

# 利用 AI 進行股市回測

鄭楷勳(113061201)

## 1 摘要

本研究透過使用 AI(claude 3.7 sonnet)，生成 Pinescript 程式碼，並且用於 Tradingview 進行股市回測。研究重點在於評估五日移動平均線策略在 1900 年至 2025 年期間對 SPX(標普 500)指數的影響，同時比較定期定額與單筆投入有什麼差別。統計的結果指出，技術分析在過去的市場中具有一定效果，但在 2000 年後效果減弱，甚至出現反效果。並且在利用 AI 生成程式碼並修正錯誤後，提出 AI 的優點及問題所在，與未來潛在的應用。

## 2 介紹

金融市場的發展使技術分析逐漸受到重視，但是現在的市場趨於高效率，使技術分析的有效性備受爭議。透過使用不同的 prompts 產生精確的程式碼、將 AI 生成的程式碼應用於 Tradingview 並測試不同的交易策略、得出實際的統計成果並用於分析不同技術策略帶來的結果、找出 AI 的優勢與劣勢，並找出 AI 適用的場景。

## 3 實際操作

初步設計 prompts，使程式碼可以在 Tradingview 上運行，並用以回測當日收盤價相對於五日移動平均線的位置（高於或低於），進而統計次一交易日的漲跌次數及平均漲跌幅。雖然一開始的程式碼可以運行，卻不能逐日檢查數據，無法判斷程式是否正常運行。接著輸入新的 prompt(在圖表中顯示累積的漲跌天數變化，且看到平均漲(跌)幅變化，並且可以自由調整從哪一天開始計算，計算總共有多少個交易日)，並將程式碼輸入 Tradingview 測試，並得到以下的結果：



圖一、Tradingview Chart 頁面，上方是股價，下方是程式輸出的圖表  
(資料來源 <https://tw.tradingview.com/chart/?symbol=AMEX%3ASPY>)

雖然程式碼看似正確，並且有列出上漲與下跌的狀況，但經過逐日檢查漲跌情況，發現漲跌的計算有出現錯誤。如前一天的股價在五日均線上，如果下一個交易日的股價低於五日均線，卻沒有納入統計的狀況。接下來是新的 prompt(只要判斷前一天收盤價大於均線，隔日收盤價無論是否高於均線都要納入統計。若收盤沒有漲也沒有跌，則不計入統計)，這時候程式碼出現錯誤，如下圖

```
68 label.new(
1 個問題 (共 2 個)
Mismatched input 'end of line without line continuation' expecting ')
```

圖二、Tradingview 程式碼編輯頁面

(資料來源 <https://tw.tradingview.com/chart/?symbol=AMEX%3ASPY>)

將問題回傳給 AI，AI 也有給出錯誤的原因與解決方法

## 解決Pine Script中的行連續性錯誤

您在 `label.new()` 這行遇到的問題是Pine Script中常見的"行連續性錯誤" ( line continuation error )。這個錯誤訊息"Mismatched input 'end of line without line continuation' expecting ')'"表示Pine Script編譯器預期一個函數調用應該繼續，但沒有看到正確的行連續語法 <sup>4</sup>。

### 錯誤原因

根據搜索結果，這個問題的原因是Pine Script有特殊的行包裝規則 <sup>2</sup>：

1. 當您希望將表達式或函數調用分成多行時，必須使用不是4的倍數的空格進行縮排
2. 這意味著您應該使用1、2、3、5、6、7等空格（但不是0、4、8、12等）
3. 這是因為4的倍數縮排在Pine Script中保留用於if語句、for循環和自定義函數 <sup>2</sup>

### 解決方法

要修正 `label.new()` 這一行的錯誤，您需要確保後續行使用正確的縮排。例如：

```
text
// 錯誤的寫法
if barstate.islast
    label.new(
        bar_index, 0, // 這裡使用了4個空格的縮排，這會導致錯誤
        text="前一日高於五日均線後的漲跌統計"
    )

// 正確的寫法
if barstate.islast
    label.new(
        bar_index, 0, // 使用1或3個空格
        text="前一日高於五日均線後的漲跌統計",
        color=color.blue,
        textcolor=color.white
    )
```

圖三、Perplexity 輸出的結果(使用模型為 claude 3.7 sonnet)

(資料來源 <https://www.perplexity.ai/search>)

結果看似正確，AI 也指出了問題，不過在實際跑程式碼時，還是會出現相同的錯誤。新的 `prompt` 命令 AI 必須使用其他的程式碼來達到相同的效果，於是得到以下成果：



圖四、程式碼修改後的輸出圖(高於五日均線)

程式碼可以成功運作，接著是更進一步調整回測的區間(設定起始時間與結束時間)，又出現新的錯誤

```
27 var time last_start_date = 0 // 上次的開始日期
```

1 個問題 (共 2 個)

'time' is not a valid type keyword.

