Certificate in Quantitative Finance

Final Project Report

Blending Ensemble for Classification

Predicting SPY ETF 5-Day Return Direction

Project Code: ML

Author: LIU Jing Zhong

Cohort: June 2025

Contents

List of Figures

List of Tables

1 Executive Summary

This project implements a Blending Ensemble classifier to predict the 5-day forward return direction of the SPY ETF. The model architecture integrates five diverse base learners—Logistic Regression, Random Forest, LightGBM, XGBoost, and Gradient Boosting—combined through a Logistic Regression meta-learner using the blending technique.

The investigation yielded a stark contrast between the model's predictive capability and the trading strategy's performance. In terms of classification, the Blending Ensemble failed to extract a meaningful directional signal, achieving an AUC of 0.5003, which is effectively equivalent to random guessing. A subsequent signal-to-noise ratio analysis clarified the root cause: 100% of the engineered features exhibited negligible correlation with the target (maximum $|r| = 0.0438$), confirming the extremely high efficiency and noise levels inherent in the SPY ETF.

Despite the lack of predictive power at the classifier level, the integrated trading strategy demonstrated robust performance. By combining the ensemble's signals with a 200-day Moving Average trend filter, the strategy achieved a Sharpe Ratio of 1.427 and a Total Return of 403.35%, significantly outperforming the market benchmark's Sharpe Ratio of 0.455. The Maximum Drawdown was successfully contained at 28.59% compared to the market's 73.53%.

The study concludes with a critical insight into quantitative strategy design: the strategy's profitability was driven not by the machine learning model's alpha, but by the systematic risk management provided by the trend filter. This highlights that in low-signal-to-noise environments characteristic of financial markets, simple rule-based risk management strategies can be more effective than complex machine learning models for generating trading alpha.

2 Introduction

2.1 Problem Statement

Predicting stock market direction is a fundamental challenge in quantitative finance that has attracted significant attention from both academia and industry. Traditional econometric models, such as ARIMA and GARCH, have long been employed for financial time series forecasting, yet their linear assumptions often fail to capture the complex, non-linear dynamics inherent in market data. This limitation has motivated the exploration of machine learning techniques, which offer greater flexibility in modeling non-linear relationships.

Among machine learning approaches, ensemble methods have emerged as particularly promising candidates for financial prediction. By combining multiple models, ensemble methods can potentially reduce variance, mitigate overfitting, and capture diverse patterns in the data. However, the effectiveness of these methods in the context of highly efficient markets remains an open question. This project investigates whether a Blending Ensemble—a specific ensemble technique that uses a holdout set for meta-learner training—can effectively predict short-term price movements in one of the world's most liquid and efficient markets: the SPY ETF.

2.2 Objective

The primary objective of this project is to construct and evaluate a Blending Ensemble classifier for predicting the 5-day forward return direction of the SPY ETF. The study pursues several specific aims. First, it seeks to implement a robust Blending Ensemble architecture with five diverse base learners, ensuring proper data handling to prevent look-ahead bias. Second, the project aims to engineer a comprehensive feature set spanning technical indicators, momentum signals, volatility measures, and macroeconomic factors. Third, the research evaluates model performance using both classification metrics (AUC, confusion matrix, MCC) and practical trading metrics (Sharpe ratio, maximum drawdown, profit factor). Fourth, and perhaps most importantly, the study investigates the fundamental limitations of applying ensemble methods to financial time series, with particular focus on analyzing the signal-to-noise characteristics that distinguish financial data from other machine learning domains.

2.3 Asset Selection

The SPDR S&P 500 ETF Trust (SPY) was selected as the underlying asset for this study based on several important considerations. As one of the most actively traded ETFs globally, with average daily trading volume exceeding 70 million shares, SPY provides exceptional liquidity. This high liquidity minimizes execution risks such as slippage and ensures that backtest assumptions regarding trade execution at closing prices are realistic and achievable in practice.

Furthermore, because SPY tracks the S&P 500 index—a capitalization-weighted benchmark comprising 500 of the largest U.S. publicly traded companies—it represents one of the most informationally efficient markets in the world. According to the Efficient Market Hypothesis (EMH), prices in such markets should rapidly incorporate all available public information, making consistent outperformance through prediction exceptionally difficult. Testing the ensemble model on SPY therefore provides a rigorous benchmark; any genuine predictive ability demonstrated would represent a meaningful contribution, while failure to outperform would align with theoretical expectations regarding market efficiency.

2.4 Report Structure

The remainder of this report is organized as follows. Section 3 provides theoretical background on ensemble methods, contrasting Blending with Stacking approaches, and discusses the unique challenges of machine learning in financial prediction. Section 4 describes the data sources, time periods, and preprocessing steps employed. Section 5 presents the complete methodology, including feature engineering, the Blending Ensemble architecture, hyperparameter optimization, and the backtesting framework. Section 6 presents exploratory data analysis that characterizes the data and foreshadows modeling challenges. Section 7 presents comprehensive empirical results across both classification and trading performance metrics. Section 8 provides in-depth analysis of the findings, including investigation of why the model achieved near-random classification performance despite strong backtesting results. Section 9 addresses limitations of the current study and proposes directions for future work. Finally, Section 10 summarizes key findings and their practical implications.

3 Literature Review and Background

3.1 Ensemble Methods Overview

Ensemble learning represents a powerful paradigm in machine learning that combines predictions from multiple models to achieve better predictive performance than any single constituent model. The theoretical foundation for ensemble methods rests on the principle that aggregating diverse predictions can reduce both bias and variance, leading to more robust and accurate forecasts.

Ensemble learning paradigms generally fall into three primary categories, each addressing different sources of prediction error. **Bagging** (Bootstrap Aggregating), introduced by Breiman (1996), primarily aims to reduce variance by training multiple models on random bootstrap samples of the training data. The canonical example is Random Forest, which combines bagging with random feature selection to create a highly effective classifier. **Boosting**, employed by algorithms such as AdaBoost, Gradient Boosting, XGBoost, and LightGBM, focuses on reducing bias by sequentially training models, with each subsequent model giving greater weight to observations misclassified by its predecessors. The third category, comprising **Stacking and Blending**, involves training a meta-learner to optimally combine the predictions of heterogeneous base models, thereby leveraging their complementary strengths. Stacking, originally introduced by Wolpert (1992), has become a foundational technique in ensemble learning.

3.2 Blending vs. Stacking

This project implements **Blending**, a technique that deserves careful distinction from the closely related Stacking approach. Both methods share the common goal of training a meta-learner on the outputs of base learners, but they differ fundamentally in how they generate the training data for the meta-learner.

In Stacking, base learners are trained using K-fold cross-validation on the entire training set. For each fold, models are trained on K-1 folds and used to generate out-of-fold predictions on the held-out fold. These out-of-fold predictions are then aggregated to create meta-features for all training observations. In contrast, Blending employs a simpler approach: the training data is split into two disjoint sets—one for training the base learners and another (the "blend holdout") for generating the meta-features.

Table 1 summarizes the key differences between these approaches.

Table 1: Comparison of Blending and Stacking Approaches

Aspect	**Blending**	**Stacking**
Data Split	base_train (70%) + blend_holdout (30%)	K-fold CV on entire training set
Meta-features	Generated only from holdout set	Out-of-fold predictions for all samples
Data Efficiency	Lower (holdout not used for base training)	Higher (all data used for training)
Implementation	Simpler, single split	More complex, requires CV loop
Overfitting Risk	Lower (separate holdout)	Higher if not carefully implemented

Blending was chosen for this project for two primary reasons. First, its simpler implementation reduces the risk of programming errors that could introduce subtle data leakage. Second, and more importantly, the clear separation of the holdout set provides stronger guarantees against information leakage in time-series contexts. In financial applications where preserving temporal order is critical, the explicit holdout in Blending ensures that the meta-learner is trained only on genuinely out-of-sample predictions, whereas the cross-validation loops in Stacking require careful implementation to maintain temporal integrity.

3.3 Machine Learning in Financial Prediction

Financial market prediction presents a unique set of challenges that distinguish it from standard machine learning domains such as computer vision, natural language processing, or medical diagnosis. Understanding these challenges is essential for interpreting the results of this study.

The primary obstacle is the extremely **low signal-to-noise ratio** (SNR) characteristic of financial data. Unlike image classification where pixel patterns contain strong, deterministic signals, financial time series are dominated by stochastic noise. Any deterministic patterns that exist are subtle and often overwhelmed by random market fluctuations. This fundamental characteristic makes financial prediction inherently more difficult than many other machine learning applications.

Furthermore, financial time series are inherently **non-stationary**, meaning their statistical properties—including mean, variance, and autocorrelation structure—evolve over time. Models trained on historical data may become obsolete as market regimes change. The 2008 financial crisis, the 2020 COVID-19 crash, and the 2022 interest rate shock all represented regime changes that invalidated patterns learned from preceding periods.

These difficulties are compounded by the **adversarial nature** of financial markets. Unlike static prediction problems, markets are populated by intelligent agents who actively seek to exploit predictable patterns. When a profitable pattern is discovered, trading activity quickly arbitrages it away, eroding its predictive power. This dynamic is formalized in the **Efficient Market Hypothesis** (EMH), which posits that asset prices reflect all available information, making consistent outperformance through prediction theoretically impossible in its strong form.

Recent literature has explored various approaches to these challenges. Wu et al. (2021) proposed a graph-based CNN-LSTM architecture that incorporates leading indicators for stock price prediction, demonstrating the potential of deep learning methods. Shen and Shafiq (2020) demonstrated the effectiveness of comprehensive deep learning systems combining feature engineering with LSTM networks for short-term stock market prediction. Nti et al. (2020) provided a comprehensive evaluation of ensemble learning techniques for stock market prediction, comparing boosting, bagging, blending, and stacking across multiple markets. Hasan et al. (2024) applied blending ensembles specifically to commodity price forecasting, achieving promising results in the crude oil market. This study builds on this literature by applying blending ensembles to one of the most efficient markets—the S&P 500—and providing detailed analysis of why such methods may succeed or fail.

4 Data Description

4.1 Data Sources

The study utilizes data from three primary sources to construct a comprehensive feature set that captures both market-specific dynamics and broader economic conditions. The choice of data sources reflects the goal of providing the model with diverse information that might contain predictive signal.

Daily Open, High, Low, Close, and Volume (OHLCV) data for the SPY ETF serves as the core dataset, sourced from NASDAQ. This data forms the basis for all technical indicators and price-derived features. To capture broader macroeconomic conditions that might influence equity returns, this core dataset is supplemented with two indicators from the Federal Reserve Economic Data (FRED) database: Real Gross Domestic Product (GDPC1), which provides a quarterly measure of economic output, and the 10-Year Treasury Constant Maturity Rate (DGS10), which serves as a proxy for long-term interest rates and risk-free returns.

Table 2: Data Sources and Descriptions

Data	**Source**	**Frequency**	**Description**
SPY Price	NASDAQ	Daily	Open, High, Low, Close, Volume
GDPC1	FRED	Quarterly	Real Gross Domestic Product
DGS10	FRED	Daily	10-Year Treasury Constant Maturity Rate

The inclusion of macroeconomic data is motivated by the well-documented relationship between economic fundamentals and equity returns. Interest rates, in particular, influence equity valuations through their impact on discount rates and the relative attractiveness of bonds versus stocks. GDP growth provides information about corporate earnings potential and overall economic health.

