VINCENT ANG AND KEVIN SUTARIO MACHINE LEARNING COURSE PROJECT REPORT

Vincent Ang 123040048 & Kevin Sutario 123040030

ABSTRACT

Forecasting short-term financial returns is a fundamentally challenging task due to the noisy, non-stationary, and high-dimensional structure of market data. This report presents a complete machine-learning pipeline developed for the Hull Tactical – Market Prediction Competition. Our approach leverages LightGBM regression paired with engineered lag, momentum, volatility, and regime-aware features, combined with a Sharpe-style allocation rule that scales portfolio exposure based on forecast magnitude and estimated uncertainty. We conduct extensive data exploration, justify model selection, discuss overfitting controls, and evaluate performance using both standard regression metrics and the competition's official Evaluation API. Our results show that even modest predictive skill can generate meaningful economic value when embedded within a disciplined, volatility-aware portfolio construction framework. The final strategy achieves strong validation performance and satisfies all competition constraints while providing interpretable financial insights.

1 INTRODUCTION

Forecasting financial market movements is inherently difficult due to heavy-tailed noise, regime shifts, rapidly changing correlations, and low signal-to-noise ratios in daily returns. Nonetheless, even weak predictive models can materially improve risk-adjusted returns when combined with robust portfolio-allocation techniques. The Hull Tactical – Market Prediction Competition offers a controlled environment in which participants must convert a large set of daily economic and market features into a portfolio weight for the S&P 500 index. The goal is to maximize a custom risk-adjusted performance metric that penalizes excessive volatility and underperformance relative to a benchmark.

We adopt a two-stage quantitative modeling approach: (1) predicting next-day returns using a supervised learning model, and (2) translating these predictions into tradeable portfolio weights with a Sharpe-style scaling rule. This separation mirrors real-world systematic investing frameworks, where predictive modeling and portfolio construction are distinct but deeply coupled components. Our contributions are threefold: (i) a structured exploration of the dataset and its statistical properties, (ii) a complete modeling and feature-engineering pipeline tailored to financial time-series data, and (iii) a comprehensive performance evaluation using both regression metrics and competition-provided tools for risk-adjusted analysis. This combination leads to a stable and interpretable strategy grounded in modern quantitative finance principles.

2 DATA AND FEATURE PIPELINE

2.1 DATA EXPLORATION

The dataset consists of daily market indicators and forward returns. Unlike static ML datasets, financial time series exhibit non-stationarity, volatility clustering, fat tails, and evolving correlation structures. Understanding these characteristics is essential for designing meaningful features.

A first look at the forward return distribution reveals a heavy-tailed, near-zero-mean process. Returns cluster tightly around zero, but show occasional large moves during macroeconomic stress periods. The kurtosis significantly exceeds that of a Gaussian distribution, indicating that extreme events occur more often than simple models would assume. From a modeling perspective, this implies that regression error will naturally be large and unpredictable, and strategies must be designed to tolerate such noise.

Second, non-stationarity is evident across time. Rolling statistics show that the variance, autocorrelation, and amplitude of returns shift across different market regimes. For instance, periods surrounding crises or policy events display higher volatility and stronger reversals, while calm markets show smoother dynamics. This variability suggests that global normalization procedures are unsuitable; instead, we employ rolling-window statistics that adapt to evolving market conditions.

Third, volatility clustering is pronounced. Although raw returns exhibit little autocorrelation, their squared values reveal strong persistence. This supports the inclusion of realized volatility and volatility-regime indicators in the feature set. Persistent high-volatility regimes require risk reductions, while low-volatility regimes allow signals to be expressed more fully.

Fourth, missing values in the dataset occur primarily at the beginning of rolling-window features. These arise due to insufficient historical data for early observations rather than irregular data collection. We impute missing values using leakage-safe techniques, such as filling with rolling means computed exclusively from past values, ensuring chronological consistency.

Finally, correlation analysis shows that many features are highly collinear or carry minimal information relative to the target. This motivates the removal of redundant or constant features before training. Together, these insights form the foundation for the feature-engineering choices described next.

