

# 程式交易概論

製作人: 黃宥輔



# 程式交易(Program Trading)是什麼





- 近年來,隨著網路與電腦迅速發展,傳統的金融商品交易也漸 漸轉由電腦處理,衍生出程式交易
- 程式交易為利用程式、軟體進行自動化或半自動化的金融商品 買賣操作

#### 程式交易特點



	<del>缺</del> 點				
效率高,能在短時間内 比對上百支金融商品資料	電腦設備需要穩定的環境				
嚴守紀律,消除因為人性的不安定因素	若邏輯不完善可能會導致預期外的狀況				
自動化交易省時省力	以歷史資料回測建構的策略 不一定代表未來的狀況				
能在快速變動的市場中搶得先機	某些傳統策略較難轉換為程式邏輯, 如形態學				
可以利用歷史資料進行回溯測試(回測),建構交易策略	學習門檻較高				

國立高雄科技大學AI金融科技中心版權所有

#### 程式交易主要做法



- 1. 套裝軟體: Multichart、XQ...
- 2. 程式語言串接API

#### 套裝軟體與串接API



比較項目	套裝軟體	Python接API
資料來源	内建	自行抓取
技術指標	豊富	TA-LIB
難易度	較簡易	較困難
靈活度	較低	較高
費用與限制	有	免費

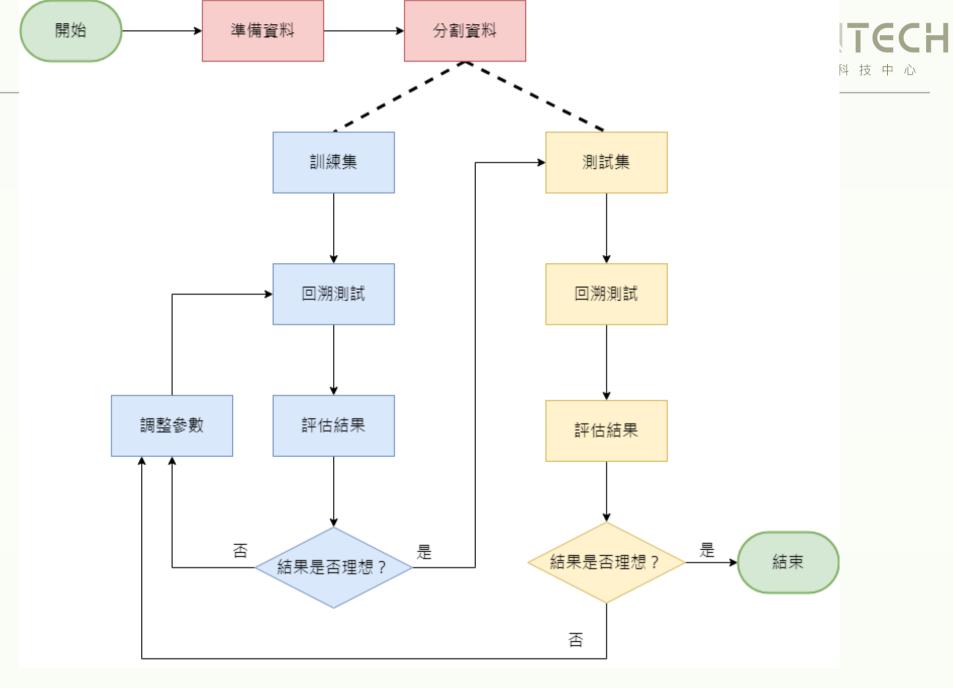
#### 回測要件





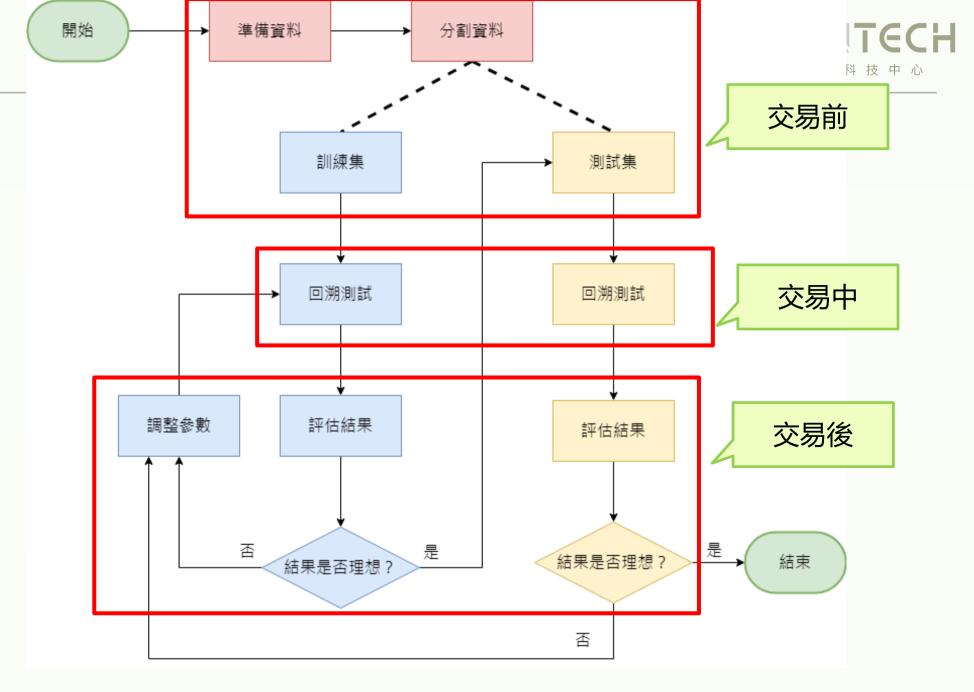
時機	基本項目	進階項目				
交易前	1. 選擇標的 2. 選擇交易頻率 3. 決定實驗期間					
交易中	1. 進出場條件、趨勢判斷 2. 進出場時機 3. 手續費與交易稅 4. 資金限制	<ol> <li>1. 滑價</li> <li>2. 還原股價</li> <li>3. 加減碼規則(部位管理)</li> </ol>				
交易後	1. 產出績效指標	1. 自動化優化參數				

#### 程式交易流程



國立高雄科技大學AI金融科技中心版權所有

#### 程式交易流程



國立高雄科技大學AI金融科技中心版權所有

#### 交易前 - 選擇標的





- 僅挑選金融商品種類,不挑選特定公司
- 挑選特定產業
- 依據條件挑選金融商品,如本益比、營收成長率等
- 挑選基金成分股,如 0050

#### 交易前 - 選擇交易頻率





- 高頻交易(high-frequency trading, HFT)
  - ▶ 指在短時間對一個金融商品進行多次的建倉與平倉
  - ▶ 通常採日內的資料
  - > 需要非常高等級的設備
- 短線價差交易
  - ▶ 持有股票賺取價差為目的進行買賣
- 長線投資
  - > 投資穩定成長或是穩定發放股息股利

#### 交易前 - 決定實驗期間



- 巴菲特:投資表現最少以3年衡量,5年為佳,並且與大盤比較, 若表現不如大盤,那何必白費力氣呢?
- 台灣的大盤可以以大盤指數或 0050 作為比較基準

#### 交易前 - 範例說明



標的:經 NSGA2 篩選過後,相關係數最小的 5 檔 0050 成分股

交易頻率:日

實驗期間:

● 訓練集 - 2015/01/01 到 2016/12/31

● 測試集 - 2017/01/01 到 2017/12/31

#### 交易前 - 準備資料(主程式)



```
# 傳入參數,呼叫程式
28
    train data, test data = get_data(stock_id = 1101,
29
            train start date = "2015-01-01",
30
            train end date = "2016-12-31",
31
            test start date = "2017-01-01",
32
            test end date = "2018-01-01",
33
                                            傳入
34
                                               股票代號
35
                                               訓練集開始日期
    # 輸出結果
36
                                               訓練集結束日期
    print(train data)
37
                                               測試集開始日期
    print("-----
38
                                               測試集結束日期
    print(test data)
39
                          輸出結果
```

## 交易前 - 準備資料(get\_data函數)



```
1 # -*- codina: utf-8 -*-
    import pandas as pd # 資料整理工具
    def get_data(**para):
       # 將傳入的參數整理成變數
       stock id = para["stock id"] # 股票代號
 6
       train start date = para["train start date"] # 訓練集開始日期
       train end date = para["train end date"] # 訓練集結束日期
8
       test start date = para["test start date"] # 測試集開始日期
9
       test end date = para["test end date"] # 測試集開結束日期
10
11
       # 讀取資料
12
       path = "股價資料 整合/"
13
       data = pd.read excel( f"{path}{stock id}.xlsx" )
14
15
       # 帶入日期,篩選訓練集的資料
16
17
       train data = data[ (data["日期"] >= train start date) &
                        (data["日期"] <= train end date)
18
19
20
       # 帶入日期,篩選測試集的資料
21
       test data = data[ (data["日期"] >= test start date) &
22
                          (data["日期"] <= test end date)
23
24
25
        return train data, test data
26
```

