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
Demo3 – TA library

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

- Leading and Lagging Indicators
- TA-Lib introduction and installation
- STOCH RSI
- Bollinger Band
- MACD
- ADOSC
- ATR & TR

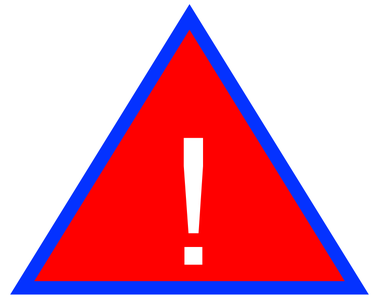
Recap

In last chapter, we learnt simple backtest for multiple Moving Average combination. Google Colab is a great choice for long running computing on cloud.

In this chapter, we will discover more Technical Analysis method with library. TA is not just used in finance and stock market analysis, it is statistic tools to analyse time series problems. We often use stock data because its variety and convenience of 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.



Leading Indicator and Lagging Indicator

Leading and lagging indicators are two types of measurements used when assessing performance in a business or economy.

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.

Leading Indicator and Lagging Indicator

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

Relying solely on either could have negative effects on a strategy, which is why many traders will aim to find a balance of the two.

TA library

There are two commonly used TA library that trader would use:

- Pandas-ta
  - Provide more than 130+ indicator. Easier to use.
  - Pandas TA - https://github.com/twopirllc/pandas-ta
- ta-lib
  - Fast library and available in other language like C, Java, Perl
  - Since its backend is written in C, the speed is faster
  - TA Lib - https://github.com/TA-Lib/ta-lib-python

Ta-lib

TA-Lib is widely used by trading software developers requiring to perform technical analysis of financial market data.

- Includes 150+ indicators such as ADX, MACD, RSI, Stochastic, Bollinger Bands, etc.
- Candlestick pattern recognition
- Open-source API for C/C++, Java, Perl, Python and 100% Managed .NET
- Although it might be a little tricky while installation, it definitely a great tool afterwards.

Ta-lib

In this chapter, we focus using ta-lib. First let's install the ta library.

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

Add two lines of scripts in .zshrc

Then source .zshrc

```
156  # TA-Lib
157  export TA_INCLUDE_PATH="$(brew --prefix ta-lib)/include"
158  export TA_LIBRARY_PATH="$(brew --prefix ta-lib)/lib"
159
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL COMMENTS

```
source .zshrc
```

Finally, in ipython, pip install ta-lib.

```
1  pip install ta-lib
```

```
Collecting ta-lib
  Using cached TA-Lib-0.4.26.tar.gz (272 kB)
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Installing backend dependencies ... done
  Preparing metadata (pyproject.toml) ... done
```

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