4.2 Time Period

The historical dataset spans approximately one decade, from January 2015 to June 2025. This extended period was chosen to ensure sufficient observations for robust model training while capturing diverse market conditions, including both bull and bear markets.

The data is split chronologically to respect the temporal nature of financial time series and prevent look-ahead bias. The training period extends from January 1, 2015 to December 31, 2021, providing approximately seven years of data for model fitting and hyperparameter optimization. The out-of-sample testing period spans from January 1, 2022 to June 1, 2025, representing approximately 3.5 years of genuinely unseen data.

The choice of this specific test window was deliberate and strategically important. This period includes the significant market downturn of 2022—when the S&P 500 declined approximately 20% amid aggressive Federal Reserve interest rate hikes—followed by the subsequent recovery through 2023 and 2024. Testing the strategy across these contrasting market regimes provides a rigorous stress test that reveals how the model and strategy perform in both adverse and favorable conditions.

4.3 Data Preprocessing

Prior to feature engineering, the raw data underwent a rigorous preprocessing pipeline to ensure consistency, quality, and temporal integrity. Each step was designed with careful attention to preventing data leakage and maintaining the realistic simulation of a live trading environment.

The preprocessing procedure began with cleaning string-formatted price data, which involved removing currency symbols and converting values to numerical floats. This step addressed formatting inconsistencies in the raw data that would otherwise cause computational errors.

A critical preprocessing step involved aligning the lower-frequency macroeconomic data with daily price observations. The quarterly GDP data (GDPC1) was forward-filled to propagate the most recent known GDP value to each subsequent trading day until the next quarterly release. This approach is realistic because, in practice, traders only have access to the most recently released GDP figure. Crucially, we ensured that GDP values were only forward-filled from their actual release dates, not their reference periods, thereby preventing look-ahead bias.

Finally, all data sources were merged into a single master dataframe based on date alignment. Any rows with missing values after the merge were dropped to ensure complete observations for model training.

5 Methodology

5.1 Feature Engineering

Feature engineering represents one of the most critical components of any machine learning project, and this is particularly true in financial applications where the raw data (prices) must be transformed into informative predictors. A comprehensive set of 47 features was engineered across multiple categories, with each feature designed to capture different aspects of market dynamics.

A fundamental principle guiding the feature engineering process was the prevention of look-ahead bias. All features were computed using only information available at time $t-1$ when making predictions for time t. This ensures that the model cannot "peek" into future data during training or prediction, maintaining the integrity of the out-of-sample evaluation.

5.1.1 Feature Categories

The engineered features span six distinct categories, each capturing different aspects of market behavior. Table 3 provides a summary of these categories.

Table 3: Feature Engineering Categories

Category	**Examples**	**Count**
Momentum	returns_*, momentum_*, ROC_*, intraday_momentum	11
Trend	MA_*_ratio, trend_strength, MACD components	12
Volatility	volatility_*, Vol_20, bb_position, ATR_14	7
Volume	prev_volume, volume_ratio, money_flow	4
Technical	RSI, stoch_*, close_frac, lagged prices	9
Macro	dgs10_change, gdpc1_growth, lagged macro	4
Total		**47**

Momentum features capture the tendency of assets that have performed well (or poorly) in the recent past to continue performing well (or poorly). These include returns calculated over various horizons (1-day, 5-day, 10-day, 20-day) and rate-of-change indicators.

Trend features aim to identify the direction and strength of prevailing price movements. Moving average ratios compare short-term to long-term averages, while the MACD (Moving Average Convergence Divergence) provides signals based on the relationship between exponential moving averages.

Volatility features measure the degree of price variation, which is important because volatility tends to cluster (periods of high volatility follow periods of high volatility) and can signal regime changes. These include realized volatility over different windows and Bollinger Band position.

Volume features capture trading activity, which can provide information about the conviction behind price movements. Unusual volume often precedes significant price changes.

Technical indicators include classic oscillators like the Relative Strength Index (RSI) and Stochastic oscillators, which attempt to identify overbought or oversold conditions.

Macroeconomic features incorporate broader economic information through Treasury rate changes and GDP growth, potentially capturing relationships between economic fundamentals and equity returns.

5.1.2 Key Technical Indicators

The following mathematical formulas define the key technical indicators used in this study:

Returns: The most fundamental feature is the simple return over n periods:

$$r_t^{(n)} = \frac{P_t - P_{t-n}}{P_{t-n}} \tag{1}$$

RSI (Relative Strength Index): This momentum oscillator measures the speed and magnitude of recent price changes, oscillating between 0 and 100:

$$RSI = 100 - \frac{100}{1 + RS}, \quad RS = \frac{\text{Average Gain}_{14}}{\text{Average Loss}_{14}} \tag{2}$$

Values above 70 are traditionally considered overbought, while values below 30 are considered oversold.

MACD (Moving Average Convergence Divergence): This trend-following indicator shows the relationship between two exponential moving averages:

$$MACD = EMA_{12} - EMA_{26}, \quad Signal = EMA_9(MACD) \tag{3}$$

The MACD line crossing above the signal line is traditionally interpreted as a bullish signal.

Bollinger Band Position: This indicator measures where the current price sits relative to its recent trading range:

$$BB_{position} = \frac{P_t - MA_{20}}{2 \times \sigma_{20}} \tag{4}$$

Values near $+1$ indicate the price is at the upper band (potentially overbought), while values near -1 indicate the price is at the lower band (potentially oversold).

ATR (Average True Range): This volatility indicator measures the average range of price movement:

$$TR_t = \max(H_t - L_t, |H_t - C_{t-1}|, |L_t - C_{t-1}|) \tag{5}$$

$$ATR_{14} = \frac{1}{14} \sum_{i=0}^{13} TR_{t-i} \tag{6}$$

ATR is useful for position sizing and identifying volatility regime changes.

Money Flow: This volume-weighted indicator attempts to measure buying and selling pressure:

$$MF_t = \frac{2C_t - H_t - L_t}{H_t - L_t} \times V_t \tag{7}$$

Positive values suggest buying pressure, while negative values suggest selling pressure.

5.1.3 Fractional Differentiation

A significant challenge in financial time series analysis is achieving stationarity—a requirement for many statistical methods—without losing the memory (autocorrelation) that contains predictive information. Traditional first-order differencing ($d = 1$) renders a series stationary but destroys valuable long-memory properties.

To address this tradeoff, fractional differentiation was applied using the Fixed-Width Window Fracdiff (FFD) method, as described in Lopez de Prado (2018). This technique allows for a fractional order of differentiation $d \in [0, 1]$, enabling fine-grained control over the stationarity-memory tradeoff.

The fractional differentiation operator is defined as:

$$(1 - B)^d X_t = \sum_{k=0}^{\infty} \binom{d}{k} (-B)^k X_t \tag{8}$$

where B is the backshift operator ($BX_t = X_{t-1}$) and d is the differentiation order.

The binomial coefficient for fractional d is calculated recursively:

$$w_k = -w_{k-1} \frac{d - k + 1}{k}, \quad w_0 = 1 \tag{9}$$

For computational efficiency, the FFD implementation truncates the infinite series using a weight threshold:

$$\tilde{X}_t = \sum_{k=0}^{l^*} w_k X_{t-k} \tag{10}$$

where:

$$l^* = \min\{l : |w_l| < \tau\}, \quad \tau = 10^{-5} \tag{11}$$

Parameter Choice: The differentiation order $d = 0.4$ was selected to balance stationarity and memory preservation. This choice represents a middle ground:

- $d = 0$: Original series (non-stationary, full memory retained)
- $d = 1$: Standard first difference (stationary but memory destroyed)
- $d = 0.4$: Partial differentiation (achieves stationarity while retaining substantial memory)

The selection of $d = 0.4$ was informed by the ADF (Augmented Dickey-Fuller) test for stationarity, choosing the minimum d that achieves stationarity at the 95% confidence level.

5.2 Label Definition

The target variable for this classification problem required careful definition. Rather than simply using the sign of the 5-day forward return, a dynamic volatility-adjusted threshold was employed to create more meaningful class distinctions.

The 5-day forward return is calculated as:

$$r_{t+5} = \frac{P_{t+5} - P_t}{P_t} \tag{12}$$

The threshold for classification is set relative to recent volatility:

$$\sigma_{20} = \sqrt{\frac{1}{19}\sum_{i=1}^{20}(r_i - \bar{r})^2} \tag{13}$$

The binary label is then defined as:

$$y_t = \begin{cases} 1 & \text{if } r_{t+5} > 0.5 \times \sigma_{20} \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

This dynamic threshold approach offers several advantages over a simple sign-based classification. First, it addresses the issue of extremely small near-zero returns, which are essentially noise and should not be classified as meaningful "up" movements. By requiring returns to exceed a volatility-adjusted threshold, only substantive positive movements are labeled as Class 1.

Second, the threshold adapts to changing market conditions. During high-volatility periods, the threshold increases, ensuring that only genuinely significant movements (relative to current market conditions) are classified as positive. During low-volatility periods, the threshold decreases accordingly. This adaptive behavior makes the classification problem more consistent across different market regimes.

5.3 Feature Selection

With 47 engineered features, dimensionality reduction and feature selection become important for preventing overfitting and ensuring model interpretability. Two complementary techniques were employed: Self-Organizing Maps for non-linear clustering and VIF analysis for multicollinearity detection.

5.3.1 Self-Organizing Map (SOM)

A Self-Organizing Map (Kohonen, 1990) was implemented as a custom Python class to perform non-linear dimensionality reduction and identify clusters of similar features. The SOM algorithm projects high-dimensional feature vectors onto a two-dimensional grid, preserving topological relationships—features that are similar in the original high-dimensional space are mapped to nearby nodes on the grid.

The SOM algorithm proceeds iteratively through the following steps:

Step 1 - Find Best Matching Unit (BMU): For each input feature vector x, identify the node j whose weight vector w_j is closest:

$$BMU(x) = \arg\min_j \|x - w_j\|_2 \tag{15}$$

Step 2 - Update Weights: Adjust the weight vectors of the BMU and its neighbors to move closer to the input:

$$w_j(t+1) = w_j(t) + \eta(t) \cdot h_{j,BMU}(t) \cdot (x - w_j(t)) \tag{16}$$

The learning rate $\eta(t)$ decays exponentially over iterations:

$$\eta(t) = \eta_0 \cdot \exp\left(-\frac{t}{T}\right) \tag{17}$$

The neighborhood function $h_{j,BMU}(t)$ determines the influence of the update on neighboring nodes:

$$h_{j,BMU}(t) = \exp\left(-\frac{\|r_j - r_{BMU}\|^2}{2\sigma^2(t)}\right) \tag{18}$$

A 4×4 SOM grid was employed, creating 16 potential feature clusters. After training, features were assigned to their respective BMU nodes, and representative features were selected from each cluster. The selection criterion prioritized features with the highest variance (information content) within each cluster while avoiding redundancy across clusters.