國立高雄科技大學 AI 金融科技中心版權所有

# 交易前 - 準備資料(get data函數)





```
# -*- codina: utf-8 -*-
import pandas as pd # 資料整理工具
def get_data(**para):
   # 將傳入的參數整理成變數
   stock_id = para["stock_id"] # 股票代號
   train start date = para["train_start_date"] # 訓練集開始日期
   train end date = para["train_end_date"] # 訓練集結束日期
   test start date = para["test start date"] # 測試集開始日期
```

return train data, test data

10 11

12

13

14 15 16

17

18

19

20

21

22

23

24 25

26

將傳入的參數指派給變數

```
# 讀取資料
path = "股價資料 整合/"
data = pd.read_excel( f"{path}{stock_id}.xlsx" )
```

test end date = para["test end date"] # 測試集開結束日期

根據股票代號, 讀取資料

```
train_data = data[ (data["日期"] >= train_start_date) &
                (data["日期"] <= train end date)
# 帶入日期,篩選測試集的資料
test data = data[ (data["日期"] >= test start date) &
                  (data["日期"] <= test_end_date)
```

根據分割日期, 將資料進行篩選

回傳訓練集、測試集

國立高雄科技大學 AI 金融科技中心版權所有

# 交易前 - 準備資料





	Console 1/A 🗵	F-1-1-X 2011 - 1- 1- 1- 1- 1- 1- 1- 1- 1- 1- 1- 1-	干囲貝付・アダ	, waii -	C./ 03E	ı ə/ uəei	/ Deskit	DD/ AIGO	□木/土/	<b>8</b>
	<u></u> 蟲程式交易最									
	日其	期 成交	泛股數	成交金額	開想	<b>盤價</b>	最高價	最低	價	收
盤價	漲跌價差	成交筆數								
0	2015-01-05	2842933	122244519	43.40	43.40	42.80	43.00	-0.40	1489	
1	2015-01-06	7654419	324150031	42.50	42.60	42.10	42.25	-0.75	3660	
2	2015-01-07	8719024	368171551	42.25	42.55	41.85	42.10	-0.15	3348	
3	2015-01-08	8697776	369425923	42.25	42.65	42.10	42.25	0.15	3700	
4	2015-01-09	10494129	441333468	42.30	42.60	41.80	41.80	-0.45	3975	
	• • •	• • •	• • •	• • •	• • •	• • •	• • •	• • •	• • •	
483	2016-12-26	4685948	160372703	34.40	34.85	34.00	34.25	-0.10	2303	
484	2016-12-27	2949091	100534741	34.30	34.30	34.05	34.10	-0.15	1188	
485	2016-12-28	6390960	218855112	34.20	34.50	34.15	34.50	0.40	2590	
486	2016-12-29	5504753	188700567	34.50	34.50	34.20	34.20	-0.30	2756	
487	2016-12-30	6121039	213121341	34.50	35.15	34.30	35.15	0.95	2939	
	_	_								
[488	3 rows x 9 c	columns]								

# 交易前 - 準備資料

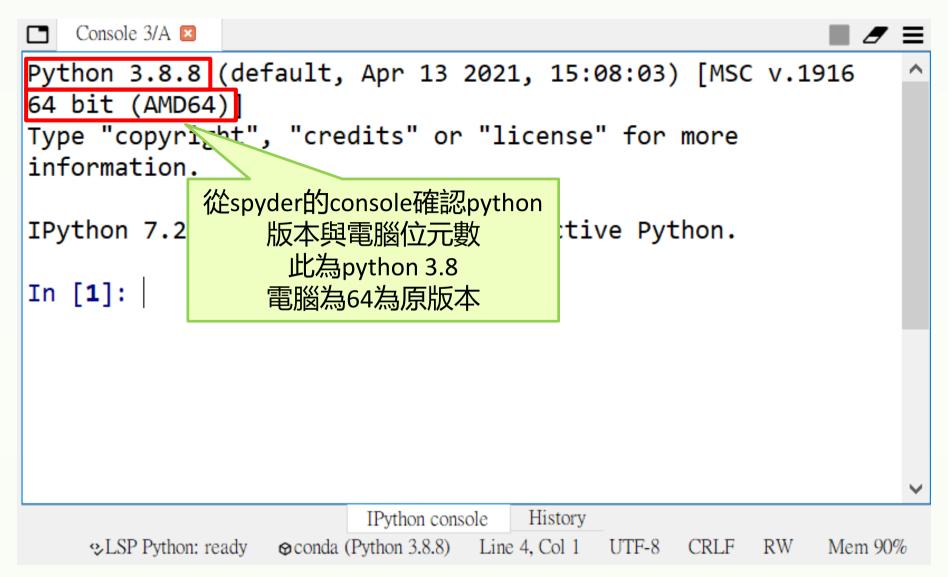




	 	武法別	<b>₽</b> /₁	<b></b>	<b>見見</b> 44	↓ <b>/</b> Ⅲ	見戸傅	具.任.		11/2
571. /ast \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\	日期	成交股	赵	成交金額	開盤	1月	最高價	最低	貝	收
盤價 漲跌價差	差 成交氫	丰數								
488 2017-01-	03 289	90982 10	1450502	35.45	35.45	34.80	35.15	0.00	1669	
489 2017-01-	04 329	96820 11	.5895011	35.20	35.30	35.00	35.25	0.10	1737	
490 2017-01-	05 409	98017 14	3758836	35.00	35.25	34.90	35.25	0.00	1788	
491 2017-01-	06 354	12625 12	4391293	35.30	35.30	34.90	35.25	0.00	1774	
492 2017-01-	09 487	77900 17	0970138	35.25	35.35	34.90	34.90	-0.35	1515	
	• •	• • •		• • •	• • •	• • •	• • •	• • •		
729 2017-12-	25 1515	3859 54	3236324	35.30	36.25	35.10	36.15	1.00	5787	
730 2017-12-	26 1886	02178 68	3520276	36.10	36.50	35.95	36.40	0.25	7927	
731 2017-12-	27 886	98450 31	.9904500	36.10	36.45	36.00	36.20	-0.20	3132	
732 2017-12-	28 761	L2103 27	5264758	36.10	36.25	36.00	36.20	0.00	2577	
733 2017-12-	29 1303	39478 47	4385049	36.20	36.80	36.10	36.45	0.25	3330	













TA-Lib: a wrapper for the TA-LIB Technical Analysis Library.

TA Lib-0.4.24-pp38-pypy38 pp73-win amd64.whl

TA Lib-0.4.24-cp310-cp310-win amd64.whl

TA Lib-0.4.24-cp310-cp310-win32.whl

TA Lib-0.4.24-cp39-cp39-win amd64.whl

TA Lib-0.4.24-cp39-cp39-win32.whl

TA Lib-0.4.24-cp38-cp38-win amd64.whl

TA Lib-0.4.24-cp38-cp38-win32.whl

TA\_Lib-0.4.24-cp37-cp37m-win\_amd64.whl

TA Lib-0.4.24-cp37-cp37m-win32.whl

TA Lib-0.4.19-cp36-cp36m-win\_amd64.whl

TA Lib-0.4.19-cp36-cp36m-win32.whl

TA Lib-0.4.17-cp35-cp35m-win amd64.whl

TA Lib-0.4.17-cp35-cp35m-win32.whl

TA Lib-0.4.17-cp34-cp34m-win amd64.whl

TA Lib-0.4.17-cp34-cp34m-win32.whl

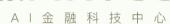
TA Lib-0.4.17-cp27-cp27m-win amd64.whl

TA\_Lib-0.4.17-cp27-cp27m-win32.whl

進入此網頁













https://www.lfd.uci.edu/~gohlke/pvthonlibs/#ta-lib

**TA-Lib**: a wrapper for the TA-LIB Technical Analysis Library.

TA Lib-0.4.24-pp38-pypy38 pp73-win amd64.whl

TA Lib-0.4.24-cp310-cp310-win amd64.whl

TA Lib-0.4.24-cp310-cp310-win32.whl

TA Lib-0.4.24-cp39-cp39-win amd64.whl

TA Lib-0.4.24-cp39-cp39-win32.whl

TA Lib-0.4.24-cp38-cp38-win amd64.whl

TA Lib-0.4.24-cp38-cp38-win32.whl

TA Lib-0.4.24-cp37-cp37m-win amd64.whl

TA Lib-0.4.24-cp37-cp37m-win32.whl

TA Lib-0.4.19-cp36-cp36m-win amd64.whl

TA Lib-0.4.19-cp36-cp36m-win32.whl

TA Lib-0.4.17-cp35-cp35m-win\_amd64.whl

TA Lib-0.4.17-cp35-cp35m-win32.whl

TA Lib-0.4.17-cp34-cp34m-win amd64.whl

TA Lib-0.4.17-cp34-cp34m-win32.whl

TA Lib-0.4.17-cp27-cp27m-win amd64.whl

TA Lib-0.4.17-cp27-cp27m-win32.whl

根據安裝的Python版本與 電腦位元數,挑選whl檔 範例為Python 3.8版 64位元的電腦 故選擇此檔案





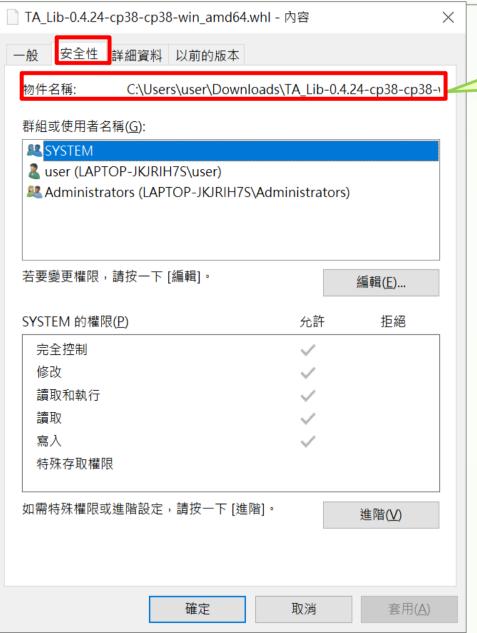
AI金融科技中心







到安全性的頁籤中可以找到完整 路徑,將其複製



國立高雄科技大學 AI 金融科技中心版權所有





在Anaconda Prompt中輸入指令安裝TA-Lib

pip install C:\Users\user\Downloads\TA\_Lib-0.4.24-cp38-cp38-win\_amd64.whl

Anaconda Prompt (anaconda3)