```
1 df_nvda = yf.download('NVDA', start='2022-04-01',
2                       end='2023-01-01')
3 df_nvda
```

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

Date	Open	High	Low	Close	Adj Close	Volume
2022-04-01	273.750000	274.959991	262.670013	267.119995	266.870575	51723500
2022-04-04	267.279999	275.579987	266.130005	273.600006	273.344543	39712000

STOCHASTIC RSI

Stochastic RSI (StochRSI) is an indicator used in technical analysis that ranges between zero and one (or zero and 100 on some charting platforms) and is created by applying the Stochastic oscillator formula to a set of relative strength index (RSI) values rather than to standard price data.

StochRSI reading above 0.8 is considered overbought, while a reading below 0.2 is considered oversold. On the zero to 100 scale, above 80 is overbought, and below 20 is oversold.

Lowest RSI = Lowest RSI reading over last 14 periods (or chosen lookback period);

Highest RSI = Highest RSI reading over last 14 period (or lookback period).

STOCHASTIC RSI

The Formulas For the Stochastic RSI (StochRSI) are:

$$\text{StochRSI} = \frac{RSI - \min[RSI]}{\max[RSI] - \min[RSI]}$$

where:

$RSI = \text{Current RSI reading}$

$\min[RSI] = \text{Lowest RSI reading over the last 14 periods}$
$(\text{or your chosen lookback interval})$

$\max[RSI] = \text{Highest RSI reading over the last 14 periods}$
$(\text{or your chosen lookback interval})$

STOCHASTIC & RSI

Using talib.STOCHRSI to prepare k and d line. Function setting as follow:

k, d = talib.STOCH(df_nvda['High'], df_nvda['Low'], df_nvda['Close'],

fastk_period, slowk_period, slowk_matype, slowd_period, slowd_matype)

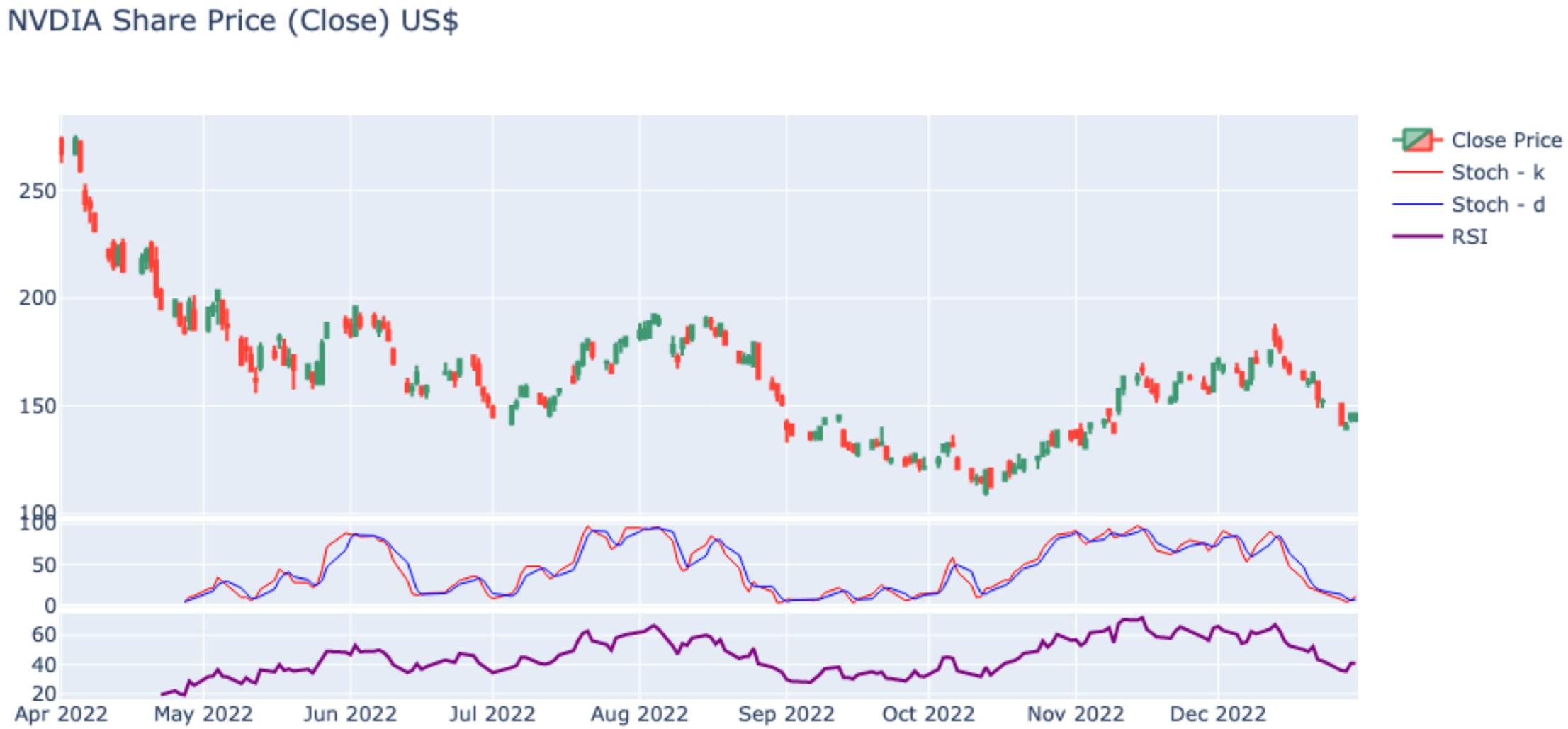
K line is the fast line.

```
rsi = talib.RSI(df_nvda['Close'], timeperiod=14)
k, d = talib.STOCH(df_nvda['High'], df_nvda['Low'], df_nvda['Close'],14)
```

STOCHASTIC & RSI

```
fig = make_subplots(rows=3, cols=1, shared_xaxes=True,
                    vertical_spacing=0.01, row_heights=[0.7, 0.15, 0.15])
fig.add_trace(go.Candlestick(x=df_nvda.index,
              open=df_nvda['Open'], high=df_nvda['High'],
              low=df_nvda['Low'], close=df_nvda['Close'],
              showlegend=True, name='Close Price'),row=1,col=1 )
fig.add_trace(go.Scatter(x=df_nvda.index,
                         y=k, name='Stoch - k',
                         line=dict(color='red', width=1)
                        ), row=2, col=1)
fig.add_trace(go.Scatter(x=df_nvda.index,
                         y=d, name='Stoch - d',
                         line=dict(color='blue', width=1)
                        ), row=2, col=1)
fig.add_trace(go.Scatter(x=df_nvda.index, y=rsi, name='RSI',
                    marker=dict(color ='purple' )), row=3, col=1)
fig.update_layout(title="NVDIA Share Price (Close) US$",
                  xaxis_rangeslider_visible=False)
fig.show()
```

STOCHASTIC & RSI



Bollinger Bands

Bollinger Bands are envelopes plotted at a standard deviation level above and below a simple moving average of the price. Because the distance of the bands is based on standard deviation, they adjust to volatility swings in the underlying price. Bollinger Bands use 2 parameters, Period and Standard Deviations.

Typical values used:

Short term: 10 day moving average, bands at 1.5 standard deviations. (1.5 times the standard dev. +/- the SMA)

Medium term: 20 day moving average, bands at 2 standard deviations.

Long term: 50 day moving average, bands at 2.5 standard deviations.

Bollinger Bands

