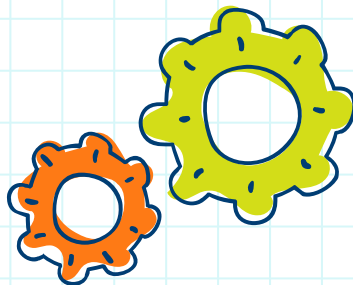
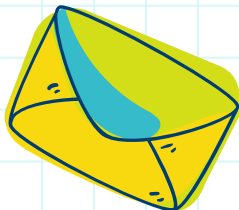
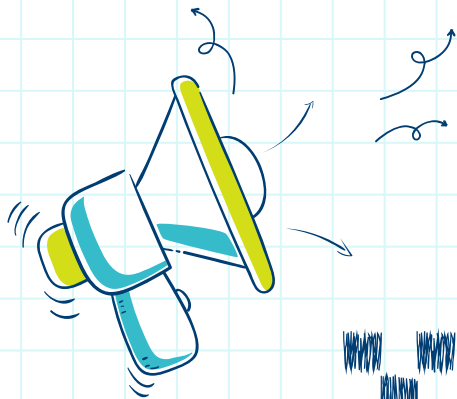


股票程式期末報告

410721204 資工四 熊亭媛



Step



01

介紹策略

02

Code

03

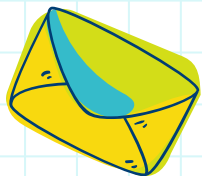
策略績效

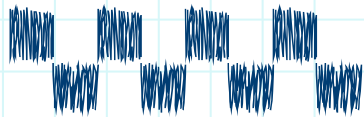
04

交易資訊

05

自我觀點



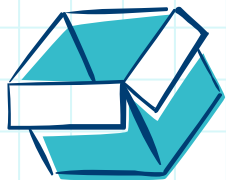


01、02

介紹策略

Code





策略一



01

買點:

當 $K > D$ 時,
當天股價 $<$ 20日平均

02

賣點:

K線和D線死亡交叉
($K > 85$)

03

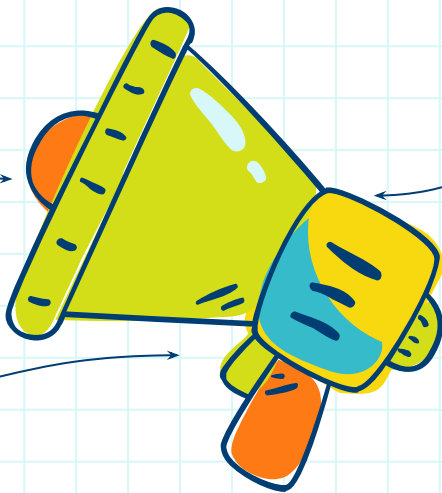
買入方法

手頭大於10萬時,取一半購買
手頭小於10萬時,則 all in

04

賣出方法

每次賣出手上5成的股票
當手上股票低於3千股時,便全部售出

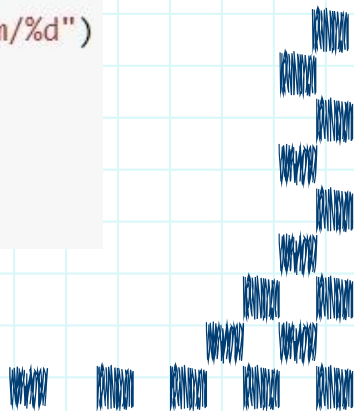


Code

```
import math
import pandas as pd
import numpy as np
from datetime import datetime
import mplfinance as mpf

#讀入股票資料
for file in lists:
    print(file)
    stocks = pd.read_csv(file)
    stocks.columns = ['Date', 'Open', 'High', 'Low', 'Close', 'Volume']
    stocks.iloc[:, 0] = pd.to_datetime(stocks.iloc[:, 0], format = "%Y/%m/%d")
    stocks = stocks.set_index(pd.DatetimeIndex(stocks["Date"]))

    stocks_close = [float(line) for line in stocks['Close']] #收盤價
    stocks_high = [float(line) for line in stocks['High']] #最高價
    stocks_low = [float(line) for line in stocks['Low']] #最低價
```



