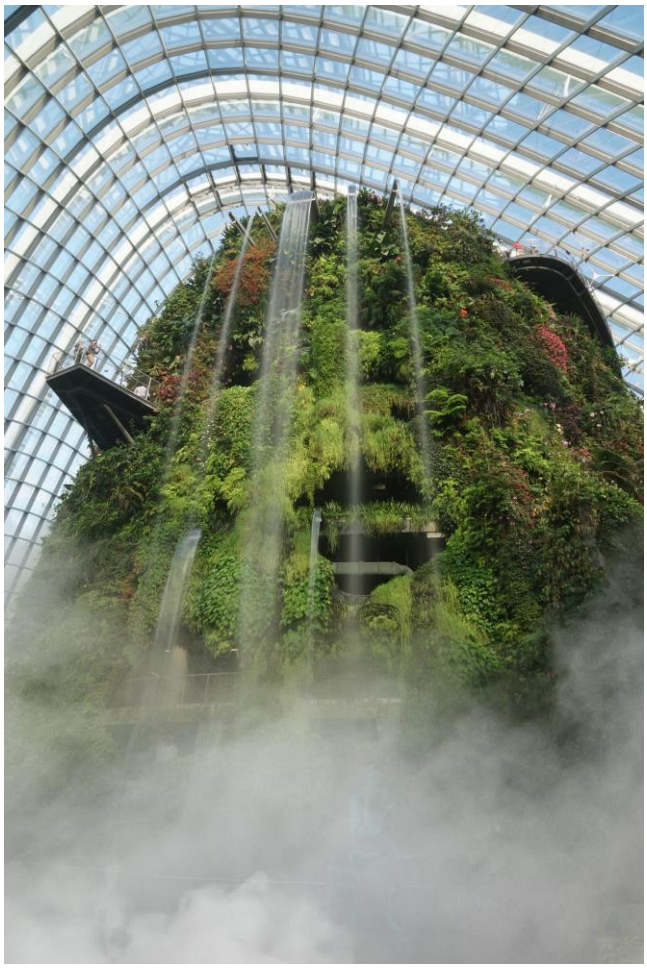
LEARNING TA-LIB IN PYTHON

DR. ALVIN ANG



Learning TA-Lib in Python

by Dr. Alvin Ang

Dr. Alvin Ang

4 min read · 1 hour ago

Introduction to Technical Analysis in Python using TA-Lib

A step-by-step tutorial about how to use TA-Lib for technical analysis in Python

itnext.io

Technical Analysis of Stocks using TA-Lib

TA-Lib technical analysis python library

towardsdatascience.com

Note that this notebook is based on:

TA-Lib : Technical Analysis Library

Technical analysis open-source software library to process financial data. Provides RSI, MACD, Stochastic, moving...

ta-lib.org

NOT THIS:

Welcome to Technical Analysis Library in Python's documentation! - Technical Analysis Library in...

It is a Technical Analysis library to financial time series datasets (open, close, high, low, volume). You can use it...

technical-analysis-library-in-python.readthedocs.io

TA-Lib and TA are different libraries!!!

All the Technical Indicators can be found here:

Function List

Technical analysis open-source software library to process financial data. Provides RSI, MACD, Stochastic, moving...

ta-lib.org

The file is here:
https://www.alvinang.sg/s/Learning_TA_Lib_in_Python_by_Dr_Alvin_Ang.ipynb

1. Pip Install and Import TA-Lib

```
!wget http://prdownloads.sourceforge.net/ta-lib/ta-lib-0.4.0-src.tar.gz
!tar -xzvf ta-lib-0.4.0-src.tar.gz
%cd ta-lib
!./configure --prefix=/usr
!make
!make install
```

```
!pip install Ta-Lib
!pip install yfinance
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

plt.style.use('fivethirtyeight')

import yfinance as yf
import talib as ta
```