(base) C:\Users\user>pip install C:\Users\user\Downloads\TA\_Lib-0.4.24-cp38-cp38-win\_amd64.whl

Processing c:\users\user\downloads\ta\_lib-0.4.24-cp38-cp38-win\_amd64.whl

Requirement already satisfied: numpy in c:\users\user\anaconda3\lib\site-packages (from TA-Lib==0.4.24) (1.20.1)

Installing collected packages: TA-Lib

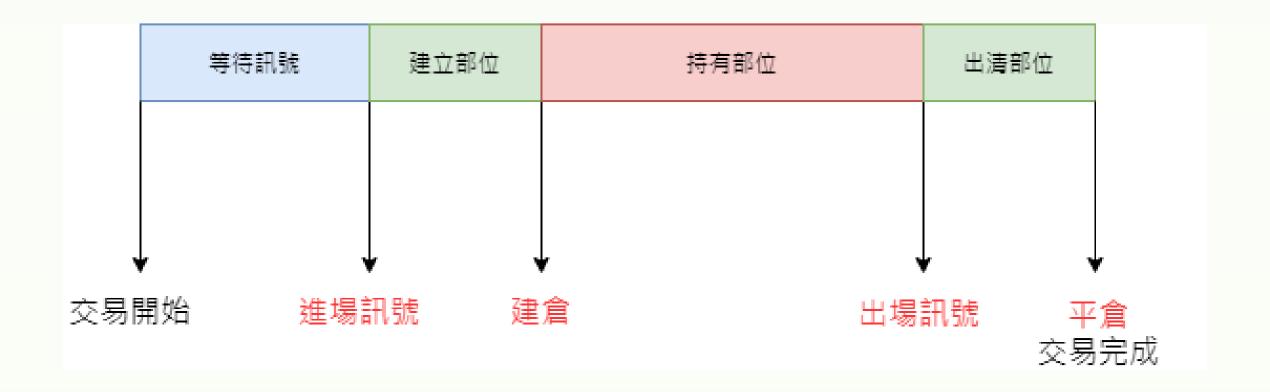
Successfully installed TA-Lib-0.4.24

(base) C:\Users\user>

#### 交易中 - 進出場時機

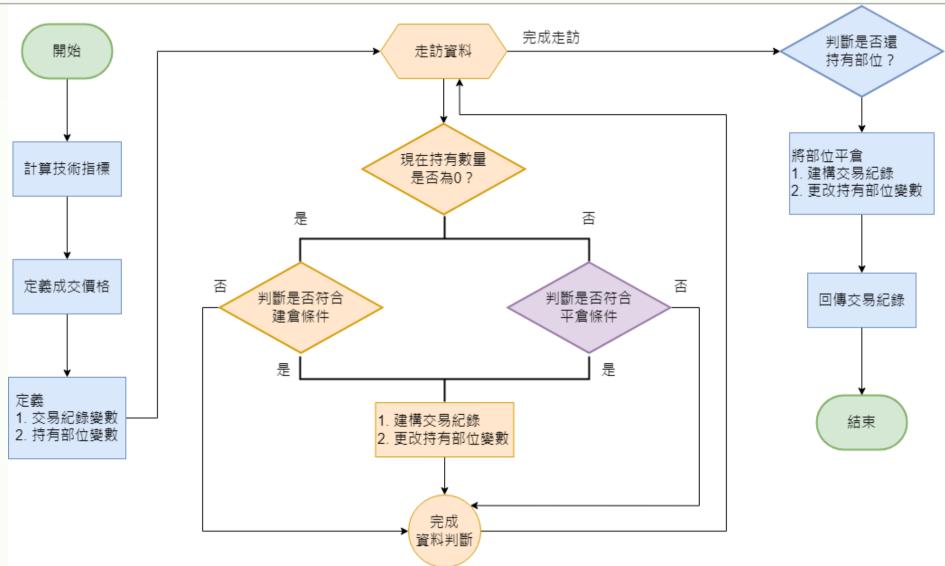






#### 交易中 - 建構回測策略





#### 交易中 - 整體程式擺放順序





#### 引用套件(import)

獲取資料函數(get\_data)

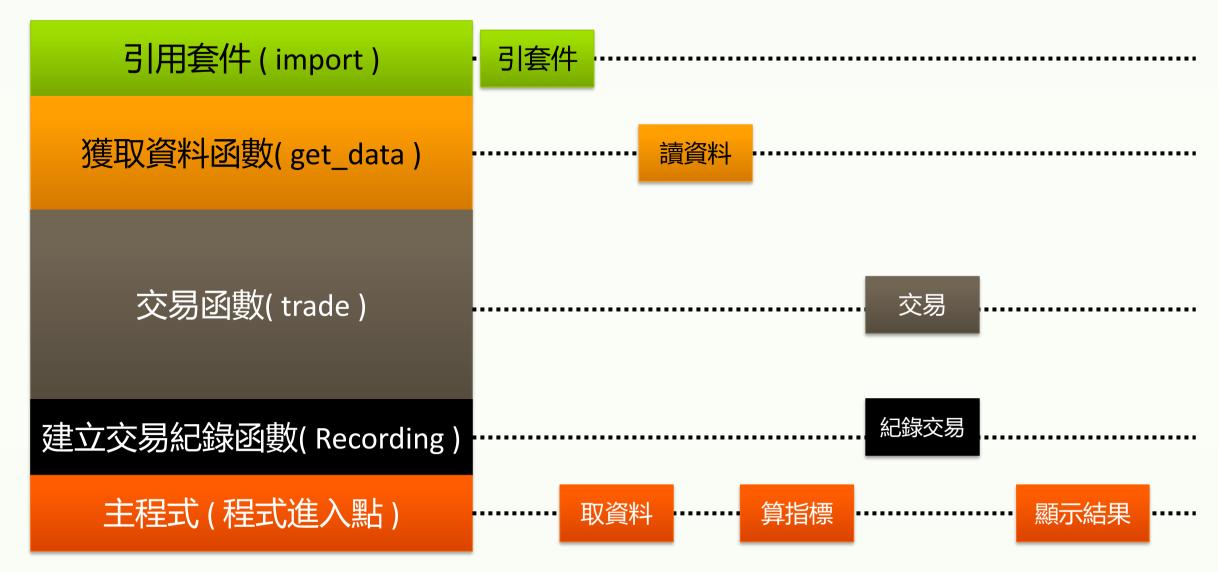
交易函數(trade)

建立交易紀錄函數(Recording)

主程式(程式進入點)

#### 交易中 - 整體程式擺放順序





#### 交易中 - RSI 指標



• 相對強弱指標(Relative Strength Index, RSI), 一種用來評估買賣 雙方強弱力度的技術指標,公式如下

• RSI = 
$$\frac{\text{UP}}{\text{UP+DOWN}} \times 100$$

- 其中UP為n日內平均上漲值; DOWN為n日內平均下跌值
- 範圍在 0 到 100 之間
- 常用週期為6日、9日、12日等週期,發明人Wilder 偏好用14日

#### 交易中 - 主程式



```
# 以隔天的開盤價當作成交價格
121
     train_data["成交價格"] = train_data["開盤價"].shift(-1)
122
123
     # 最後一天以收盤價當作交易價格
     train data["成交價格"].fillna(method = "ffill", inplace = True)
124
     # 計算 RSI5
125
     train data["RSI 5"] = talib.RSI(train data["收盤價"], 5)
126
127
128
     # 以隔天的開盤價當作成交價格
     test_data["成交價格"] = test_data["開盤價"].shift(-1)
129
     # 最後一天以收盤價當作交易價格
130
     test_data["成交價格"].fillna(method = "ffill", inplace = True)
131
132
     # 計算 RSI5
133
     test data["RSI 5"] = talib.RSI(test data["收盤價"], 5)
134
     trade(train_data)
135
     print(record df)
136
137
     trade(test data)
138
     print(record df)
139
```