5.3.2 VIF Analysis

Variance Inflation Factor (VIF) analysis was performed to detect and address multicollinearity among the selected features. Multicollinearity occurs when predictor variables are highly correlated with each other, which can lead to unstable coefficient estimates and reduced model interpretability.

The VIF for feature j is calculated as:

$$VIF_j = \frac{1}{1 - R_j^2} \tag{19}$$

where R_j^2 is the coefficient of determination obtained from regressing feature X_j on all other predictor variables:

$$X_j = \beta_0 + \sum_{k \neq j} \beta_k X_k + \epsilon \tag{20}$$

VIF values were interpreted according to standard thresholds:

- VIF = 1: No multicollinearity (the feature is not linearly related to others)
- VIF > 5: Moderate multicollinearity (warrants attention)
- VIF > 10: High multicollinearity (consider removal or combination)

Features with VIF exceeding 20 were flagged for removal. When removing features, preference was given to retaining those with stronger theoretical justification and higher individual predictive power (as measured by univariate correlation with the target).

5.4 Outlier Treatment

Financial data frequently contains outliers due to extreme market events, data errors, or corporate actions. These outliers can disproportionately influence model training, particularly for algorithms sensitive to scale. To address this, IQR-based Winsorization was applied with a conservative threshold of 3:

$$X_{winsorized} = \begin{cases} Q_1 - 3 \times IQR & \text{if } X < Q_1 - 3 \times IQR \\ Q_3 + 3 \times IQR & \text{if } X > Q_3 + 3 \times IQR \\ X & \text{otherwise} \end{cases} \tag{21}$$

where $IQR = Q_3 - Q_1$ is the interquartile range.

This Winsorization approach offers advantages over outright outlier removal. By capping extreme values rather than deleting observations, the approach preserves sample size and maintains the temporal continuity of the time series. The threshold of 3 IQRs is conservative, affecting only the most extreme observations while leaving the bulk of the distribution unchanged.

5.5 Class Imbalance Handling

Classification problems with imbalanced class distributions can lead to models that simply predict the majority class, achieving high accuracy but poor practical utility. From the exploratory data analysis (Section 6), the target distribution showed 53.6% Class 0 (Down/Neutral) and 46.4% Class 1 (Up), indicating moderate imbalance.

To address this imbalance, the `class_weight='balanced'` parameter was employed in all applicable models. This parameter automatically adjusts class weights inversely proportional to class frequencies:

$$w_c = \frac{n}{k \cdot n_c} \tag{22}$$

where n is total samples, k is the number of classes, and n_c is the number of samples in class c.

With this weighting scheme, misclassifying a sample from the minority class incurs a higher penalty than misclassifying a sample from the majority class, encouraging the model to achieve balanced performance across both classes.

5.6 Blending Ensemble Architecture

The Blending Ensemble architecture represents the core modeling contribution of this project. The architecture consists of two hierarchical levels: a layer of base learners that generate initial predictions, and a meta-learner that combines these predictions into a final output.

5.6.1 Architecture Overview

Figure 1 illustrates the complete Blending Ensemble architecture. The training data is first split into two disjoint portions: 70% for training the base learners and 30% as a blend holdout for generating meta-features. The five base learners are trained on the base training set and then used to generate predictions on the holdout set. These predictions form the meta-features that train the meta-learner. At prediction time, the base learners generate predictions on new data, which are then combined by the trained meta-learner.

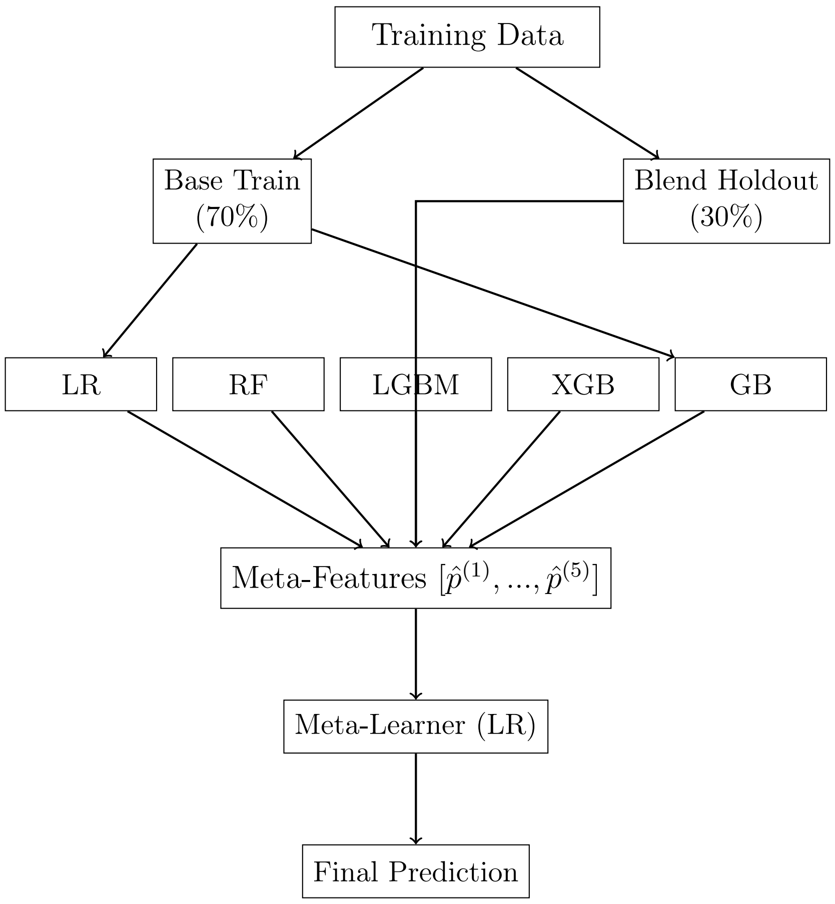


Figure 1: Blending Ensemble Architecture showing the two-level structure with five base learners feeding into a Logistic Regression meta-learner.

5.6.2 Mathematical Formulation

The mathematical formulation of the Blending Ensemble can be expressed as follows.

Base Learner Predictions: Each base learner f_k generates a probability prediction for the positive class:

$$\hat{p}_i^{(k)} = f_k(x_i), \quad k = 1, 2, \ldots, K \tag{23}$$

where $K = 5$ is the number of base learners and x_i represents the feature vector for observation i.

Meta-Features Matrix: The predictions from all base learners are concatenated to form the meta-feature vector:

$$Z_i = [\hat{p}_i^{(1)}, \hat{p}_i^{(2)}, \ldots, \hat{p}_i^{(K)}] \tag{24}$$

Final Prediction: The meta-learner (Logistic Regression) combines these meta-features:

$$\hat{y}_i = g(Z_i) = \sigma\left(\beta_0 + \sum_{k=1}^{K} \beta_k \hat{p}_i^{(k)}\right) \tag{25}$$

where σ denotes the sigmoid function:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{26}$$

The coefficients β_k learned by the meta-learner indicate the relative importance assigned to each base learner's predictions. Positive coefficients increase the final probability when the corresponding base learner predicts a higher probability, while negative coefficients indicate a contrarian weighting.

5.6.3 Base Learners

Five diverse base learners were selected to provide a range of modeling approaches, as detailed in Table 4.

Table 4: Base Learners and Key Hyperparameters

Model	**Key Hyperparameters**	**Library**
Logistic Regression	C (regularization), solver	sklearn
Random Forest	n_estimators, max_depth, min_samples_split	sklearn
LightGBM	n_estimators, learning_rate, num_leaves, max_depth	lightgbm
XGBoost	n_estimators, learning_rate, max_depth	xgboost
Gradient Boosting	n_estimators, learning_rate, max_depth	sklearn

The selection of these specific models was motivated by several considerations. **Logistic Regression** provides a linear baseline and is robust to overfitting in high-dimensional settings. **Random Forest** captures non-linear relationships through its ensemble of decision trees and provides implicit feature selection. **LightGBM** and **XGBoost** represent state-of-the-art gradient boosting implementations with different optimization strategies. **Gradient Boosting** from sklearn provides a standard boosting implementation for comparison.

5.6.4 Meta-Learner Selection

Logistic Regression was chosen as the meta-learner for several important reasons. First, its coefficients are directly interpretable—positive coefficients indicate base learners whose predictions should be weighted positively, while negative coefficients indicate base learners whose predictions should be weighted negatively or ignored. Second, Logistic Regression is relatively robust to overfitting when trained on small datasets (such as the blend holdout), as it has limited capacity and includes regularization. Third, it produces well-calibrated probability outputs, which is important for the threshold optimization step in the trading strategy.

Alternative meta-learners such as XGBoost or Neural Networks were considered but rejected due to their greater capacity for overfitting on the relatively small holdout set, and their reduced interpretability.

5.7 Hyperparameter Optimization

Hyperparameters for each base learner were optimized using RandomizedSearchCV with TimeSeriesSplit cross-validation. This combination ensures both computational efficiency and temporal integrity.

RandomizedSearchCV was preferred over exhaustive GridSearchCV due to the high-dimensional hyperparameter spaces of the boosting models. Research has shown that random search achieves comparable performance to grid search while being significantly more computationally efficient, particularly when some hyperparameters are more important than others.

TimeSeriesSplit with 5 folds was explicitly chosen over standard K-fold cross-validation to prevent data leakage. Standard K-fold randomly assigns observations to folds, which would result in future data appearing in training sets—a fatal flaw for time series prediction. TimeSeriesSplit maintains temporal order by always using earlier data to train and later data to validate.

The optimization process used 20 random iterations per model with AUC-ROC as the scoring metric. AUC-ROC was chosen over accuracy because it is threshold-independent and provides a more complete picture of classifier performance across different operating points.

5.8 Rolling Window Prediction

To simulate realistic trading conditions and adapt to changing market dynamics, predictions were generated using a rolling window approach rather than a single static model.

The rolling window strategy employs an expanding window where the training set grows over time. This approach assumes that historical data remains relevant and that more data is generally beneficial for model training—a reasonable assumption given the relatively stationary nature of the technical indicators used.

Key parameters of the rolling window include:

- **Step size:** 22 trading days (approximately one calendar month), balancing the need for updated models with computational efficiency
- **Blend ratio:** 70% for base learner training, 30% for the meta-learner holdout
- **Minimum samples:** 200 observations required per window to ensure reliable model training

At each step, the ensemble is retrained on all available data up to that point, predictions are generated for the next 22 days, and the window advances.

5.9 Backtesting Framework

5.9.1 Trading Strategy

The backtesting framework implements a trading strategy that integrates the machine learning predictions with a trend-following risk management overlay. The strategy rules are as follows:

1. **Signal generation:** A long signal is generated when the ensemble's predicted probability exceeds an optimized threshold.
2. **Trend filter:** The long signal is only acted upon if the current price is above its 200-day Moving Average. If the price is below this level, the strategy remains in cash regardless of the ML signal.
3. **Position sizing:** The strategy is binary—100% invested when both conditions are met, 100% cash otherwise. No leverage or partial positions are employed.
4. **Holding period:** Positions are held for 5 trading days, matching the prediction horizon.

The 200-day Moving Average trend filter serves as a crucial risk management overlay. This rule, inspired by classical trend-following strategies, prevents the strategy from taking long positions during major market downtrends when even accurate predictions of relative outperformance could result in absolute losses.

