Python初級數據分析員證書

（六）數據分析 及可視化專案

13. 數據分析專案

Demo4 – TA Systematic Testing

Review

- Statistics
- Hypothesis testing
- Algebra
- Linear regression
- Propositional logic
- Python
- R
- SQL

- Pandas, NumPy, SciPy
- Data Visualization, Matplotlib, Seaborn, Plotly
- Dashboard Visualization, Business Intelligence
- Storytelling

Chapter Summary

- Recap Leading & Lagging Indicator, TA-Lib
- Label Target
- Create signal and features
- Indicators’ accuracy testing

Recap

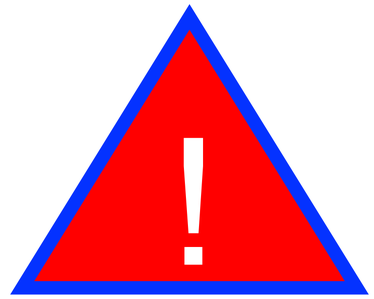
In last chapter, we discovered some Technical Analysis method with TA-Lib with plotly and tried its simplicity coding.

In this chapter, we will focus on TA data analysis testing.

Remember that TA is not just used in finance and stock market analysis, it is statistic tools to analyse time series problems. We often use stock for example because its variety and convenience of data source.

Disclaimer

All content on from our course is for informational and educational purposes of a general nature only, and does not address any circumstances of any particular individual or entity. Do not construe any such information or material as investment, financial, professional or any other advice.



Recap Leading and Lagging indicators

Leading indicators are considered to point toward future events.

- Heads-up for economists and investors who hope to anticipate trends.
- Bond yields are thought to be a good leading indicator of the stock market because bond traders anticipate and speculate about trends in the economy.

Lagging indicators are seen as confirming a pattern that is in progress.

- Be known after the event, but that doesn't make them useless.
- They can clarify and confirm a pattern that is occurring over time.
- The unemployment rate is one of the most reliable lagging indicators.

Popular leading indicators in finance

- The relative strength index (RSI)
- The stochastic oscillator
- Williams %R
- On-balance volume (OBV)

Popular lagging indicators in finance

- Moving averages
- The MACD indicator
- Bollinger bands

Test indicators' accuracy

To test the accuracy of indicators, we cannot just do it visually by graph.

Instead, we need to proof them in statistics, and/or hypothesis testing.

Before we go on perform the test, we need to make some assumptions:

1. Trend will happen in following 1 days after the indicators signal rose
2. Each indicator is independent to historic price
3. There is no other new big impact (such as terror attack, riot, war, etc.) during the 5 days forecasting phrase
4. Pricing history may repeat in normal term
5. Investors are rational

Brainstorm and draft workflow

- Label the T+1 days price movement as target in current T (today) row
- Add indicator parameters as features
- Add indicator signals as features
- Compare the signal to the targeted movement
- Analyse the statistics

In machine learning, label or labelling process is defining the output already known. Features is the input variables. Target is the final output you want to predict. Our final goal is to predict the price movement.

Label the target

This is the most important process of any analysis. Also we need to decide what to be our study matter. For example, we could use our history data to predict tomorrow return, T+1, T+2, T+3….’s return, predict only discrete value, up or down, etc.

In this chapter, we study the T+1’s trend (up or down) only.

Import data

```
1 import pandas as pd
2 import numpy as np
3 import talib
4 from talib import MA_Type
5 import math
6 import yfinance as yf
7 import plotly.express as px
8 import plotly.graph_objects as go
9 from plotly.subplots import make_subplots
```

```
1 df_nvda = yf.download('NVDA', start='2003-01-01',
2                       end='2023-01-01')
3 df_nvda
```

```
[**********************100%**********************]  1 of 1 completed
```

Date	Open	High	Low	Close	Adj Close	Volume
2003-01-02	1.000000	1.037500	0.965833	1.025833	0.941301	130254000
2003-01-03	1.029167	1.062500	1.008333	1.025833	0.941301	103749600