2. Download DBS Stock Data

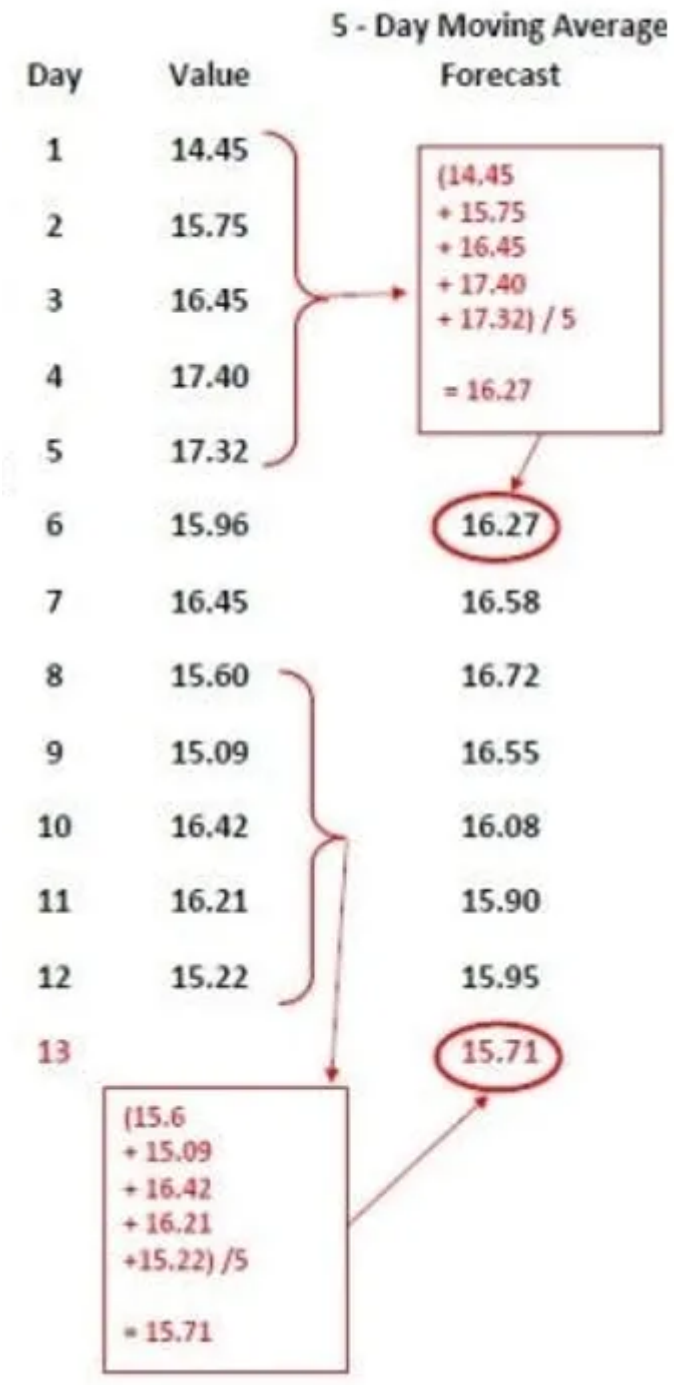
2. Download DBS Stock Data

```
[ ]  df = yf.download('D05.SI', start='2022-01-01', end = '2022-12-31' )
```


Date	Open	High	Low	Close	Adj Close	Volume
2022-01-03	32.799999	32.830002	32.720001	32.790001	30.541002	1610600
2022-01-04	33.049999	33.779999	33.009998	33.709999	31.397894	5285200
2022-01-05	33.799999	33.849998	33.070000	33.240002	30.960133	4654600
2022-01-06	32.980000	33.820000	32.959999	33.820000	31.500353	4700100
2022-01-07	34.150002	34.369999	34.040001	34.369999	32.012627	5028600
...	...	...	...	...	...	...
2022-12-23	34.000000	34.230000	33.980000	34.040001	33.089344	1341300
2022-12-27	34.570000	34.570000	34.130001	34.299999	33.342083	1666900
2022-12-28	34.020000	34.349998	34.020000	34.200001	33.244873	1563100
2022-12-29	34.000000	34.080002	33.779999	33.930000	32.982418	3464924
2022-12-30	34.000000	34.080002	33.759998	33.919998	32.972694	1820100