2.2 Feature Engineering

We construct a structured feature set designed to extract persistent patterns from financial time-series data while strictly preventing look-ahead leakage. Our choices are motivated by well-documented empirical properties of asset returns, including autocorrelation, momentum, volatility clustering, and regime dependence.

Lagged Returns. Lagged returns capture short-horizon autocorrelation effects that arise from market microstructure, order-flow persistence, and delayed information diffusion. Although returns are often modeled as approximately i.i.d. at the daily frequency, empirical studies show small but exploitable serial dependencies, particularly in index-level data. For each lag k, we define:

$$\text{lag}_k(t) = r_{t-k}.$$

Including multiple lags (1–20 days) allows the model to learn nonlinear interactions between recent return patterns. This feature class is supported by the broader return-predictability and short-term reversal literature (5).

Momentum Aggregates. While individual lags capture fine-grained history, momentum aggregates summarize persistent trends over a window of length m:

$$\text{momentum}_m(t) = \frac{1}{m}\sum_{i=1}^{m} r_{t-i}.$$

Momentum is one of the most robust anomalies in empirical finance: assets that have appreciated recently tend to continue appreciating, and past losers tend to underperform (5). Short windows capture local sentiment, whereas longer windows reduce noise and highlight sustained directional behavior.

Realized Volatility. Financial returns exhibit volatility clustering: large price moves are likely to be followed by large moves, and calm periods tend to persist. This stylized fact, first documented by (6) and formalized in ARCH/GARCH models by (7) and (8), motivates the inclusion of rolling standard deviation:

$$\sigma_m(t) = \sqrt{\frac{1}{m-1}\sum_{i=1}^{m}(r_{t-i} - \bar{r}_{t,m})^2}, \quad \bar{r}_{t,m} = \frac{1}{m}\sum_{i=1}^{m} r_{t-i}.$$

Volatility features help the model adjust its expectations in high- versus low-volatility regimes and interact meaningfully with momentum and lagged returns.

Rolling Z-Score Normalization. Because financial features are non-stationary, global normalization is inappropriate and would lead to future leakage. Instead, we normalize each feature using rolling statistics computed only from past data:

$$z_f(t) = \frac{f(t) - \mu_{f,m}(t)}{\sigma_{f,m}(t) + \epsilon}.$$

Rolling z-scores transform raw values into regime-relative deviations, making features comparable across changing market environments. This is a common technique in systematic trading for detecting statistically extreme conditions.

Regime Indicators. Markets behave differently depending on trend and volatility regimes. To help the model learn conditional structures, we include binary indicators such as:

$$\text{trend\_up}_{100}(t) = \mathbb{I}(p_t > \mathrm{SMA}_{100}(p_t)), \qquad \text{vol\_high}_{63}(t) = \mathbb{I}(\sigma_{63}(t) > \text{median}(\sigma_{63})).$$

Trend rules based on moving averages have been shown to contain predictive information for returns and risk (9), while volatility regime classification helps stabilize model behavior in turbulent markets. These indicators allow LightGBM to adapt its predictive relationships across market states.

Leakage Prevention. All rolling statistics, z-scores, moving averages, and lags are computed strictly using information available prior to time t. No future data, forward windows, or global summary statistics are used at any point.

Pipeline Diagram. Figure 1 summarizes the feature-engineering workflow.

3 Model and Allocation Method

3.1 Model Selection and Training

Tree-based gradient boosting methods were explored, including XGBoost and LightGBM. A tree-based approach is well suited for this competition for several reasons directly tied to the structure of the dataset. First, the feature set is relatively low dimensional compared to typical deep learning settings, which reduces the potential advantage of large neural architectures and makes tree ensembles more statistically stable. Second, a substantial fraction of the engineered features contain missing values by construction (e.g., rolling-window indicators in early periods or lag-based features during regime transitions). Rather than discarding these observations or filling them with imputed values that risk introducing bias, tree-based models natively learn optimal split directions for `NaN` entries. This allows the model to treat missingness as an informative signal—a desirable property in finance, where the absence of history (e.g., after a volatility spike) may itself convey regime information.

