Open in app ↗

Medium Search Write S

Member-only story

Striking Backtesting Performance of the PLTR Elliott Wave Algo-Trading Strategy vs Buy&Hold

Discover the dynamic fractal pattern of market action to identify the top profitable PLTR trading strategy that significantly outperforms the passive benchmark

Alexzap Follow 6 min read · 2 days ago

17 1

"The stock market is not a gamble; it is a business. And it should be conducted as such, and it must be, to be successful." — Ralph Nelson Elliott

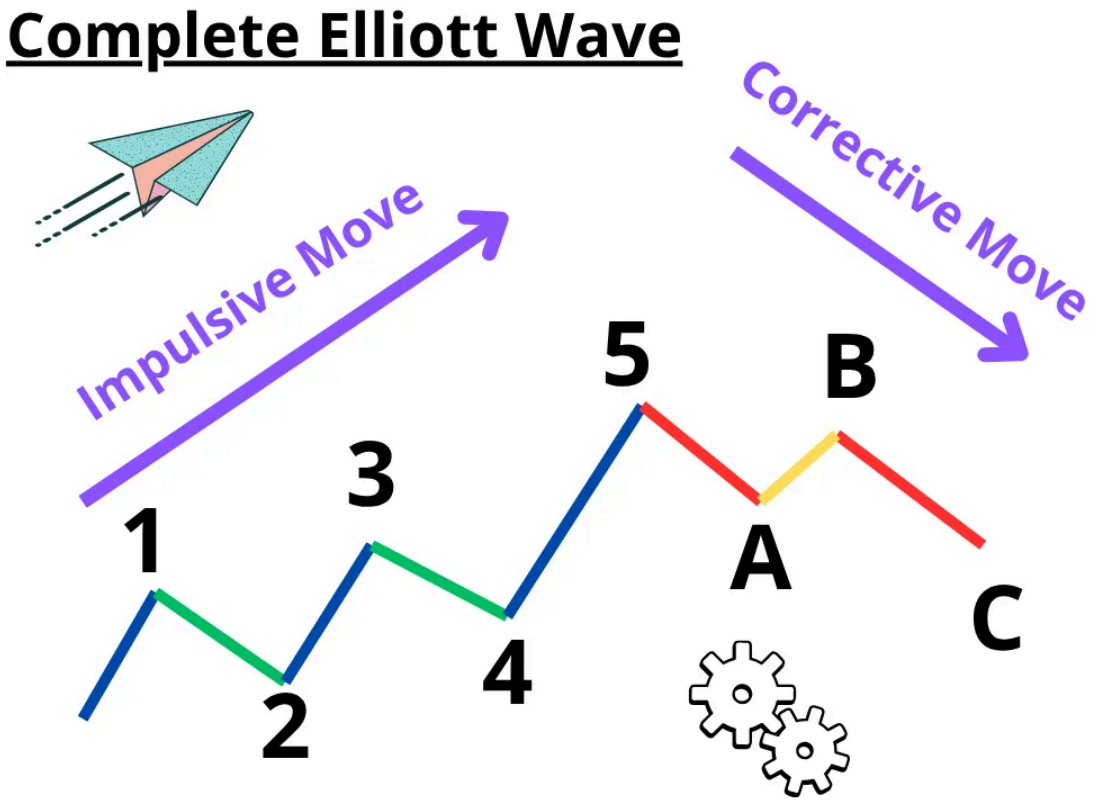


Image Design by the Author via Canva.

- In this post, we'll take a look at the high-frequency trading strategy based on the Elliot Wave Oscillator (EWO) [1].
- EWO is fundamentally a trend-following, momentum indicator.
- The theory assumes that stock price movements can be predicted because they move in repeating up-and-down patterns (aka waves) created by investor sentiment.
- In the sequel, we'll use the wave patterns to identify potential trade opportunities, such as buying at the end of a corrective

wave and selling at the end of an impulse wave.

- For the present use-case example, we'll focus on Palantir Technologies Inc. (PLTR) as the time series. Here's why:

1. The stock has gained 81% in 2025
2. It has a 21-day ATR of 4.67%
3. It holds an Accumulation/Distribution Rating of B-plus
4. It is among **AI stocks** to watch.