```
1  from talib import MA_Type
2
3  upper, middle, lower = talib.BBANDS(df_nvda['Close'], matype=MA_Type.T3)
```

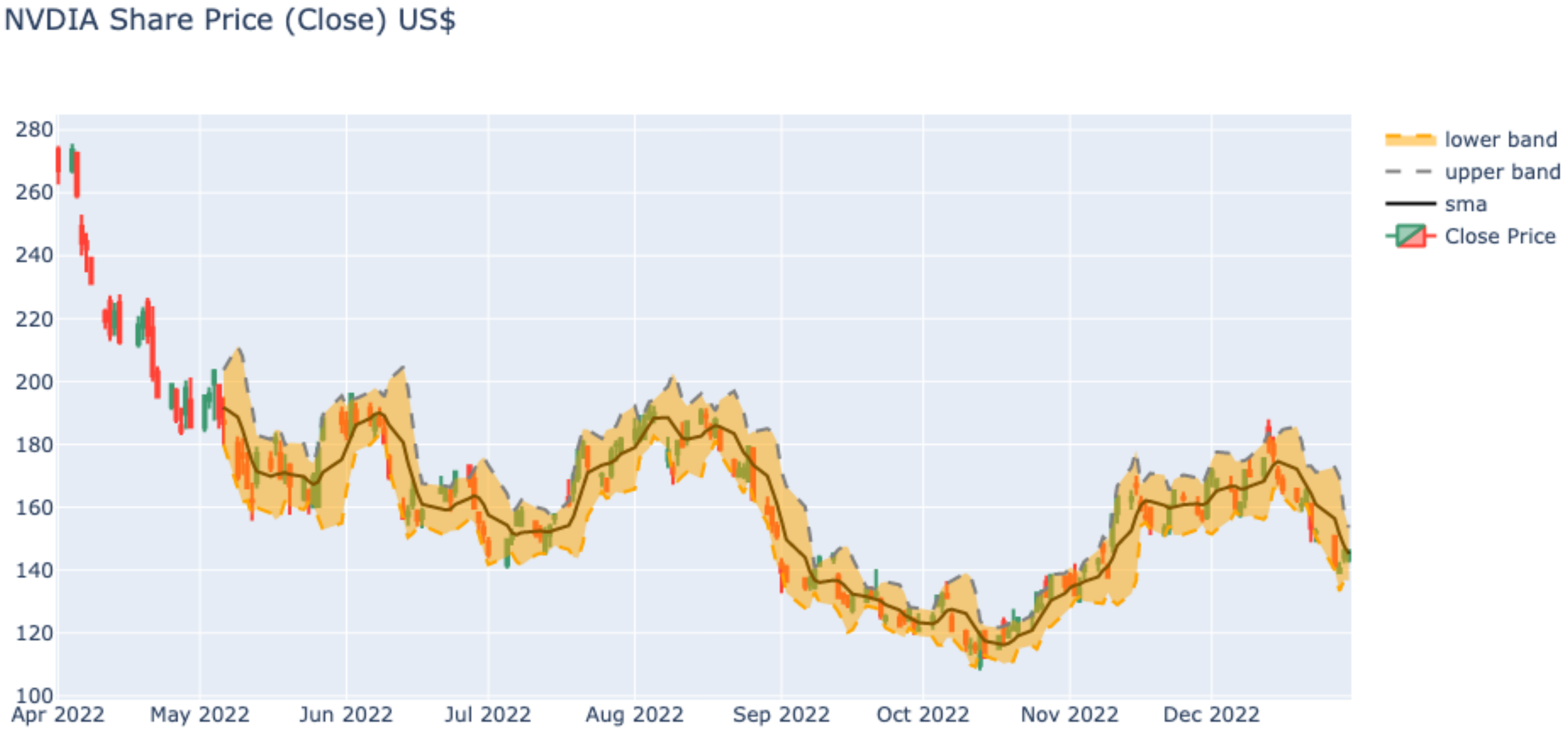
Note (for your ref):

MA_Type.T3 = talib.MA(closed,timeperiod=10, matype=3)

Bollinger Bands

```
fig=go.Figure(data=[go.Candlestick(x=df_nvda.index,
            open=df_nvda['Open'], high=df_nvda['High'],
            low=df_nvda['Low'], close=df_nvda['Close'],
            showlegend=True, name='Close Price')] )

# Moving Average
fig.add_trace(go.Scatter(x = df_nvda.index, y = middle,
            line_color = 'black', name = 'sma'))

# Upper Bound
fig.add_trace(go.Scatter(x = df_nvda.index, y = upper,
            line_color = 'gray', line = {'dash': 'dash'},
            name = 'upper band', opacity = 0.5))