#### 交易中 - 主程式





```
# 以隔天的開盤價當作成交價格
121
    train_data["成交價格"] = train_data["開盤價"].shift(-1)
122
123
    # 最後一天以收盤價當作交易價格
    |train_data["成交價格"].fillna(method = "ffill", inplace = True)
124
    # 計算 RSI5
125
    train data["RSI 5"] = talib.RSI(train_data["收盤價"], 5)
126
127
128
    # 以隔天的開盤價當作成交價格
    |test_data["成交價格"] = test_data["開盤價"].shift(-1)
129
    # 最後一天以收盤價當作交易價格
130
    test_data["成交價格"].fillna(method = "ffill", inplace = True)
131
    # 計算 RSI5
132
    test_data["RSI_5"] = talib.RSI(test_data["收盤價"], 5)
133
134
    trade(train_data)
135
```

接續取得資料 定義成交價格與 計算RSI指標

將資料帶入 trade 函數取得交易紀錄 (下一個部分會開始建構trade函數)

136

137

138

print(record df)

trade(test data)



```
29
    def trade(data_df):
       global record_df, hold_data # 設定全域變數,讓其他函數也可以使用者兩個變數
30
31
       # 定義交易紀錄
32
       record_df = pd.DataFrame( columns = ["time", "price", "LS", "num"] )
33
34
       # 定義持有資訊
35
       hold data = {
36
37
           "price" : 0,
           "num": 0 # 做多部位為正數,做空部位為負
38
39
```





#### 將交易紀錄、持有記錄設為全域變數

```
29
    def trade(data df):
        global record df, hold data #
30
31
32
        # 定義交易紀錄
        record df = pd.DataFrame( columns = ["time", "price", "LS", "num"] )
33
34
        # 定義持有資訊
35
        hold data = {
36
            "price" : 0,
37
            "num": ∅ # 做多部位為正數,做空部位為負
38
39
```

#### 交易紀錄、紀錄

- 交易時間(time)
- 成交價格(price)
- 多空方向(LS)
- 成交數量(num)

#### 持有資訊, 紀錄未平倉的

- 建倉時的成交價格(price)
- 建倉時的成交數量(num) 其中成交數量做多以正數, 做空以負數代表

國立高雄科技大學 AI 金融科技中心版權所有





```
def trade(data df):
       for index, row in data df.iterrows():
41
42
           # 沒有持有部位,建倉判斷
           if hold data["num"] == 0:
43
              condition1 = row["RSI 5"] < 20 # 若RSI小於20,則做多(Long)
44
45
              condition2 = row["RSI 5"] > 80 # 若RSI大於80 , 則做空(Short)
46
47
              # 符合做多條件
              if condition1:
48
                  trade price = row["成交價格"] # 取出當下的成交價格
49
50
                  num = 1 # 交易數量
                  Recording(row["日期"], trade price, "L", num) # 紀錄做多交易
51
52
53
                  # 更動持有狀況,紀錄成交價格與交易數量
                  hold data["price"], hold data["num"] = trade price, num
54
55
              # 符合做空條件
56
              elif condition2:
57
                  trade price = row["成交價格"] # 取出當下的成交價格
58
                  num = 1 # 成交數量
59
                  Recording(row["日期"], trade price, "S", num) # 紀錄做空交易
60
61
62
                  # 更動持有狀況,紀錄成交價格與交易數量,做空數量以負數代表
                  hold data["price"], hold data["num"] = trade price, -num
63
```

國立高雄科技大學 AI 金融科技中心版權所有

#### 逐列走訪資料





```
def trade(data df):
      for index, row in data df.iterrows():
41
                                           當下持有股票數量為0,則進入建倉判斷
42
           # 沒有持有部位,建倉判斷
          if hold data["num"] == 0:
43
              condition1 = row["RSI 5"] < 20 # 若RSI小於20,則做多(Long)
44
45
              condition2 = row["RSI 5"] > 80 # 若RSI大於80,則做空(Short)
46
                符合做多條件
47
              if condition1:
48
49
                 trade_price = row["成交價格"] # 取出當下的成交價格
50
                 num = 1 # 交易數量
                 Recording(row["日期"], trade price, "L", num) # 紀錄做多交易
51
52
                 # 更動持有狀況,紀錄成交價格與交易數量
53
                 hold data["price"], hold data["num"] = trade price, num
54
55
                符合做空條件
56
              elif condition2:
57
                 trade_price = row["成交價格"] # 取出當下的成交價格
58
                 num = 1 # 成交數量
59
                 Recording(row["日期"], trade price, "S", num) # 紀錄做空交易
60
61
62
                 # 更動持有狀況,紀錄成交價格與交易數量,做空數量以負數代表
                 hold data["price"], hold data["num"] = trade price, -num -
63
```

建倉條件分別為

- 做多條件 RSI < 20
- 做空條件 RSI > 80

若符合建倉條件 則利用Recording函數 紀錄交易、 更新持有資料 (hold data)

做空數量以負數表示

## 交易中 - 紀錄函數(Recording)



```
def Recording(time, price, LS, num):
101
         if record df.empty: # 若當下的交易紀錄是空的,則索引值設為 @
102
            index count = 0
103
         else: # 若交易紀錄不是空的,則索引值設為最大值 + 1
104
            index count = record df.index[-1] + 1
105
106
         # 將資料寫入交易紀錄
107
         record df.at[index count, "time"] = time
108
         record df.at[index count, "price"] = price
109
         record df.at[index count, "LS"] = LS
110
         record_df.at[index count, "num"] = num
111
```

#### 交易中 - 紀錄函數(Recording)





AI.FINTEC

AI金融科技中心

#### 傳入

- 1. 交易時間(time)
- 2. 成交價格(price)
- 3. 多空方向(LS)
- 4. 成交數量(num)

設定索引值

```
def Recording(time, price, LS, num):
101
         if record df.empty: # 若當下的交易紀錄是
102
             index count = 0
103
         else: # 若交易紀錄不是空的,則索引值設為最大值 + 1
104
            index count = record df.index[-1] + 1
105
106
         # 將資料寫入交易紀錄
107
                                                       將資料記入
         record df.at[index count, "time"] = time
108
                                                        record df
         record df.at[index count, "price"] = price
109
         record df.at[index count, "LS"] = LS
110
        record_df.at[index_count, "num"] = num
111
```





```
def trade(data df):
41
        for index, row in data df.iterrows():
            else:
66
               # 做多平倉條件
67
               condition1 = hold data["num"] > 0 # 持有數量大於0
68
               condition2 = row["RSI 5"] > 80 # RSI大於80
69
70
               # 做空平倉條件
71
               condition3 = hold data["num"] < 0 # 持有數量小於@
72
               condition4 = row["RSI 5"] < 20 # RSI//於20
73
                           for index, row in data df.iterrows():
74
               # 做多平倉條件
75
               if condition1 and condition2:
76
                   num = hold data["num"] # 當下持有數量
77
                   trade price = row["成交價格"] # 取出當下的成交價格
78
                   Recording(row["日期"], trade_price, "S", num) # 紀錄平倉交易
79
                   hold data["price"], hold data["num"] = "", 0 # 清空持有紀錄
80
81
               # 做空平倉條件
82
               elif condition3 and condition4:
83
84
                   num = -hold_data["num"] # 當下持有數量,將負數轉正
85
                   trade_price = row["成交價格"] # 取出當下的成交價格
                   Recording(row["日期"], trade price, "L", num) # 紀錄平倉交易
86
                   hold data["price"], hold data["num"] = "", 0 # 清空持有紀錄
87
```

87





#### AI.FINTECH

AI金融科技中心

```
def trade(data df):
41
        for index, row in data df.iterrows():
66
            else:
67
               condition1 = hold data["num"] > 0 # 持有數量大於0
68
               condition2 = row["RSI 5"] > 80 # RSI 大於80
69
70
               # 做空平倉條件
71
               condition3 = hold data["num"] < 0 # 持有數量小於0
72
               condition4 = row["RSI 5"] < 20 # RSI小於20
73
74
75
               # 做多平倉條件
               if condition1 and condition2:
76
77
                   num = hold data["num"] # 當下持有數量
                   trade_price = row["成交價格"] # 取出當下的成交價格
78
                   Recording(row["日期"], trade price, "S", num) # 紀錄平倉交易
79
80
                   hold data["price"], hold data["num"] = "", 0 # 清空持有紀錄
81
82
               # 做空平倉條件
83
               elif condition3 and condition4:
                   num = -hold_data["num"] # 當下持有數量,將負數轉正
84
85
                   trade_price = row["成交價格"] # 取出當下的成交價格
                   Recording(row["日期"], trade_price, "L", num) # 紀錄平倉交易
86
```

hold data["price"], hold data["num"] = "", 0 # 清空持有紀錄

平倉條件分別為做多平倉:

- 1. 持有數量 > 0
- 2. RSI > 80

做空平倉

- 1. 持有數量 < 0
- 2. RSI < 80

若符合條件,紀錄交易、清空持有紀錄



```
def trade(data df):
89
       # 最後一日平倉
       last_date = data_df.iloc[-1]["日期"] # 最後一日日期
90
       last_trade_price = data_df.iloc[-1]["成交價格"] # 最後一日成交價格
91
92
93
       if hold_data["num"] > 0: # 若持有數量不為大於0 , 將做多部位平倉
94
           Recording(last_date, last_trade_price, "S", hold_data["num"])
95
       elif hold_data["num"] < 0: # 若持有數量不為小於0,將做空部位平倉
96
           Recording(last_date, last_trade_price, "L", -hold_data["num"])
97
98
       return record_df # 回傳交易紀錄
99
```



def trade(data df):