Label Target

```
1 df_nvda['Log_rtn'] = df_nvda['Close'].pct_change(1)
2 df_nvda['Target'] = df_nvda['Log_rtn'].shift(-1)
```

```
1 df_nvda
```

Date	Open	High	Low	Close	Adj Close	Volume	Log_rtn	Target
2003-01-02	1.000000	1.037500	0.965833	1.025833	0.941301	130254000	NaN	0.000000
2003-01-03	1.029167	1.062500	1.008333	1.025833	0.941301	103749600	0.000000	0.081235
2003-01-06	1.050833	1.125833	1.050000	1.109167	1.017768	103342800	0.081235	-0.003006
2003-01-07	1.130833	1.156667	1.093333	1.105833	1.014709	149827200	-0.003006	-0.099473

Add indicators

We add indicators to our DF and drop the NaN.

```
df_nvda['rsi'] = talib.RSI(df_nvda['Close'], timeperiod=14)
df_nvda['k'], df_nvda['d'] = talib.STOCH(df_nvda['High'], df_nvda['Low'], df_nvda['Close'],14)
df_nvda['macd'], df_nvda['macdsignal'], df_nvda['macdhist'] = talib.MACD(df_nvda['Close'],
                                                    fastperiod=12, slowperiod=26, signalperiod=9)
df_nvda['adosc'] = talib.ADOSC(df_nvda['High'], df_nvda['Low'],df_nvda['Close'],
                     df_nvda['Volume'],fastperiod=3, slowperiod=10)
df_nvda['atr'] = talib.ATR(df_nvda['High'], df_nvda['Low'],df_nvda['Close'], timeperiod=14)
df_nvda['tr'] = talib.TRANGE(df_nvda['High'], df_nvda['Low'],df_nvda['Close'])
```

```
df_nvda = df_nvda.dropna()
```