# Lower Bound fill in between with parameter 'fill': 'tonexty'
fig.add_trace(go.Scatter(x = df_nvda.index, y = lower,
            line_color = 'orange', line = {'dash': 'dash'},
            fill = 'tonexty', name = 'lower band', opacity = 0.4))

fig.update_layout(title="NVDIA Share Price (Close) US$",
                  xaxis_rangeslider_visible=False)
```

Bollinger Bands



MACD

Moving average convergence divergence (MACD) is one of the most popular technical indicators in trading. The MACD is appreciated by traders worldwide for its simplicity and flexibility, as it can be used as a trend or momentum indicator and signal opportunities to enter and exit positions.

MACD

There are three different elements involved with the histogram, which is mapped out around a baseline:

- The MACD line (subtracting a long-term exponential moving average (EMA) from a shorter-term EMA)
- The signal line (subtracting the two EMAs and creating a 9-day moving average)
- The histogram (subtracting the MACD line from the signal line)

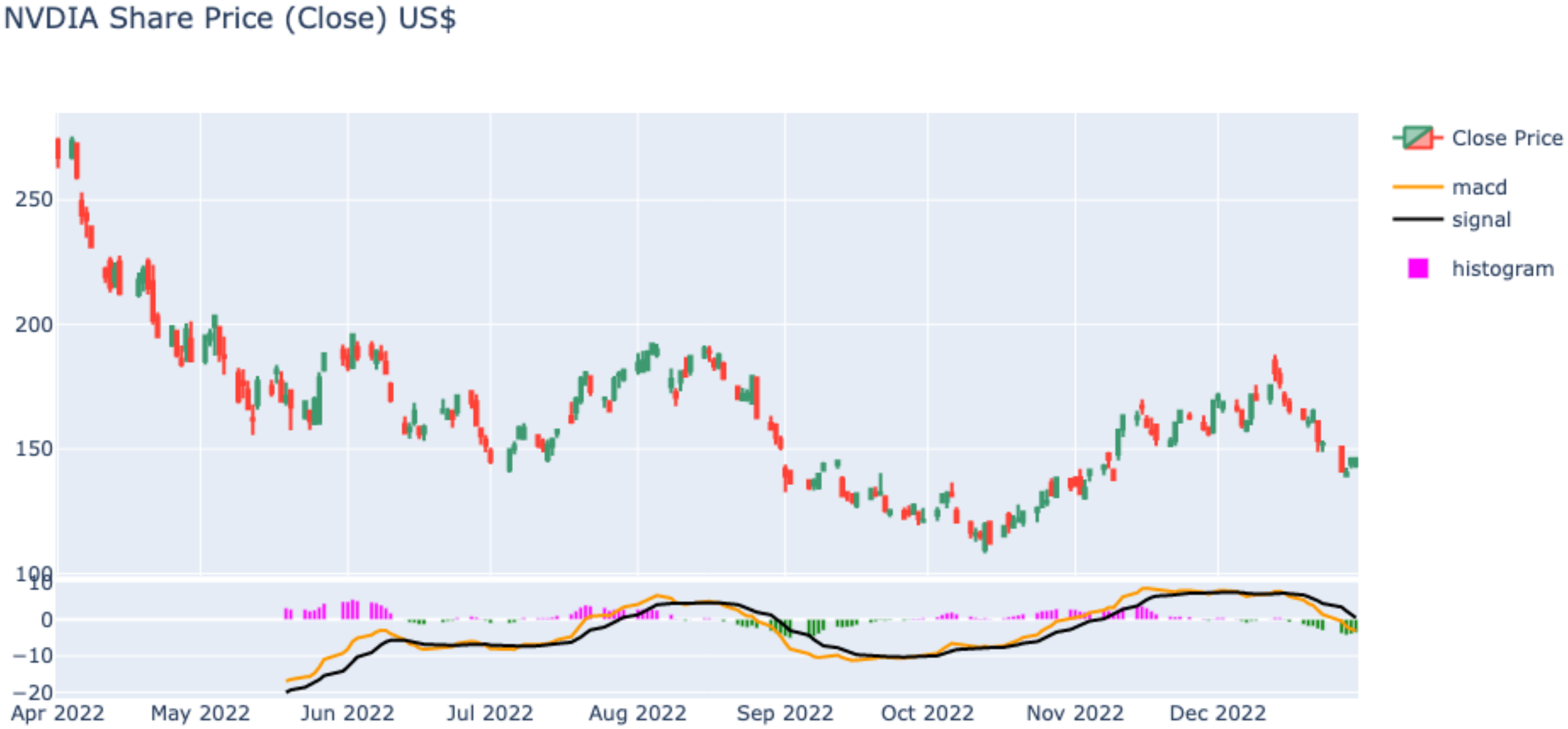
TA lib setting

```
Macd, macdsignal, macdhist = talib.MACD(close, fastperiod=12, slowperiod=26, signalperiod=9)
```

MACD

