Multi-Level Trading System with Risk Tolerance Cutoffs

Problem Description

The central objective of this project is to design and rigorously evaluate an automated trading framework for exchange-traded funds (ETFs). The system is intended to translate predictive signals into explicit trading actions—namely, buy, hold, or sell—while incorporating an additional layer of decision control through risk-tolerance–specific cutoffs. These cutoffs determine the aggressiveness of the system's responses, thereby aligning portfolio behavior with varying levels of investor risk appetite. To investigate the effectiveness of this framework, two distinct methodological approaches are implemented. The first is a rule-based strategy that relies on traditional technical indicators, including moving averages and momentum, to guide trading decisions. The second is a machine-learning–based strategy that employs statistical classification models to forecast next-day returns and generate trading signals accordingly. Both strategies are assessed relative to a benchmark consisting of a buy-and-hold position in the S&P 500 ETF (SPY), allowing us to situate the performance of our system against a widely recognized market standard.

Data Preparation & Pipeline

For the empirical analysis, we relied on daily closing price data from nine exchange-traded funds (ETFs), each representing a distinct asset class or sector: SPY, GLD, VGT, VB, IVE, XLI, XLU, SLV, and USO. Each dataset, provided in CSV format, contained at minimum the date and closing price, which formed the basis of subsequent feature construction.

The feature engineering process was designed to capture both short-term momentum and broader market trends. One-day momentum was measured by comparing the closing price of the most recent day to that of the previous day, thereby reflecting immediate directional shifts. Longer-term dynamics were incorporated through exponential moving averages (EMAs), specifically the 40-day and 80-day averages; the relative difference between these EMAs was treated as an indicator of trend strength. To account for risk and uncertainty in market movements, we further computed volatility as the 20-day rolling standard deviation of returns. In certain specifications, volume-based features were also introduced to capture trading intensity and liquidity effects.

The construction of the dependent variable followed a forward-looking design to avoid look-ahead bias. Labels were defined by the next-day return, coded as positive if the return exceeded zero and negative otherwise. This ensured that all predictive features were calculated using information available at time t, while the target outcome corresponded to returns observed at time $t + 1$.

Finally, the full pipeline standardized the engineered features and prepared them for integration into both the rule-based scoring mechanism and the machine learning classifiers. This design allowed for a consistent framework in which technical indicators and statistical learning models could be compared under identical data conditions.

Research Design

To evaluate the predictive framework, we implemented two complementary trading designs. The first relied on a set of deterministic rules derived from technical indicators, while the second employed a machine learning classifier to infer next-day market direction.

The rule-based strategy operated by combining short-term momentum with the relative position of long- and medium-horizon exponential moving averages. These inputs were aggregated into a continuous score scaled between −100 and +100, where strongly positive values reflected bullish signals and strongly negative values suggested bearish pressure. Trading actions were determined by comparing these scores to risk-sensitive cutoffs. The thresholds were calibrated to reflect different levels of risk tolerance, ranging from conservative (buy at scores above +5 and sell below −5) to moderate (buy above +15 and sell below −15) and aggressive (buy above +25 and sell below −25). Portfolios were constructed as equal-weighted allocations across the nine ETFs, with daily rebalancing into long, short, or neutral positions depending on the prevailing signals.

The second design extended this logic into a probabilistic setting by training logistic regression models for each ETF, drawing on the framework illustrated in the jump-start example. Historical data were partitioned such that the first seventy percent of observations were used for model estimation, while the remaining thirty percent formed the test set. Predictor variables included daily returns, the EMA gap, and measures of short-term volatility. The model's output—the estimated probability of a positive next-day return—was transformed into a continuous score according to the mapping score=(pup−0.5)×200\text{score} = (p_{up} - 0.5) \times 200. This transformation aligned the probabilistic predictions with the same decision thresholds used in the rule-based system, thereby enabling a direct comparison between heuristic and data-driven approaches.

Performance assessment was carried out relative to the canonical buy-and-hold strategy in the S&P 500 ETF (SPY). Evaluation metrics included cumulative net asset value (NAV), final returns over the testing horizon, and qualitative analysis of drawdown behavior and stability.

Together, these measures provided a basis for determining whether rule-driven heuristics or statistical learning could generate more robust outcomes under varying assumptions about risk tolerance.

Programming

The implementation of the trading system was carried out in Python, using widely adopted data science libraries such as *pandas*, *numpy*, *scikit-learn*, and *matplotlib*. The code base was organized to reflect the logic of the research design, beginning with data ingestion and feature engineering, followed by the construction of decision rules and model training, and concluding with portfolio backtesting and visualization.