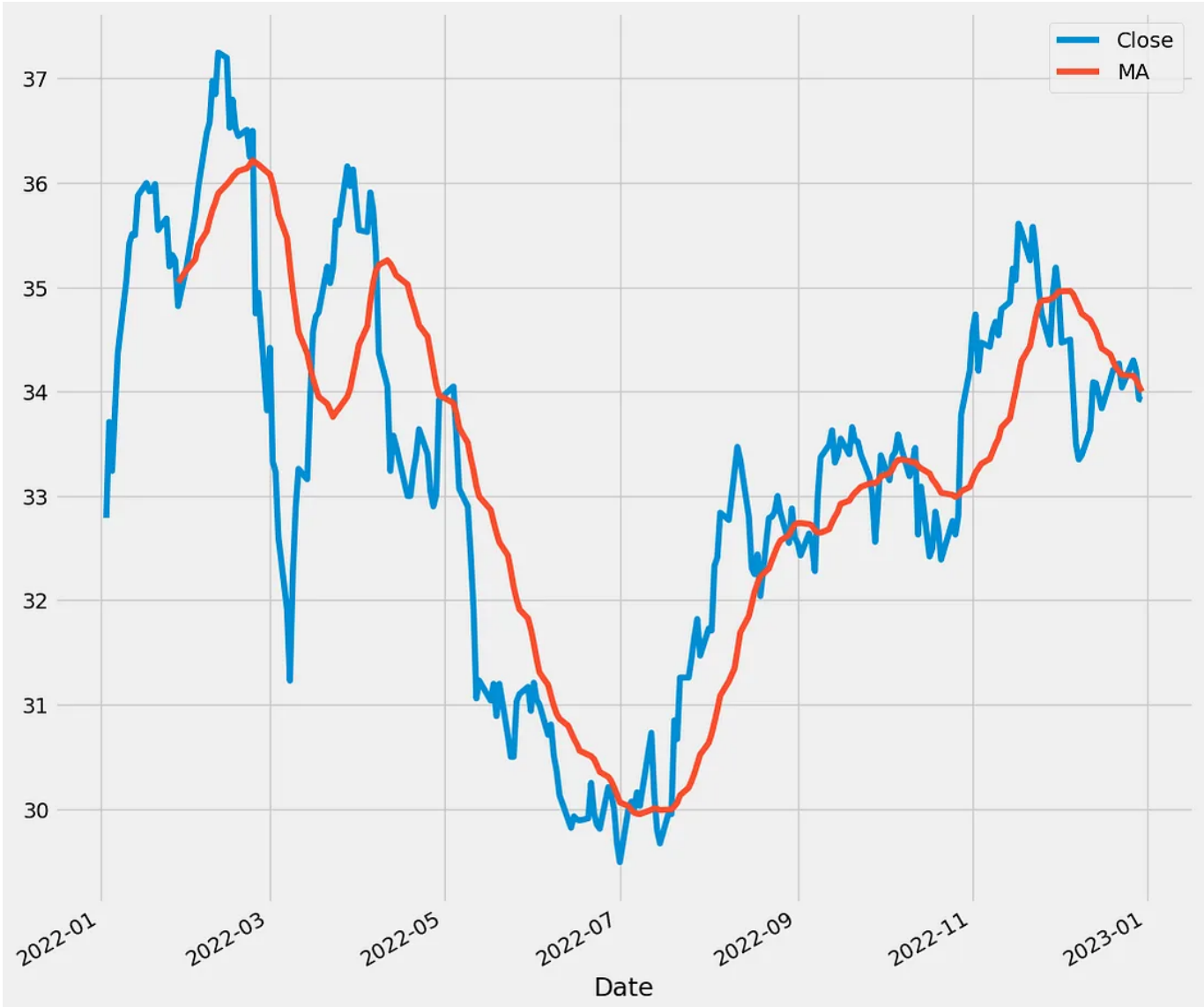
3. Simple Moving Average (SMA)

How does SMA work?