```
# Colorize the histogram values
colors = np.where(macdhist < 0, '#008000', '#FF00FF')

fig = make_subplots(rows=2, cols=1, shared_xaxes=True,
                    vertical_spacing=0.01, row_heights=[0.8, 0.2])
fig.add_trace(go.Candlestick(x=df_nvda.index,
            open=df_nvda['Open'], high=df_nvda['High'],
            low=df_nvda['Low'], close=df_nvda['Close'],
            showlegend=True, name='Close Price'),row=1,col=1 )
# MACD
fig.append_trace(
    go.Scatter(x=df_nvda.index, y=macd,
        line=dict(color='#ff9900', width=2), name='macd',
        legendgroup='2',), row=2, col=1)

# MACD Signal
fig.append_trace(
    go.Scatter(x=df_nvda.index, y=macdsignal,
        line=dict(color='#000000', width=2),
        legendgroup='2', name='signal'), row=2, col=1)

# histogram
fig.append_trace(
    go.Bar(x=df_nvda.index, y=macdhist, name='histogram',
        marker_color=colors,), row=2, col=1)

fig.update_layout(title="NVDIA Share Price (Close) US$",
                  xaxis_rangeslider_visible=False)
fig.show()
```

MACD


NVDIA Share Price (Close) US$
Close Price
macd
signal
histogram
250
200
150
100
10
0
−10
−20
Apr 2022
May 2022
Jun 2022
Jul 2022
Aug 2022
Sep 2022
Oct 2022
Nov 2022
Dec 2022

ADOSC

Accumulation/Distribution Oscillator (ADOSC) is a volume indicator. The ADOSC is a cumulative indicator that uses volume and price to assess whether a stock is being accumulated or distributed.

The Accumulation/Distribution Indicator (A/D) Formula

$$\text{MFM} = \frac{(\text{Close} - \text{Low}) - (\text{High} - \text{Close})}{\text{High} - \text{Low}}$$

where:

$\text{MFM} = \text{Money Flow Multiplier}$

$\text{Close} = \text{Closing price}$

$\text{Low} = \text{Low price for the period}$

$\text{High} = \text{High price for the period}$

$$\text{Money Flow Volume} = \text{MFM} \times \text{Period Volume}$$

$$\text{A/D} = \text{Previous A/D} + \text{CMFV}$$

where:

$\text{CMFV} = \text{Current period money flow volume}$

ADOSC

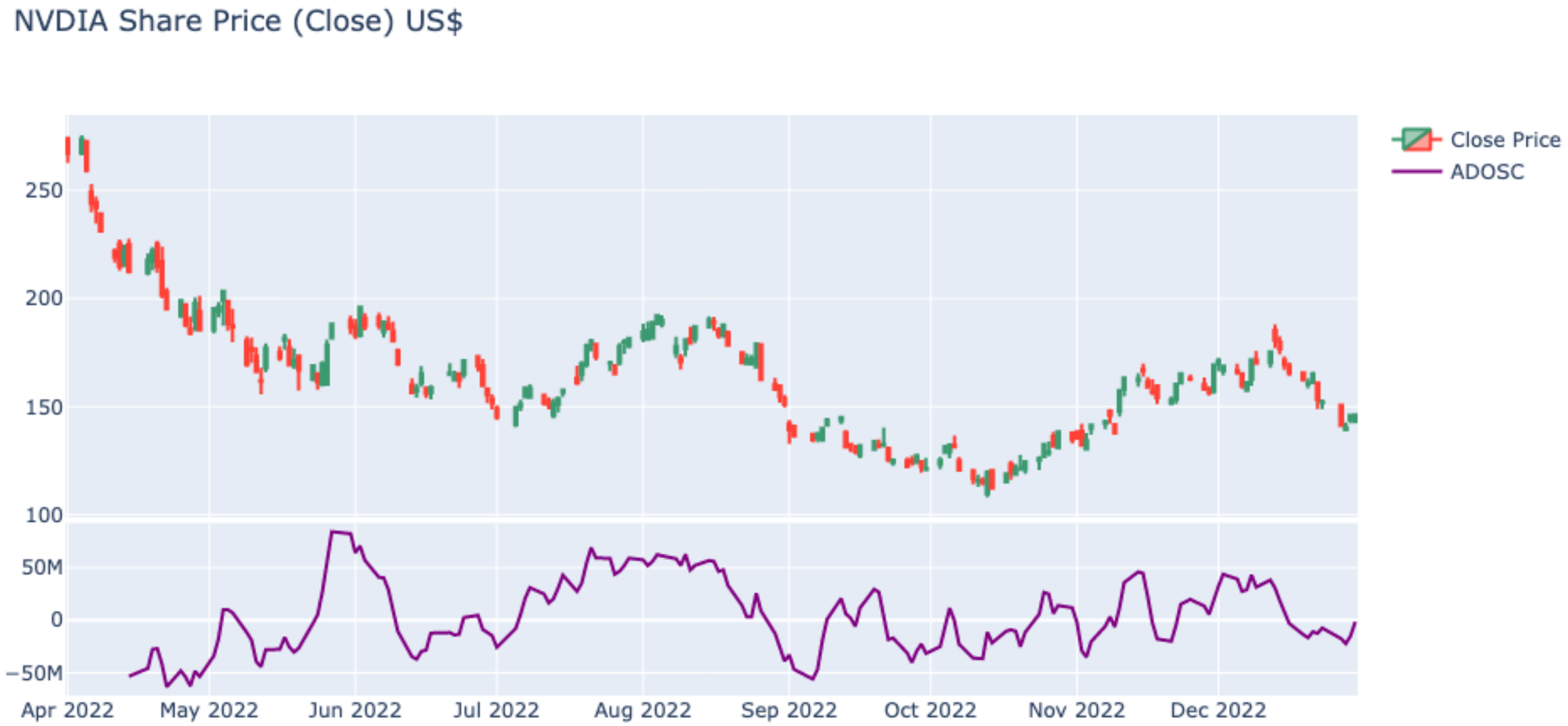
TA Lib setting

real = talib.ADOSC(high, low, close, volume, fastperiod=3, slowperiod=10)