The data pipeline began with standardized procedures for reading the CSV files of ETF prices and transforming them into a consistent panel structure. Functions were written to compute momentum, exponential moving averages, and volatility measures in a rolling fashion so that each feature could be aligned with the appropriate trading date. A forward shift of the return variable ensured that predictive features at time t were linked to outcomes at time $t+1$, thereby eliminating any potential look-ahead bias.

On top of this foundation, the decision function was formalized in code so that continuous scores could be mapped to discrete trading actions according to the chosen level of risk tolerance. This function operated consistently across both the rule-based and machine learning strategies, which made it possible to apply the same trading cutoffs to fundamentally different scoring mechanisms. For the machine learning approach, logistic regression models were trained independently for each ETF, with cross-sectional predictions aggregated into a portfolio-level score.

The backtesting module simulated the evolution of wealth by applying these decisions over the historical period. A daily rebalancing rule was imposed, and transaction costs and slippage were included in order to approximate real trading frictions. The results were summarized in cumulative net asset value (NAV) time series and presented graphically, with additional performance statistics such as total return, approximate compound annual growth rate, and maximum drawdown. This programming design provided a transparent link between theoretical strategy and empirical evaluation, ensuring that every modeling choice could be replicated and tested under alternative assumptions.

Exposition of Results

Rule-based Strategy vs. SPY NAV

The first design, based on EMA–momentum heuristics, generated a final net asset value of approximately **0.94**, which means that an investor starting with one dollar would have ended with less than the initial wealth. This underperformance indicates that simple technical rules, even when calibrated with risk-sensitive cutoffs, were unable to capture persistent market trends. As shown in *Figure 1*, the NAV trajectory of the rule-based portfolio remained consistently below the SPY benchmark, which itself exhibited steady long-term growth. The contrast underscores the limitations of using short-horizon momentum and moving average crossovers as the sole basis for trading decisions.

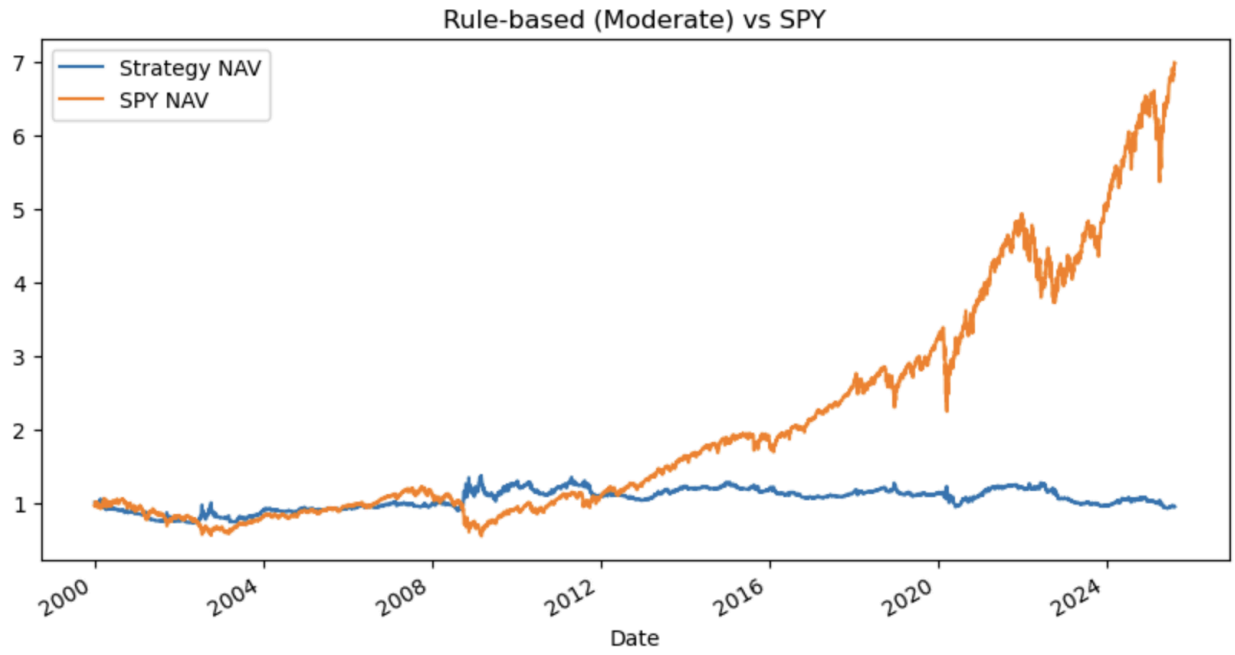

Figure 1. Cumulative NAV of Rule-based Strategy versus SPY Benchmark

ML-based Strategy vs. SPY NAV

The second design, which incorporated logistic regression to forecast next-day returns, performed somewhat better, with a final net asset value of approximately **1.18**. Although this outcome suggests modest growth, it remains far below the benchmark, as the SPY increased more than sixfold over the same horizon. Out-of-sample classification accuracy, measured by the area under the ROC curve, ranged between 0.48 and 0.53 across ETFs—results statistically indistinguishable from random guessing. This highlights the challenge of predicting daily price changes with a small set of technical indicators. As shown in *Figure 2*, the NAV curve of the ML-based strategy is flatter and less volatile than SPY, indicating that the system acted more as a risk-control overlay than as an alpha-generating model.