圖五、程式碼編輯頁

經過修改後，可以顯示出統計時間，並得出以下結果



圖六、可調整計算區間的輸出圖

並且利用以下參數設定可以自訂數據回測的區間

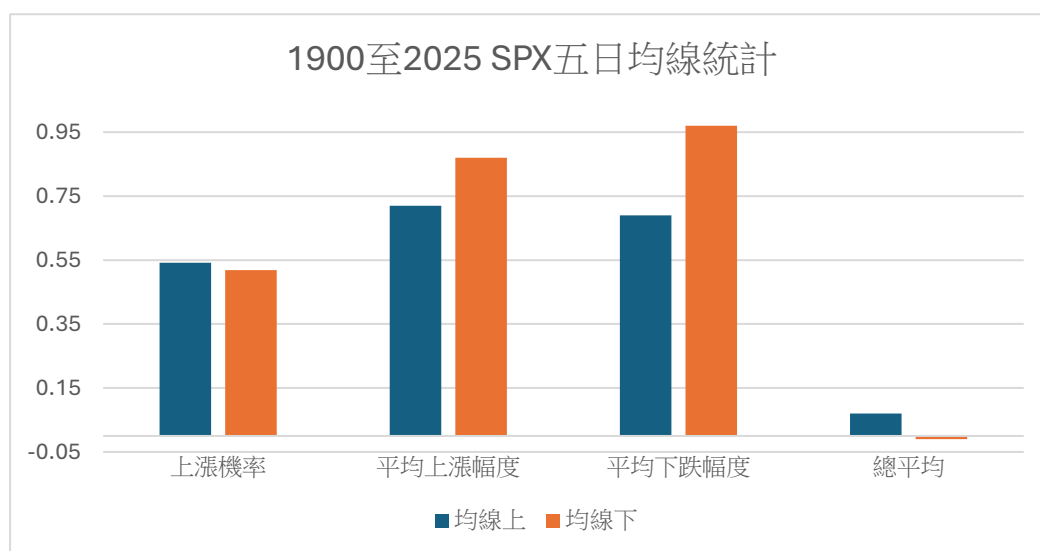
```
// 參數設定
ma_length = input.int(5, title="移動平均線長度", minval=1)
start_date = input.time(timestamp("2023-01-01"), title="開始日期", confirm=true)
end_date = input.time(timestamp("2025-05-11"), title="結束日期", confirm=true)
```

接著再製作一個低於五日均線的版本，就可以找出均線對股價的影響，如下



圖七、程式碼修改後的輸出圖(低於五日均線)

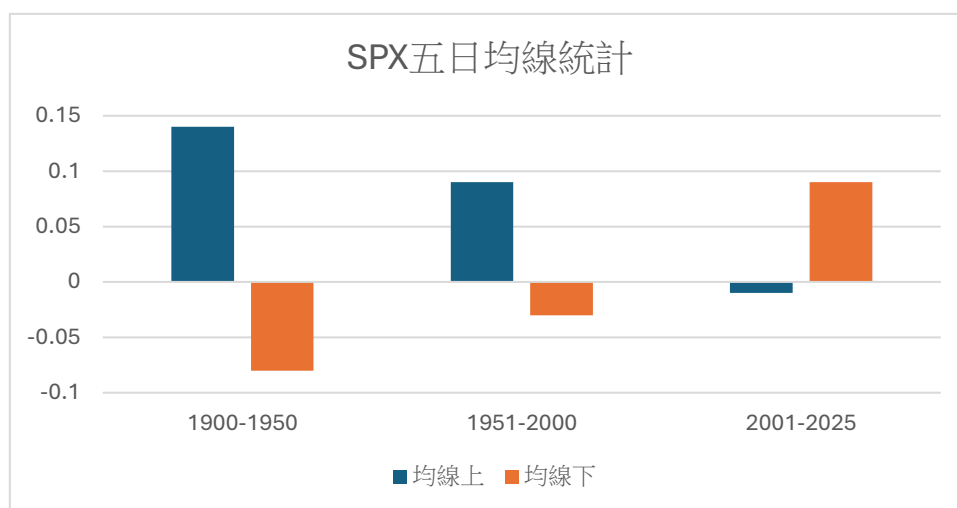
最後統整所有資料，得到以下結論(以下內容使用 SPX 作為統計資料)