```
adosc = talib.ADOSC(df_nvda['High'], df_nvda['Low'],df_nvda['Close'],
                    df_nvda['Volume'],fastperiod=3, slowperiod=10)
```

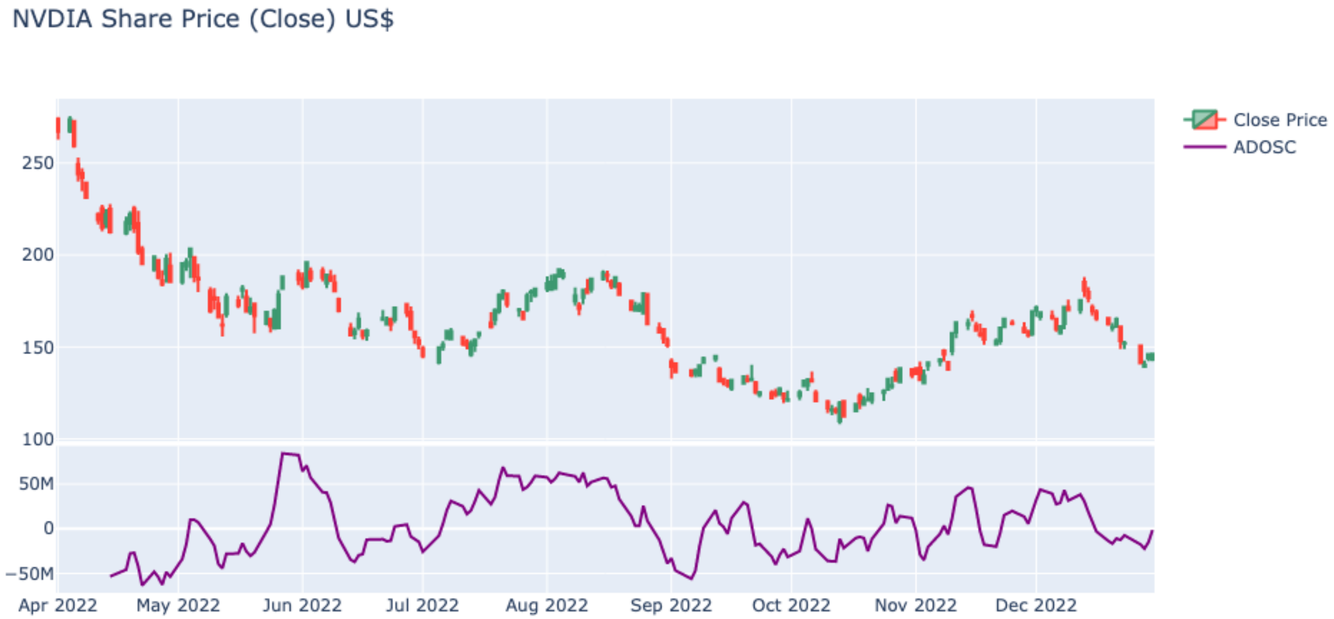
```
fig = make_subplots(rows=2, cols=1, shared_xaxes=True,
                    vertical_spacing=0.01, row_heights=[0.7, 0.3])
fig.add_trace(go.Candlestick(x=df_nvda.index,
              open=df_nvda['Open'], high=df_nvda['High'],
              low=df_nvda['Low'], close=df_nvda['Close'],
              showlegend=True, name='Close Price'),row=1,col=1 )
fig.add_trace(go.Scatter(x=df_nvda.index, y=adosc, name='ADOSC',
                    marker=dict(color ='purple' )), row=2, col=1)
fig.update_layout(title="NVDIA Share Price (Close) US$",
                  xaxis_rangeslider_visible=False)
fig.show()
```

ADOSC


NVDIA Share Price (Close) US$
Close Price
ADOSC
250
200
150
100
50M
0
−50M
Apr 2022
May 2022
Jun 2022
Jul 2022
Aug 2022
Sep 2022
Oct 2022
Nov 2022
Dec 2022

ADOSC

- The accumulation/distribution (ADOSC) indicates supply and demand of an asset or security by looking at where the price closed within the period's range and then multiplying that by volume.
- The ADOSC indicator is cumulative, meaning one period's value is added or subtracted from the last.
- In general, a rising ADOSC helps confirm a rising price trend, while a falling ADOSC helps confirm a price downtrend.



ADOSC – Further Consideration

You may check its holder of of the stock on morningstars or yfinance. And calculate the approximately market shares and the non-institute share portion. That makes hint for you to estimate the minimum and maximum ADOSC.

Major Holders

Breakdown

4.18%	% of Shares Held by All Insider
68.08%	% of Shares Held by Institutions
71.06%	% of Float Held by Institutions
4,015	Number of Institutions Holding Shares

https://finance.yahoo.com/quote/NVDA/holders?p=NVDA

ATR

The average true range (ATR) is a technical analysis indicator introduced by market technician J. Welles Wilder Jr. in his book *New Concepts in Technical Trading Systems* that measures market volatility by decomposing the entire range of an asset price for that period.

ATR & TR

- ATR is typically derived from the 14-day simple moving average of a series of true range indicators.
- The ATR was initially developed for use in commodities markets but has since been applied to all types of securities.
- ATR shows investors the average range prices swing for an investment over a specified period.
- True Range is the daily based swing range.

ATR

TA lib setting