最後一日的日期與成交價格

```
89
       # 最後一月平倉
       last_date = data_df.iloc[-1]["日期"] # 最後一日日期
90
       last_trade_price = data_df.iloc[-1]["成交價格"] # 最後一日成交價格
91
92
93
       if hold_data["num"] > 0: # 若持有數量不為大於0 , 將做多部位平倉
94
           Recording(last_date, last_trade_price, "S", hold_data["num"])
95
       elif hold_data["num"] < 0: # 若持有數量不為小於0,將做空部位平倉
96
           Recording(last_date, last_trade_price, "L", -hold_data["num"])
97
98
       return record_df # 回傳交易紀錄
99
```

若持有數量大於0或小於0,則進行平倉

回傳交易紀錄

## 執行結果





	Console 1/A 🗵				
		time	price	LS	num
0	2015-01-12	00:00:00	40.7	L	1
1	2015-01-28	00:00:00	42.7	S	1
2	2015-03-31	00:00:00	44.0	S	1
3	2015-05-07	00:00:00	42.5	L	1
4	2015-06-01	00:00:00	41.1	L	1
5	2015-10-07	00:00:00	36.55	S	1
6	2015-10-12	00:00:00	37.35	S	1
7	2015-11-11	00:00:00	34.5	L	1
8	2015-11-12	00:00:00	34.25	L	1
9	2016-02-16	00:00:00	28.3	S	1
10	2016-02-18	00:00:00	29.0	S	1
11	2016-05-04	00:00:00	30.4	L	1
12	2016-05-05	00:00:00	30.5	L	1
13	2016-06-03	00:00:00	30.7	S	1
14	2016-07-05	00:00:00	33.95	S	1
15	2016-09-14	00:00:00	34.0	L	1
16	2016-12-19	00:00:00	36.2	L	1
17	2016-12-30	00:00:00	34.5	S	1

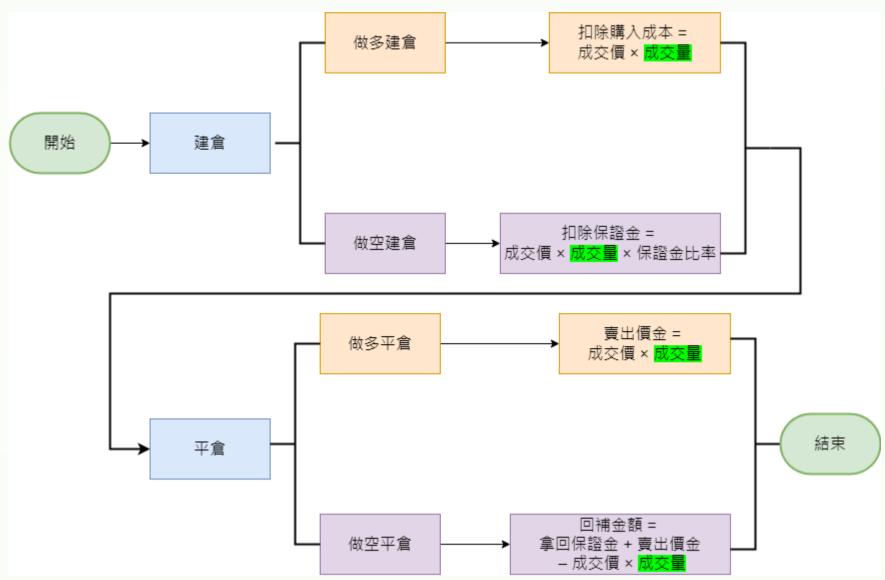
#### 訓練集交易結果

time price LS num 0 2017-01-12 00:00:00 36.3 S 1 1 2017-12-29 00:00:00 36.2 L 1

測試集交易結果

#### 交易中 - 加入資金管控





#### 交易中 - 加入資金管控



- 原始資金 1,000,000
- 做多建倉,成交價 65 元,<mark>成交量</mark> = 1,000,000 // 65 = 15,384 股
- 現金 = 1,000,000 65 × 15,384 = 40

- 做多平倉,成交價 70 元,成交量(持有量) 15,384 股
- 現金 =  $40 + 70 \times 15,384 = 1,076,920$

#### 交易中 - 加入資金管控



- 現金 1,076,920
- 做空建倉,成交價 75 元,<mark>成交量</mark> = 1,076,920 // 75 = 14,358 股
- 保證金 = 75 × 14,358 × 100% = 1,076,850
- 現金 = 1,076,920 保證金 = 70

- 做空平倉,成交價 72 元,<mark>成交量(持有量) 14,358</mark> 股
- 現金 = 70 + 保證金 + (75 72) × 14,358 = 1,119,994

#### 交易中 - 主程式(加入資金管控)





```
# 以隔天的開發價當作成交價格
141
      train data["成交價格"] = train data["開盤價"].shift(-1)
142
      # 最後一天以收繳價當作交易價格
143
      train_data["成交價格"].fillna(method = "ffill", inplace = True)
144
      # 計算 RSI5
145
      train data["RSI 5"] = talib.RSI(train data["收盤價"], 5)
146
147
      # 以隔天的開盤價當作成交價格
148
      |test_data["成交價格"] = test_data["開盤價"].shift(-1)
149
      # 最後一天以收盤價當作交易價格
150
      test data["成交價格"].fillna(method = "ffill", inplace = True)
151
      # 計算 RSI5
152
      test data["RSI_5"] = talib.RSI(test_data["收盤價"], 5)
153
154
      # 定義初始資金
155
      cash = 1000000
156
157
      trade(train data, cash)
158
      print(record df)
159
160
      trade(test data, cash)
161
      print(record_df)
162
```

#### 交易中 - 主程式(加入資金管控)





```
# 以隔天的開發價當作成交價格
141
142
     train_data["成交價格"] = train_data["開盤價"].shift(-1)
     # 最後一天以收魁價當作交易價格
143
     train data["成交價格"].fillna(method = "ffill", inplace = True)
144
     # 計算 RSI5
145
     train data["RSI 5"] = talib.RSI(train data["收盤價"], 5)
146
147
     # 以隔天的開盤價當作成交價格
148
149
     |test_data["成交價格"] = test_data["開盤價"].shift(-1)
     # 最後一天以收盤價當作交易價格
150
     test data["成交價格"].fillna(method = "ffill", inplace = True)
151
     # 計算 RSI5
152
     test data["RSI 5"] = talib.RSI(test data["收盤價"], 5)
153
154
     # 定義初始資金
155
                         持有資金
156
     cash = 1000000
157
     trade(train data, cash)
158
     print(record df)
159
                                 帶入資金到交易函數中
160
     trade(test data, cash)
161
     print(record_df)
162
```





```
def trade(data df, cash):
29
       global record_df, hold_data # 設定全域變數,讓其他函數也可以使用者兩個變數
30
31
32
       # 定義交易紀錄
       record_df = pd.DataFrame( columns = ["time", "price", "LS", "num"] )
33
34
       # 定義持有資訊
35
       hold data = {
36
           "price" : 0,
37
           "num": 0 # 做多部位為正數,做空部位為負
38
39
```





#### 傳入原始金額

```
def trade(data df, cash):
29
       global record_df, hold_data # 設定全域變數,讓其他函數也可以使用者兩個變數
30
31
32
       # 定義交易紀錄
       record_df = pd.DataFrame( columns = ["time", "price", "LS", "num"] )
33
34
       # 定義持有資訊
35
       hold data = {
36
           "price" : 0,
37
           "num": 0 # 做多部位為正數,做空部位為負
38
39
```