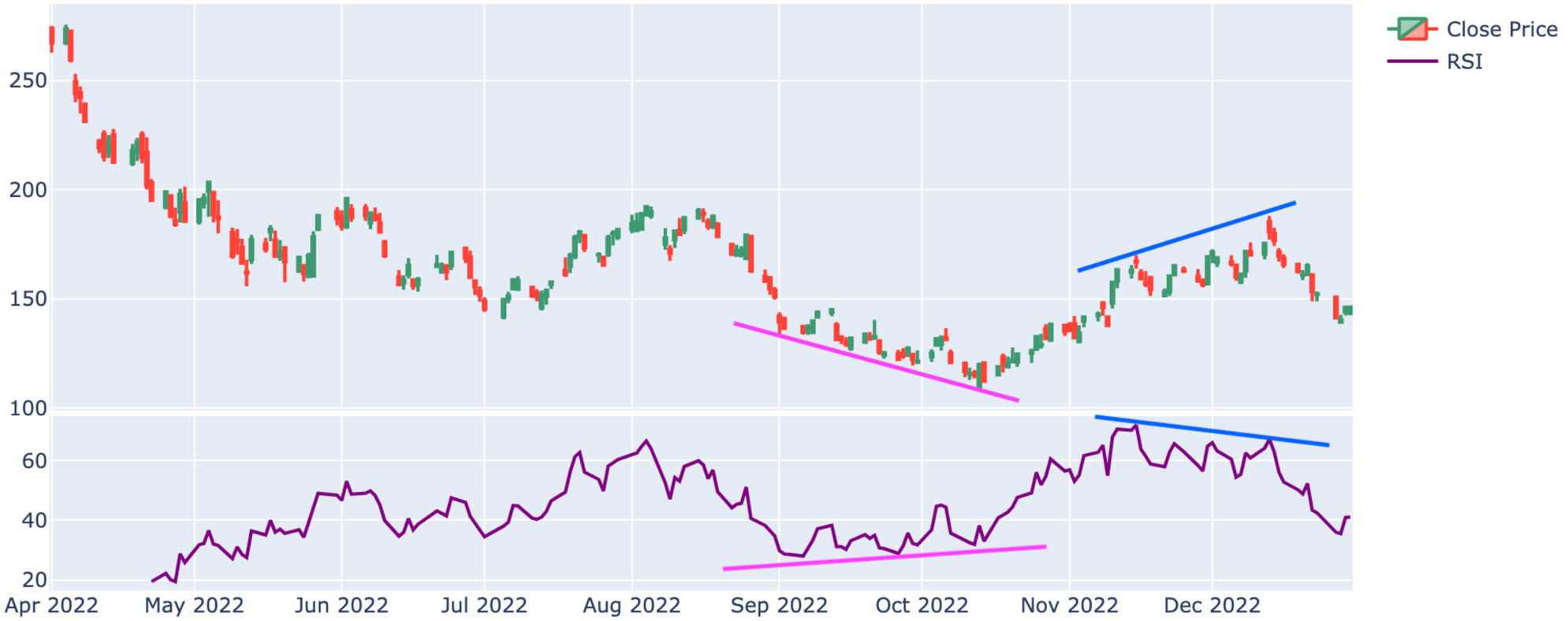
Define how the signal raise

There a few ways to to define the signal. Here we use the straightforward one. For RSI below 30, we define it as oversold and buy signal, over 70 is sell signal. 1 denoted as buy, -1 denoted as sell, 0 denoted as hold.

```
# RSI 30 oversold, 70 overbought
df_nvda.loc[df_nvda.index, ['rsi_signal']] = np.where(df_nvda['rsi']<30, 1,
                                                      np.where(df_nvda['rsi']>70, -1, 0)).tolist()
```

Define how the signal raise

In some scenario, the divergence of price trend and indicator, is a better prediction. But it take more complicated coding and computing.



Define how the signal raise

For Stochastic, if both k and d below 20, and k is larger than d, it is buy signal.

```
# STOCHASTIC k & d below 20, and k > d, buy for oversold
df_nvda.loc[df_nvda.index, ['stoch_signal']] = np.where(
    (df_nvda['k']<20)&(df_nvda['d']<20)&(df_nvda['k']>df_nvda['d']), 1,
        np.where((df_nvda['k']>80)&(df_nvda['d']>80)&(df_nvda['k']<df_nvda['d']),-1,0)).tolist()
```

Define how the signal raise

For MACD, we denote buy as MACD histogram is larger than 0.

Sell as MACD histogram is smaller than 0.

```
# MACD-histogram > 0, buy signal
df_nvda.loc[df_nvda.index, ['macd_signal']] = np.where(df_nvda['macdhist']>0, 1,
                                                       np.where(df_nvda['macdhist']<0, -1, 0)).tolist()
```

Define how the signal raise

ADOSC is **Accumulation/Distribution Oscillator.**

For ADOSC, the threshold is 0. If its previous value is under 0, and now over 0, it is buy signal. Vice versa for sell.

```
# ADOSC: 0 is threshold line, from <0 to >0, buy signal
df_nvda.loc[df_nvda.index, ['adosc_signal']] = np.where(
            (df_nvda['adosc'].shift(1)<0) & (df_nvda['adosc']>0), 1,
                np.where((df_nvda['adosc'].shift(1)>0) & (df_nvda['adosc']<0), -1, 0)).tolist()
```

Define how the signal raise

ATR is Average True Range of 14-days period. TR is today's True Range.

For ATR, if TR is larger than ATR, it is a buy signal.

```
# if current close higher than average range, buy signal
df_nvda.loc[df_nvda.index, ['atr_signal']] = np.where(df_nvda['tr']>df_nvda['atr'], 1,
                                                      np.where(df_nvda['tr']<df_nvda['atr'], -1, 0)).tolist()
```

Create features

We gather the signals as features.

```
1  feats = df_nvda[['rsi_signal','stoch_signal','macd_signal',
2                   'adosc_signal','atr_signal','Log_rtn','Target']].copy()
```

```
1  feats.sample(3)
```

Date	rsi_signal	stoch_signal	macd_signal	adosc_signal	atr_signal	Log_rtn	Target
2004-11-12	-1.0	0.0	1.0	0.0	-1.0	0.001660	0.014917
2006-12-21	0.0	-1.0	1.0	0.0	-1.0	-0.008625	-0.023728
2013-03-27	0.0	0.0	-1.0	1.0	1.0	0.012000	0.014229