```
ATR = talib.ATR(high, low, close, timeperiod=14)
TR = talib.TRANGE(high, low, close)
```

The Average True Range (ATR) Formula

The formula to calculate ATR for an investment with a previous ATR calculation is :

$$\frac{\text{Previous ATR}(n-1) + \text{TR}}{n}$$

where:

$n = \text{Number of periods}$

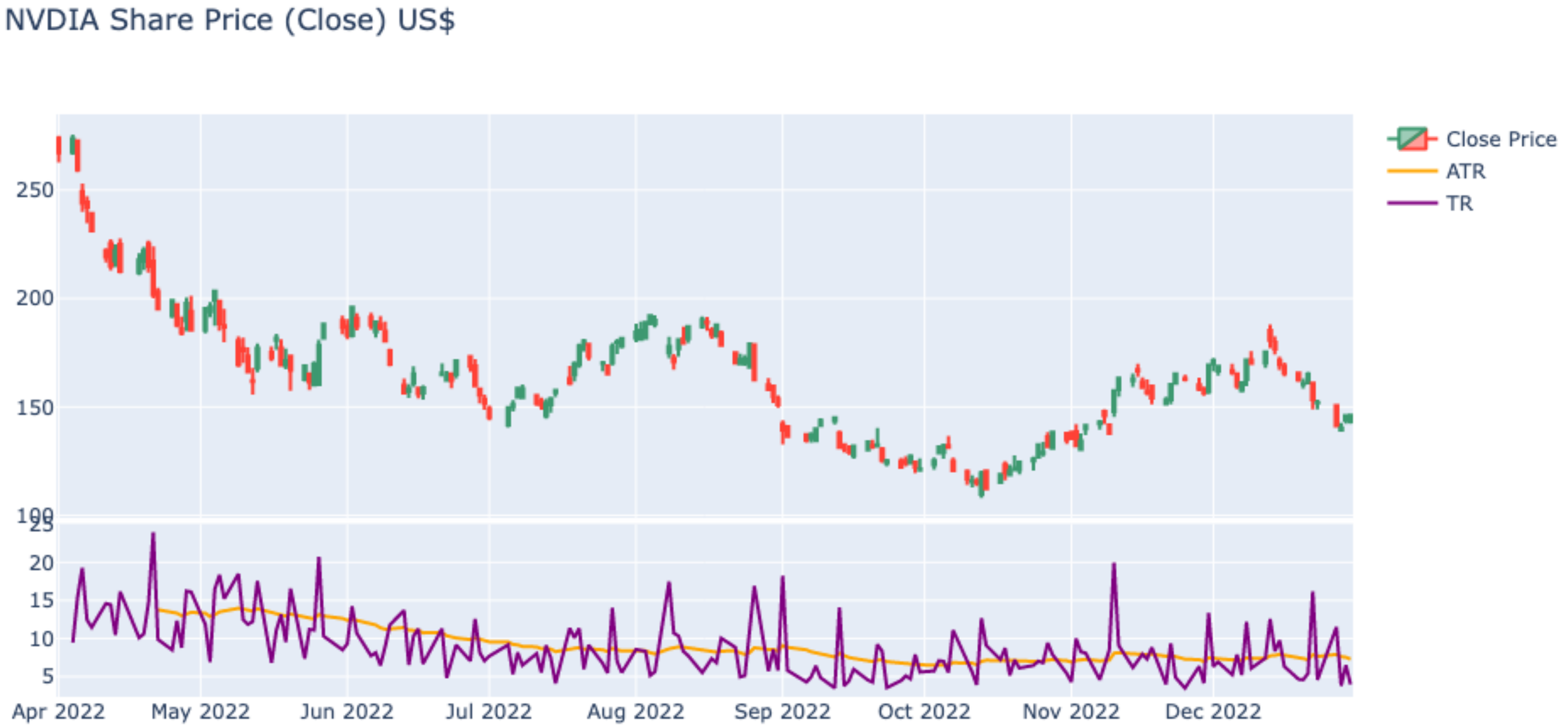
$\text{TR} = \text{True range}$

ATR & TR

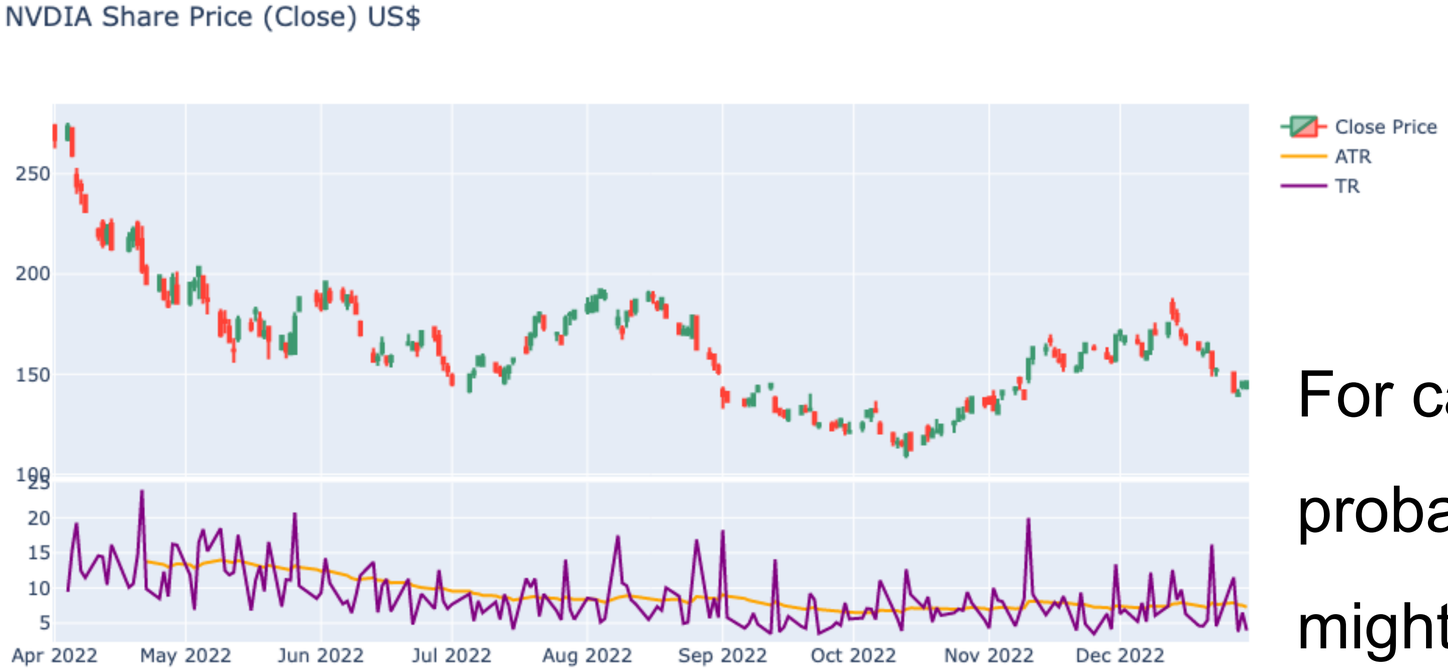
```
atr = talib.ATR(df_nvda['High'], df_nvda['Low'],df_nvda['Close'], timeperiod=14)
tr = talib.TRANGE(df_nvda['High'], df_nvda['Low'],df_nvda['Close'])
```