69

```
def trade(data df):
41
        for index, row in data df.iterrows():
42
           # 沒有特有部份,建會判斷
43
           if hold data["num"] == 0:
               condition1 = row["RSI 5"] < 20 # 若RSI小於20,則做多(Long)
44
               condition2 = row["RSI 5"] > 80 # 若RSI大於80 , 則做空(Short)
45
46
               # 若RSI / \於20 , 則做多(Long)
47
               if condition1:
48
49
                  trade_price = row["成交價格"] # 取出當下的成交價格
                  num = cash // trade price # 整數除法
50
                  Recording(row["日期"], trade price, "L", num) # 紀錄做多交易
51
52
53
                  # 更動持有狀況,紀錄成交價格與交易數量
54
                  hold data["price"], hold data["num"] = trade price, num
55
                  # 資金更動,扣除胃股票的資金
56
                                                      做多買入時扣除資金
                  cash-= trade_price * num
57
58
59
               # 若RSI大於80 , 則做空(Short)
               elif condition2:
60
61
                  trade price = row["成交價格"] # 取出當下的成交價格
                  num = cash // trade price # 整數除法,
62
63
                  Recording(row["日期"], trade price, "S", num) # 紀錄做空交易
64
65
                  # 更動持有狀況,紀錄成交價格與交易數量,做空數量以負數代表
                  hold_data["price"], hold_data["num"] = trade_price, -num
66
67
                  # 資金更動,假設保證金率為1,目沒有酬券手續費
68
```

cash-= trade price \* num

根據持有資金決定成交量

根據持有資金決定成交量

做空賣出時扣除保證金, 假設保證金 率為100%,沒有融券手續費與利息



```
29 def trade(data_df):
```

for index, row in data\_df.iterrows():

```
71
            # 持有部位,平倉判斷
72
            else:
                # 做多平倉條件
73
                condition1 = hold data["num"] > 0 # 特有數量大於0
74
                condition2 = row["RSI 5"] > 80 # RSI大於80
75
76
                # 做空平倉條件
77
78
                condition3 = hold data["num"] < 0 # 特有數量小於0
                condition4 = row["RSI 5"] < 20 # RSI小於20
79
 80
                # 做多平倉條件
 81
                if condition1 and condition2:
 82
                   num = hold_data["num"] # 當下持有數量
 83
                   trade_price = row["成交價格"] # 取出當下的成交價格
 84
                   Recording(row["日期"], trade_price, "S", num) # 紀錄平倉交易
 85
 86
                   cash+= trade_price * num # 平倉取得金額
 87
 88
                   hold_data["price"], hold_data["num"] = "", 0 # 清空持有紀錄
 89
90
                # 做空平倉條件
 91
                elif condition3 and condition4:
 92
                   num = abs( hold data["num"] )# 當下持有數量,將負數轉正
 93
                   hold_price = hold_data["price"] # 建倉成交價
 94
 95
                   trade_price = row["成交價格"] # 取出當下的成交價格
                   Recording(row["日期"], trade_price, "L", num) # 紀錄平倉交易
 96
97
                   # 拿同保證金、賣出價金,減除同補成本
98
                   cash+= hold price * num + ( hold price - trade price ) * num
99
100
                   hold data["price"], hold data["num"] = "", 0 # 清空持有紀錄
101
```

99 100

101



```
def trade(data df):
41
         for index. row in data df.iterrows():
71
           # 持有部位,平倉判斷
72
           else:
               # 做多平倉條件
73
               condition1 = hold data["num"] > 0 # 特有數量大於0
74
75
               condition2 = row["RSI 5"] > 80 # RSI大於80
76
               # 做空平倉條件
77
               condition3 = hold data["num"] < 0 # 特有數量小於0
78
               condition4 = row["RSI 5"] < 20 # RSI小於20
79
80
               # 做多平會條件
81
               if condition1 and condition2:
82
                  num = hold_data["num"] # 當下持有數量
83
                  trade price = row["成交價格"] # 取出當下的成交價格
84
                  Recording(row["日期"], trade price, "S", num) # 紀錄平倉交易
85
86
                                                                    做多平倉時回補資金
                  cash+= trade_price * num # 平倉取得金額
87
88
                  hold data["price"], hold_data["num"] = "", 0 # 清空特有紀錄
89
90
               # 做空平倉條件
91
               elif condition3 and condition4:
92
                  num = abs( hold data["num"] )# 當下持有數量,將負數轉正
93
                  hold_price = hold_data["price"] # 建倉成交價
94
95
                  trade_price = row["成交價格"] # 取出當下的成交價格
                  Recording(row["日期"], trade_price, "L", num) # 紀錄平倉交易
96
97
                  # 拿回保證金、賣出價金,減除回補成本
98
```

cash+= hold\_price \* num + ( hold\_price - trade\_price ) \* num

hold\_data["price"], hold\_data["num"] = "", 0 # 清空持有紀錄

做空平倉時,拿回保證金、 當初賣出時的價金, 並扣除回補時的價金





#### 29 def trade(data df):

```
# 最後一日平倉
103
         last_date = data_df.iloc[-1]["日期"] # 最後一日日期
104
         last_trade_price = data_df.iloc[-1]["成交價格"] # 最後一日成交價格
105
106
107
         if hold data["num"] > 0: # 若特有數量不為大於0 ,將做多部位平倉
108
             Recording(last date, last trade price, "S", hold data["num"])
109
             cash+= last_trade_price * hold_data["num"] # 平倉取得金額
110
         elif hold_data["num"] < 0: # 若持有數量不為小於0,將做空部位平倉
111
             num = abs( hold data["num"] ) # 建倉成交量
112
             hold price = hold data["price"] # 建倉成交價
113
             Recording(last date, last trade price, "L", num)
114
             # 拿回保證金、賣出價金,減除回補成本
115
             cash+= hold price * num + ( hold price - last trade price ) * num
116
         print(cash)
117
         return record_df # 回傳交易紀錄
118
```





def trade(data df):

• • •

```
# 最後一日平倉
103
         last_date = data_df.iloc[-1]["日期"] # 最後一日日期
104
         last_trade_price = data_df.iloc[-1]["成交價格"] # 最後一日成交價格
105
106
107
         if hold_data["num"] > 0: # 若持有數量不為大於0,將做多部位平倉
108
             Recording(last date, last trade price, "S", hold data["num"])
109
             cash+= last_trade_price * hold_data["num"] # 平倉取得金
110
         elif hold_data["num"] < 0: # 若特有數量不為小於0,將做空部位平倉
111
             num = abs( hold_data["num"] ) # 建倉成交量
112
             hold_price = hold_data["price"] # 建倉成交價
113
             Recording(last_date, last_trade_price, "L", num)
114
             # 拿回保證金、賣出價金,減除回補
115
            cash+= hold price * num + ( hold price - last trade price ) * num
116
         print(cash)
117
         return record_df # 回傳交易紀錄
118
```

最後一日持有部位處理

做多平倉時回補資金

做空平倉時,拿回保證金、 當初賣出時的價金, 並扣除回補時的價金

#### 執行結果





#### 訓練集交易後剩餘 資金與交易結果

	Console 1/A			
子又	内里农/4.	作土.	八又勿	7
78	2210 55			

		-			
7822	210.55				
		time	price	LS	num
0	2015-01-12	00:00:00	40.7	L	24570.0
1	2015-01-28	00:00:00	42.7	S	24570.0
2	2015-03-31	00:00:00	44.0	S	23844.0
3	2015-05-07	00:00:00	42.5	L	23844.0
4	2015-06-01	00:00:00	41.1	L	26396.0
5	2015-10-07	00:00:00	36.55	S	26396.0
6	2015-10-12	00:00:00	37.35	S	25831.0
7	2015-11-11	00:00:00	34.5	L	25831.0
8	2015-11-12	00:00:00	34.25	L	30318.0
9	2016-02-16	00:00:00	28.3	S	30318.0
10	2016-02-18	00:00:00	29.0	S	29587.0
11	2016-05-04	00:00:00	30.4	L	29587.0
12	2016-05-05	00:00:00	30.5	L	26774.0
13	2016-06-03	00:00:00	30.7	S	26774.0
14	2016-07-05	00:00:00	33.95	S	24210.0
15	2016-09-14	00:00:00	34.0	L	24210.0
16	2016-12-19	00:00:00	36.2	L	22672.0
17	2016-12-30	00:00:00	34.5	S	22672.0

1002754.799999998

time price LS num 0 2017-01-12 00:00:00 36.3 S 27548.0

1 2017-12-29 00:00:00 36.2 L 27548.0

測試集交易後剩餘 資金與交易結果

## 交易中 - 交易函數(加入交易成本)



- 手續費率 = 1.425 ‰ = 0.001425 (依據券商不同而有差異)
- 買賣都需要繳交手續費,不足 20 元,以 20 元計價

- 證交稅率 = 3 ‰ = 0.003
- 僅賣出時需要繳納

#### 交易中 - 加入交易成本





- 原始資金 1,000,000
- 做多建倉,成交價 65 元,

- 手續費 = 65 × 15,362 × 手續費率 = 1,423
- 現金 = 1,000,000 65 × 15,362 1,423 = 47

- 做多平倉,成交價 70 元,成交量(持有量) 15,384 股
- 手續費 = 70 × 15,362 × 手續費率 = 1,532
- 證交稅 = 70 × 15,362 × 證交稅率 = 3,226
- 現金 =  $47 + 70 \times 15,362 1,532 3,226 = 1,070,652$

#### 交易中 - 加入交易成本





- 現金 1,070,652
- 做空建倉,成交價 75 元,

- 保證金 = 75 × 14,212 × 100% = 1,065,900
- 手續費 = 75 × 14,212 × 手續費率 = 1,519
- 證交稅 = 75 × 14,212 × 證交稅率 = 3,198
- 現金 = 1,070,652 1,065,900 1,519 3,198 = 35