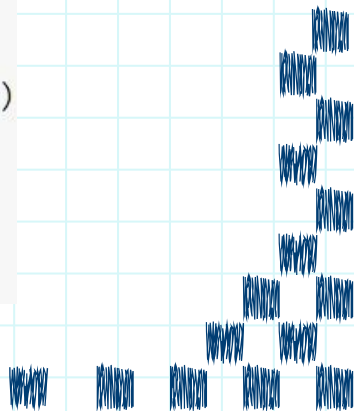
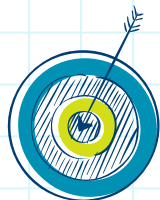
Code

```
rsv = []          #RSV list
Kpoint = [50]     #K值 list, 初始設定50
Dpoint = [50]     #D值 list

#計算RSV
for i in range(11, len(stocks['Date'])):
    low = min(stocks_low[i-8 : i+1])
    high = max(stocks_high[i-8 : i+1])
    rsv.append(round(100 * ((stocks_close[i] - low) / (high - low)), 2))

#計算 K值和 D值
for i in range(0, len(rsv) - 1):
    Kpoint.append(round((Kpoint[i] * (2 / 3) + rsv[i + 1] * (1 / 3)), 2))
    Dpoint.append(round((Dpoint[i] * (2 / 3) + Kpoint[i + 1] * (1 / 3)), 2))

#補 0
for i in range(0, 11):
    Kpoint.insert(i, np.nan)
    Dpoint.insert(i, np.nan)
```



Code

#買賣點所使用的陣列

```
points_buy = [np.nan] * len(stocks)
points_sell = [np.nan] * len(stocks)
```

#初始金額

```
initial = 500000
money = initial
```

#持有股票數量(股)

```
stock = 0
```

#每次購入張數

```
unit = 1
```

#目前獲利

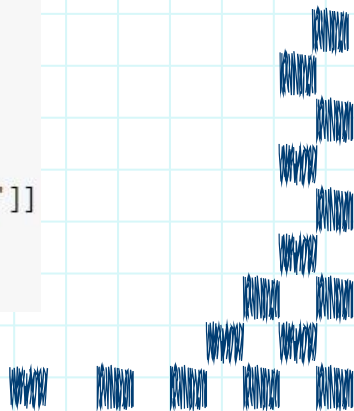
```
gain = 0
```

#紀錄交易資訊

```
detail = [['日期', '買/賣(+/-)', '價格', '價格(稅)', '實際花費', '目前獲利', '剩餘金額']]
```

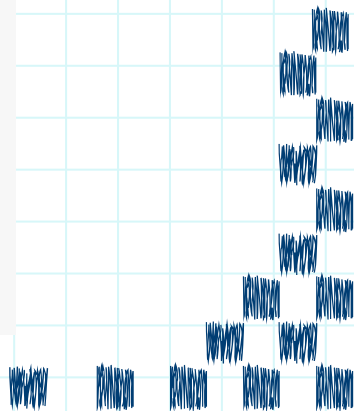
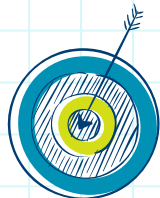
#暫時交易資訊

```
temp_detail = []
```



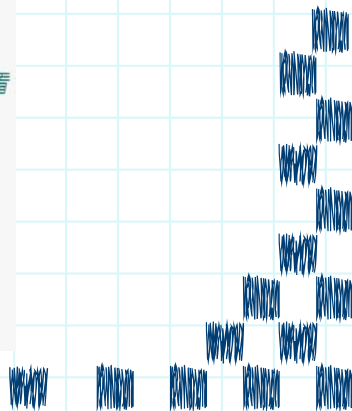
Code

```
for i in range(9, stocks.shape[0]): #KD_n=9
    # 當K > D時, 且當天股價 < 20日平均
    if((Kpoint[i-1] < Dpoint[i-1]) and #昨日 k < d
        (Kpoint[i] > Dpoint[i]) and #今日 k > d
        (Kpoint[i-1] < Kpoint[i]) and #昨日 k < 今日k
        (stocks_close[i] < (sum(stocks_close[i-19:i+1])/20))): #今日收盤< 20日的收盤平均
        price = round(stocks_close[i] * 1.001425, 1) #計算含稅的股價
        if money > 100000: #如果購買力大於10萬, 則每次取持有的0.4購買
            num = math.floor((money*0.4)/price)
        else:
            num = math.floor(money/price) #如果購買力小於10萬, 則all in
        buy = round(num*price) #計算實際花費
        money = money-buy #更新 money
        gain -= buy #淨收益計算(含稅)
        points_buy[i] = stocks_low[i] * 0.98 #購買日最低價 * 0.98方便觀看
        #交易資訊(日期)
        temp_detail.append(str(datetime.date(stocks['Date'][i]).month) + '/' +
                           str(datetime.date(stocks['Date'][i]).day))
        temp_detail.append('+' + str(num)) #買/賣 num股
        stock += num #更新持有股數 +num股
        temp_detail.append(stocks_close[i]) #購買日買入價格: "收盤價"
        temp_detail.append(price) #抽稅之後的價格(含稅)
        temp_detail.append('-' + str(buy)) #實際花費 buy元
        temp_detail.append(gain) #目前獲利
        temp_detail.append(money) #剩餘金額
        detail.append(temp_detail) #紀錄交易資訊至正是表格中
        temp_detail = [] #清除暫存交易資訊
```