- Let's delve into the specifics of the ultimate 4 step Elliott Wave trading strategy for maximizing PLTR stock returns in 2025.
- Step 1: Importing the necessary Python libraries and using the TwelveData API reading the PLTR stock data from 2025–01–01 to 2025–06–26 [2]

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import requests
from math import floor
from termcolor import colored as cl

plt.style.use('fivethirtyeight')
plt.rcParams['figure.figsize'] = (20,10)

# EXTRACTING STOCK DATA
```

```
import pandas as pd
import numpy as np
import requests
import matplotlib.pyplot as plt
from math import floor
from termcolor import colored as cl

plt.style.use('fivethirtyeight')
plt.rcParams['figure.figsize'] = (20,10)

def get_historical_data(symbol, start_date):
    api_key = 'YOUR API KEY'
    api_url = f'https://api.twelvedata.com/time_series?symbol={symb
    raw_df = requests.get(api_url).json()
    df = pd.DataFrame(raw_df['values']).iloc[::-1].set_index('datet
    df = df[df.index >= start_date]
    df.index = pd.to_datetime(df.index)
    return df

df = get_historical_data('PLTR', '2025-01-01')
df.tail()
```

datetime	open	high	low	close	volume
2025-06-20	140.69000	142.24001	136.74200	137.30000	87067000.0
2025-06-23	138.89999	142.14999	135.96001	139.92000	70501100.0
2025-06-24	140.98000	143.66000	137.80000	143.23000	58574200.0
2025-06-25	144.49001	147.67000	141.53000	142.89999	61244300.0
2025-06-26	144.93000	148.20000	142.93000	144.42000	3010721.0

- Using Plotly to visualize the PLTR candlestick chart with volume [4]

```
from plotly.subplots import make_subplots
import plotly.graph_objects as go
```

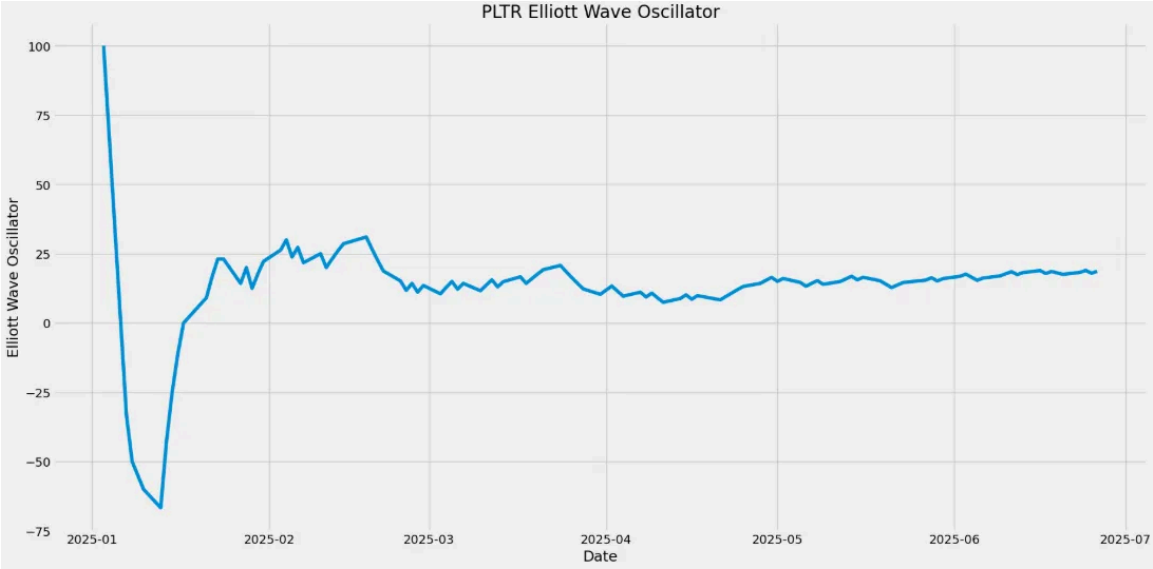
```
fig = make_subplots(rows=2, cols=1, shared_xaxes=True, vertical_spa
fig.add_trace(go.Candlestick(x=df.index,
                             open=df['open'],
                             high=df['high'],
                             low=df['low'],
                             close=df['close'],
                             name='PLTR'),
              row=1, col=1)


# Plotting volume chart on the second row
fig.add_trace(go.Bar(x=df.index,
                     y=df['volume'],
                     name='Volume',
                     marker=dict(color='black', opacity=1.0)),
              row=2, col=1)