- 做空平倉,成交價 72 元,成交量(持有量) 14,212 股
- 現金 = 35 + 保證金 + (75 72) × 14,358 = 1,109,009

#### 交易中 - 主程式(加入交易成本)





```
# 以隔天的開盤價當作成交價格
177
     train data["成交價格"] = train_data["開盤價"].shift(-1)
178
     # 最後一天以收盤價當作交易價格
179
     train data["成交價格"].fillna(method = "ffill", inplace = True)
180
181
     # 計算 RSI5
182
     train data["RSI 5"] = talib.RSI(train data["收盤價"], 5)
183
184
     # 以隔天的開盤們當作成交價格
185
     test data["成交價格"] = test data["開盤價"].shift(-1)
186
     # 最後一天以收盤價當作交易價格
187
     test data["成交價格"].fillna(method = "ffill", inplace = True)
188
     # 計算 RSI5
     test data["RSI_5"] = talib.RSI(test_data["收盤價"], 5)
189
190
191
     # 定義初始資金
192
     cash = 1000000
193
     fee rate = 0.001425 # 手續費率,千分之1.425
194
195
     min fee = 20 # 不足20元以20元計價
196
     tax rate = 0.003 # 誇交稅
197
     trade(train data, cash)
198
199
     print(record df)
200
     trade(test data, cash)
201
202
     print(record_df)
```

#### 交易中 - 主程式(加入交易成本)





```
# 以隔天的開盤們當作成交價格
177
     train data["成交價格"] = train data["開盤價"].shift(-1)
178
     # 最後一天以收盤價當作交易價格
179
     train data["成交價格"].fillna(method = "ffill", inplace = True)
180
181
     # 計算 RSI5
     train data["RSI 5"] = talib.RSI(train data["收盤價"], 5)
182
183
184
     # 以隔天的開盤們當作成交價格
185
     test data["成交價格"] = test data["開盤價"].shift(-1)
186
     # 最後一天以收盤價當作交易價格
     test data["成交價格"].fillna(method = "ffill", inplace = True)
187
188
     # 計算 RSI5
189
     test data["RSI 5"] = talib.RSI(test data["收盤價"], 5)
190
191
     # 定義初始資金
192
     cash = 1000000
193
     fee_rate = 0.001425 # 手續費率,千分之1.425
194
     min fee = 20 # 不足20元以20元計價
195
196
     tax rate = 0.003 # 讚交稅
197
     trade(train data, cash)
198
     print(record df)
199
200
     trade(test data, cash)
201
202
     print(record_df)
```

設定證手續費率、最低手續費、證交稅率

#### 交易中 - 交易函數(加入交易成本)





```
29 def trade(data_df):
```

```
41
        for index, row in data df.iterrows():
42
           # 沒有持有部位,建倉判斷
           if hold data["num"] == 0:
43
               condition1 = row["RSI_5"] < 20 # 若RSI小於20,則做多(Long)
44
45
               condition2 = row["RSI 5"] > 80 # 若RSI大於80,則做空(Short)
46
               # 若RSI小於20 , 則做多(Lona)
47
               if condition1:
48
49
                  trade price = row["成交價格"] # 取出當下的成交價格
                  num = cash // (trade price * ( 1 + fee rate ) ) # 整數除法,根據現有資金決定成交股數
50
51
                  Recording(row["日期"], trade price, "L", num) # 紀錄做多交易
52
                  # 更動持有狀況,紀錄成交價格與交易數量
53
54
                  hold data["price"], hold data["num"] = trade price, num
55
56
                  # 交易手續費
                  fee = fee cal( trade price, num )
57
58
59
                  # 資金更動,扣除買股票的資金
                  cash-= trade price * num + fee
60
```

#### 交易中 - 交易函數(加入交易成本)





def trade(data df): for index, row in data df.iterrows(): 41 42 # 沒有持有部位,建倉判斷 if hold data["num"] == 0: 43 condition1 = row["RSI\_5"] < 20 # 若RSI小於20,則做多(Long) 44 condition2 = row["RSI\_5"] > 80 # 若RSI大於80 , 則做空(Short) 45 46 做多計算成交量時, 47 # 若RSI/小於20 , 則做多(Long) 將手續費視為成交價的增加 if condition1: 48 49 trade\_price = row["成交價格"] # 取出當下的成交價格 num = cash // (trade price \* ( 1 + fee rate ) ) # 整數除法,根據現有資金決定成交股數 50 51 Recording(row["日期"], trade price, "L", num) # 紀錄做多交易 52 53 # 更動持有狀況,紀錄成交價格與交易數量 54 hold data["price"], hold data["num"] = trade price, num 55 交易手續費計算函數 56 fee = fee\_cal( trade\_price, num ) 57 (下一頁詳細說明) 58 59 做多買入時扣除資金與手續費 cash-= trade price \* num + fee 60





```
# 計算手續費
146
     def fee cal(trade price, num):
147
         # 交易手續費
148
         fee = trade_price * num * fee_rate
149
150
         # 手續費不足20元則以20元計價
151
         if fee < min fee:</pre>
152
             fee = min_fee
153
154
         return fee # 回傳手續費
155
```

傳入成交價、成交量 計算手續費,並判別若受續費 小於min\_fee,即 20元, 則以20元計算





```
def trade(data df):
        for index, row in data df.iterrows():
41
42
           # 沒有持有部位,建倉判斷
43
            if hold data["num"] == 0:
                 . . .
               # 若RSI/小於20 , 則做多(Long)
               if condition1:
48
                     . . .
               # 若RSI大於80 , 則做空(Short)
62
               elif condition2:
63
                  trade_price = row["成交價格"] # 取出當下的成交價格
64
65
                  num = cash // (trade price * ( 1 + fee rate + tax rate) ) # 整數除法,根據現有資金決定成交股數
                  Recording(row["日期"], trade_price, "S", num) # 紀錄做空交易
66
67
                  # 更動持有狀況,紀錄成交價格與交易數量,做空數量以負數代表
68
                  hold data["price"], hold data["num"] = trade price, -num
69
70
71
                  # 交易手續費
                  fee = fee cal( trade price, num )
72
73
                  # 證交稅
74
                  tax = trade price * num * tax rate
75
76
77
                  # 資金更動,假設保證金率為1,且沒有融券手續費、利息
                  cash-= trade price * num + fee + tax
78
```





```
def trade(data df):
       for index, row in data df.iterrows():
           # 沒有持有部位,建倉判斷
           if hold data["num"] == 0:
                . . .
              # 若RSI/小於20 , 則做多(Long)
               if condition1:
48
                                                                         做空計算成交量時,
                     . . .
              # 若RSI大於80 , 則做空(Short)
62
              elif condition2:
63
                  trade price = row["成交價格"] # 取出當下的成交價格
65
                  num = cash // (trade price * ( 1 + fee rate + tax rate) ) # 整數除法,根據現有資金決定成交股數
66
                  Recording(row|"日期"|, trade price, "S", num) # 紀錄做等父易
67
68
                  # 更動持有狀況,紀錄成交價格與交易數量,做空數量以負數代表
                  hold_data["price"], hold_data["num"] = trade_price, -num
69
70
                  # 交易手續費
71
                  fee = fee cal( trade price, num )
72
73
                  # 證交稅
74
75
                  tax = trade price * num * tax rate
76
                  # 資金更動,假設保證金率為1,且沒有融券手續費、利息
77
                  cash-= trade price * num + fee + tax
78
```

將手續費與證交稅視為成交價的增加

計算手續費、證交稅

做空賣出時扣除資金、手續費與證交稅





```
def trade(data df):
         for index, row in data_df.iterrows():
41
42
             # 沒有持有部位,建倉判斷
             if hold data["num"] == 0:
43
 80
             # 持有部位,平倉判斷
 81
             else:
                      . . .
90
                # 做多平倉條件
                if condition1 and condition2:
91
                    num = hold_data["num"] # 當下持有數量
92
                    trade price = row["成交價格"] # 取出當下的成交價格
93
94
                    Recording(row["日期"], trade_price, "S", num) # 紀錄平倉交易
95
96
                    # 交易手續費
                    fee = fee cal( trade price, num )
97
98
99
                    # 證莎稅
                    tax = trade_price * num * tax_rate
100
101
                    cash+= trade price * num - fee - tax # 平倉取得金額
102
103
                    hold_data["price"], hold_data["num"] = "", 0 # 清空持有紀錄
104
```





```
def trade(data df):
        for index, row in data_df.iterrows():
41
42
            # 沒有持有部位,建倉判斷
            if hold data["num"] == 0:
43
            # 持有部位,平倉判斷
 80
            else:
 81
                     . . .
90
               # 做多平倉條件
               if condition1 and condition2:
91
                  num = hold_data["num"] # 當下持有數量
92
                  trade price = row["成交價格"] # 取出當下的成交價格
93
94
                   Recording(row["日期"], trade_price, "S", num) # 紀錄平倉交易
95
                  # 交易手續費
96
                  fee = fee cal( trade price, num )
97
                                                              做多平倉時,計算手續費、證交稅
98
99
                  # 證莎稅
                  tax = trade_price * num * tax_rate
100
                                                                          做多平倉時回補資金並扣
101
                  cash+= trade price * num - fee - tax # 平倉取得金額
                                                                              除手續費、證交稅
102
103
                  hold_data["price"], hold_data["num"] = "", 0 # 清空持有紀錄
104
```