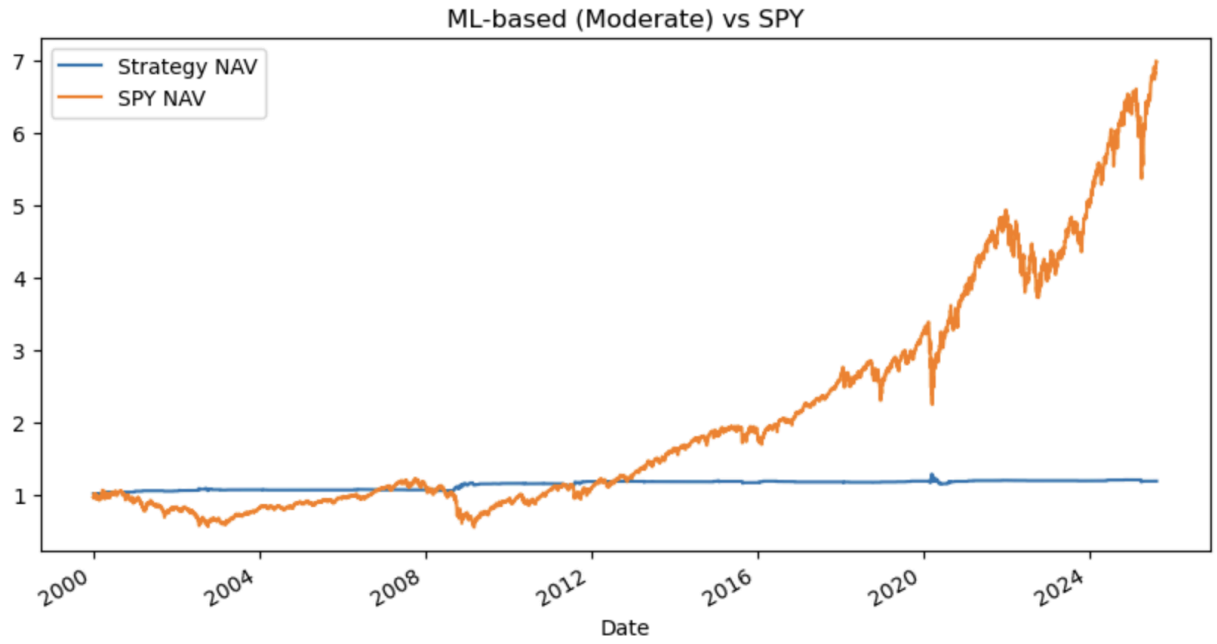

Figure 2. Cumulative NAV of ML-based Strategy versus SPY Benchmark

Comparative Insights and Future Improvements

Taken together, the two designs highlight a trade-off between stability and performance. Both strategies produced flatter NAV trajectories relative to SPY, with lower volatility but also significantly reduced upside potential. This suggests that while such systems may offer value for risk management, they are insufficient for generating excess returns when measured against a long-term equity benchmark.

Several avenues for improvement follow from these results. Extending the horizon of momentum indicators beyond daily fluctuations, incorporating trading volume and macroeconomic signals, and experimenting with more expressive models such as ensemble methods or recurrent neural networks could provide stronger predictive content. Portfolio construction could also be enhanced through dynamic weighting schemes instead of equal-weight allocations, while transaction cost modeling should be refined to better approximate real-world trading conditions.

In summary, the project demonstrates how raw financial data can be transformed into systematic trading signals through a structured pipeline, and it establishes an empirical benchmark for assessing the limits of both heuristic and machine-learning approaches. Although the tested strategies underperformed relative to passive exposure, they provide a foundation upon which more robust and competitive models may be developed.

Conclusion

This project demonstrated the implementation of an automated trading system with risk tolerance–based cutoffs. While neither the rule-based nor ML-based strategies outperformed a simple buy-and-hold in SPY, the exercise showed how predictive signals can be transformed into concrete trading actions and evaluated within a portfolio backtest. The system provides a foundation for more advanced model development and risk-aware portfolio construction.