Code

```
if(Kpoint[i] > 85):# K線和D線死亡交叉 (K線>80)
    # k downcross d
    if((Kpoint[i-1] > Dpoint[i-1]) and #昨日 k > d
        (Kpoint[i] < Dpoint[i]) and #今日 k < d
        (Kpoint[i-1] > Kpoint[i])): #昨日 k > 今日k
        #交易資訊(日期)
        temp_detail.append(str(datetime.date(stocks['Date'][i]).month) + '/' +
                            str(datetime.date(stocks['Date'][i]).day))
    if stock < 3000: #如果手上股數小於3000股就全部賣出
        sell = round(stocks_close[i] * 0.995575 * stock)
        temp_detail.append('-' + str(stock)) #股票全部賣出
        stock=0
    else:
        sell = round(stocks_close[i] * 0.995575 * stock*0.5) #如果手上股數大於3000股就賣出一半
        stock = stock*0.5 #持股一半
        temp_detail.append('-' + str(stock)) #賣出的股票
    money = money + sell # 更新 money
    points_sell[i] = stocks_high[i] * 1.02 #繪製點所使用的陣列，賣出日最高價 * 1.02方便觀看
    gain += sell #淨收益計算
    temp_detail.append(stocks_close[i]) #賣出日賣出價格：“收盤價”
    temp_detail.append(round(stocks_close[i] * 0.995575)) #抽稅之後的價格(含稅)
    temp_detail.append('+' + str(sell)) #實際收入（四捨五入到個位數）
    temp_detail.append(gain) #目前獲利
    temp_detail.append(money) #剩餘金額
    detail.append(temp_detail) #紀錄交易資訊至正是表格中
    temp_detail = []
```

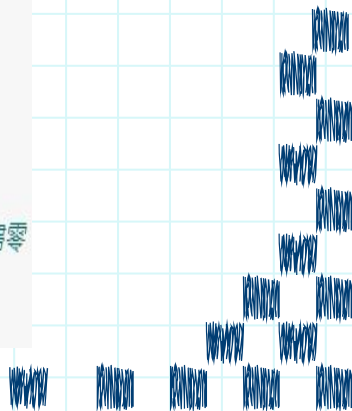
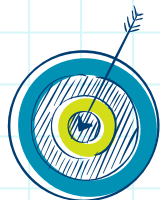


Code

```
#將持有的股數全數出清
if(stock!=0):
    sell = round(stocks_close[i] * 0.995575 * stock)
    money = money + sell
    #繪製點所使用的陣列，賣出日最高價 * 1.02方便觀看
    points_sell[i] = stocks_high[i] * 1.02
    #淨收益計算
    gain += sell
    #交易資訊(日期)
    temp_detail.append(str(datetime.date(stocks['Date'][i]).month) + '/' +
                       str(datetime.date(stocks['Date'][i]).day))
    temp_detail.append('-' + str(stock))
    temp_detail.append(stocks_close[i])
    temp_detail.append(round(stocks_close[i] * 0.995575))
    temp_detail.append('+' + str(sell))
    temp_detail.append(gain)
    temp_detail.append(money)
    stock = 0
    detail.append(temp_detail)
    temp_detail = []

#計算實際收入
#更新 money

#股票全部賣出
#賣出日賣出價格: "收盤價"
#抽稅之後的價格(含稅)
#實際收入 (四捨五入到個位數)
#目前獲利
#剩餘金額
#因為全部股票賣出，所以當前股票張數歸零
#紀錄交易資訊至正是表格中
#清除暫存交易資訊
```



