Automated Algorithmic Trading Using Multi-Level Return Predictions

Ben Caldwell
MSDS 451: Financial Machine Learning

August 12, 2025

Abstract

This report outlines an algorithmic trading strategy built using multi-level return predictions and variable risk tolerances. Strategies are driven by both machine learning models (Random Forest and LSTM) and rule-based options logic. Using ExxonMobil (XOM) as the underlying asset, I developed and backtested five strategies from 2010–2025, focusing on trade automation rather than predictive performance. My backtest includes realistic constraints such as options premium decay and volatility filters. Sharpe ratio, drawdown, and cumulative return are used to evaluate effectiveness.

1 Problem Description

The goal of this assignment is to develop a fully automated trading algorithm that makes buy/sell/hold decisions using signals derived from multi-level return predictions. These predictions may stem from a mean-reversion, momentum, or machine learning-based model. The algorithm adapts to a trader's risk tolerance and executes long, short, or options-based trades accordingly. Unlike previous assignments, the emphasis is on trading automation and performance analysis rather than model training.

2 Data Preparation and Pipeline

I used daily stock data from Yahoo Finance for ExxonMobil (XOM) from January 1, 2010 to January 1, 2025. The features engineered include:

- Simple and exponential moving averages (SMA_10, EMA_20)
- Bollinger Bands (Upper/Lower)
- Momentum and rolling standard deviation
- Relative Strength Index (RSI)

Signals were labeled based on future returns using multi-level thresholds corresponding to risk levels.

3 Research Design

I implemented and compared five distinct strategies:

1. **Random Forest (RF)** — Supervised classification model predicting trade signals
2. **LSTM** — Deep learning model using temporal dependencies to predict signals
3. **Options/Short (Rule-Based)** — Executes options-based logic with short selling based on labeled signals
4. **Options/Short Realistic** — Adds realistic trade constraints (e.g., avoiding back-to-back directional changes)
5. **Options with Premium Decay** — Includes exponential options premium decay and volatility thresholds using VIX data

All strategies were backtested using a rolling portfolio logic and evaluated using key metrics.

4 Programming

The program was developed in Python using modular, reusable functions. The pipeline includes:

- A feature engineering module with technical indicators
- LSTM and Random Forest models for signal generation
- Backtesting engines for traditional and options-based strategies
- Performance evaluation using returns, volatility, Sharpe ratio, and drawdown
- Realistic logic for options premium decay and volatility gating

Code is organized for reproducibility and flexibility across tickers and timeframes. The project uses libraries such as `yfinance`, `scikit-learn`, `keras`, `numpy`, and `matplotlib`.

Strategy	Cumulative Return	Annualized Return	Volatility	Sharpe Ratio	Max Drawdown
Random Forest	38.53x	27.79%	13.84%	1.84	-33.78%
LSTM	0.90x	-0.71%	9.75%	-0.02	-36.06%
Options Realistic	17.36B	387.10%	48.83%	3.49	-34.76%
Options Decay	0.003x	-32.03%	8.11%	-4.72	-99.68%
Buy & Hold	2.83x	7.23%	25.01%	0.40	-62.40%

Table 1: Strategy Performance Summary (XOM: 2010–2025)

5 Performance Metrics

6 Discussion

The results reveal a stark contrast between model-driven and rule-based strategies. The Random Forest strategy outperformed the buy-and-hold baseline, achieving a Sharpe ratio of 1.84 with moderate drawdowns. The LSTM model failed to generalize, producing poor returns and higher risk. The naive options strategy with idealistic execution yielded astronomical returns, indicating unrealistic assumptions.

The realistic options strategy, which incorporated entry/exit rules and reduced overtrading, performed impressively well, returning over 17 billion dollars on a $10,000 investment. However, the volatility and high Sharpe ratio raise questions about slippage, market impact, and survivorship bias.

By contrast, the options decay strategy, which incorporated premium erosion and VIX filters, underperformed dramatically, demonstrating the difficulty of profiting from options without precise timing and volatility modeling.

7 Conclusion

This assignment highlights the power and pitfalls of automated trading strategies. While model-driven signals can offer alpha, incorporating trading frictions like premium decay and volatility is crucial for realism. Going forward, improvements could include transaction cost modeling, better volatility forecasting, and multi-asset portfolios.

References

- Hull, J. C. (2017). *Options, Futures, and Other Derivatives.* Pearson Education.
- Chan, E. (2013). *Algorithmic Trading: Winning Strategies and Their Rationale.* Wiley.
- yFinance API: `https://pypi.org/project/yfinance/`
- Scikit-learn Documentation: `https://scikit-learn.org`
- Keras Documentation: `https://keras.io`