Check the indicators results

For RSI indicator performance, we may check how its buy/sell signal compare to actual Target.

```
# correctly predicted buy
feats['Target'].loc[(feats['rsi_signal']==1)&(feats['Target']>0)].count()
```

```
78
```

```
# mean of correct predict buy log_return
feats['Target'].loc[(feats['rsi_signal']==1)&(feats['Target']>0)].mean()
```

```
0.03375603065712058
```

```
# incorrectly predicted buy
feats['Target'].loc[(feats['rsi_signal']==1)&(feats['Target']<0)].count()
```

```
77
```

```
# mean of incorrect predict buy log_return
feats['Target'].loc[(feats['rsi_signal']==1)&(feats['Target']<0)].mean()
```

```
-0.029394032496178
```

Define a function to return the result

```python
def indicator_result(indicator, feats):
    buy_correct_time = feats['Target'].loc[(feats[indicator]==1)&(feats['Target']>0)].count()
    buy_correct_mean = feats['Target'].loc[(feats[indicator]==1)&(feats['Target']>0)].mean()
    buy_incorrect_time = feats['Target'].loc[(feats[indicator]==1)&(feats['Target']<0)].count()
    buy_incorrect_mean = feats['Target'].loc[(feats[indicator]==1)&(feats['Target']<0)].mean()
    buy_correct_ratio = buy_correct_time / (buy_correct_time + buy_incorrect_time)

    sell_correct_time = feats['Target'].loc[(feats[indicator]==-1)&(feats['Target']<0)].count()
    sell_correct_mean = feats['Target'].loc[(feats[indicator]==-1)&(feats['Target']<0)].mean()
    sell_incorrect_time = feats['Target'].loc[(feats[indicator]==-1)&(feats['Target']>0)].count()
    sell_incorrect_mean = feats['Target'].loc[(feats[indicator]==-1)&(feats['Target']>0)].mean()
    sell_correct_ratio = sell_correct_time / (sell_correct_time + sell_incorrect_time)
    return [ indicator, f"buy_correct_ratio: {buy_correct_ratio:.4f}",
                        f"buy_correct_mean: {buy_correct_mean*100:.4f}%",
                        f"sell_correct_ratio: {sell_correct_ratio:.4f}",
                        f"sell_correct_mean: {sell_correct_mean*100:.4f}%"]
```

Indicator's result

RSI

```
1 indicator_result('rsi_signal', feats)
```

```
['rsi_signal',
 'buy_correct_ratio: 0.5032',
 'buy_correct_mean: 3.3756%',
 'sell_correct_ratio: 0.4790',
 'sell_correct_mean: -1.5680%']
```

Indicator's result

STOCHASTIC

```
1  indicator_result('stoch_signal', feats)
```

```
['stoch_signal',
 'buy_correct_ratio: 0.5286',
 'buy_correct_mean: 2.8469%',
 'sell_correct_ratio: 0.4675',
 'sell_correct_mean: -1.7943%']
```

Indicator's result

MACD

```
1 indicator_result('macd_signal', feats)
```

```
['macd_signal',
 'buy_correct_ratio: 0.5177',
 'buy_correct_mean: 2.2111%',
 'sell_correct_ratio: 0.4691',
 'sell_correct_mean: -2.2187%']
```

Indicator's result

ADOSC

```
1 indicator_result('adosc_signal', feats)
```

```
['adosc_signal',
 'buy_correct_ratio: 0.5451',
 'buy_correct_mean: 1.8422%',
 'sell_correct_ratio: 0.4721',
 'sell_correct_mean: -2.4982%']
```

Indicator's result

ATR/TR

```
1 indicator_result('atr_signal', feats)
```

```
['atr_signal',
 'buy_correct_ratio: 0.5165',
 'buy_correct_mean: 2.2973%',
 'sell_correct_ratio: 0.4709',
 'sell_correct_mean: -2.0649%']
```

Discovery the statistics

It looks like the overall buy signal is slightly overperform which over 50% a bit. However the sell signal is slightly underperform.

