electrical and Computer Engineering Innovations,* pp. 1-10, 21 11 2020.

[20] Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, Xifeng Yan, ″Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting,″ 於 *Neural Information Processing Systems*, 2019.

[21] Bryan Lim, Sercan Ö. Arık, Nicolas Loeff, Tomas Pfister, ″Temporal Fusion Transformers for interpretable multi-horizon time series forecasting, ″ *International Journal of Forecasting,* pp. 1748-1764, 12 2021.

[22] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, Wancai Zhang, Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting, 2020.

[23] F. Chollet, Deep Learning with Python, Second Edition 2nd, 2021.