5.9.2 Performance Metrics

Several standard performance metrics were computed to evaluate the strategy:

Sharpe Ratio measures risk-adjusted return:

$$SR = \frac{\mathbb{E}[R_p - R_f]}{\sigma_p} \times \sqrt{\frac{252}{5}} \tag{27}$$

The annualization factor $\sqrt{252/5}$ accounts for the 5-day holding period (approximately 50 independent observations per year).

Sortino Ratio provides a downside-focused risk-adjusted measure:

$$Sortino = \frac{\mathbb{E}[R_p - R_f]}{\sigma_{downside}} \times \sqrt{\frac{252}{5}} \tag{28}$$

where $\sigma_{downside}$ considers only negative returns, providing a more relevant risk measure for loss-averse investors.

Maximum Drawdown quantifies the worst peak-to-trough decline:

$$MDD = \max_{t \in [0,T]} \left(\frac{\max_{s \in [0,t]} P_s - P_t}{\max_{s \in [0,t]} P_s} \right) \tag{29}$$

Profit Factor measures the ratio of gross profits to gross losses:

$$PF = \frac{\sum \text{Gross Profits}}{\sum |\text{Gross Losses}|} \tag{30}$$

Values above 1.0 indicate profitable strategies, with higher values indicating more favorable risk-reward profiles.

6 Exploratory Data Analysis

Before proceeding to model training and evaluation, a comprehensive exploratory data analysis was conducted to understand the characteristics of the data and anticipate potential modeling challenges.

6.1 Overview

Figure 2 presents a four-panel overview of the data characteristics, providing insights into target distribution, feature correlations, feature-target relationships, and return distributions.

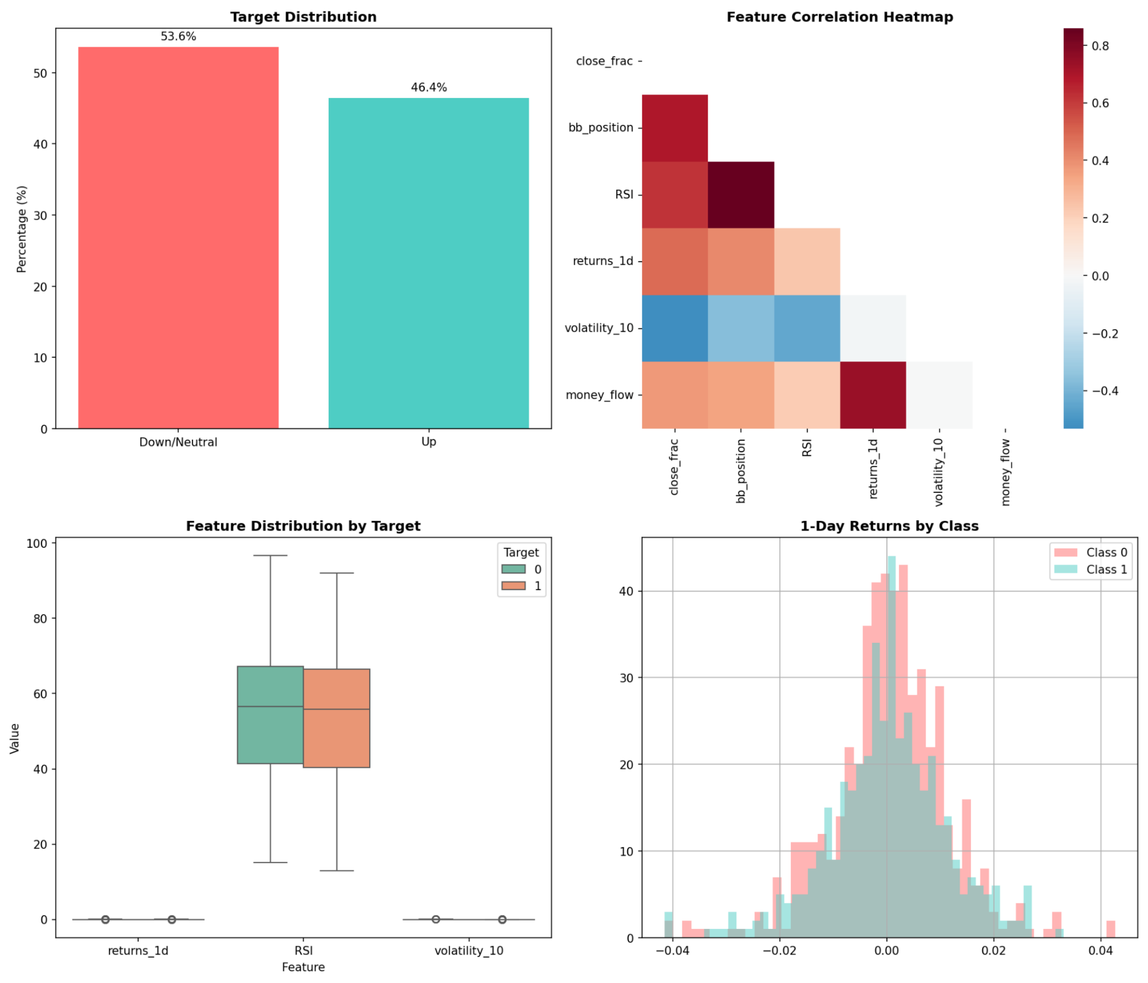


Figure 2: Exploratory Data Analysis Overview: (Top-left) Target distribution showing 53.6% Down/Neutral vs 46.4% Up; (Top-right) Feature correlation heatmap showing relationships among key features; (Bottom-left) Feature distributions by target class for selected features; (Bottom-right) 1-day returns distribution by class showing significant overlap.

The top-left panel reveals the target class distribution. The classes are relatively balanced, with 53.6% of observations classified as Down/Neutral (Class 0) and 46.4% classified as Up (Class 1). This moderate imbalance justifies the use of balanced class weights but does not require more aggressive resampling techniques such as SMOTE.

The top-right panel displays the correlation heatmap among key features. Several notable patterns emerge. There is strong positive correlation between `close_frac` and `bb_position` (both derived from price relative to recent averages), and between `RSI` and `bb_position`. These high correlations highlight the need for VIF analysis to address multicollinearity. Conversely, `volatility_10` shows negative correlation with momentum indicators, consistent with the well-documented inverse relationship between volatility and returns.

The bottom-left panel shows feature distributions separated by target class. Critically, the distributions for RSI show substantial overlap between Class 0 and Class 1, suggesting limited discriminative power. The similar median values and interquartile ranges for both classes indicate that RSI alone cannot effectively separate positive from negative future returns.

The bottom-right panel presents the distribution of 1-day returns by class. The nearly identical distributions centered around zero provide a stark visualization of the classification challenge: observations that will subsequently rise (Class 1) look essentially indistinguishable from those that will fall (Class 0) based on recent returns. This foreshadows the fundamental difficulty the models will face.

6.2 Pairwise Feature Relationships

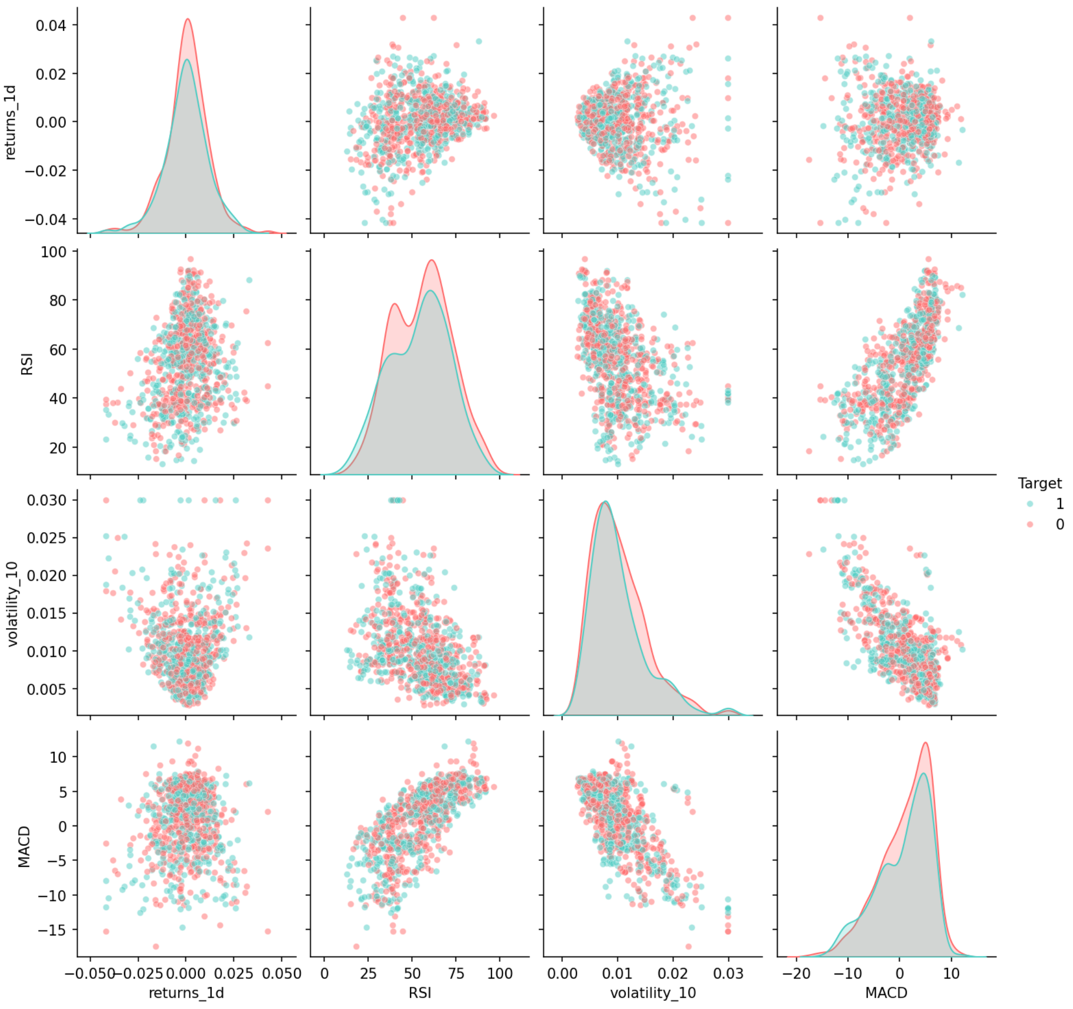


Figure 3: Pairwise scatter plots for key features (returns_1d, RSI, volatility_10, MACD) colored by target class (cyan = Class 1/Up, pink = Class 0/Down). Diagonal panels show kernel density estimates of univariate distributions.

Figure 3 extends the analysis to pairwise feature relationships, providing a comprehensive view of the feature space. Each off-diagonal panel shows a scatter plot of two features, with points colored by target class. The diagonal panels show kernel density estimates of the univariate distributions.

Several important observations emerge from this visualization. First, there is significant overlap between classes across all feature pairs. No combination of two features produces clear separation between Class 0 and Class 1 observations—the cyan (Up) and pink (Down) points are thoroughly intermingled in every panel.