圖八、1900 至 2025 年 SPX 五日均線統計數據

若前一天的收盤價高於五日均線，隔日平均上漲 0.07%；如果低於五日均線，隔日則平均下跌 0.01%。得到這個結論後，我想再進一步探討技術分析在現今更效率化的市場當中，技術線的效果是否也如過去一百多年的一樣顯著。

如果將統計時間分成 1900-1950、1951-2000、2001-2025，可以得到下圖



圖九、不同時間區間的 SPX 五日均線統計圖

從這個圖表的資料可以知道，雖然技術分析在 2000 年以前效果顯著，但統計資料指出在 2000 年以後，反而收盤價低於均線時，隔天上漲的機率更高。因此，技術分析在過去的市場是有效的，而對現在的市場則是效果不顯著(甚至會得到反效果)。詳細資料如圖

|           |     |             |           |           |       |      |      |
|-----------|-----|-------------|-----------|-----------|-------|------|------|
| 1900-2025 |     | 上漲機率        | 平均上漲幅度    | 平均下跌幅度    | 總平均   | 上漲天數 | 下跌天數 |
|           | 均線上 | 0.541960094 | 0.72      | 0.69      | 0.07  | 7388 | 6244 |
|           | 均線下 | 0.518390912 | 0.87      | 0.97      | -0.01 | 5567 | 5172 |
| 1900-1950 |     | 上漲機率        | 平均上漲幅度    | 平均下跌幅度    | 總平均   | 上漲天數 | 下跌天數 |
|           | 均線上 | 0.526997158 | 1.14      | 0.99      | 0.14  | 1669 | 1498 |
|           | 均線下 | 0.521436848 | 1.24      | 1.52      | -0.08 | 1350 | 1239 |
| 1951-2000 |     | 上漲機率        | 平均上漲幅度    | 平均下跌幅度    | 總平均   | 上漲天數 | 下跌天數 |
|           | 均線上 | 0.557329881 | 0.58      | 0.53      | 0.09  | 3874 | 3077 |
|           | 均線下 | 0.499459459 | 0.65      | 0.7       | -0.03 | 2772 | 2778 |
| 2001-2025 |     | 上漲機率        | 平均上漲幅度    | 平均下跌幅度    | 總平均   | 上漲天數 | 下跌天數 |
|           | 均線上 | 0.525042686 | 0.63      | 0.71      | -0.01 | 1845 | 1669 |
|           | 均線下 | 0.555769231 | 0.97      | 1.01      | 0.09  | 1445 | 1155 |
|           |     | 1900-1950   | 1951-2000 | 2001-2025 |       |      |      |
|           | 均線上 | 0.14        | 0.09      | -0.01     |       |      |      |
|           | 均線下 | -0.08       | -0.03     | 0.09      |       |      |      |