# Configuring layout
fig.update_layout(title='PLTR Candlestick Chart',
                  yaxis=dict(title='Price (USD)'),
                  height=1000,
                 template = 'plotly_white')

# Configuring axes and subplots
fig.update_xaxes(rangeslider_visible=False, row=1, col=1)
fig.update_xaxes(rangeslider_visible=False, row=2, col=1)
fig.update_yaxes(title_text='Price (USD)', row=1, col=1)
fig.update_yaxes(title_text='Volume', row=2, col=1)

fig.show()
```

PLTR candlestick chart with volume

- Step 2: Calculating and plotting the Elliott Wave Oscillator [1]

```
# Calculate the difference between the current and previous day's c
df['diff'] = df['close'].diff()

# Identify upward and downward trends
df['up'] = np.where(df['diff'] > 0, 1, 0)
df['down'] = np.where(df['diff'] < 0, 1, 0)

# Calculate the running total of up and down days
df['up_days'] = df['up'].cumsum()
```

```
df['down_days'] = df['down'].cumsum()

# Calculate the Elliott Wave Oscillator
df['ewo'] = 100 * (df['up_days'] - df['down_days']) / (df['up_days'

# Plot the Elliott Wave Oscillator
import matplotlib.pyplot as plt

plt.plot(df['ewo'])
plt.xlabel('Date')
plt.ylabel('Elliott Wave Oscillator')
plt.title('PLTR Elliott Wave Oscillator')
plt.show()
```

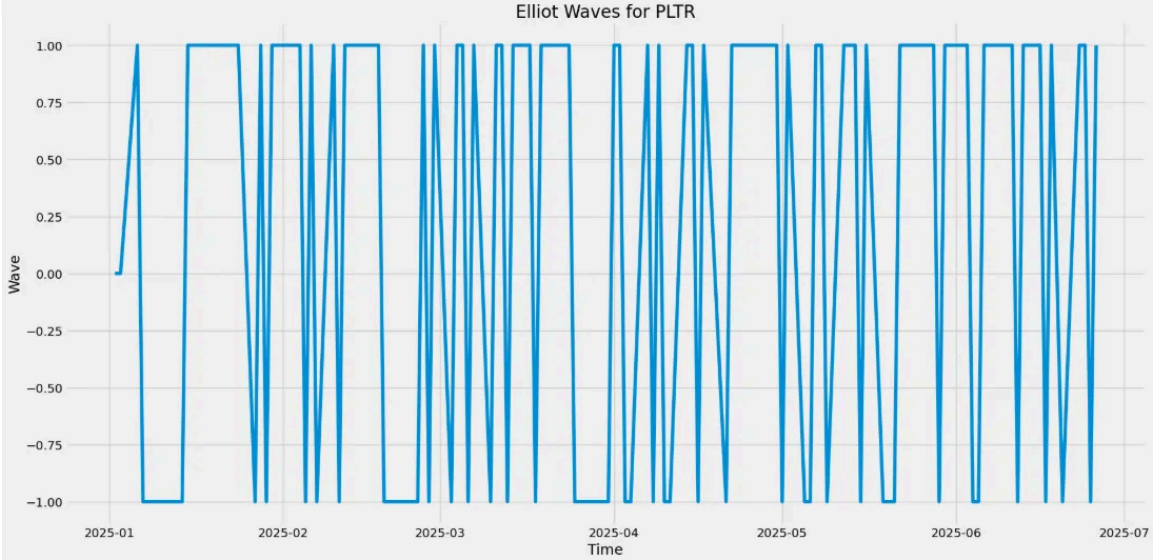


PLTR Elliott Wave Oscillator

- Step 3: Identifying the Elliot Waves for PLTR [1]

```
waves = []
for i in range(0, len(df)):
```

```
    if i==0:
        waves.append(0)
    if df["close"][i] > df["close"][i-1] and i>0:
        waves.append(1)
    elif df["close"][i] < df["close"][i-1] and i>0:
        waves.append(-1)

#Add waves to the SP500 dataframe

x=0
waves.insert(0, x)

df["Waves"] = waves

#Plot the waves

plt.plot(df["Waves"])
plt.title("Elliot Waves for PLTR")
plt.xlabel("Time")
plt.ylabel("Wave")
plt.show()

#Identify the waves

wave_count = 0
for i in range(1, len(df)):
    if df["Waves"][i] == 1:
        wave_count += 1
    elif df["Waves"][i] == -1:
        wave_count -= 1

#identify future evolution of spot based on wave count

if wave_count > 0:
    print("PLTR is likely to increase in the future.")
elif wave_count < 0:
    print("PLTR is likely to decrease in the future.")
else:
    print("PLTR is likely to remain stable in the future.")


