CASH Algo Equity Trading Challenge 2023 Finalize desk

TEAM: Trade Rider

LEUNG TSUN MING
WONG KING HEI
LAU HONG WO, Peter

Content

1. Ideas summartization & Background
2. Implementation
    a. Code
    b. Result
    c. Pros & Cons
3. Live test results & feedbacks
4. Consultion

Ideas summartization

Brief ideas

- MACD and RSI indicators
- Pairs Trading
- Price Pattern recognition
- Regression model
- Monte Carlo simulation
- ADR indicator

Ideas used

- Various indicators
- Price Pattern recognition
- Monte Carlo simulation
- ATR prediction

Finalize

Indicators (MACD & RSI)

Background

Assets: Optimized **30** common stocks' portfolio

Time period: from **mid-2022** to **end-2022**

Interest rate benchmark: **HKXHKD**

Software used:

1. Google Colab (for testing)
2. ALGOGENE platform (for backtesting and live test)

Idea used (finalized)

Indicators (Mixed with RSI and MACD)

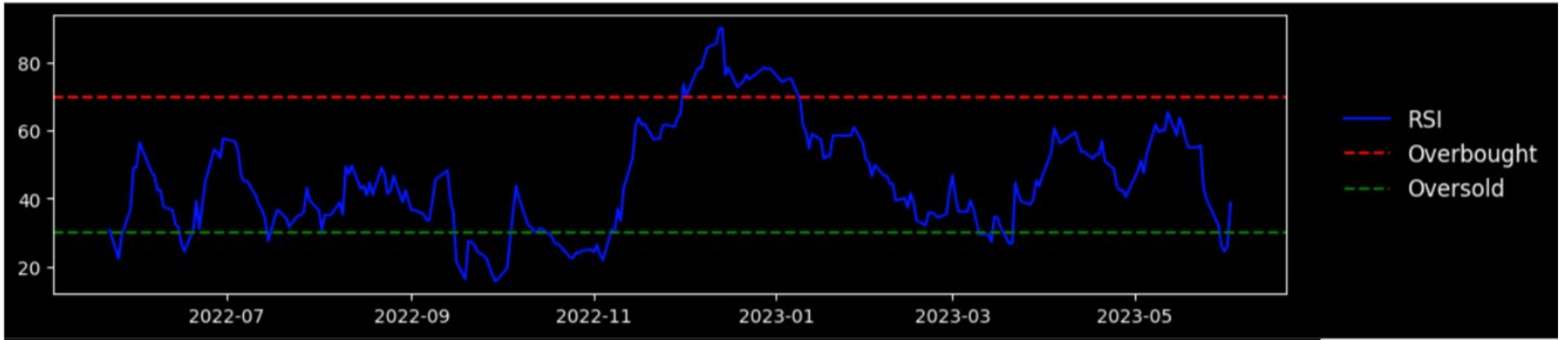
Aims: Different indicators have different conditions to active trading

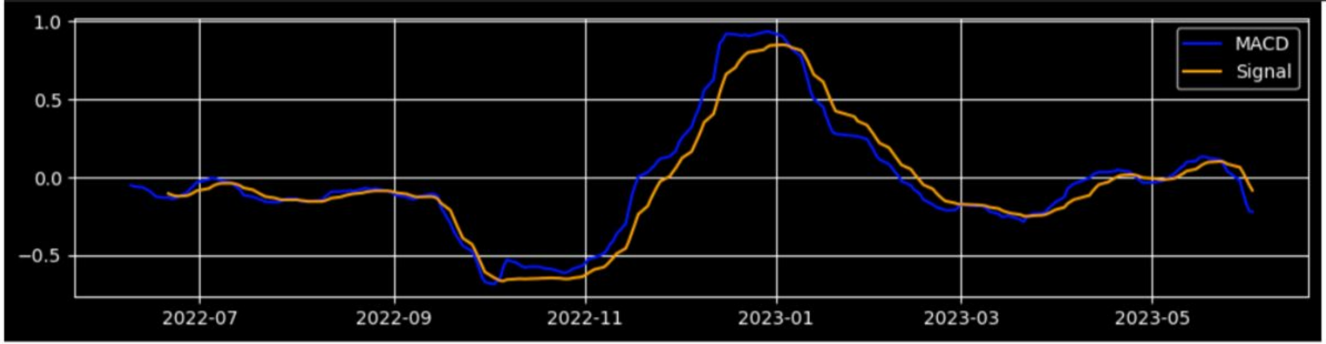
Brief ideas: Comparing different indicators to optimize the result