3. Simple Moving Average (SMA)

```
[ ]   1 df['MA'] = ta.SMA(df['Close'],20)
      2 df[['Close','MA']].plot(figsize=(12,12))
      3 plt.show()
```




```
1 #u see now that the new MA column has been added
2 df
```

Date	Open	High	Low	Close	Adj Close	Volume	MA
2022-01-03	32.799999	32.830002	32.720001	32.790001	30.541002	1610600	NaN
2022-01-04	33.049999	33.779999	33.009998	33.709999	31.397894	5285200	NaN
2022-01-05	33.799999	33.849998	33.070000	33.240002	30.960133	4654600	NaN
2022-01-06	32.980000	33.820000	32.959999	33.820000	31.500353	4700100	NaN
2022-01-07	34.150002	34.369999	34.040001	34.369999	32.012627	5028600	NaN
...	...	...	...	...	...	...	...
2022-12-23	34.000000	34.230000	33.980000	34.040001	33.089344	1341300	34.1595
2022-12-27	34.570000	34.570000	34.130001	34.299999	33.342083	1666900	34.1520
2022-12-28	34.020000	34.349998	34.020000	34.200001	33.244873	1563100	34.1145
2022-12-29	34.000000	34.080002	33.779999	33.930000	32.982418	3464924	34.0515
2022-12-30	34.000000	34.080002	33.759998	33.919998	32.972694	1820100	34.0000

4. Exponential Moving Average (EMA)

How does EMA work?

Day	Value	Exponential Smoothing Forecast
1	14.45	
2	15.75	14.45
3	16.45	15.36
4	17.40	16.12
5	17.32	17.02
6	15.96	17.23
7	16.45	16.34
8	15.60	16.42
9	15.09	15.85
10	16.42	15.32
11	16.21	16.09
12	15.22	16.17
13		15.51

0.7 (15.75)
+ 0.3 (14.45)

