



# STATFLOW STRATEGY

**PRESENTED BY: KPO MEMBERS**

DOMAIN 2: QUANTITATIVE TRADING



# ▶▶▶ Architectural Design of Backtest Library

co utils.py 👤

co strategy.py 👤

co model.py 👤

co metrics.py 👤

co core.py 👤

co \_\_init\_\_.py 👤

1. Core.py—This is likely where the main engine lives — stuff like data handling, orchestration of strategy logic, or managing the pipeline (e.g., loading prices → calculating metrics → deciding trades).

2. metrics.py - it's the calculator. It computes things like momentum, volatility, returns, maybe drawdowns. Used to filter stocks or generate signals.

3. model.py - define a machine learning model, or a rule-based decision system. Could be what actually makes the buy/sell call after seeing metrics.

4. strategy.py - “If momentum > X and volatility < Y → buy stock”

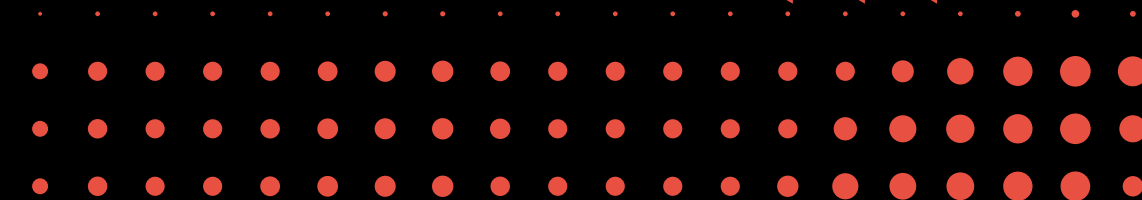
5. utils.py - Smoothing data, Converting dates, Normalizing values

6. \_\_init\_\_.py - This makes your folder a Python package. It's like the “hey, I'm ready to be imported” file. Might also expose key functions or classes from the library.



# What is the **StatFlow Strategy**?

Volatility-Aware Momentum (VAM) is a trading strategy that combines momentum signals with a volatility filter. It enters trades only when the price is trending strongly and market volatility is within a safe range this help to reduce false signals and manage risk effectively.







## Business Value

- combines momentum and volatility filters to avoid false signals
- Proactively avoids trades during high volatility, reducing drawdowns
- More Predictable Returns: Reduces surprises from sudden market swings.
- Scalable Automation: Easily expanded to multiple assets or strategies.





# How the **StatFlow Strategy Works**

Buy Signal Formula:

If  $\text{Momentum}_t > M_{threshold}$  AND  $\text{Volatility}_t < V_{threshold}$ , then BUY

Sell Signal Formula:

If  $\text{Momentum}_t < -M_{threshold}$  AND  $\text{Volatility}_t < V_{threshold}$ , then SELL





## How the **StatFlow Strategy Works**

```
threshold = merged_df['combo_signal'].quantile(0.80)
```

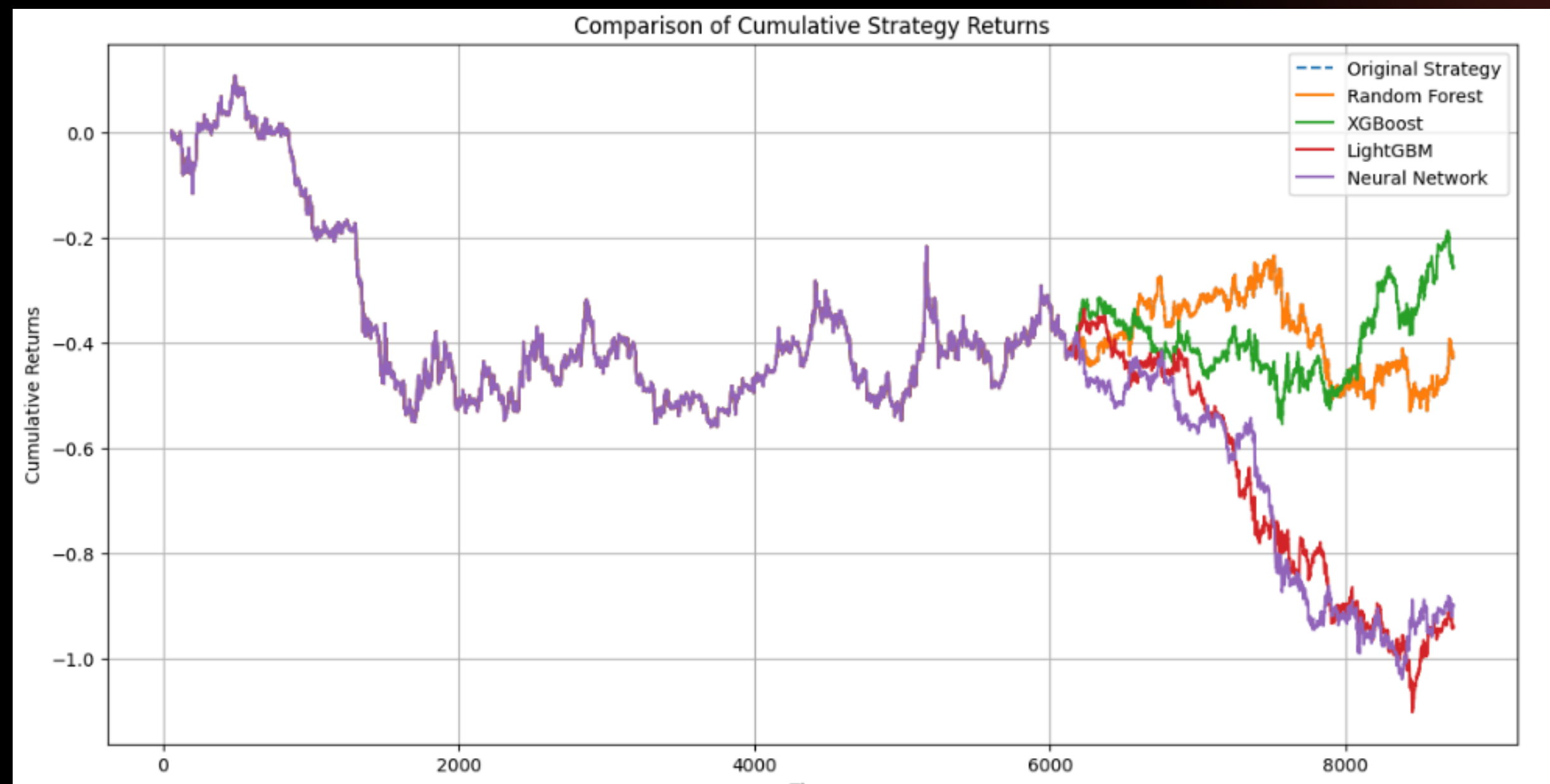
Use a dynamic cutoff — only trade the top 20% strongest signals; it reacts only to high-confidence setups.