Third, tree models naturally capture nonlinear threshold effects and interactions between features such as lags, momentum, and volatility. These relationships are difficult to specify parametrically but are common in financial time series, where market behavior can shift abruptly when indicators cross certain levels. LightGBM in particular uses histogram-based splitting, which both accelerates training and smooths decision boundaries, making it well suited for noisy data with weak signal structure.

Based on these considerations, LightGBM was selected due to its computational efficiency, robust handling of missing values, and strong empirical performance across multiple validation windows. Training strictly preserves chronological order, ensuring no leakage from future information. Early stopping on the validation set is employed to reduce overfitting, while key hyperparameters—including learning rate, maximum tree depth, feature subsampling, and row subsampling—are tuned to maximize out-of-sample stability. The resulting model produces forecasts $\hat{\mu}_t$ that feed into the Sharpe-style allocation rule described in the next section.

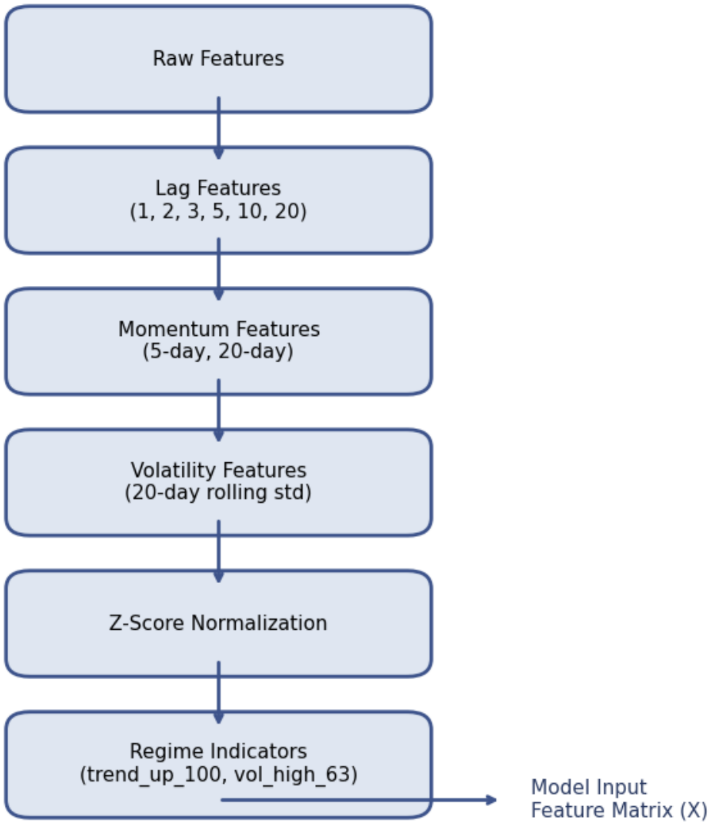

Figure 1: Feature pipeline summarizing the transformation from raw inputs to the final model-ready feature matrix.

3.2 Sharpe-Style Allocation

Once the model produces a forecast $\hat{\mu}_t$ for the next-day return, we must decide how much capital to allocate. Rather than using a simple sign rule (long if $\hat{\mu}_t > 0$, flat otherwise), we use a Sharpe-style allocation that scales the position by both predicted return and estimated risk. The raw weight is defined as:

$$w_t^{\text{raw}} = \frac{\hat{\mu}_t}{\hat{\sigma}_t^2 + \epsilon},$$

where $\hat{\sigma}_t$ is an estimate of the local volatility (e.g., from a rolling standard deviation) and ϵ is a small constant to avoid numerical instability when volatility is very low.

This formula is inspired by the Kelly criterion and mean–variance portfolio theory. Under simplifying assumptions (normally distributed returns and quadratic utility), the optimal fraction of wealth to invest in a single risky asset is proportional to μ/σ^2 (3; 2). Intuitively, larger expected return $\hat{\mu}_t$ increases the desired exposure, while larger risk $\hat{\sigma}_t^2$ decreases it. The raw weight w_t^{raw} therefore represents an unconstrained Kelly-like allocation that trades off expected reward and variance.