PLTR is likely to increase in the future.
```

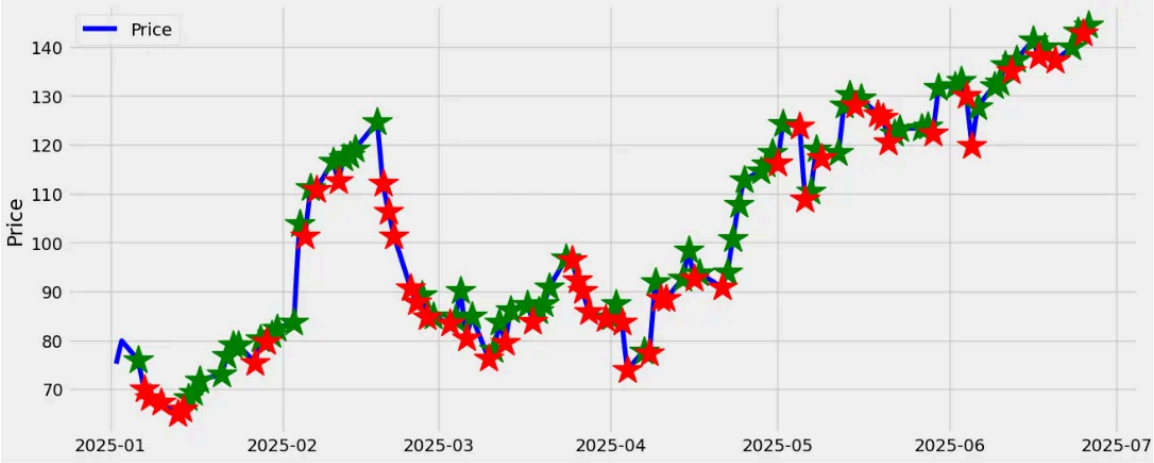


Elliot Waves for PLTR

- Visualizing the Elliot Wave buy and sell signals for PLTR [3]

```
buysignals = df[df["Waves"] == 1]
sellsignals = df[df["Waves"] == -1]

plt.figure(figsize=(14,6))
plt.plot(df["close"], color="b", label="Price")

plt.ylabel("Price")

for idx in buysignals.index.tolist():
  plt.plot(
      idx,
      df.loc[idx]["close"],
      "g*",
      markersize=25
  )

for idx in sellsignals.index.tolist():
  plt.plot(
      idx,
      df.loc[idx]["close"],
```

```
        "r*",
        markersize=25
    )