Code

#支出、收入、淨收益、投資報酬率

cost = initial

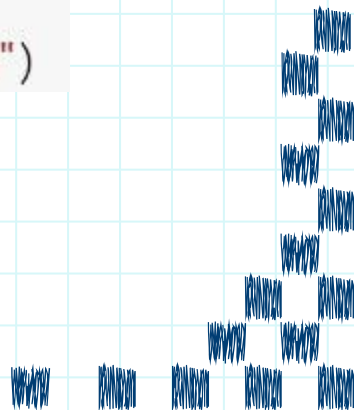
income = money


print("支出:", cost, "元")

print("收入:", income, "元")

print("淨收益:", income - cost, "元")

print("投資報酬率:", round(100 * (income - cost) / cost, 2), "%")





策略一 績效

0050_2018.csv
支出： 500000 元
收入： 628574 元
淨收益： 128574 元
投資報酬率： 25.71 %

1304_2011.csv
支出： 500000 元
收入： 290771 元
淨收益： -209229 元
投資報酬率： -41.85 %

1762_2010.csv
支出： 500000 元
收入： 486832 元
淨收益： -13168 元
投資報酬率： -2.63 %

2065_2017.csv
支出： 500000 元
收入： 665619 元
淨收益： 165619 元
投資報酬率： 33.12 %

2236_2015.csv
支出： 500000 元
收入： 284489 元
淨收益： -215511 元
投資報酬率： -43.1 %

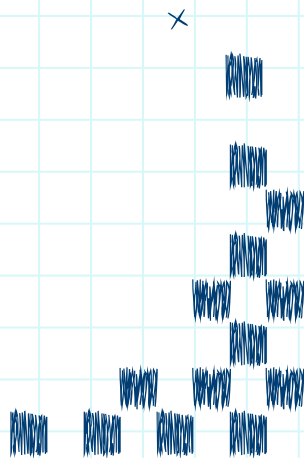
2330_2020.csv
支出： 500000 元
收入： 875252 元
淨收益： 375252 元
投資報酬率： 75.05 %

2603_2020.csv
支出： 500000 元
收入： 2498042 元
淨收益： 1998042 元
投資報酬率： 399.61 %

2748_2016.csv
支出： 500000 元
收入： 348180 元
淨收益： -151820 元
投資報酬率： -30.36 %

2886_2019.csv
支出： 500000 元
收入： 509728 元
淨收益： 9728 元
投資報酬率： 1.95 %

2926_2013.csv
支出： 500000 元
收入： 183584 元
淨收益： -316416 元
投資報酬率： -63.28 %





策略二



01

買點:

rsi線向下突破 25 ,
rsi線持續在 25 以下

02

賣點:

K線和D線死亡交叉
(K線>85)

03

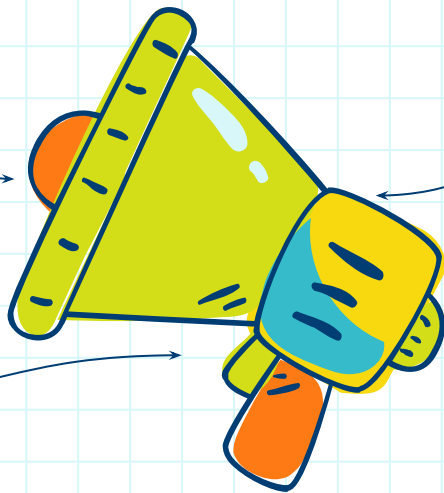
買入方法

手頭大於 10 萬時,取一半購買
手頭小於 10 萬時,則 all in

04

賣出方法

每次賣出手上 5 成的股票
當手上股票校於 3 千股時,便全部售出



Code

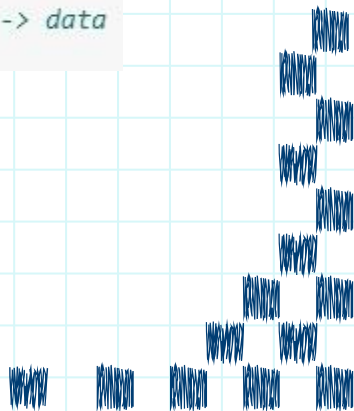
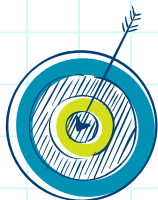
```
stocks_close = [float(line) for line in stocks['Close']]
stocks_high = [float(line) for line in stocks['High']]
stocks_low = [float(line) for line in stocks['Low']]
```