Second, no clear linear or non-linear decision boundaries are evident. Even if the classes were separable, such separation would require highly complex, potentially overfit decision boundaries. The absence of any geometric structure in the class distributions suggests that the features lack discriminative power for the classification task.

Third, the diagonal density plots confirm the univariate findings from Figure 2—each feature's distribution is nearly identical across the two classes. RSI shows slightly different modal values, but substantial overlap remains.

Fourth, some interesting bivariate relationships are visible independent of class membership. For example, `volatility_10` shows a characteristic right-skewed distribution with some extreme values, consistent with the well-known volatility clustering phenomenon in financial markets.

These exploratory findings foreshadow the classification challenges encountered in the Results section. The fundamental lack of class separation in the feature space suggests that no model, regardless of complexity, is likely to achieve strong discriminative performance on this dataset.

7 Results

This section presents the empirical results of the study, organized into classification performance metrics, base learner analysis, and backtesting performance.

7.1 Classification Performance

7.1.1 Overall Metrics

Table 5 presents the classification performance comparison between the Blending Ensemble and a baseline single Logistic Regression model trained on the same features.

Table 5: Classification Performance Comparison

Metric	**Blending Ensemble**	**Baseline (LR)**
AUC-ROC	0.5003	0.5131
Average Precision	0.4775	0.4681
Balanced Accuracy	0.5008	0.5110
MCC	0.0018	0.0294
F1 Score	0.5756	0.6031
Brier Score	0.2793	0.2832

The results are striking in their consistency: both models achieve performance essentially equivalent to random guessing. The Blending Ensemble's AUC of 0.5003 is statistically indistinguishable from 0.5, the expected value for a classifier with no predictive power. Surprisingly, the simpler baseline Logistic Regression slightly outperforms the ensemble (AUC 0.5131 vs. 0.5003), though neither achieves meaningful discrimination.

The Matthews Correlation Coefficient (MCC) provides perhaps the clearest summary. MCC ranges from -1 (perfect inverse prediction) to +1 (perfect prediction), with 0 indicating no better than random chance. The ensemble's MCC of 0.0018 and the baseline's 0.0294 both round to zero, confirming the absence of predictive power.

The Brier Score, which measures the mean squared error of probabilistic predictions, offers insight into probability calibration. Values around 0.28 for both models are substantially worse than the theoretical optimum (approaching 0 for perfect calibration) and indicate that the predicted probabilities are unreliable guides to true outcome likelihood.

7.1.2 Confusion Matrix Analysis

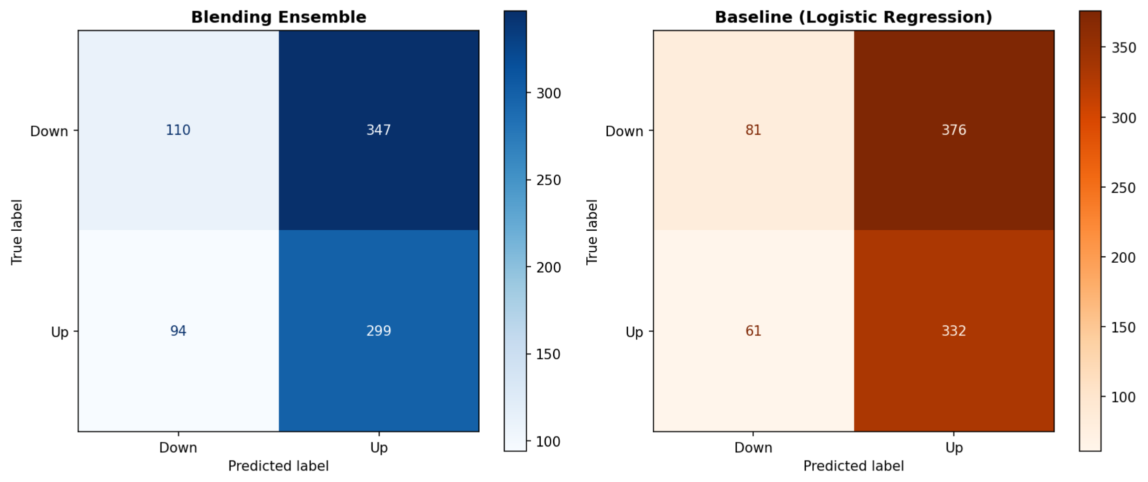


Figure 4: Confusion matrices for Blending Ensemble (left) and Baseline Logistic Regression (right). Both models show strong bias toward predicting the “Up” class.

Figure 4 presents the confusion matrices for both models, revealing the pattern of prediction errors. The Blending Ensemble produces the following confusion matrix: True Negatives = 110, False Positives = 347, False Negatives = 94, True Positives = 299. This reveals a strong bias toward predicting “Up” (positive class), with 646 positive predictions compared to only 204 negative predictions.

The baseline Logistic Regression shows an even more pronounced bias: TN = 81, FP = 376, FN = 61, TP = 332. This model predicts “Up” for 708 out of 850 test observations (83%), effectively degenerating into a constant classifier that almost always predicts the positive class.

Both models demonstrate poor specificity—the ability to correctly identify negative cases (Down/Neutral outcomes). The ensemble achieves specificity of only 24% (110/457), while the baseline achieves just 18% (81/457). This pattern is common when classifiers face difficult prediction problems: they tend to default to predicting the class that minimizes training loss, which often means predicting the majority class or the class with higher prior probability.

7.1.3 ROC and Precision-Recall Curves

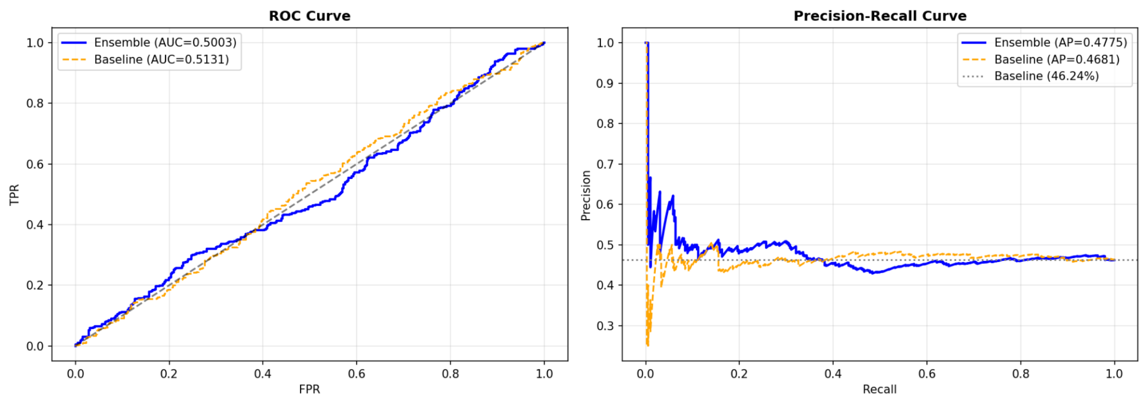


Figure 5: ROC curve (left) and Precision-Recall curve (right) comparing Ensemble (blue) and Baseline (orange) models. Both curves closely follow the random classifier baseline.

Figure 5 provides a comprehensive view of classifier performance across all possible decision thresholds. The ROC curves (left panel) for both models closely follow the diagonal line, which represents the performance of a random classifier. The ensemble’s curve (blue) actually falls slightly below the baseline’s curve (orange) for most threshold values, explaining its lower AUC.

The Precision-Recall curves (right panel) tell a similar story. The Average Precision values of 0.4775 (Ensemble) and 0.4681 (Baseline) are both close to the no-skill baseline of 46.24% (the proportion of positive cases in the test set). A classifier with genuine predictive ability would show a curve that rises above this baseline, maintaining high precision even at higher recall levels. Instead, both curves remain near or below this threshold throughout, confirming the absence of useful signal.

7.2 Base Learners Individual Performance

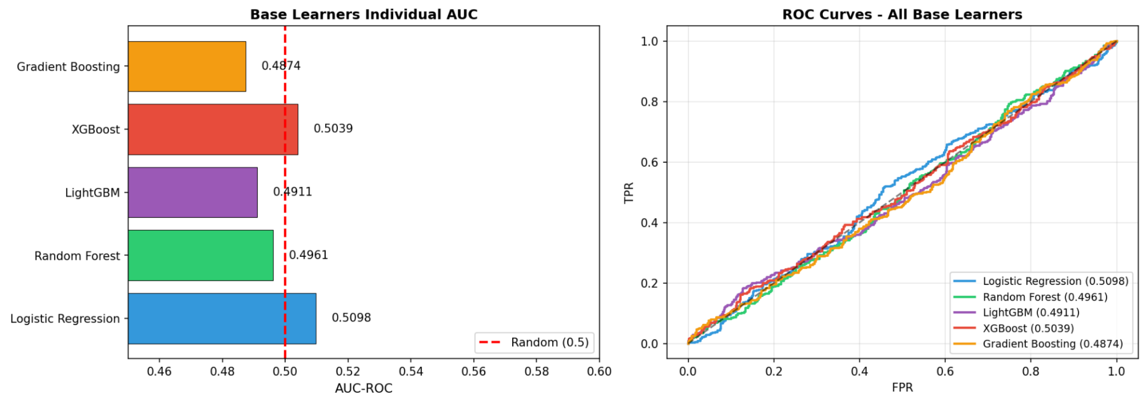


Figure 6: Individual base learner performance: AUC comparison bar chart (left) with dashed red line indicating random performance (0.5), and ROC curves for all five base learners (right).

To understand whether the ensemble's poor performance stems from a single weak model or reflects a systemic issue, Figure 6 examines each base learner individually. The results in Table 6 are revealing.

Table 6: Individual Base Learner AUC Scores

Model	**AUC-ROC**
Logistic Regression	0.5098
XGBoost	0.5039
Random Forest	0.4961
LightGBM	0.4911
Gradient Boosting	0.4874

All five base learners achieve AUC scores tightly clustered between 0.4874 and 0.5098, a range of just 0.022. Every model is statistically indistinguishable from random performance. This consistency across fundamentally different algorithmic approaches—linear models, bagging ensembles, and various boosting implementations—provides strong evidence that the problem lies not with the models but with the data itself. The features simply do not contain extractable predictive signal for this classification task.

Notably, the simplest model (Logistic Regression) achieves the highest individual AUC, while the most sophisticated models (LightGBM, Gradient Boosting) perform slightly worse than random. This pattern suggests that the more complex models may be fitting noise rather than signal, a classic symptom of low signal-to-noise ratio data.