ax = plt.gca()
for index, label in enumerate(ax.xaxis.get_ticklabels()):
    if index % 1 != 0:
        label.set_visible(False)
plt.legend()
plt.show()
```

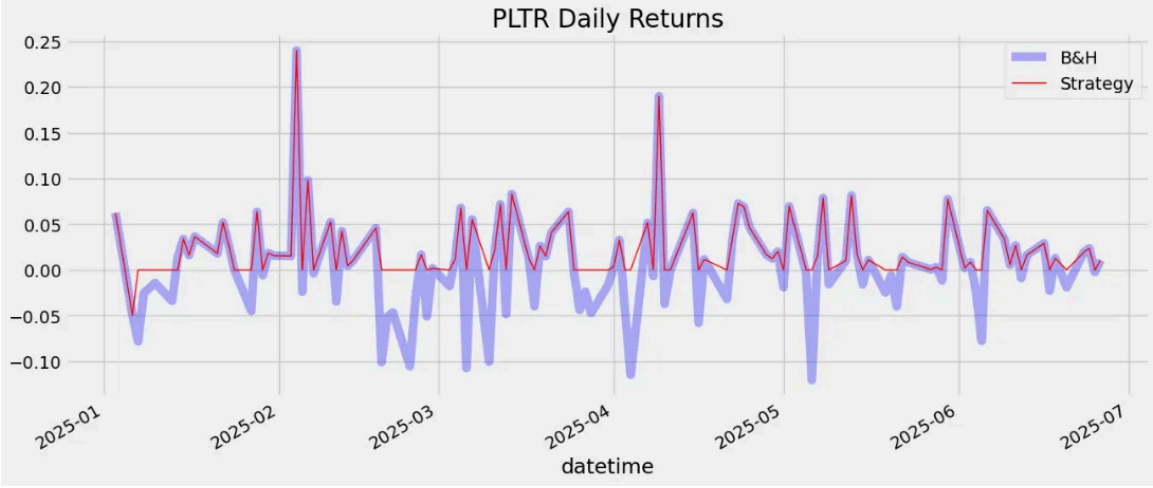


Elliot Wave buy and sell signals vs close price for PLTR

- Step 4: Comparing PLTR daily and cumulative returns (backtesting [2]): Elliot Wave strategy vs B&H benchmark

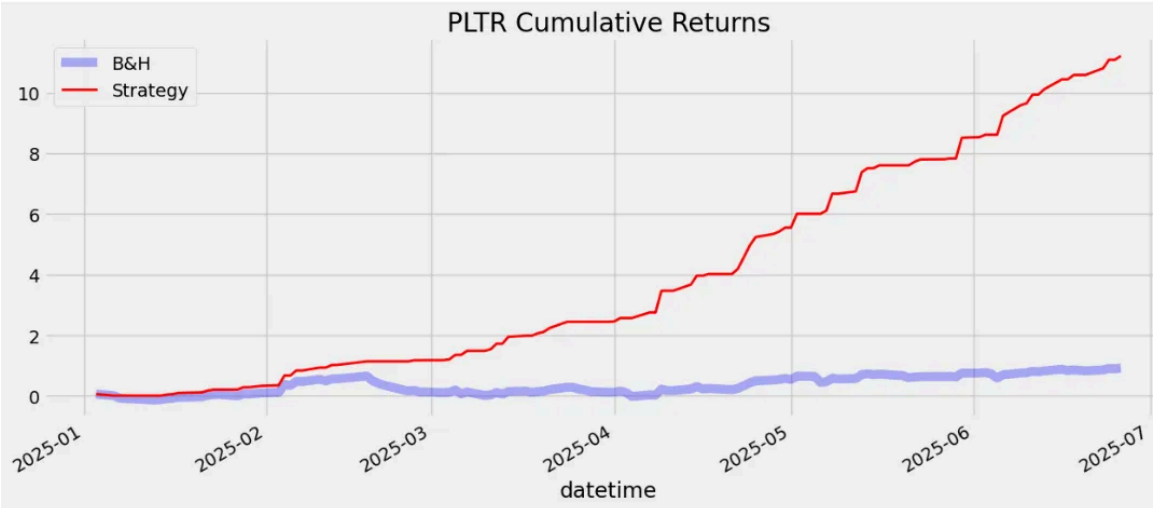
```
mysignal=df["Waves"]
position = []
for i in range(len(mysignal)):
    if mysignal[i] == 1:
        position.append(1)
    else:
```

```
            position.append(0)

    for i in range(len(df['close'])):
        if mysignal[i] == 1:
            position[i] = 1
        elif mysignal[i] == -1:
            position[i] = 0
        else:
            position[i] = position[i-1]

    rets = df.close.pct_change().dropna()
    strat_rets = position[1:]*rets
    plt.figure(figsize=(14,6))

    #plt.title('Daily Returns')
    rets.plot(color = 'blue', alpha = 0.3, linewidth = 7,label='B&H')
    strat_rets.plot(color = 'r', linewidth = 1,label='Strategy')
    plt.legend()
    plt.title("PLTR Daily Returns")
    plt.show()
```



PLTR Daily Returns

```
rets_cum = (1 + rets).cumprod() - 1
strat_cum = (1 + strat_rets).cumprod() - 1

plt.figure(figsize=(14,6))

plt.title('Cumulative Returns')
rets_cum.plot(color = 'blue', alpha = 0.3, linewidth = 7,label='B&H
strat_cum.plot(color = 'r', linewidth = 2,label='Strategy')
plt.legend()
plt.title("PLTR Cumulative Returns")
plt.show()
```



PLTR Cumulative Returns

- Comparing cumulative returns from the last available 5 days