1. MACD
2. RSI
3. Bollinger Band
4. STOCH
5. TRIX

Coding and results (Brief)

```
# !pip install ta
import numpy as np
import ta
import yfinance as yf
import matplotlib.pyplot as plt


# Calculate RSI and MACD
df['rsi'] = ta.momentum.RSIIndicator(df['Adj Close'], window=14).rsi()
df['macd'] = ta.trend.MACD(df['Adj Close']).macd()
df['signals'] = ta.trend.MACD(df['Adj Close']).macd_signal()

macd = np.array(df['macd'])
signals = np.array(df['signals'])

# RSI plotting
plt.figure(figsize=(12, 6))
plt.subplot(2, 1, 1)
plt.plot(df.index, df['rsi'], label='RSI',color='b')
plt.axhline(y=70, color='red', linestyle='--', label='Overbought')
plt.axhline(y=30, color='green', linestyle='--', label='Oversold')

plt.legend(bbox_to_anchor=(1.04,0.5), loc="center left", borderaxespad=0,frameon=False,fontsize = 12)

# MACD plotting
plt.figure(figsize=(12, 6))
plt.subplot(2, 1, 2)
plt.plot(df.index, df['macd'], label='MACD',color='b')
plt.plot(df.index, ta.trend.MACD(df['Close']).macd_signal(), label='Signal',color='orange')
idx = np.argwhere(np.diff(np.sign(np.array(macd)- np.array(signals)))).flatten()
plt.plot(macd[idx[0]],signals[idx[0]], 'ko')
plt.legend()
plt.grid(True)

# Visulaize
plt.show()
```


80
60
40
20
2022-07
2022-09
2022-11
2023-01
2023-03
2023-05
RSI
Overbought
Oversold


1.0
0.5
0.0
−0.5
2022-07
2022-09
2022-11
2023-01
2023-03
2023-05
MACD
Signal

Setting (General)

Data interval: Starts from 1/1/2014

Volume: **0.1** (higher means risk ↑)

OrderType: Market order

Trading signals:

a. Buy (**Golden cross** appears)
b. Sell (**Death cross** appears)
c. Close (Either RSI > 50 after buy or RSI < 50 after sell)

Setting (indicators)

MACD:

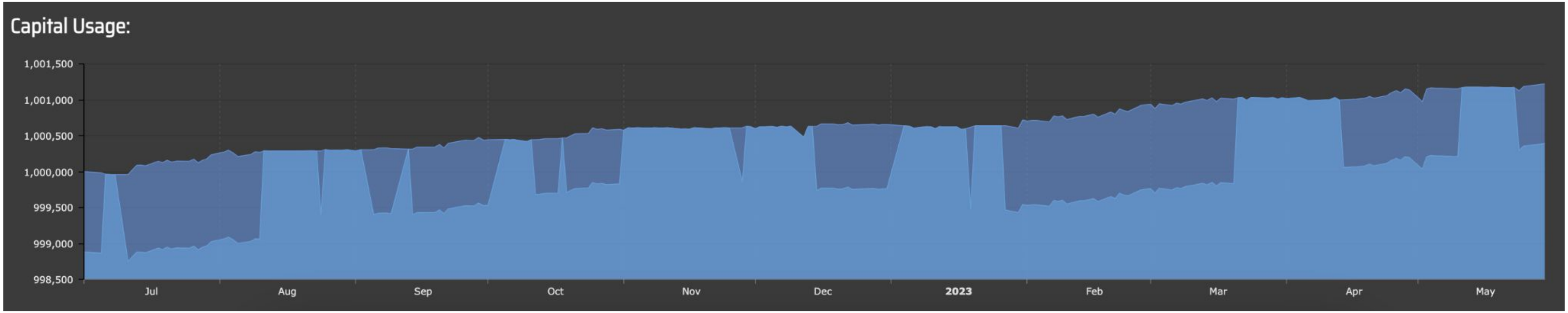
- FestPeriod: 7-day EMA
- SlowPeriod: 35-day EMA
- Signal: 5-day EMA

RSI:

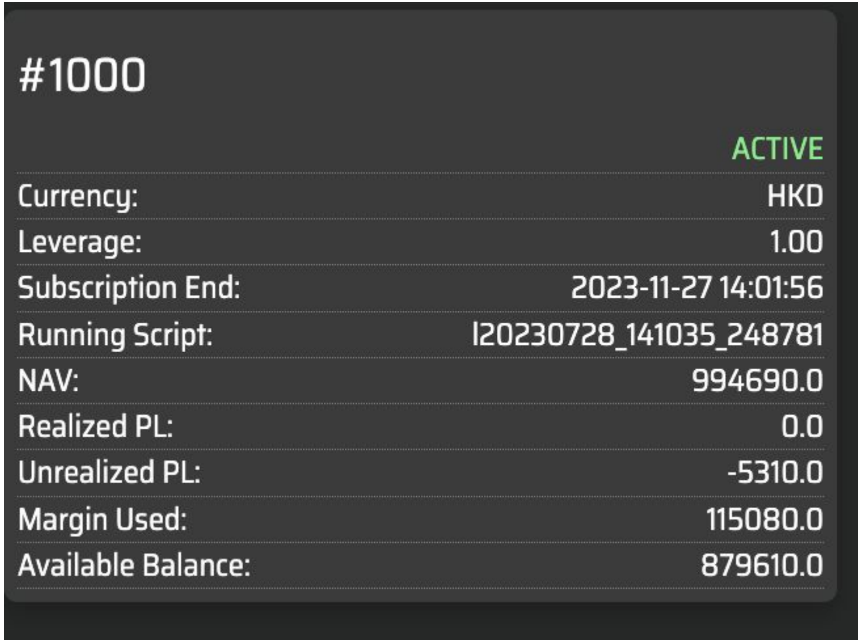
- Period: 14-day
- Overbought signal: 80 (optimized)
- Oversold signal: 20 (optimized)

Coding and Result (backtestting)

```python
def on_marketdatafeed(self, md, ab):

    if md.timestamp >= self.timer+timedelta(hours=24): # Day trade
        self.timer = md.timestamp
        lastprice = md.lastPrice

        # retrieve recent observations
        res = self.evt.getHistoricalBar({"instrument": self.myinstrument},245, 'D')
        res_rsi = self.evt.getHistoricalBar({"instrument":self.myinstrument}, self.rsi_period+1, "D")
        arr = [res_rsi[t]['c'] for t in res_rsi]
        arrClose = numpy.array([res_rsi[t]['c'] for t in res_rsi])
        arrHigh = numpy.array([res_rsi[t]['h'] for t in res_rsi])
        arrLow = numpy.array([res_rsi[t]['l'] for t in res_rsi])
        self.arr_close = numpy.array([res[t]['c'] for t in res])

        # calculate the current RSI value
        RSI_cur = RSI(numpy.array(arr), self.rsi_period)[-1]
        ATR_cur = ATR(arrHigh, arrLow, arrClose, self.ATR_period)[-1]
        # fit SMA line
        self.arr_macd, self.arr_signal, self.arr_hist = talib.MACD(self.arr_close,self.fastperiod, self.slowperiod, self.signal)
        # check number of record is at least greater than both self.fastperiod, self.slowperiod
        if not numpy.isnan(self.arr_macd[-1]) and not numpy.isnan(self.arr_macd[-2]) and not numpy.isnan(self.arr_signal[-1]) and not numpy.isnan(self.arr_signal[-2]) :
            # send a buy order for Golden Cross
            if (self.arr_macd[-1] > self.arr_signal[-1] and self.arr_macd[-2] < self.arr_signal[-2]) or (self.arr_hist[-2] < 0 and self.arr_hist[-1] > 0 ) and RSI_cur>self.rsi_overbought: # DIF>DNA

                self.test_sendOrder(lastprice, 1, 'open')


            # send a sell order for Death Cross
            if (self.arr_macd[-1] < self.arr_signal[-1] and self.arr_macd[-2] > self.arr_signal[-2]) or (self.arr_hist[-2] > 0 and self.arr_hist[-1] < 0) and RSI_cur<self.rsi_oversold: # DIF<DNA

                self.test_sendOrder(lastprice, -1, 'open') #sell signal
```


Capital Usage:
1,001,500
1,001,000
1,000,500
1,000,000
999,500
999,000
998,500
Jul
Aug
Sep
Oct
Nov
Dec
2023
Feb
Mar
Apr
May

Result (live test)



Pros & Cons

Pros:

- These indicators have used commonly in finance
- Easy to implantant
- Mixed indicators complementary to each other

Cons:

- Other events (eg. TRIX-related) may be ignored
- Risk (reversal signal…) appears
- Mostly used in **short-term** only

Conclusion

- Other indicators or strategies can be also considered
- Optimize other parameters (eg. Volume) is necessarily
- Other assets’ portfolios (ETF,bonds) can be used

Q & A

THE END