圖十、詳細數據

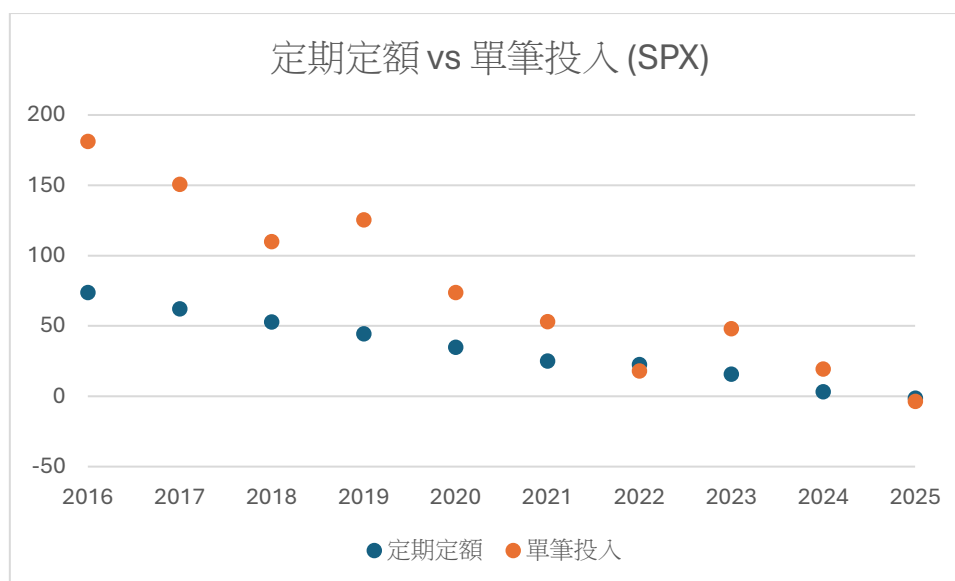
接著是定期定額與單筆投入的比較。如定期定額的部分，AI 成功寫出程式碼，並且可以寫出每日總資產與總投入金額的變化，如下圖(假設每日投入的金額為 1000 元)



圖十一、定期定額程式碼輸出圖

(資料來源 <https://tw.tradingview.com/chart/?symbol=SP%3ASPX>)

並且我統計了從不同年份的 1/1 號開始計算，並於 2025/5/9 截止，定期定額與單筆投入的績效表現，如下圖



圖十二、定期定額與單筆投入的績效統計圖

可以看出在較短的時間內(1-5 年)，定期定額的可以帶來比較穩定的收益，而當時間拉長來看，考慮到現金使用效率的部分，單筆投入的報酬優於定期定額。

除了使用了最新的 AI 模型生成程式碼，我也測試了較差的模型(舊的 GPT-4)，寫程式的錯誤率非常高，相對於 o4-mini, claude 3.7 sonnet，GPT-4 需要大量的修正才能輸出正確的程式碼，並在過程中會出現多個不同的錯誤，如出現錯誤的標籤、指令等，如圖



圖十三與十四、Tradingview 報錯與錯誤輸出

最後成功產生類似功能的程式碼，卻花了超過十倍的時間(相對最新的 AI 模型而言)，顯示出這幾年內 AI 在程式設計的部分大幅進步，雖然還是有些不足之處，但已經有能力生成基礎的程式碼。

#### 4 討論：

在這個研究中，可以看出 AI 在金融分析領域的潛力與不足之處。AI 在生成程式碼的侷限主要包含以下兩點：

1. AI 生成的程式碼頻繁出現語法或邏輯錯誤(較新的推理模型有改善，但還是存在問題)，增加了除錯的負擔。例如，AI 多次生成不存在的函數（如 `math.add` 或 `ta.int`），導致程式無法直接執行。使用者需要明確

在提示中排除錯誤函數，AI 才會嘗試新的解決方案，AI 生成的程式碼品質高度依賴於提示的精確性與使用者的技術反饋能力。

2. 程式碼品質與提示的明確性高度相關。當提示過於籠統時，AI 傾向於依據內建邏輯生成程式碼，而非主動澄清需求，導致結果偏離使用者的預期。對於複雜任務，使用者必須將任務分解為多個子步驟，並逐一提供詳細提示，才能提高程式碼的正確率。這增加了檢查與修正程式碼的時間成本。

雖然 AI 有其侷限性，仍展現顯著的優勢：

1. AI 顯著降低了程式設計的技術門檻。即使使用者對 Pinescript 了解不多，透過精確提示仍可生成基本程式碼，並在發現問題後快速修正。這使得不具程式背景的金業從業人員也能參與技術分析。
2. AI 大幅提升了回測效率。相較於手動編寫程式碼需耗費數十分鐘，AI 可以在數十秒內生成具備基本功能的程式碼。雖然正確性還是需要檢查，但 AI 的高速除錯能力顯著縮短了策略測試週期，允許使用者快速驗證多種投資策略。
3. AI 生成的程式碼有助於量化分析。透過程式回測，可精確計算技術指標的勝率、報酬率等關鍵指標，提供客觀的策略評估依據。如本研究量化了五日移動平均線策略的表現。

AI 對金融交易長期的影響：AI 不但可以提升寫程式的效率，還可以降低門檻，促使競爭力增加。不只如此，AI 可以培養跨領域學習的能力，讓沒有程式背景的人也可以結合程式與自己的金融知識，促使更多人探索金融科技的技術。