```
#Strategy

strat_cum.tail()
```

```
datetime
2025-06-20    10.586790
2025-06-23    10.807892
2025-06-24    11.087224
2025-06-25    11.087224
2025-06-26    11.215795

#B&H

rets_cum.tail()

datetime
2025-06-20    0.826041
2025-06-23    0.860886
2025-06-24    0.904908
2025-06-25    0.900519
2025-06-26    0.920734
```

- It is clear that the PLTR profitability ratio Strategy/B&H is 12.17 on 2025–06–26.

Conclusions

- In this post, we have shown how to backtest the EWO trading strategy while avoiding subjective interpretation of repeated patterns found in larger/smaller time frames.
- This is where Elliott Wave meets quant trading.
- In practice, multiple factors should line up to help confirm trade signals with confidence: support and resistance levels, different technical indicators, fundamental analysis, etc.

References

1. How to build and use the Elliot Wave oscillator and waves in python ?
2. Algorithmic Trading with Python
3. Visualizing Trading Signals in Python
4. Data Science for Financial Markets

Explore More

- Unlocking Palantir Technologies' Potential with Profitable Algo Trading Strategies in Python — 1. ADX RSI
- Unlocking Palantir Technologies' Potential with Profitable Algo Trading Strategies in Python — 2. Middle-Term Hull Volume Moving Average
- Unlocking Palantir Technologies' Potential with Profitable Algo Trading Strategies in Python — 3. Dynamic Risk Analytics

Contacts

- Website
- GitHub
- Facebook
- X/Twitter

- Pinterest
- Mastodon
- Tumblr

Disclaimer

- I declare that this article is written by me and not with any generative AI tool such as ChatGPT.
- I declare that no data privacy policy is breached, and that any data associated with the contents here are obtained legitimately to the best of my knowledge.
- The following disclaimer clarifies that the information provided in this article is for educational use only and should not be considered financial or investment advice.
- The information provided does not take into account your individual financial situation, objectives, or risk tolerance.
- Any investment decisions or actions you undertake are solely your responsibility.
- You should independently evaluate the suitability of any investment based on your financial objectives, risk tolerance, and investment timeframe.
- It is recommended to seek advice from a certified financial professional who can provide personalized guidance tailored

to your specific needs.

- The tools, data, content, and information offered are impersonal and not customized to meet the investment needs of any individual. As such, the tools, data, content, and information are provided solely for informational and educational purposes only.
- *All images unless otherwise noted are by the author.*