#計算RSV

```
for i in range(11, len(stocks['Date'])):
    low = min(stocks_low[i-8 : i+1])
    high = max(stocks_high[i-8 : i+1])
    rsv.append(round(100 * ((stocks_close[i] - low) / (high - low)), 2))
```

#使用talib

```
stocks.columns = ['date', 'open', 'high', 'low', 'close', 'volume'] #Data -> data
rsiLine = abstract.RSI(stocks, 9) #9日平均, 計算RSI
```



Code

```
#買入點：rsi線持續在25以下
for i in range(9, len(rsiLine)):
    if(rsiLine[i - 1] < 25 and rsiLine[i] < 25) :
        price = round(stocks_close[i] * 1.001425, 1) #計算含稅的股價
        if money > 100000: #如果購買力大於10萬, 則每次取持有的0.4購買
            num = math.floor((money*0.4)/price)
        else:
            num = math.floor(money/price) #如果購買力小於10萬, 則all in

        buy = round(num*price) #計算實際花費
        money = money-buy #更新 money
        gain -= buy #淨收益計算(含稅)
        points_buy[i] = stocks_low[i] * 0.98 #購買日最低價 * 0.98方便觀看

#紀錄交易資訊
temp_detail.append( str(datetime.date(stocks['date'][i]).month) +
                    '/' + str(datetime.date(stocks['date'][i]).day) )
temp_detail.append('+ ' + str(num)) #股數(股)
total_stock.append(num)
stock += num
temp_detail.append(stocks_close[i]) #購買日買入價格: "收盤價"
temp_detail.append(price) #抽稅之後的價格(含稅)
temp_detail.append('- ' + str(buy)) #實際花費 buy元
temp_detail.append(gain) #目前獲利
temp_detail.append(money) #剩餘金額
detail.append(temp_detail) #紀錄交易資訊至正是表格中
temp_detail = [] #清除暫存交易資訊
```

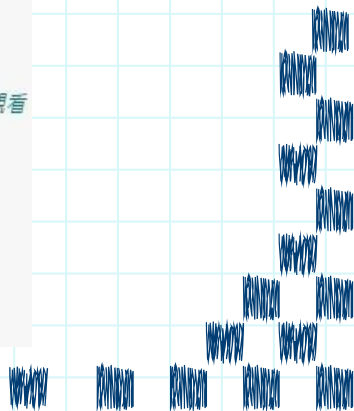
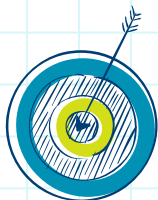



Code

```
if(Kpoint[i] > 85):# K線和D線死亡交叉 (K線>85 )
    # k downcross d
    if((Kpoint[i-1] > Dpoint[i-1]) and #昨日 k > d
        (Kpoint[i] < Dpoint[i]) and #今日 k < d
        (Kpoint[i-1] > Kpoint[i])): #昨日 k > 今日k
        #交易資訊(日期)
        temp_detail.append(str(datetime.date(stocks['date'][i]).month) +
                            '/' + str(datetime.date(stocks['date'][i]).day))

    if stock < 3000: #如果手上股數小於3000股就全部賣出
        sell = round(stocks_close[i] * 0.995575 * stock) #全部賣出
        temp_detail.append('-' + str(stock)) #股票全部賣出
        stock=0
    else:
        sell = round(stocks_close[i] * 0.995575 * stock*0.5) #如果手上股數大於3000股就賣出一半
        stock = stock*0.5 #賣出、持股各一半
        temp_detail.append('-' + str(stock)) #賣出的股票

    money = money + sell
    points_sell[i] = stocks_high[i] * 1.02 #更新 money
    gain += sell #繪製點所使用的陣列，賣出日最高價 * 1.02方便觀看
    temp_detail.append(stocks_close[i]) #淨收益計算
    temp_detail.append(round(stocks_close[i] * 0.995575)) #賣出日賣出價格: "收盤價"
    temp_detail.append('+' + str(sell)) #抽稅之後的價格(含稅)
    temp_detail.append(gain) #實際收入 (四捨五入到個位數)
    temp_detail.append(money) #目前獲利
    detail.append(temp_detail) #剩餘金額
    temp_detail = [] #紀錄交易資訊至正是表格中
```