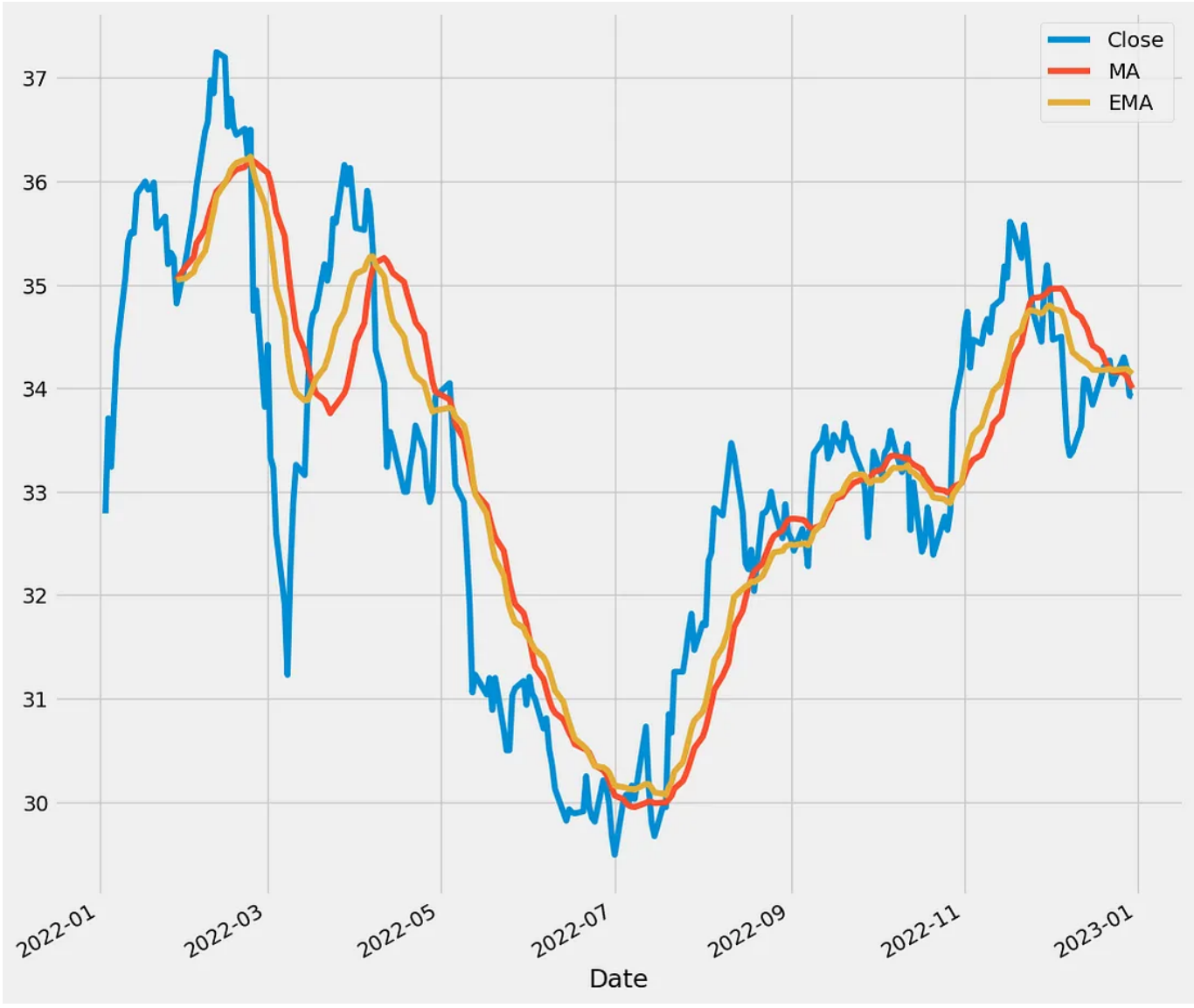
= 15.36

0.7 (15.22)
+ 0.3 (16.17)

= 15.51

4. Exponential Moving Average (EMA)

```
1 df['MA'] = ta.SMA(df['Close'],timeperiod=20)
2 df['EMA'] = ta.EMA(df['Close'], timeperiod = 20)
3 df[['Close','MA','EMA']].plot(figsize=(12,12))
4 plt.show()
```



```
1 #u see now that the new EMA column has been added
2 df
```

Date	Open	High	Low	Close	Adj Close	Volume	MA	EMA
2022-01-03	32.799999	32.830002	32.720001	32.790001	30.541002	1610600	NaN	NaN
2022-01-04	33.049999	33.779999	33.009998	33.709999	31.397894	5285200	NaN	NaN
2022-01-05	33.799999	33.849998	33.070000	33.240002	30.960133	4654600	NaN	NaN
2022-01-06	32.980000	33.820000	32.959999	33.820000	31.500353	4700100	NaN	NaN
2022-01-07	34.150002	34.369999	34.040001	34.369999	32.012627	5028600	NaN	NaN
...	...	...	...	...	...	...	...	...
2022-12-23	34.000000	34.230000	33.980000	34.040001	33.089344	1341300	34.1595	34.174597
2022-12-27	34.570000	34.570000	34.130001	34.299999	33.342083	1666900	34.1520	34.186540
2022-12-28	34.020000	34.349998	34.020000	34.200001	33.244873	1563100	34.1145	34.187822
2022-12-29	34.000000	34.080002	33.779999	33.930000	32.982418	3464924	34.0515	34.163267