## 5 結論

本研究利用 AI 生成 Pinescript 程式碼，並將不同策略套用在 SPX 的歷史數據中，驗證了五日移動平均線策略及定期定額投資與單筆投資的表現差異。透過 AI 的幫助，大幅降低回測的門檻，並提高寫程式的效率，顯著的簡化金融科技的應用與技術學習。

然而 AI 還是存在許多問題，AI 常常出現語法錯誤或其他錯誤(在新的模型中有部分改善)，並且非常依賴 prompt 的精確度。當問題更加複雜時，AI 的錯誤率也會大幅上升，無法有效的解決高難度的問題。對使用者而言，AI 加速開發速度的同時也帶來更高的技術要求，而對一般學生而言，AI 促進跨領域學習的同時需要大量的指導。在精確的 prompt 與有效的指導下，AI 可以很有效的幫助學習。

## 6 程式碼 (有些 label 太長就沒有完整的貼上來)

高於五日均線(低於五日均線則修改判斷式即可)

```
1  //@version=5
2  indicator("前一日高於五日均線後的漲跌統計", overlay=false)
3
4  // 參數設定
5  ma_length = input.int(5, title="移動平均線長度", minval=1)
6  start_date = input.time(timestamp("2024-01-01"), title="開始日期", confirm=true)
7  end_date = input.time(timestamp("2025-05-09"), title="結束日期", confirm=true)
8
9  // 計算五日移動平均線
10 ma5 = ta.sma(close, ma_length)
11
12 // 判斷前一天收盤價是否高於均線
13 prev_above_ma = close[1] > ma5[1]
14
15 // 計算今天相對昨天的價格變化百分比
16 price_change_pct = (close - close[1]) / close[1] * 100
17
18 // 初始化計數器
19 var int cum_up_days = 0 // 累積上漲天數
20 var int cum_down_days = 0 // 累積下跌天數
21 var float cum_up_pct = 0.0 // 累積上漲百分比
22 var float cum_down_pct = 0.0 // 累積下跌百分比
23 var float avg_up_pct = 0.0 // 平均上漲百分比
24 var float avg_down_pct = 0.0 // 平均下跌百分比
25 var int days_counted = 0 // 已統計的交易日數量
26 var int qualified_days = 0 // 符合前一日高於均線條件的交易日數量
27 var int last_start_date = 0 // 上次的開始日期
28 var int last_end_date = 0 // 上次的結束日期
29
30 // 檢測日期範圍是否已經更改
31 if start_date != last_start_date or end_date != last_end_date
32     last_start_date := start_date
33     last_end_date := end_date
34     days_counted := 0
35     qualified_days := 0
36     cum_up_days := 0
37     cum_down_days := 0
38     cum_up_pct := 0.0
39     cum_down_pct := 0.0
40     avg_up_pct := 0.0
41     avg_down_pct := 0.0
42
43 // 確定當前bar是否在計算範圍內
44 in_calculation_period = time >= start_date and time <= end_date
45
46 // 如果在計算期間，增加已計算的交易日數量
47 if in calculation period
```

```

48     days_counted := days_counted + 1
49
50     // 只有在前一天收盤價高於均線時才進行漲跌統計
51     if prev_above_ma
52         qualified_days := qualified_days + 1
53
54         if price_change_pct > 0
55             cum_up_days := cum_up_days + 1
56             cum_up_pct := cum_up_pct + price_change_pct
57             avg_up_pct := cum_up_pct / cum_up_days
58         else if price_change_pct < 0
59             cum_down_days := cum_down_days + 1
60             cum_down_pct := cum_down_pct + price_change_pct
61             avg_down_pct := cum_down_pct / cum_down_days
62         // 如果價格變化為0，則不計入統計
63
64     // 繪製累積天數
65     plot(cum_up_days, title="當前一日高於均線後上漲天數", color=■color.green, linewidth=2)
66     plot(cum_down_days, title="當前一日高於均線後下跌天數", color=■color.red, linewidth=2)
67
68     // 繪製平均漲跌幅
69     plot(avg_up_pct, title="平均上漲幅度 %", color=■color.lime, linewidth=2)
70     plot(avg_down_pct, title="平均下跌幅度 %", color=■color.maroon, linewidth=2)
71
72     // 計算比率
73     var float up_ratio = 0.0
74     var float avg_change = 0.0
75     if (cum_up_days + cum_down_days > 0)
76         up_ratio := 100 * cum_up_days / (cum_up_days + cum_down_days)
77         avg_change := (cum_up_pct + cum_down_pct) / (cum_up_days + cum_down_days)
78
79     // 添加標籤顯示統計結果（單行格式避免行連續性錯誤）
80     if barstate.islast
81         label.new(bar_index, 0, text="統計範圍：從 " + str.format("{0,date,yyyy-MM-dd}", start
--