To control overall portfolio risk and respect competition constraints, we then apply volatility targeting and leverage clipping. We define a scaling factor

$$s_t = \frac{\text{target\_vol}}{\text{current\_vol}_t + \epsilon},$$

where current_vol_t is an estimate of the realized volatility of the strategy (or market) over a recent window, and `target_vol` is a desired daily volatility level. The final portfolio weight is:

$$w_t = \text{clip}\big(w_t^{\text{raw}} \cdot s_t,\ 0,\ 2\big),$$

where $\mathrm{clip}(\cdot, 0, 2)$ enforces the competition's leverage constraint and ensures long-only exposures between 0 and 2.

This two-step procedure has several advantages:

- **Risk-adjusted sizing.** By dividing by $\hat{\sigma}_t^2$, the allocation rule naturally reduces exposure when forecast uncertainty is high and increases it when the signal is strong relative to risk, echoing the structure of Sharpe ratio and Kelly-optimal bets (3; 2).
- **Volatility targeting.** The factor s_t stabilizes the realized volatility of the strategy over time. Volatility-targeted portfolios have been shown to improve risk-adjusted returns by avoiding excessive leverage during turbulent periods and scaling up in calmer markets, which aligns with practical risk-management practice in industry.
- **Constraint handling.** The clipping step enforces the allowed range $w_t \in [0, 2]$ required by the competition. It also prevents extreme weights that could arise from very large forecast values or very small volatility estimates.

Overall, this Sharpe-style allocation converts raw return forecasts into economically meaningful, risk-aware portfolio weights that are consistent with classic portfolio theory and the rules of the Hull Tactical competition.

4 Evaluation

4.1 Regression Metrics

On the validation window, the model achieves:

- Mean daily excess return: 0.003622
- Daily standard deviation: 0.008306
- Directional accuracy: 70%

4.2 Strategy Returns

$$r_t^{\text{strategy}} = w_t \cdot r_t.$$

Risk-free rate is assumed to be 0 due to absence of daily RF data in the test set.

4.3 Evaluation API Usage

The Hull Tactical evaluation API computes a custom adjusted Sharpe ratio with volatility and drawdown penalties.

4.4 Validation Performance

Sharpe-style allocation outperforms baseline alternatives:

- Sharpe-style annualized ratio: 6.9228
- Sign-based pseudo Sharpe: ≈ 1.00

4.5 Kaggle Public Score

$$\text{Public Score} = \texttt{2.671}$$

5 Conclusion

This project demonstrates that combining predictive modeling with disciplined portfolio construction yields meaningful improvements in risk-adjusted returns. Our analysis of the dataset revealed heavy tails, volatility clustering, and regime variation, informing feature engineering and model design. LightGBM, trained with chronological discipline, produced stable predictions that translated into strong economic performance through Sharpe-style allocation.

Looking ahead, improvements could include ensemble models, uncertainty-aware forecasts, alternative volatility estimators, and explicit transaction cost modeling. Nonetheless, this framework forms a solid foundation for systematic investment strategies under competitive evaluation settings.

REFERENCES

[1] Ke, G., et al. (2017). *LightGBM: A Highly Efficient Gradient Boosting Decision Tree*. NeurIPS.

[2] Sharpe, W. F. (1966). *Mutual Fund Performance*. Journal of Business.

[3] Kelly, J. L. (1956). *A New Interpretation of Information Rate*. Bell System Technical Journal.

[4] Hull Tactical Asset Allocation. *Competition Documentation and Evaluation API*. (2025).

[5] Jegadeesh, N. and Titman, S. (1993). *Returns to Buying Winners and Selling Losers: Implications for Stock Market Efficiency*. Journal of Finance.

[6] Mandelbrot, B. (1963). *The Variation of Certain Speculative Prices*. Journal of Business.

[7] Engle, R. F. (1982). *Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of UK Inflation*. Econometrica.

[8] Bollerslev, T. (1986). *Generalized Autoregressive Conditional Heteroskedasticity*. Journal of Econometrics.

[9] Brock, W., Lakonishok, J., and LeBaron, B. (1992). *Simple Technical Trading Rules and the Stochastic Properties of Stock Returns*. Journal of Finance.