WiDS Kalman Filtered Trend Trader: Assignment 2

Aarav Malde and Krishang Krishna

16th December, 2025

Problem Statement

Financial markets exhibit non-stationary dynamics where relationships between price, momentum, volatility, and other market variables evolve over time. Traditional static models often fail to adapt to such time-varying behavior. In this assignment, you will design, implement, and evaluate a **trading strategy** for a stock using **Kalman Filters**.

You are required to develop a **trading strategy for Microsoft Corp. (MSFT)** using historical daily market data. The strategy should combine **feature-based machine learning prediction** with **Kalman filtering** to estimate time-varying latent parameters governing the stock's price over time. Trading decisions must be generated algorithmically and evaluated via backtesting.

Key Objectives

1. Model the time-varying relationship between MSFT's price and engineered market features using a Kalman Filter.
2. Integrate machine learning techniques (any of your choice, but avoid Reinforcement Learning) to predict future price movements or returns.
3. Design a rule-based trading strategy using model predictions.
4. Backtest the strategy on historical data and evaluate its performance using quantitative metrics.

Steps to be Followed

Step 1: Fetch Historical Stock Data

Obtain stock price data for **Microsoft Corp. (MSFT)** over a sufficiently long period (e.g., 2015–2024) from Yahoo Finance.

Step 2: Feature Engineering

Construct a feature set that may include, but is not limited to:

- Moving Averages (e.g., 5-day, 20-day, 60-day averages)
- Log returns and lagged returns
- Rate of Change (ROC)
- Rolling volatility measures
- Moving averages and momentum indicators
- Volume-based features

Step 3: Kalman Filter Model

Formulate a **state-space model** where, the latent state represents time-varying parameters (e.g., regression coefficients relating features to price or returns). The observation equation links observed prices or returns to the feature set and the state transition equation allows model parameters to evolve over time.
Use the Kalman Filter to recursively estimate latent parameters and generate filtered and/or predicted prices or returns. Clearly define the State vector, Observation matrix, Process noise assumptions, Observation noise assumptions in your report.

Step 4: Machine Learning Integration

Use Kalman-filtered states as inputs to a machine learning model (e.g., linear regression, ridge regression, tree-based model, any model of your choice) to predict the future price ratio of MSFT.
Clearly explain the modeling choice and discuss its advantages in your report.

Step 5: Generate Trading Signals

Use the predicted ratio to generate buy/sell signals:

- **Buy Signal**: If the predicted ratio is significantly higher than the current ratio.
- **Sell Signal:** If the predicted ratio is significantly lower than the current ratio.

Ensure that the Long/Short signals are derived from predicted returns or prices, you can consider implementing threshold-based entry and exit rules. Create risk constraints such as maximum exposure or stop-loss rules. Ensure that your signals aren't causal i.e. your signals should not be based of decisions made with data from the future. Ensure its based only on the present and the past.

Step 6: Simulate the Trading Strategy

Simulate the strategy over the historical period:

- Take positions based on the trading signals.
- Calculate profit and loss (PnL) for each trade.
- Accumulate the PnL to evaluate overall performance.

Step 7: Backtesting and Evaluation

Backtest the trading strategy over the historical dataset:

- Simulate the trading strategy over the historical dataset using the generated signals.
- Calculate profit and loss (PnL) for each trade.
- Evaluate the strategy using quantitative performance metrics, including Cumulative return, Sharpe ratio, Maximum drawdown, Win/loss ratio and anything else you feel like can be interpreted to improvise yout strategy.

Also Compare the strategy's performance against a **buy-and-hold benchmark** for MSFT.

Deliverables

1. Well-documented Python code implementing data processing, the Kalman Filter, the machine learning model, and the trading simulation. Please comment your code properly so that it's easy for me to understand what you've done.
2. A concise pdf report explaining what you have done, assumptions taken, trading logic, and performance analysis. Include results and thoughts.
3. Include visualizations on Kalman-filtered parameters over time, Trading signals and Equity curve of the strategy. Feel free to include as many plots as you feel necessary to express your performance.

Hints

1. Use the Python library pykalman for Kalman Filtering.
2. Ensure your backtesting framework accounts for transaction costs.
3. Use a rolling window to calculate the spread's mean and standard deviation dynamically.

May The Profit Be With You!