# Machine Learning

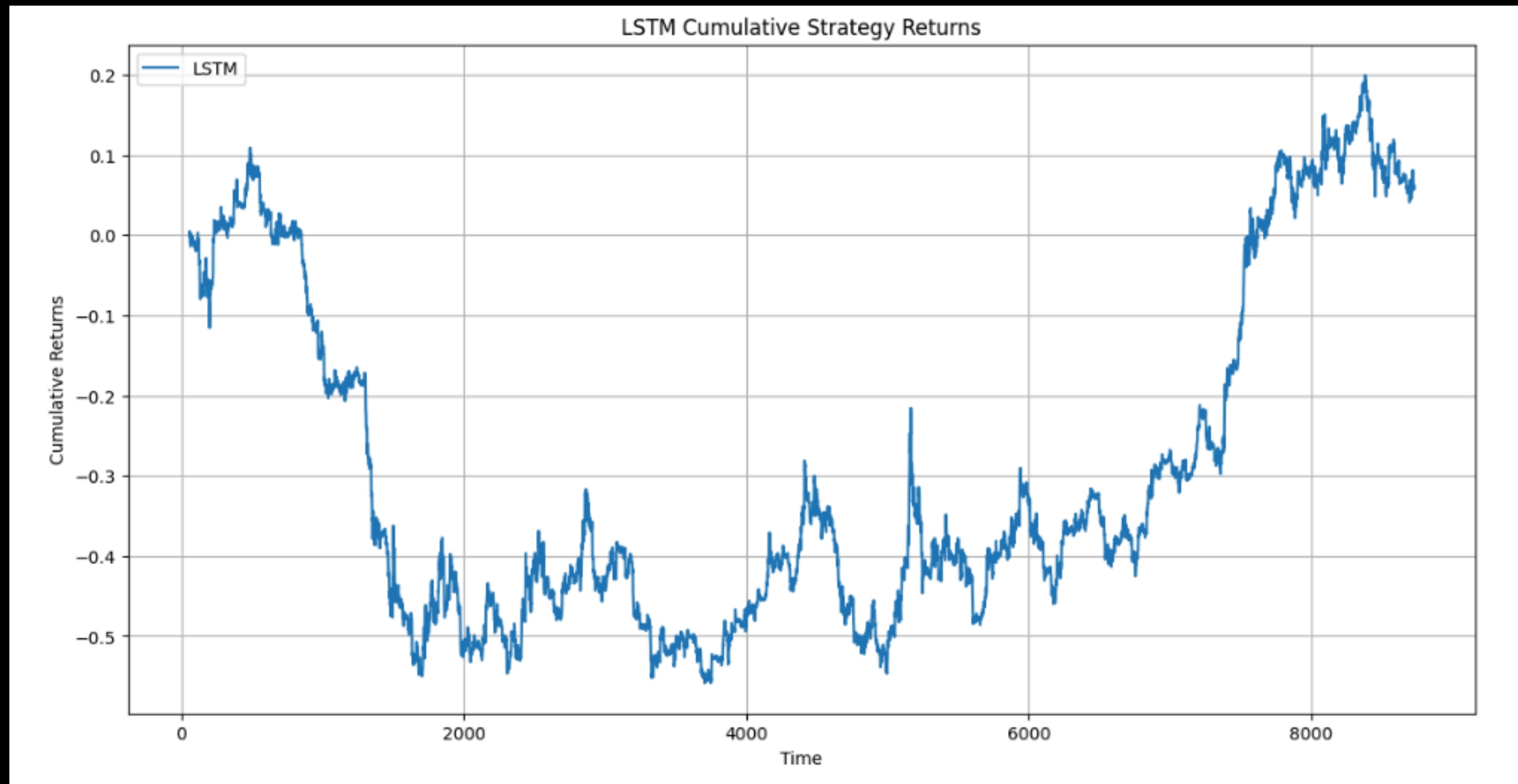


```
RandomForestClassifier Accuracy: 0.5186753946861764  
XGBoost Accuracy: 0.5097963887821744  
LightGBM Accuracy: 0.4978870533999232  
Neural Network Accuracy: 0.48021513638109875
```





# Chooosen Model



LSTM Accuracy: 0.5196153846153846

# What Our Model do?

- **Learns from past 10 time steps of flow + momentum features.**
- **Predicts whether BTC price will rise next time step.**
- **Generates binary trading signals.**
- **Simulates a basic long-short strategy.**
- **Plots cumulative P&L to evaluate performance.**



**THANK  
YOU**