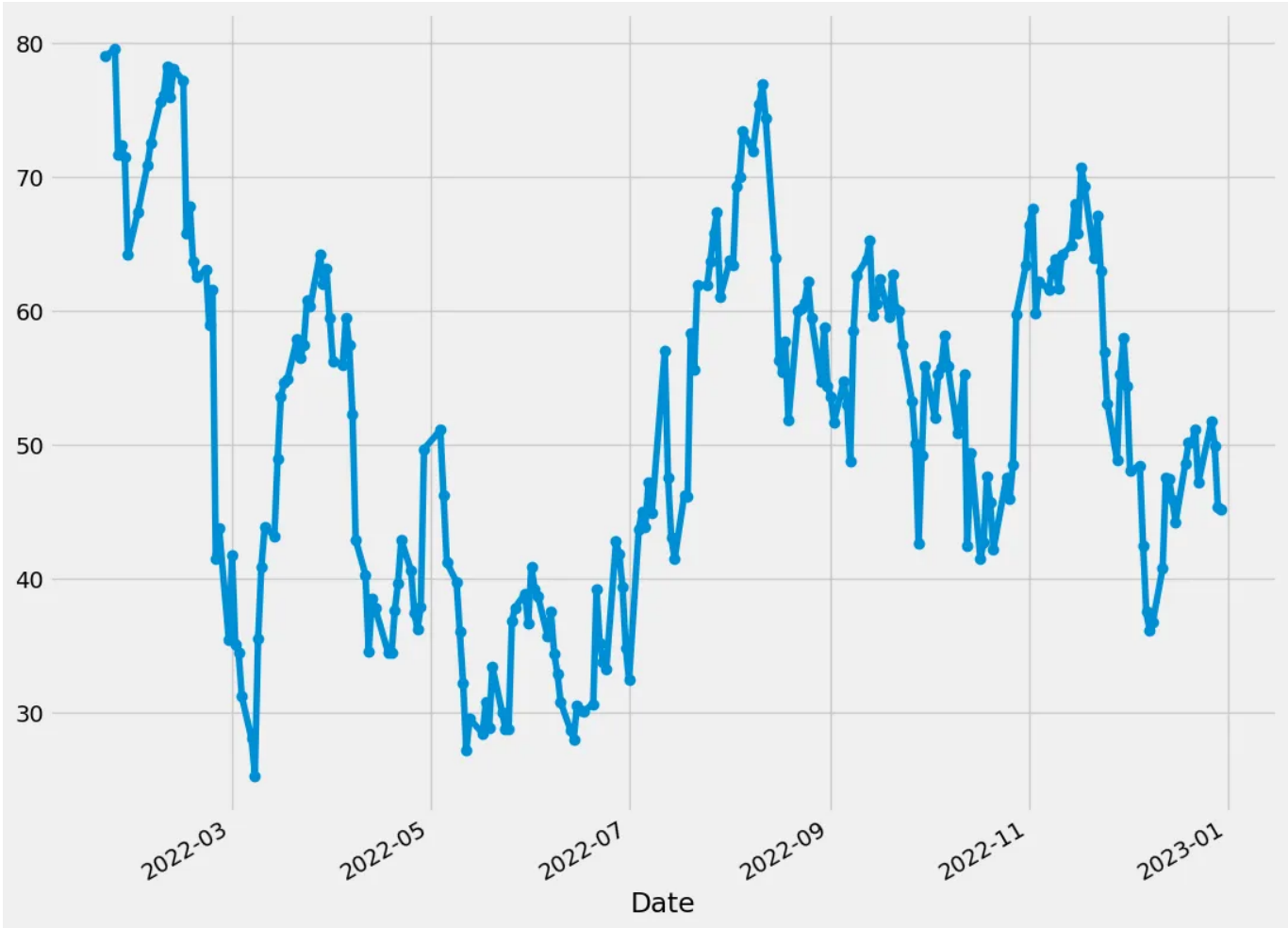
5. Relative Strength Index (RSI)

How does RSI work?

- RSI value lies between 0~100.
- If RSI < *30*, it means it is oversold and is a good chance to ***BUY.***
- If RSI > *70*, it means it is overbought and is a good chance to ***SELL.***
- RSI = 50 is the Centerline.
- If RSI is above Centerline (>50), it means its ***BULLISH.***
- If RSI is below Centerline (<50), it means its ***Bearish.***

5. Relative Strength Index (RSI)

```
1 df['RSI'] = ta.RSI(df['Close'],14)
2 df['RSI'].plot(figsize=(12,10),marker='o')
3 plt.show()
```




```
1 #u see now the new RSI column has been added
2 df
```