```

## 定期定額

```

1 // @version=5
2 indicator("Dollar Cost Average", "DCA", overlay=true)
3 float startingAmount = input.float(0.0, "Starting Investment")
4 float investmentAmount = input.float(1000.0, "Repeating Investment")
5 int frequency = input.int(1, "Repeats every (in days)", inline="frequency")
6 float feeRatio = input.float(0, "Broker Fee")
7 int startDate = input.time(timestamp("Jan 01 2023"), "Starting Date")
8 int endDate = input.time(timestamp("May 11 2025"), "Ending Date")
9 float cellWidth = input.float(0.0, "Cell Width", group="advanced")
10 float cellHeight = input.float(10.0, "Cell Height", group="advanced")
11
12 var float assets = 0.0
13 var float feeAmount = 0.0
14 var float entryPrice = 0.0
15 var float invested = 0.0
16 var float netProfit = 0.0
17 var float totalFee = 0.0
18 var float grossProfit = 0.0
19 var float grossLoss = 0.0
20 var int totalTrades = 0
21 var float portfolioValue = 0.0
22
23
24 bool investmentWindow = time >= startDate and time < endDate
25
26 if investmentWindow
27     if startingAmount > 0.0 and invested == 0.0
28         totalTrades := totalTrades + 1
29         feeAmount := startingAmount * feeRatio
30         invested := startingAmount - feeAmount
31         assets := invested / close
32         entryPrice := close
33         totalFee := feeAmount
34
35     if bar_index % frequency == 0
36         totalTrades := totalTrades + 1
37         feeAmount := investmentAmount * feeRatio
38         invested := invested + investmentAmount - feeAmount
39         assets := assets + (investmentAmount - feeAmount) / close
40         entryPrice := (entryPrice * (assets - (investmentAmount - feeAmount) / close) + close * (investmentAmount - feeAmount) / close) / assets
41         totalFee := totalFee + feeAmount
42
43     netProfit := close * assets - invested
44     portfolioValue := close * assets
45
46
47 plot(investmentWindow ? portfolioValue : na, "Portfolio Value", ■color.yellow)
48 plot(investmentWindow ? invested : na, "Invested Amount", ■color.lime, 2)

```

## 單筆投入

```
1  // @version=5
2  indicator("日期間價格變化百分比", overlay=true)
3
4  // 日期參數設定
5  start_date = input.time(timestamp("2016-01-01"), title="開始日期", confirm=true)
6  end_date = input.time(timestamp("2025-05-10"), title="結束日期", confirm=true)
7
8  // 確保日期設定有效
9  if start_date > end_date
10     runtime.error("開始日期不能晚於結束日期")
11
12  // 初始化變數儲存價格
13  var float start_price = na
14  var float end_price = na
15  var bool start_found = false
16  var bool end_found = false
17
18  // 找到開始日期的收盤價 (第一個大於或等於開始日期的交易日)
19  if not start_found and time >= start_date
20     start_price := close
21     start_found := true
22
23  // 記錄結束日期的收盤價 (最後一個小於或等於結束日期的交易日)
24  if time <= end_date
25     end_price := close
26     end_found := true
27
28  // 計算價格變化百分比並在圖表上顯示結果
29  if barstate.islast
30     if start_found and end_found
31         // 修改後的計算公式：(結束日的股價-開始日的股價)/開始日的股價*100
32         float price_change_pct = (end_price - start_price) / start_price * 100
33
34         // 使用單行格式顯示價格變化資訊 (避免行連續性問題)
35         label.new(bar_index, close, text="價格變化百分比分析:\n從 " + str.format("{0,d",
36     else
37         // 若找不到指定日期的資料則顯示錯誤訊息 (同樣使用單行格式)
38         label.new(bar_index, close, text="未找到指定日期範圍內的價格數據", color=■ colc
```