Python | Algorithmic Trading | Backtesting | Elliottwave | Palantir

Written by Alexzap

3.1K followers · 1.97K following

Data scientist, shareholder, investor, fintech, passionate about ML/AI, Python, and open-source knowledge sharing https://newdigitals.org/

Follow

Responses (1)

Steven Feng CAI

What are your thoughts?

Altan Alpay
5 hours ago

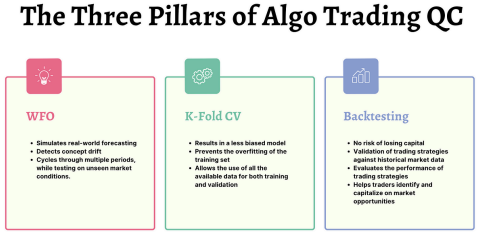
as of june 30, this is a great lesson in how not to do a strat. it is multiplying past returns after labeling if it was up or down... and it gets great returns :D.

It is also calculating 'ewo' and not using it later.

Intuitively whenever you see a strat that is 10x+ of buynhod you should look for bias.

1 reply Reply

More from Alexzap



In InsiderFinance Wire by Alexzap

In InsiderFinance Wire by Itay V

Is Walk-Forward Optimization (WFO) the Future of...

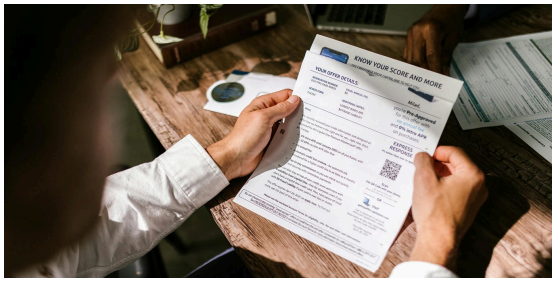
Unlock the Great Value of WFO for Live Trading with Sudden Market...

Jun 2 111

I tried trading with Agentic AI, and it's mind blowing

Discover how agentic systems are transforming the retail-trading scene...

May 27 705 11

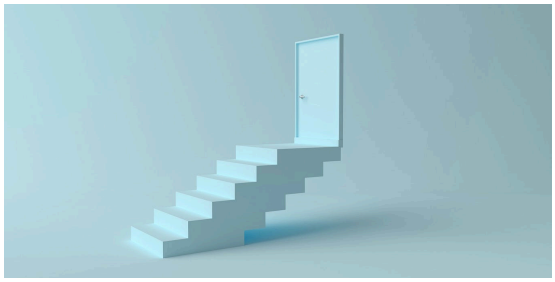


In InsiderFinance Wire by Steeve Towa

While Everyone's Focused on AI and Crypto, the Real...

How a 200-year-old trade finance instrument is getting a blockchain...

Jun 3 536 20



Alexzap

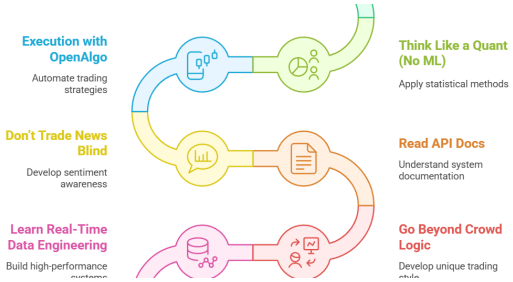
Comparing Profitability of 4 Moving Average (MA) Algo-...

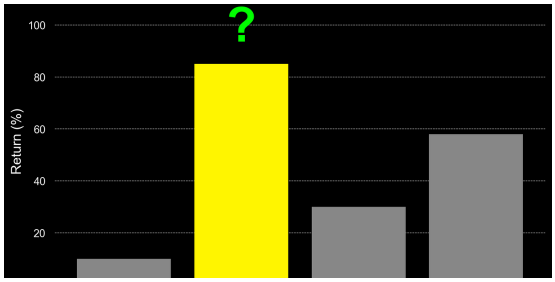
How DEMA Crossover Strategy Significantly Outperforms SMA, EMA...

Jun 16 96 1

See all from Alexzap

Recommended from Medium





Rajandran R (Creator - OpenAlgo)

Algorithmic Trading Roadmap 2025: From Curio...

The dream of algorithmic trading is simple: let code take over the...

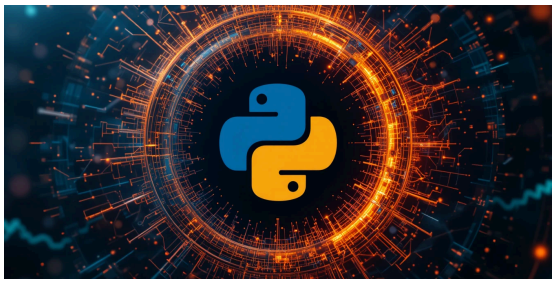
Jun 22 23 3

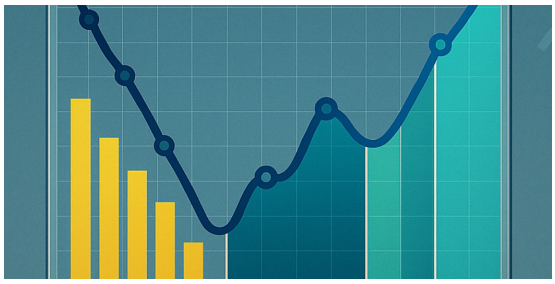
In InsiderFinance Wire by Brian Hulela

This #1 Optimization Strategy Will Skyrocket Your Trading...

A Comparative Analysis of Buy and Hold vs. Optimized Trading Strategies

5d ago 2 1





Unicorn Day

The Quest for the Perfect Trading Score: Turning...

Anitha Rajasekaran

Moments That Matter: Identifying Change Points in...

Navigating the financial markets... it feels like being hit by a tsunami of da...

3d ago 43

Behind every sudden spike or subtle dip lies a story—Change Point...

Mar 31 6 1

Swapnilphutane

How I Built a Multi-Market Trading Strategy That Pass...

When I first got into trading, I had no plans of building a full-blown system....

6d ago 16 1

Sriram Murali

Why Most Pricing Models Fail and How Bayesian Modeling...

(Part 2: Feature Engineering and Modeling)

Jun 18 12

See more recommendations