Date	Open	High	Low	Close	Adj Close	Volume	MA	EMA	RSI
2022-01-03	32.799999	32.830002	32.720001	32.790001	30.541002	1610600	NaN	NaN	NaN
2022-01-04	33.049999	33.779999	33.009998	33.709999	31.397894	5285200	NaN	NaN	NaN
2022-01-05	33.799999	33.849998	33.070000	33.240002	30.960133	4654600	NaN	NaN	NaN
2022-01-06	32.980000	33.820000	32.959999	33.820000	31.500353	4700100	NaN	NaN	NaN
2022-01-07	34.150002	34.369999	34.040001	34.369999	32.012627	5028600	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...
2022-12-23	34.000000	34.230000	33.980000	34.040001	33.089344	1341300	34.1595	34.174597	47.162229
2022-12-27	34.570000	34.570000	34.130001	34.299999	33.342083	1666900	34.1520	34.186540	51.743046
2022-12-28	34.020000	34.349998	34.020000	34.200001	33.244873	1563100	34.1145	34.187822	49.949402
2022-12-29	34.000000	34.080002	33.779999	33.930000	32.982418	3464924	34.0515	34.163267	45.375742
2022-12-30	34.000000	34.080002	33.759998	33.919998	32.972694	1820100	34.0000	34.140099	45.210589

250 rows × 9 columns

6. Bollinger Bands (BB)

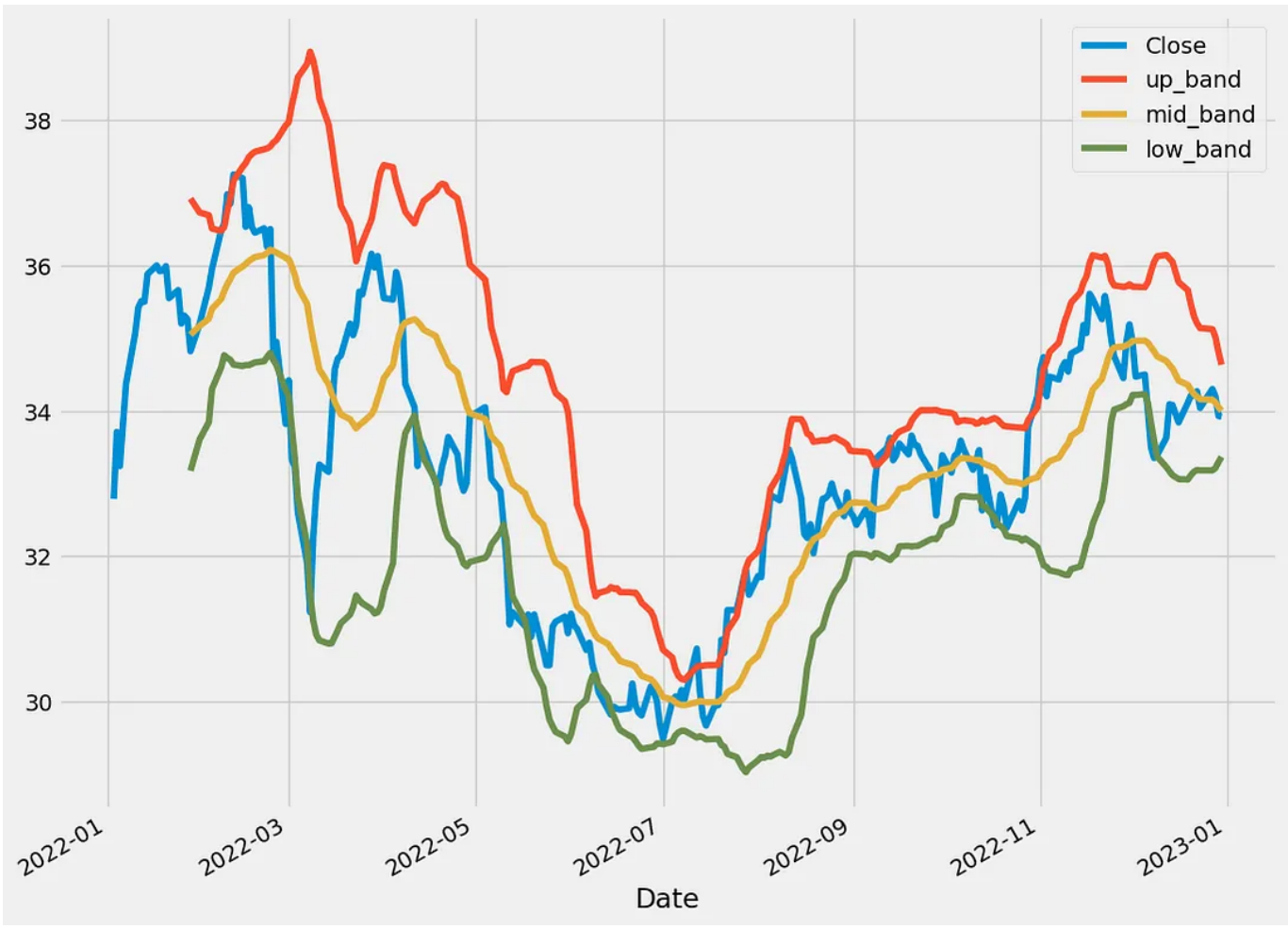
How does BB work?