策略二 績效

0050_2018.csv
支出： 500000 元
收入： 718160 元
淨收益： 218160 元
投資報酬率： 43.63 %

1304_2011.csv
支出： 500000 元
收入： 283673 元
淨收益： -216327 元
投資報酬率： -43.27 %

1762_2010.csv
支出： 500000 元
收入： 502390 元
淨收益： 2390 元
投資報酬率： 0.48 %

2065_2017.csv
支出： 500000 元
收入： 619185 元
淨收益： 119185 元
投資報酬率： 23.84 %

2236_2015.csv
支出： 500000 元
收入： 275478 元
淨收益： -224522 元
投資報酬率： -44.9 %

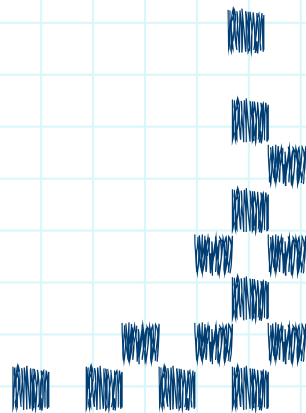
2330_2020.csv
支出： 500000 元
收入： 876715 元
淨收益： 376715 元
投資報酬率： 75.34 %

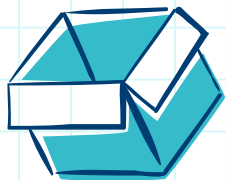
2603_2020.csv
支出： 500000 元
收入： 3997282 元
淨收益： 3497282 元
投資報酬率： 699.46 %

2748_2016.csv
支出： 500000 元
收入： 407638 元
淨收益： -92362 元
投資報酬率： -18.47 %

2886_2019.csv
支出： 500000 元
收入： 567030 元
淨收益： 67030 元
投資報酬率： 13.41 %

2926_2013.csv
支出： 500000 元
收入： 206226 元
淨收益： -293774 元
投資報酬率： -58.75 %





策略三



01

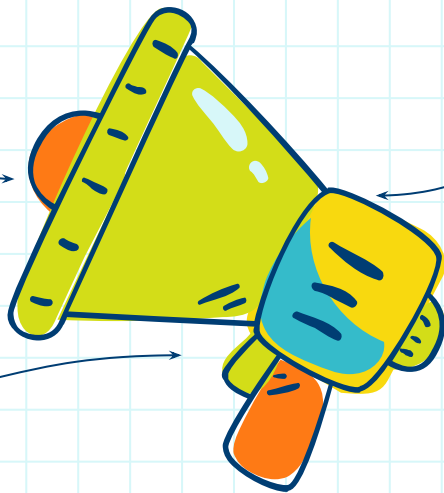
買點:

MACD 快線(DIF)
向上突破慢線(DEM)

02

賣點:

MACD 黃金交叉點
快線(DIF)向下突破慢線(DEM),
且兩者皆 <0 ,
且收盤價 >20 日的平均



03

買入方法

手頭大於10萬時,取一半購買
手頭小於10萬時,則 all in

04

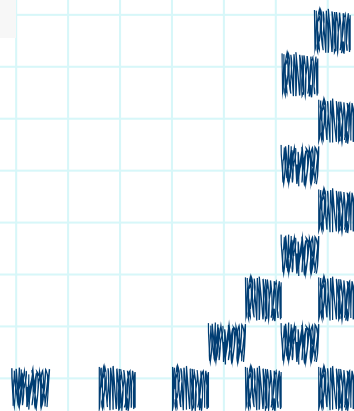
賣出方法

每次賣出手上5成的股票
當手上股票校於3千股時,便全部售出

Code

#製作MACD圖

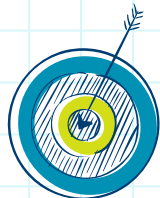
```
ema5 = stocks['Close'].ewm(span = 5, adjust = False).mean() #5日移動平均線  
ema20 = stocks['Close'].ewm(span = 20, adjust = False).mean() #20日移動平均線  
DIF = ema5 - ema20 #快線  
DEM = DIF.ewm(span = 7, adjust = False).mean() #慢線span = 7
```