Can you suggest a way to make a better signal?

```
1 indicator_result('adosc_signal', feats)
```

```
['adosc_signal',
 'buy_correct_ratio: 0.5451',
 'buy_correct_mean: 1.8422%',
 'sell_correct_ratio: 0.4721',
 'sell_correct_mean: -2.4982%']
```

Comprehensive Analyse

We perform buy if triggered by only 1 indicator, we may have 713 correct, and 656 wrong.

```
feats['Comp_signal'] = np.where((np.where(feats['rsi_signal']==1,1,0) +
            np.where(feats['stoch_signal']==1,1,0)+
            np.where(feats['macd_signal']==1,1,0)+
            np.where(feats['adosc_signal']==1,1,0)+
            np.where(feats['atr_signal']==1,1,0)
        )>1 , 1, 0 )
```

```
feats['Target'].loc[(feats['Comp_signal']==1) & (feats['Target']>0)].count()
```

```
713
```

```
feats['Target'].loc[(feats['Comp_signal']==1) & (feats['Target']<0)].count()
```

```
656
```

Comprehensive Analyse

We perform buy if triggered by at least 2 indicator, we may have 51 correct, and 47 wrong.

```
1  feats['Comp_signal'] = np.where((np.where(feats['rsi_signal']==1,1,0) +
2              np.where(feats['stoch_signal']==1,1,0)+
3              np.where(feats['macd_signal']==1,1,0)+
4              np.where(feats['adosc_signal']==1,1,0)+
5              np.where(feats['atr_signal']==1,1,0)
6          )>2 , 1, 0 )
```

```
1  feats['Target'].loc[(feats['Comp_signal']==1) & (feats['Target']>0)].count()
```

```
51
```

```
1  feats['Target'].loc[(feats['Comp_signal']==1) & (feats['Target']<0)].count()
```

```
47
```

Considerations

Consideration(buy side):

- For some indicator is leading, we shall delay the Target day to T+3 or more?
- For some indicator is lagging, we shall keep T+1? Or set it as confirmation of leading indicator only?
- Different asset may vary
- Is there any other indicator? ie. Earning, dividend, ESG, interest rate

Considerations

Consideration(sell side):

- Some asset may have long term steady growth
- Sell side may lost its accuracy due to pessimistic investor
- Sell signal may be favour in junk stock
- We didn’t analyse specific pattern, such as MACD/RSI divergence

Hard facts

- There is no single indicator work well for all stocks.
- Retail investors may have less inside information
- Indicators may have different parameter setting. ie. MACD26,12, 9, if we change signal line 9 to 5, this may trigger faster action but bring volatility to portfolio as well
- Compare overall return with long term investing
- Compare overall return with benchmark return
- Check for availability of other asset with same return but less risk

Chapter Wrapping

There are more than 130+ TA functions in TA-Lib.

- Try to use indicator in different categories.
- Try to use different parameter setting.

https://ta-lib.github.io/ta-lib-python/doc_index.html

- All Functions
  - Overlap Studies
  - Momentum Indicators
  - Volume Indicators
  - Volatility Indicators
  - Price Transform
  - Cycle Indicators
  - Pattern Recognition
  - Statistic Functions
  - Math Transform
  - Math Operators

Chapter Wrapping

- In this chapter we discovered 5 TA indicators (leading and lagging) in a stock with 20 years data.
- TA is one of the major analysis apart from fundamental analysis.
- We could adjust parameter and target to quantify the test.
- Signal from each TA rarely trigger in same time.
- If one TA overall accuracy is over 60%, is outstanding

Academic Resources

TA Lib - https://github.com/TA-Lib/ta-lib-python

TA Lib Documentation - https://ta-lib.github.io/ta-lib-python/doc_index.html

Pandas TA - https://github.com/twopirllc/pandas-ta

QuantConnect - https://www.quantconnect.com/

Google Colab - https://colab.research.google.com/

Google Scholar - https://scholar.google.com/

Research Gate - https://www.researchgate.net/

JSTOR - https://www.jstor.org/

Research SPJ - https://spj.science.org/