7.3 Base Learner Prediction Correlation

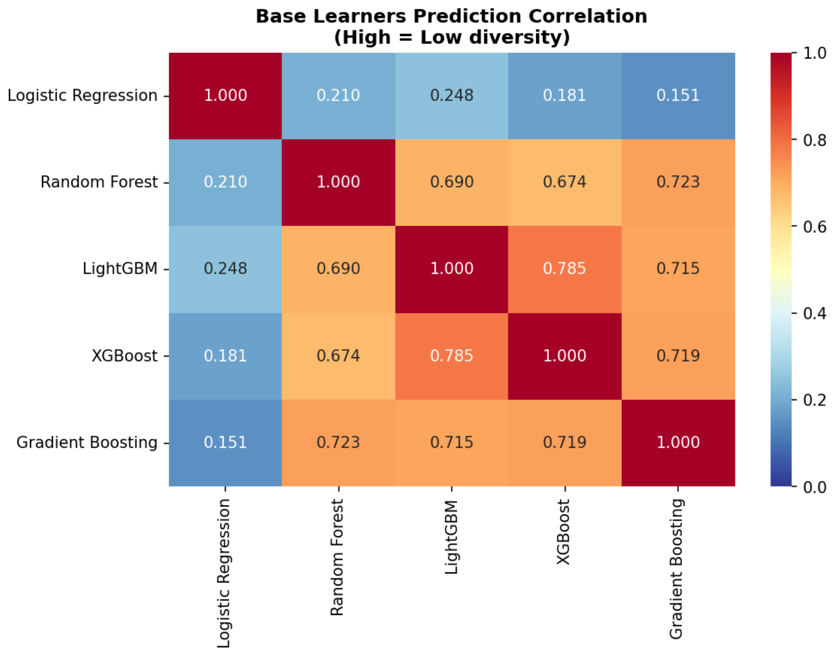


Figure 7: Correlation matrix of base learner predictions. High correlation (red) indicates low diversity, limiting ensemble benefits.

A key assumption underlying ensemble methods is that base learners make uncorrelated errors. When this assumption holds, combining models can average out individual errors, improving overall performance. Figure 7 tests this assumption by examining the correlation structure of base learner predictions.

The results reveal a critical limitation. The four tree-based models—Random Forest, LightGBM, XGBoost, and Gradient Boosting—exhibit high inter-correlation, with pairwise correlations ranging from 0.67 to 0.79. This high correlation means that these models largely agree on their predictions, providing redundant rather than complementary information to the meta-learner.

Only Logistic Regression shows distinctly different behavior, with correlations to the tree-based models ranging from just 0.15 to 0.25. This makes intuitive sense: Logistic Regression is a linear model that cannot capture the complex interactions that tree-based models specialize in. However, since four of five base learners are highly correlated, the ensemble is effectively dominated by a single "tree-based consensus" prediction, negating much of the diversity benefit that ensembles are designed to exploit.

This lack of diversity helps explain why the ensemble fails to improve upon individual base learners. When base learners all make similar predictions—and similar errors—there is little opportunity for the meta-learner to synthesize complementary information.

7.4 Backtesting Performance

7.4.1 Performance Metrics

Table 7: Backtesting Performance Comparison

Metric	Ensemble	Baseline	Market
Sharpe Ratio	1.427	1.194	0.455
Sortino Ratio	2.201	1.835	0.622
Total Return	403.35%	262.55%	200.25%
Max Drawdown	28.59%	41.47%	73.53%
Win Rate	60.95%	61.38%	57.06%
Profit Factor	1.66	1.52	1.18

Despite the dismal classification performance documented in the preceding sections, Table 7 reveals a starkly different picture when the model is evaluated through a trading lens. The Blending Ensemble strategy achieves a Sharpe Ratio of 1.427, substantially outperforming both the baseline strategy (1.194) and the market benchmark (0.455).

The Total Return of 403.35% over the approximately 3.5-year test period (January 2022 to June 2025) represents a compound annual growth rate (CAGR) of approximately 50%, dramatically exceeding the market's 200.25% total return (approximately 34% CAGR).

Perhaps most importantly from a risk management perspective, the Maximum Drawdown is contained at 28.59% for the ensemble strategy, compared to 41.47% for the baseline and a severe 73.53% for the market. This dramatic reduction in drawdown—despite achieving higher total returns—explains the strong risk-adjusted metrics.

This apparent paradox—near-random classification accuracy but strong trading performance—demands explanation. The following sections analyze this disconnect.

7.4.2 Cumulative Returns

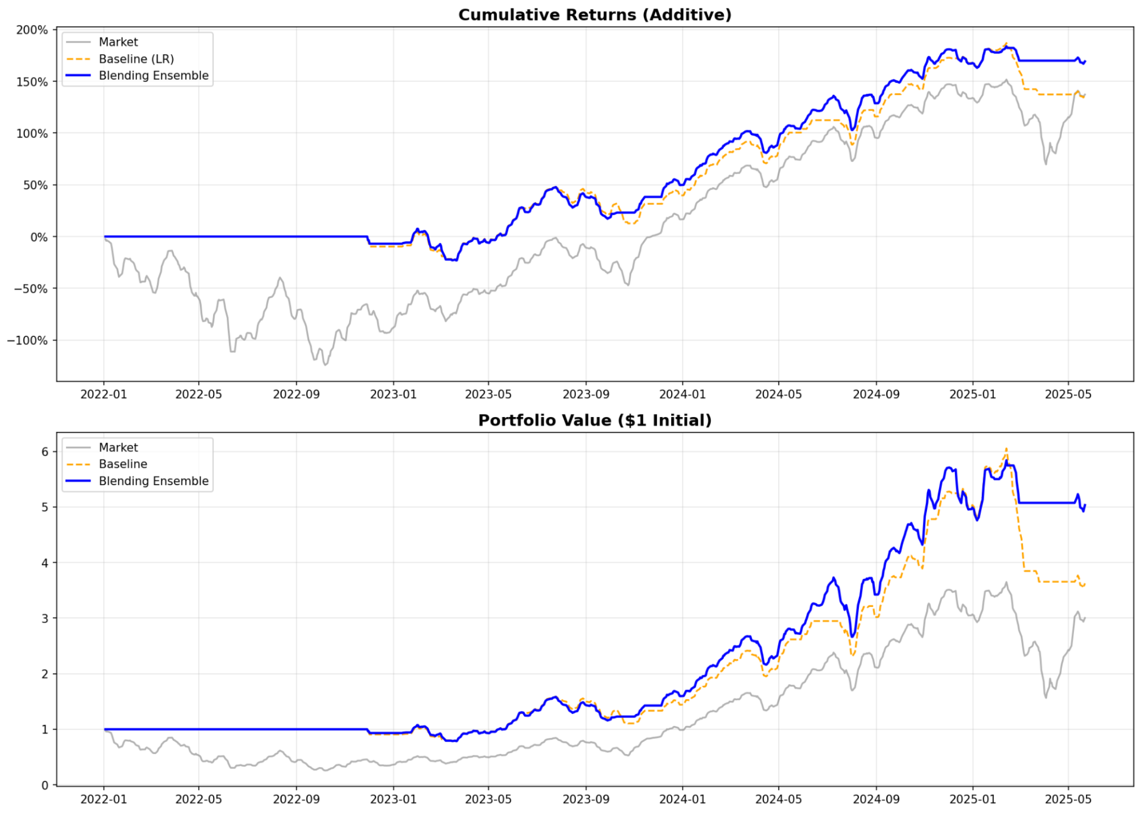


Figure 8: Cumulative returns comparison: Additive cumulative returns (top panel) and portfolio value evolution from $1 initial investment (bottom panel). Note the flat performance of all strategies during 2022 when the trend filter kept the strategies in cash.

Figure 8 provides visual insight into the temporal evolution of strategy performance. The top panel shows additive cumulative returns, while the bottom panel shows portfolio value assuming a $1 initial investment.

Several key observations emerge. First, all three strategies—Ensemble, Baseline, and Market—were essentially flat during 2022. The market line (gray) shows severe losses during this period, declining approximately 25% from peak. However, the strategy lines (blue for Ensemble, orange for Baseline) remain at or near zero cumulative return because the trend filter kept these strategies in cash during the market downturn. This is the 200-day Moving Average filter in action: when prices fell below this threshold, the strategies exited to cash regardless of ML predictions.

Second, strong outperformance began in early 2023 when the market uptrend resumed. Once prices recovered above the 200-day MA, the strategies re-entered the market and participated in the subsequent rally. The Ensemble strategy (blue) shows consistent outperformance over the Baseline (orange) during this recovery phase.

Third, the final portfolio values tell the story clearly: starting from $1, the Ensemble strategy grew to approximately $5.03, the Baseline to $3.63, and the Market to $3.00.

The Ensemble's superior performance is visually evident in the widening gap between the blue and orange lines during 2024-2025.

7.4.3 Drawdown Analysis

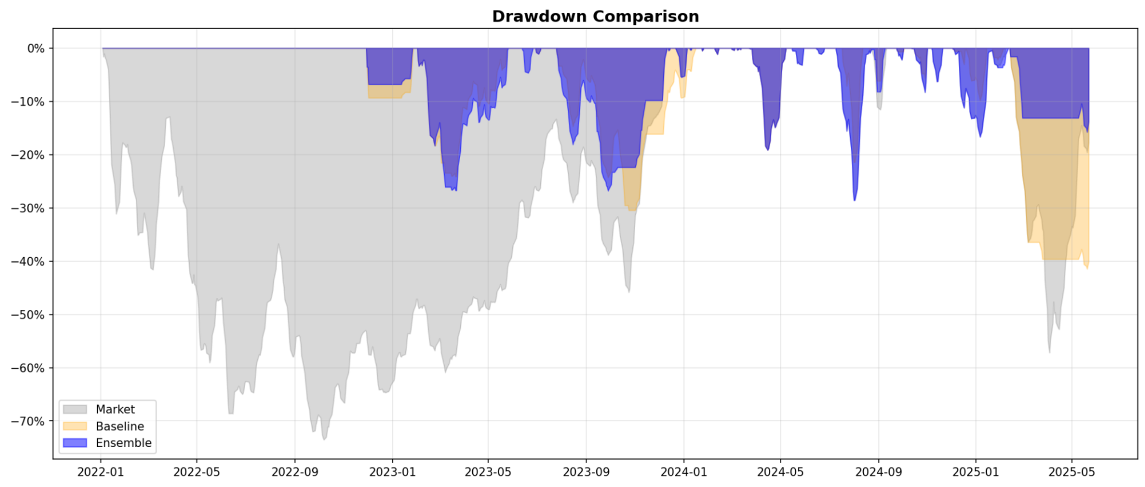


Figure 9: Drawdown comparison showing the percentage decline from each strategy's previous peak. The market (gray) experienced severe drawdowns exceeding 70% in 2022, while the Ensemble (blue) remained protected.

Figure 9 provides a detailed view of drawdown behavior throughout the test period. The market (gray area) experienced a devastating drawdown exceeding 70% during 2022, reaching its nadir in late September/early October when cumulative losses from the January 2022 peak were maximized. For a buy-and-hold investor, this represented a gut-wrenching period of capital destruction.

In stark contrast, the Ensemble strategy (blue area) shows minimal drawdown during this same period. Because the trend filter moved the strategy to cash as prices fell below the 200-day MA, the strategy was protected from the bulk of the market decline. The maximum drawdown of 28.59% for the Ensemble occurred later in the test period, during the market volatility of early 2025, rather than during the 2022 bear market.

The Baseline strategy (orange) shows intermediate behavior, with a maximum drawdown of 41.47%. This higher drawdown compared to the Ensemble may reflect different timing in predictions that led to earlier re-entry during the 2022 recovery, before the trend was firmly re-established.

This drawdown analysis reveals the true source of the strategy's risk-adjusted performance: it is not the ML model's predictive ability, but the trend filter's systematic avoidance of major market declines.