Code

```
for i in range(1, len(stocks['Date'])):
    if(DIF[i] > DEM[i] and DIF[i - 1] < DEM[i - 1]): #快線(DIF)向上突破慢線(DEM)
        price = round(stocks_close[i] * 1.001425, 1) #計算含稅的股價
        if money > 100000: #如果購買力大於10萬,則每次取持有的0.4購買
            num = math.floor((money*0.4)/price)
        else:
            num = math.floor(money/price) #如果購買力小於10萬,則all in
        buy = round(num*price) #計算實際花費
        money = money-buy #更新 money
        gain -= buy #淨收益計算(含稅)
        points_buy[i] = stocks_low[i] * 0.98 #購買日最低價 * 0.98方便觀看

#紀錄交易資訊
temp_detail.append( str(datetime.date(stocks['Date'][i]).month) +
                    '/' + str(datetime.date(stocks['Date'][i]).day) )
temp_detail.append('+ ' + str(num)) #股數(股)
total_stock.append(num)
stock += num
temp_detail.append(stocks_close[i]) #購買日買入價格: "收盤價"
temp_detail.append(price) #抽稅之後的價格(含稅)
temp_detail.append('- ' + str(buy)) #實際花費 buy元
temp_detail.append(gain) #目前獲利
temp_detail.append(money) #剩餘金額
detail.append(temp_detail) #紀錄交易資訊至正是表格中
temp_detail = [] #清除暫存交易資訊
```

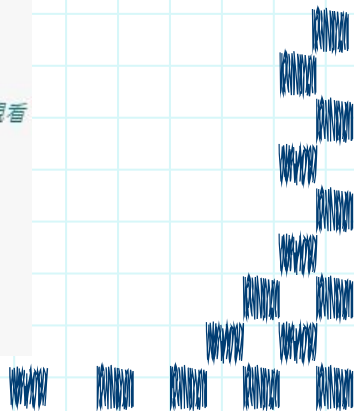



Code

```
#賣出：快線(DIF)向下突破慢線(DEM)，且兩者皆<0，且收盤價>20日的平均
if(DIF[i] < 0 and DEM[i] < 0 and (stocks_close[i] > (sum(stocks_close[i-19:i+1])/20))):
    if((DIF[i-1] > DEM[i-1]) and
       (DIF[i] < DEM[i]) and
       (DIF[i-1] > DEM[i]))):
        #交易資訊(日期)
        temp_detail.append(str(datetime.date(stocks['Date'][i]).month) +
                           '/' + str(datetime.date(stocks['Date'][i]).day))

    if stock < 3000: #如果手上股數小於3000股就全部賣出
        sell = round(stocks_close[i] * 0.995575 * stock) #全部賣出
        temp_detail.append('-' + str(stock))                #股票全部賣出
        stock=0
    else:
        sell = round(stocks_close[i] * 0.995575 * stock*0.5) #如果手上股數大於3000股就賣出一半
        stock = stock*0.5 #賣出、持股各一半
        temp_detail.append('-' + str(stock))                #賣出的股票

money = money + sell # 更新 money
points_sell[i] = stocks_high[i] * 1.02 #繪製點所使用的陣列，賣出日最高價 * 1.02方便觀看
gain += sell #淨收益計算
temp_detail.append(stocks_close[i]) #賣出日賣出價格："收盤價"
temp_detail.append(round(stocks_close[i] * 0.995575)) #抽稅之後的價格(含稅)
temp_detail.append('+' + str(sell)) #實際收入 (四捨五入到個位數)
temp_detail.append(gain) #目前獲利
temp_detail.append(money) #剩餘金額
detail.append(temp_detail) #紀錄交易資訊至正是表格中
temp_detail = []
```