119





```
def trade(data df):
         for index, row in data_df.iterrows():
 41
 42
             # 沒有持有部位,建倉判斷
             if hold data["num"] == 0:
 43
             # 持有部位,平倉判斷
 80
 81
             else:
90
                # 做多平倉條件
91
                if condition1 and condition2:
106
                # 做空平倉條件
                elif condition3 and condition4:
107
                    num = -hold_data["num"] # 當下持有數量,將負數轉正
108
                    hold price = hold data["price"] # 建倉成交價
109
                    trade_price = row["成交價格"] # <u>取出當下的成交價格</u>
110
                    Recording(row["日期"], trade_price, "L", num) # 紀錄平倉交易
111
112
113
                    # 交易手續費
114
                    fee = fee cal( trade price, num )
115
                    # 拿回保證金、賣出價金,減除回補成本
116
                    cash+= hold_price * num + (hold_price - trade_price) * num - fee
117
118
                    hold_data["price"], hold_data["num"] = "", 0 # 清空持有紀錄
```





```
def trade(data df):
        for index, row in data_df.iterrows():
 41
 42
            # 沒有持有部位,建倉判斷
            if hold data["num"] == 0:
 43
            # 持有部份,平倉判斷
 80
            else:
 81
90
               # 做多平倉條件
91
               if condition1 and condition2:
106
               # 做空平倉條件
               elif condition3 and condition4:
107
                   num = -hold_data["num"] # 當下持有數量,將負數轉正
108
                   hold price = hold data["price"] # 建倉成交價
109
                   trade_price = row["成交價格"] # <u>取出當下的成交價格</u>
110
                   Recording(row["日期"], trade_price, "L", num) # 紀錄平倉交易
111
112
                                                        做空平倉時,計算手續費
113
                   fee = fee cal( trade price, num )
114
115
                                                                                   做空平倉時,
                   # 拿回保證金、賣出價金,減除回補成本
116
                                                                              拿回保證金並結算損益
                   cash+= hold_price * num + (hold_price - trade_price) * num - fee
117
118
                                                                            大學 AI 金融科技中心版權所有
                   hold_data["price"], hold_data["num"] = "", 0 # 清空持有紀錄
119
```





def trade(data\_df):

• • •

```
121
         # 最後一日平倉
         last date = data_df.iloc[-1]["日期"] # 最後一日日期
122
123
         last_trade_price = data_df.iloc[-1]["成交價格"] # 最後一日成交價格
124
         if hold data["num"] > 0: # 若特有數量不為大於0 , 將做多部位平倉
125
126
             num = hold data["num"]
127
             Recording(last date, last trade price, "S", hold data["num"])
128
            # 交易手續費
129
            fee = fee cal( last trade price, num )
130
            # 證莎稅
            tax = last trade price * num * tax rate
131
132
             cash+= last trade price * hold data["num"] - fee - tax # 平倉取得金額
133
134
135
         elif hold data["num"] < 0: # 若持有數量不為小於0,將做空部位平倉
136
             num = -hold data["num"]
             hold price = hold_data["price"] # 建倉成交價
137
             Recording(last date, last trade price, "L", num)
138
139
            # 交易手續費
140
            fee = fee_cal( last_trade_price, num )
            # 拿回保證金、賣出價金,減除回補成本
141
             cash+= hold price * num + ( hold price - last trade price ) * num - fee
142
         print(cash)
143
         return record_df # 回傳交易紀錄
144
```



29 def trade(data\_df):

. . .

```
121
        # 最後一日平倉
                                                                         最後一日持倉處理
        last date = data df.iloc[-1]["日期"] # 最後一日日期
122
        last_trade_price = data_df.iloc[-1]["成交價格"] # 最後一日成交價格
123
124
        if hold data["num"] > 0: # 若特有數量不為大於0,將做多部位平倉
125
           num = hold data["num"]
126
           Recording(last date, last trade price, "S", hold data["num"])
127
128
129
           fee = fee cal( last trade price, num )
                                                       做多平倉時,計算手續費、證交稅
130
           # 讚交稅
           tax = last trade_price * num * tax_rate
131
                                                                             做多平倉時回補資金並扣
132
           cash+= last_trade_price * hold_data["num"] - fee - tax # 平倉取得金額
133
                                                                                 除手續費、證交稅
134
135
        elif hold data["num"] < 0: # 若持有數量不為小於0,將做空部位平倉
           num = -hold data["num"]
136
           hold price = hold_data["price"] # 建倉成交價
137
           Recording(last date, last trade price, "L", num)
138
                                                          做空平倉時,計算手續費
139
           fee = fee_cal( last_trade_price, num )
140
141
           # 拿回保證金、賣出價金,減除回補成本
           cash+= hold price * num + ( hold price - last trade price ) * num - fee
                                                                                  做空平倉時,
142
        print(cash)
143
                                                                             拿回保證金並結算損益
        return record_df # 回傳交易紀錄
144
```

## 執行結果





	Console 1/A 🗵	Console 2/A 🗵				
742189.4887862502						
		time	price	LS	num	
0	2015-01-12	00:00:00	40.7	L	24535.0	
1	2015-01-28	00:00:00	42.7	S	24535.0	
2	2015-03-31	00:00:00	44.0	S	23600.0	
3	2015-05-07	00:00:00	42.5	L	23600.0	
4	2015-06-01	00:00:00	41.1	L	26055.0	
5	2015-10-07	00:00:00	36.55	S	26055.0	
6	2015-10-12	00:00:00	37.35	S	25272.0	
7	2015-11-11	00:00:00	34.5	L	25272.0	
8	2015-11-12	00:00:00	34.25	L	29584.0	
9	2016-02-16	00:00:00	28.3	S	29584.0	
10	2016-02-18	00:00:00	29.0	S	28615.0	
11	2016-05-04	00:00:00	30.4	L	28615.0	
12	2016-05-05	00:00:00	30.5	L	25817.0	
13	2016-06-03	00:00:00	30.7	S	25817.0	
14	2016-07-05	00:00:00	33.95	S	23140.0	
15	2016-09-14	00:00:00	34.0	L	23140.0	
16	2016-12-19	00:00:00	36.2	L	21608.0	
17	2016-12-30	00:00:00	34.5	S	21608.0	

#### 訓練集交易後剩餘 資金與交易結果

```
996922.4599749999

time price LS num

0 2017-01-12 00:00:00 36.3 S 27426.0

1 2017-12-29 00:00:00 36.2 L 27426.0
```

測試集交易後剩餘資金與交易結果

## 執行結果比較





Console 1/A E 無交易成本						
782	2210.55					
		time	price	LS	num	
0	2015-01-12	00:00:00	40.7	L	24570.0	
1	2015-01-28	00:00:00	42.7	S	24570.0	
2	2015-03-31	00:00:00	44.0	S	23844.0	
3	2015-05-07	00:00:00	42.5	L	23844.0	
4	2015-06-01	00:00:00	41.1	L	26396.0	
5	2015-10-07	00:00:00	36.55	S	26396.0	
6	2015-10-12	00:00:00	37.35	S	25831.0	
7	2015-11-11	00:00:00	34.5	L	25831.0	
8	2015-11-12	00:00:00	34.25	L	30318.0	
9	2016-02-16	00:00:00	28.3	S	30318.0	
10	2016-02-18	00:00:00	29.0	S	29587.0	
11	2016-05-04	00:00:00	30.4	L	29587.0	
12	2016-05-05	00:00:00	30.5	L	26774.0	
13	2016-06-03	00:00:00	30.7	S	26774.0	
14	2016-07-05	00:00:00	33.95	S	24210.0	
15	2016-09-14	00:00:00	34.0	L	24210.0	
16	2016-12-19	00:00:00	36.2	L	22672.0	

34.5

22672.0

2016-12-30 00:00:00

	onal losging	2018 2018 Value of Science and
	Console 1/A ☑	Console 2/A
742	2189.4887862	502
		tin
0	2015-01-12	00:00:0
1	2015-01-28	00:00:0
2	2015-03-31	00:00:0
3	2015-05-07	00:00:0
4	2015-06-01	00:00:0
5	2015-10-07	00:00:0
6	2015-10-12	00:00:0
7	2015-11-11	00:00:0
8	2015-11-12	00:00:0

有交易成本

742189.4887862502					
		time	price	LS	num
0	2015-01-12	00:00:00	40.7	L	24535.0
1	2015-01-28	00:00:00	42.7	S	24535.0
2	2015-03-31	00:00:00	44.0	S	23600.0
3	2015-05-07	00:00:00	42.5	L	23600.0
4	2015-06-01	00:00:00	41.1	L	26055.0
5	2015-10-07	00:00:00	36.55	S	26055.0
6	2015-10-12	00:00:00	37.35	S	25272.0
7	2015-11-11	00:00:00	34.5	L	25272.0
8	2015-11-12	00:00:00	34.25	L	29584.0
9	2016-02-16	00:00:00	28.3	S	29584.0
10	2016-02-18	00:00:00	29.0	S	28615.0
11	2016-05-04	00:00:00	30.4	L	28615.0
12	2016-05-05	00:00:00	30.5	L	25817.0
13	2016-06-03	00:00:00	30.7	S	25817.0
14	2016-07-05	00:00:00	33.95	S	23140.0
15	2016-09-14	00:00:00	34.0	L	23140.0
16	2016-12-19	00:00:00	36.2	L	21608.0
17	2016-12-30	00:00:00	34.5	S	21608.0



# 章節到此結束,有任何問題歡迎提出來討論!