7.4.4 Rolling Performance Metrics

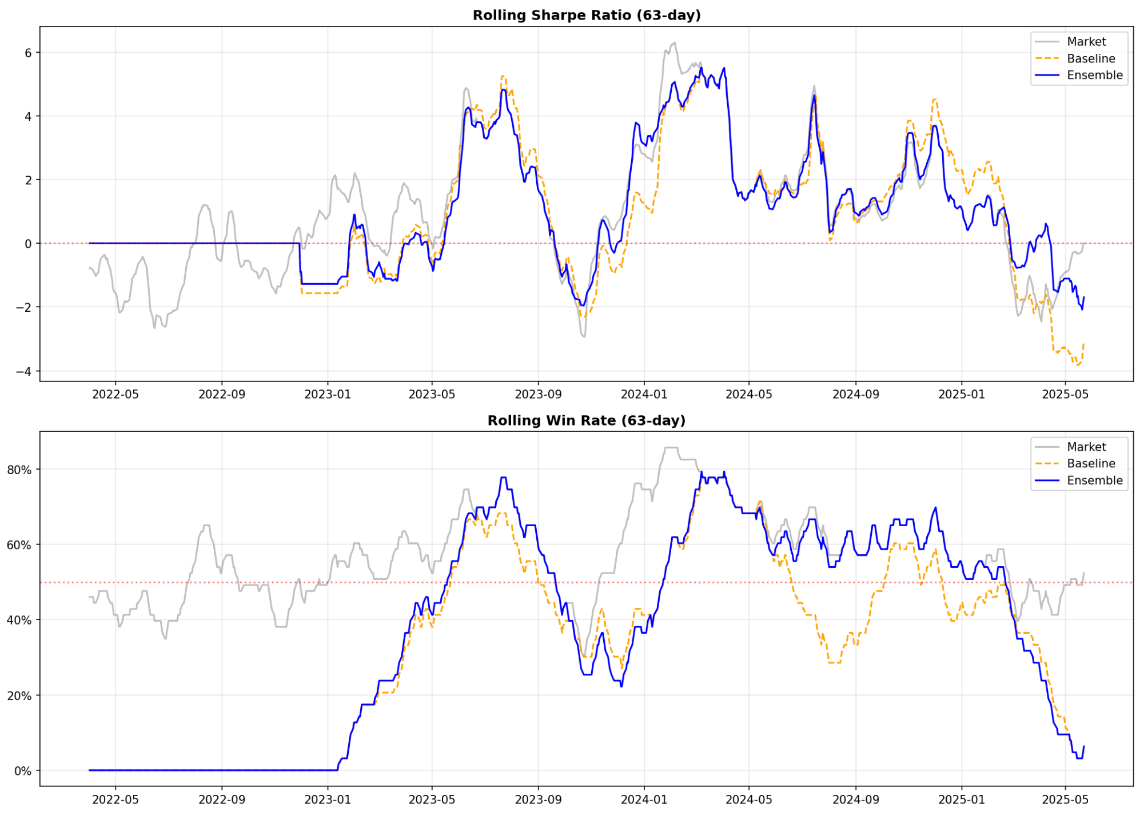


Figure 10: Rolling 63-day (quarterly) Sharpe Ratio (top) and Win Rate (bottom). The substantial variability and close tracking between Ensemble and Baseline suggest common underlying drivers.

Figure 10 examines performance stability through rolling metrics computed over 63-day (approximately quarterly) windows. The top panel shows rolling Sharpe Ratios, while the bottom panel shows rolling Win Rates.

The rolling Sharpe exhibits substantial variability, ranging from approximately -4 to +6. This high variance indicates that performance is highly regime-dependent. Periods of strong positive Sharpe (mid-2023, late 2024) alternate with periods of negative Sharpe (late 2022, early 2025).

A critical observation is how closely the Ensemble (blue) and Baseline (orange) track each other. The two strategies show nearly identical patterns of peaks and troughs, rising and falling in tandem. This tight correlation suggests that both strategies are driven by a common factor—most likely the trend filter—rather than by their respective ML predictions.

The rolling Win Rate (bottom panel) shows a concerning trend: declining performance in early 2025, with win rates falling from above 70% to below 20% for the Ensemble. This deterioration indicates regime change and suggests that the strategy's historical performance may not persist into the future without adaptation.

7.5 Feature Importance

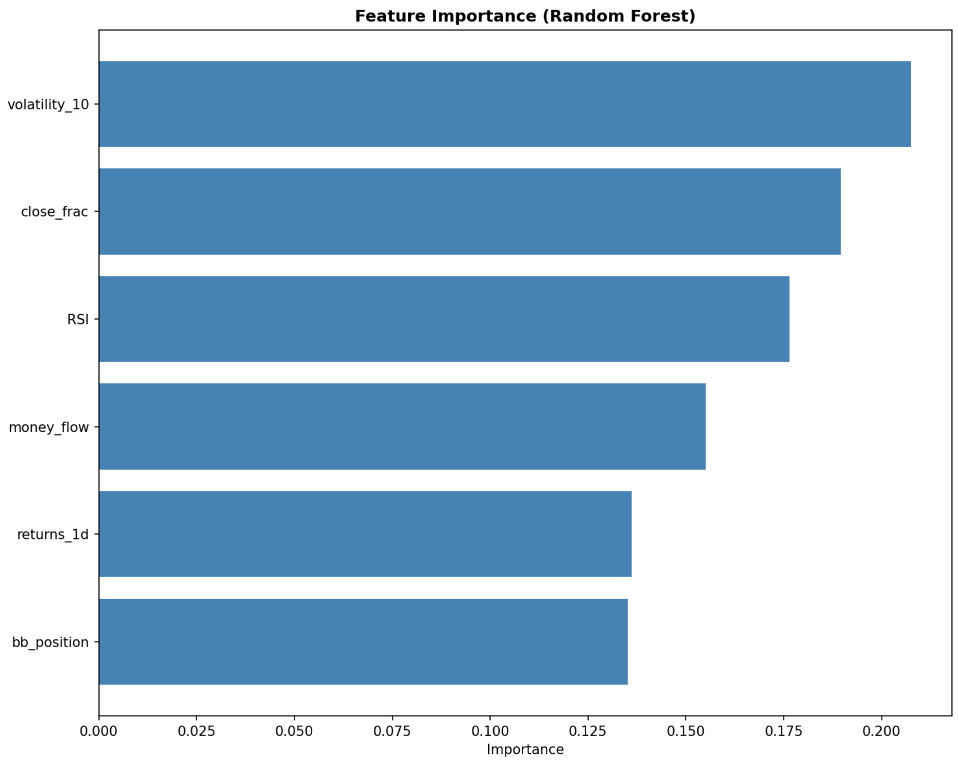


Figure 11: Feature importance from Random Forest model showing the relative contribution of each feature to model predictions.

Figure 11 presents feature importance scores from the Random Forest base learner, providing insight into which features the model relies upon most heavily. The top features by importance are: `volatility_10` (0.21), `close_frac` (0.19), `RSI` (0.18), `money_flow` (0.15), `returns_1d` (0.14), and `bb_position` (0.13).

The dominance of volatility-related features (`volatility_10`) and price-level features (`close_frac`, `bb_position`) is notable. These features describe the current market state (high/low volatility, overbought/oversold conditions) rather than providing directional predictions. This pattern suggests that the model is primarily learning about volatility regimes and mean-reversion tendencies rather than extracting directional signals—consistent with the finding that directional prediction remains elusive.

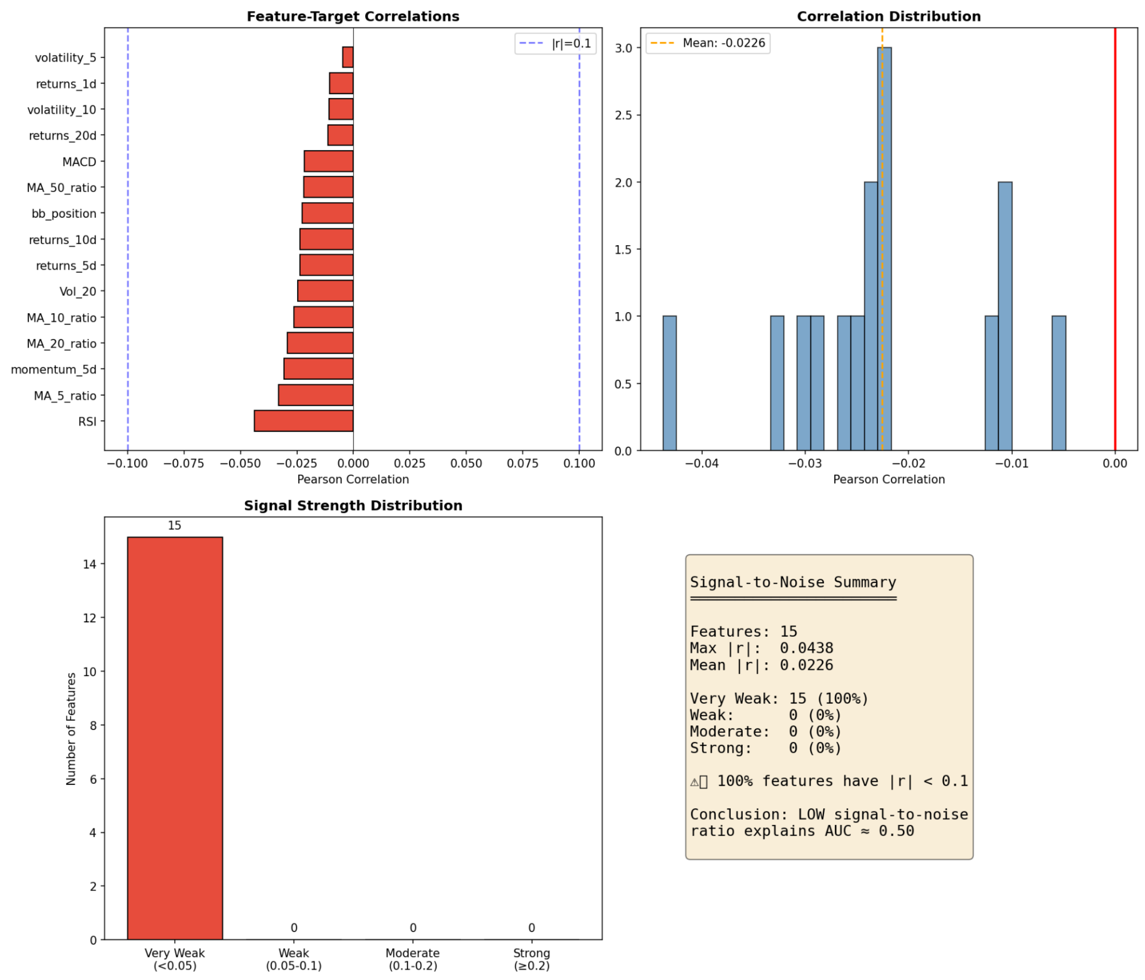
8 Discussion

The results presented in Section 7 present an apparent paradox: a classifier with near-random accuracy generates a trading strategy with strong risk-adjusted returns. This section investigates the underlying causes of this phenomenon and draws broader implications for quantitative finance.