策略三 績效

2330_2020.csv
支出：500000 元
收入：1117010 元
淨收益：617010 元
投資報酬率：123.4 %

2603_2020.csv
支出：500000 元
收入：781745 元
淨收益：281745 元
投資報酬率：56.35 %

2748_2016.csv
支出：500000 元
收入：386783 元
淨收益：-113217 元
投資報酬率：-22.64 %

2886_2019.csv
支出：500000 元
收入：538301 元
淨收益：38301 元
投資報酬率：7.66 %

2926_2013.csv
支出：500000 元
收入：251785 元
淨收益：-248215 元
投資報酬率：-49.64 %

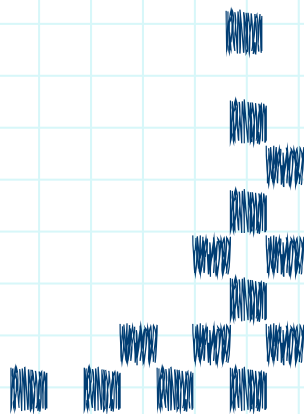
0050_2018.csv
支出：500000 元
收入：641911 元
淨收益：141911 元
投資報酬率：28.38 %

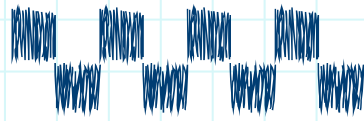
1304_2011.csv
支出：500000 元
收入：362316 元
淨收益：-137684 元
投資報酬率：-27.54 %

1762_2010.csv
支出：500000 元
收入：476127 元
淨收益：-23873 元
投資報酬率：-4.77 %

2065_2017.csv
支出：500000 元
收入：630808 元
淨收益：130808 元
投資報酬率：26.16 %

2236_2015.csv
支出：500000 元
收入：231755 元
淨收益：-268245 元
投資報酬率：-53.65 %





05

分享 自我觀點



績效


優秀

2603

1101.18%



差

2926

-63.28%

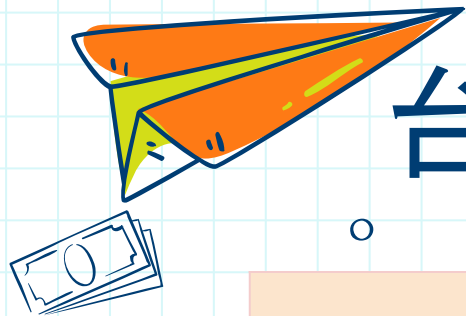


普通

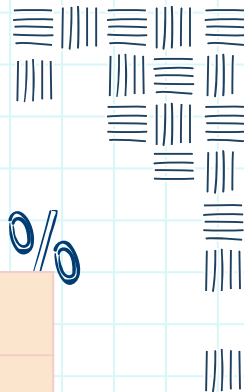
2886

14.01%





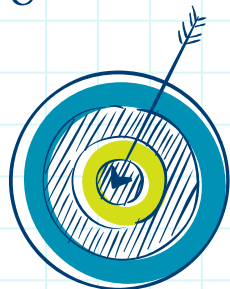
台灣經濟成長預測

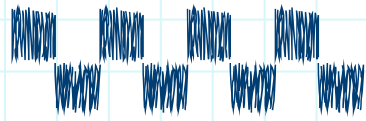


	0.98	4.37	3.8	3.5
2022	Q1	Q2	Q3	Q4

	5.08	5.86	2.12	2.55
2021	Q1	Q2	Q3	Q4

	2.51	0.35	3.92	4.94
2020	Q1	Q2	Q3	Q4





03

績效 比較表





	ETF	塑膠產業	生技產業	櫃鋼鐵產業	汽車產業
股票代碼	0050	1304	1762	2065	2236
策一	25.71%	-41.85%	-2.63%	33.12%	-43.1%
策二	43.63%	-43.27%	0.48%	23.84%	-44.9%
策三	62.58%	-39.9%	-10.19%	29.49%	-40.48%



	半導體產業	航運產業	觀光產業	金融產業	櫃文創產業
股票代碼	2330	2603	2748	2886	2926
策一	75.05%	399.61%	-30.36%	1.95%	-63.28%
策二	75.34%	699.46%	-18.47%	13.41%	-58.75%
策三	93.52%	1101.18%	-14.84%	14.01%	-59.65%

總績效	354.22%	690.77%	1135.72%
	策略一	策略二	策略三

分工內容：自己一個人一組，努力 100%

Thanks!

CREDITS: This presentation template was created by
Slidesgo, including icons by **Flaticon**, and infographics &
images by **Freepik**

Please keep this slide for attribution