```
fig = make_subplots(rows=2, cols=1, shared_xaxes=True,
                    vertical_spacing=0.01, row_heights=[0.7, 0.3])
fig.add_trace(go.Candlestick(x=df_nvda.index,
              open=df_nvda['Open'], high=df_nvda['High'],
              low=df_nvda['Low'], close=df_nvda['Close'],
              showlegend=True, name='Close Price'),row=1,col=1 )
fig.add_trace(go.Scatter(x=df_nvda.index, y=atr, name='ATR',
                    marker=dict(color ='orange' )), row=2, col=1)
fig.add_trace(go.Scatter(x=df_nvda.index, y=tr, name='TR',
                    marker=dict(color ='purple' )), row=2, col=1)
fig.update_layout(title="NVDIA Share Price (Close) US$",
                  xaxis_rangeslider_visible=False)
fig.show()
```

ATR & TR


NVDIA Share Price (Close) US$
Close Price
ATR
TR
250
200
150
100
25
20
15
10
5
Apr 2022
May 2022
Jun 2022
Jul 2022
Aug 2022
Sep 2022
Oct 2022
Nov 2022
Dec 2022

ATR & TR



For case TR greater than ATR, that probably means the current trend might change soon.

If you are an option trader, you should pay more attention to the asset volatility.

Chapter Wrapping

TA-Lib is a C backend library support Python and other main stream core language. We only demonstrated 6 out of 130+ TA with plotly graphs. Statistical TA data analysis will be discussed in later chapters.

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/