- The Middle Band is actually just a SMA of 20 periods.
- The Upper and Lower Bands are actually just 2 Standard Deviations (SD) away from the Middle Band.
- During Quiet Market, the band will ***tighten.***
- During Votalile Market, the band will ***widen.***
- You may ***Sell*** if the price touches / breaks ***below*** the Lower Band.
- You may ***Buy*** if the price touches / breaks ***above*** the Upper Band.

6. Bollinger Bands (BB)

```
df['up_band'], df['mid_band'], df['low_band'] = ta.BBANDS(df['Close'], timeperiod =20)

df[['Close','up_band','mid_band','low_band']].plot(figsize=(12,10))
plt.show()
```




```
#u see that the BBs have been created: Up Band / Mid Band / Low Bands
df
```

Date	Open	High	Low	Close	Adj Close	Volume	MA	EMA	RSI	up_band	mid_band	low_band
2022-01-03	32.799999	32.830002	32.720001	32.790001	30.541002	1610600	NaN	NaN	NaN	NaN	NaN	NaN
2022-01-04	33.049999	33.779999	33.009998	33.709999	31.397894	5285200	NaN	NaN	NaN	NaN	NaN	NaN
2022-01-05	33.799999	33.849998	33.070000	33.240002	30.960133	4654600	NaN	NaN	NaN	NaN	NaN	NaN
2022-01-06	32.980000	33.820000	32.959999	33.820000	31.500353	4700100	NaN	NaN	NaN	NaN	NaN	NaN
2022-01-07	34.150002	34.369999	34.040001	34.369999	32.012627	5028600	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...
2022-12-23	34.000000	34.230000	33.980000	34.040001	33.089344	1341300	34.1595	34.174597	47.162229	35.137631	34.1595	33.181369
2022-12-27	34.570000	34.570000	34.130001	34.299999	33.342083	1666900	34.1520	34.186540	51.743046	35.123383	34.1520	33.180617
2022-12-28	34.020000	34.349998	34.020000	34.200001	33.244873	1563100	34.1145	34.187822	49.949402	35.015088	34.1145	33.213912
2022-12-29	34.000000	34.080002	33.779999	33.930000	32.982418	3464924	34.0515	34.163267	45.375742	34.806915	34.0515	33.296085
2022-12-30	34.000000	34.080002	33.759998	33.919998	32.972694	1820100	34.0000	34.140099	45.210589	34.634067	34.0000	33.365933

250 rows × 12 columns

About Dr. Alvin Ang

www.AlvinAng.sg

Dr. Alvin Ang earned his Ph.D., Masters and Bachelor degrees from NTU, Singapore. Previously he was a Principal Consultant (Data Science) as well as an Assistant Professor. He was also 8 years SUSS adjunct lecturer. His focus and interest is in the area of real world data science. Though an operational researcher by study, his passion for practical applications outweigh his academic background He is a scientist, entrepreneur, as well as a personal/business advisor.

More about him at www.AlvinAng.sg.

Python | Python Finance | Technical Analysis | Python For Finance | Technical Analysis Tools

Written by Dr. Alvin Ang

Edit profile

67 Followers

www.AlvinAng.sg | www.linkedin.com/in/dr-alvin | www.datafrens.sg | https://public.tableau.com/app/profile/dr.alvin.ang

More from Dr. Alvin Ang