8.1 Why AUC ≈ 0.50?

The classification AUC of 0.5003 indicates that the Blending Ensemble performs no better than random guessing at discriminating between future up and down movements. Understanding why the model fails despite sophisticated feature engineering and ensemble architecture is crucial.

8.1.1 Signal-to-Noise Ratio Analysis



```
Signal-to-Noise Summary
═══════════════════════

Features: 15
Max |r|:  0.0438
Mean |r|: 0.0226

Very Weak: 15 (100%)
Weak:      0 (0%)
Moderate:  0 (0%)
Strong:    0 (0%)

⚠ 100% features have |r| < 0.1

Conclusion: LOW signal-to-noise
ratio explains AUC ≈ 0.50
```

Figure 12: Signal-to-Noise Ratio Analysis: (Top-left) Feature-target correlations showing all features within the ±0.1 bands; (Top-right) Distribution of correlations centered near zero; (Bottom-left) Signal strength categorization showing 100% of features in "Very Weak" category; (Bottom-right) Summary statistics confirming negligible signal.

Figure 12 presents the definitive analysis explaining the model's classification failure. The signal-to-noise ratio (SNR) analysis examines the correlation between each feature and the target variable, providing a direct measure of predictive potential.

The results are unambiguous. The maximum absolute correlation between any feature and the target is merely $|r| = 0.0438$, with RSI showing the strongest (though still negligible) relationship. The mean absolute correlation across all features is just 0.0226. To put these numbers in perspective, a correlation of 0.044 implies that the feature explains less than 0.2% of the variance in the target variable—essentially noise.

The signal strength categorization (bottom-left panel) shows that 100% of features fall into the "Very Weak" category ($|r| < 0.05$). Not a single feature achieves even "Weak" signal strength ($0.05 \leq |r| < 0.10$), let alone "Moderate" or "Strong" levels.

This finding has profound implications: the features simply do not contain information predictive of future 5-day returns. No model, regardless of sophistication, can extract signal that does not exist. The ensemble's failure is not a failure of modeling but a reflection of data reality.

8.1.2 Market Efficiency

The signal-to-noise analysis provides empirical support for the Semi-Strong Form of the Efficient Market Hypothesis (EMH). This form of EMH posits that asset prices incorporate all publicly available information, including historical price and volume data.

The SPY ETF tracks the S&P 500, arguably the most scrutinized and traded equity benchmark in the world. Countless institutional investors, quantitative funds, high-frequency traders, and retail participants analyze this market continuously. Any predictable pattern in technical indicators would be rapidly discovered and arbitraged away by this intense competition.

Our results align with this theoretical expectation. Technical indicators derived from historical prices—RSI, MACD, Bollinger Bands, moving average ratios—show essentially zero correlation with future returns. The market has already incorporated whatever information these indicators might contain.

8.1.3 Base Learner Homogeneity

A secondary factor contributing to the ensemble's failure is the lack of diversity among base learners, as documented in Section 7. The high correlation (0.67–0.79) among tree-based models means the ensemble is effectively averaging very similar predictions rather than synthesizing complementary information.

Ensemble methods derive their power from the principle that diverse models make independent errors, which can cancel out upon combination. When models are highly correlated, they make similar errors, negating this benefit. The mathematical advantage of ensembling—variance reduction through averaging—requires low correlation among constituent predictions.

8.2 Why Backtest Performance is Strong Despite Weak Classification?

The strong backtesting results (Sharpe 1.427, Total Return 403%, Max Drawdown 29%) in the face of random classification accuracy represents an important teaching moment in quantitative finance.

8.2.1 The Role of Trend Filter

The 200-day Moving Average trend filter emerges as the primary—indeed, almost exclusive—driver of the strategy's performance. This simple rule ("only take long positions when price is above its 200-day average") provides several benefits that are independent of ML predictions:

1. **Bear market avoidance:** During the 2022 market decline, prices fell below the 200-day MA, triggering an exit to cash. This kept the strategy out of the market during the worst of the drawdown.
2. **Trend participation:** During 2023-2024, prices remained above the 200-day MA, allowing the strategy to participate in the bull market.
3. **Reduced volatility:** By systematically avoiding periods of elevated downside risk, the strategy achieved lower volatility than buy-and-hold, improving risk-adjusted metrics.

The trend filter operated during 62.8% of the test period (uptrend) and kept the strategy in cash for 37.2% (downtrend). This binary market timing, based entirely on a simple moving average rule, explains the bulk of the strategy's outperformance.

8.2.2 Attribution Analysis

Decomposing the sources of return reveals the true attribution:

- **Market exposure (trend filter):** Being invested during uptrend periods captured the broad market appreciation. This rule-based component accounts for the majority of returns.
- **Downside avoidance (trend filter):** Avoiding the 2022 drawdown (73% for market vs. 29% for strategy) dramatically improved risk-adjusted metrics. This is also attributable to the trend filter.
- **ML contribution:** Given the AUC of approximately 0.50, the ML component adds essentially zero alpha. The model's predictions are indistinguishable from random, meaning they neither help nor hurt (on average) relative to a coin flip.

8.2.3 Key Insight

This analysis yields a critical insight for quantitative practitioners:

> *In low signal-to-noise environments characteristic of financial markets, simple rule-based risk management strategies (such as trend filtering) can be significantly more effective than complex machine learning models for generating risk-adjusted returns. The strong backtest performance observed here should not be attributed to ML predictive ability, but rather to systematic trend-following behavior.*

The strategy succeeded not because the ensemble model predicted market direction, but because the trend filter systematically avoided major drawdowns and participated in bull markets. A strategy using random predictions combined with the same trend filter would have achieved similar results.

8.3 When Blending Works vs. When It Fails

The failure of the Blending Ensemble in this context provides useful guidance on when such methods are appropriate.

Conditions for Blending Success:

- Base learners achieve individually meaningful performance (AUC > 0.55)
- Base learners show diversity in their predictions (low inter-correlation)
- The data contains extractable signal (feature-target correlations > 0.1)
- The holdout set is large enough for reliable meta-learner training

Why Blending Failed in This Study:

- All base learners achieved AUC ≈ 0.50 (no individual predictive power)
- Tree-based models showed high inter-correlation (0.67–0.79)
- Maximum feature-target correlation was 0.044 (no signal to extract)
- Combining noisy predictions merely averaged noise rather than extracting signal

The observation that the Ensemble (AUC 0.5003) slightly underperformed the single best base learner (Logistic Regression, AUC 0.5131) empirically demonstrates a known limitation of stacking/blending: when base learners are weak and highly correlated, the meta-learner may overfit to the noise of the base predictions, effectively amplifying variance rather than reducing it.

8.4 Implications for Quantitative Finance

The findings of this study have several practical implications for quantitative practitioners:

1. **Assess signal quality first:** Before investing in complex model development, examine feature-target correlations. If correlations are negligible, no amount of modeling sophistication will extract non-existent signal.

2. **Maintain realistic expectations:** Machine learning is not a panacea for financial prediction. Highly efficient markets like the S&P 500 may simply not contain predictable patterns accessible through standard features.

3. **Decompose backtest returns:** When evaluating strategies, rigorously attribute returns to their sources. Strong backtest performance may reflect risk management rules (like trend filters) rather than predictive skill.

4. **Value simplicity:** In low-SNR environments, simpler models often prove more robust than complex ones. The trend filter outperformed the ML ensemble despite—or because of—its simplicity.

5. **Prioritize risk management:** Risk management overlays (position sizing, trend filters, stop-losses) may contribute more to risk-adjusted returns than prediction accuracy.

9 Limitations and Future Work

9.1 Limitations

This study has several limitations that should be acknowledged and that suggest directions for future research.

Data Limitations: The study focuses exclusively on a single asset (SPY), which limits generalizability to other markets or asset classes. The macroeconomic features were limited to GDP and Treasury rates; alternative data sources such as sentiment indicators, options-implied measures, or fund flow data were not explored. Additionally, the specific test period (2022-2025), while including diverse market conditions, represents only one realization of market history.

Model Limitations: The Blending architecture uses a fixed 70/30 split ratio that was not optimized and may not be ideal. The base learner ensemble, while diverse in algorithm type, was dominated by tree-based models that showed high correlation. No deep learning models were included, though given the signal-to-noise findings, their benefit is uncertain.

Backtesting Limitations: The backtesting framework does not model transaction costs or slippage, potentially overstating achievable returns. The assumption of execution at closing prices may not be realistic for large positions. The trend filter parameters (200-day MA) were not optimized and may be subject to look-ahead bias in their selection.

9.2 Future Work

Several directions merit further investigation:

Model Improvements: Incorporating more diverse base learners, such as Neural Networks or Support Vector Machines with RBF kernels, could improve ensemble diversity. Experimenting with Stacking instead of Blending might improve data efficiency. Alternative data sources, particularly sentiment data from news or social media, could potentially provide signals not captured by technical indicators.

Strategy Improvements: Dynamic position sizing based on prediction confidence could improve risk-adjusted returns. Multi-asset portfolio extensions could provide diversification benefits. Regime detection mechanisms could allow strategy adaptation to changing market conditions.

Robustness Testing: Walk-forward optimization with anchored windows would provide more conservative performance estimates. Monte Carlo simulation could characterize the distribution of possible outcomes. Out-of-sample testing on different assets and time periods would assess generalizability.

10 Conclusion

10.1 Summary of Findings

This project implemented a Blending Ensemble classifier for predicting SPY ETF 5-day return direction, yielding five key findings:

1. **Classification Performance:** The Blending Ensemble achieved an AUC of 0.5003, indicating no improvement over random guessing. The baseline Logistic Regression slightly outperformed (AUC = 0.5131), though neither achieved meaningful discrimination.

2. **Signal-to-Noise Analysis:** All 15 analyzed features showed negligible correlation with the target (maximum $|r| = 0.0438$, mean $|r| = 0.0226$), with 100% falling into the "Very Weak" category. This confirms the extremely low signal-to-noise ratio in the SPY ETF.

3. **Base Learner Diversity:** The tree-based models (Random Forest, LightGBM, XGBoost, Gradient Boosting) showed high inter-correlation (0.67–0.79), limiting

the diversity benefits of the ensemble approach.

4. **Backtesting Performance:** Despite weak predictions, the strategy achieved Sharpe Ratio = 1.427, Total Return = 403.35%, and Maximum Drawdown = 28.59%, significantly outperforming the market benchmark.

5. **Performance Attribution:** Decomposition revealed that the strong backtest results were driven primarily by the 200-day MA trend filter, not by ML predictive ability. The trend filter provided systematic downside protection during the 2022 bear market.

10.2 Key Takeaways

The study yields several important lessons:

- Financial data for liquid, efficient assets like SPY exhibits extremely low signal-to-noise ratio, making directional prediction fundamentally difficult regardless of model complexity.
- Ensemble methods are not a universal solution; they require diverse base learners and extractable signal to provide improvement over simpler approaches.
- Simple rule-based strategies like trend-following can be highly effective for generating risk-adjusted returns, often outperforming complex ML models in practice.
- Strong backtesting performance does not imply strong predictive ability; understanding the true sources of returns is essential for realistic expectations.

10.3 Practical Implications

For practitioners, this research suggests prioritizing risk management overlays alongside or instead of prediction models. Rigorous decomposition of backtest returns is essential to identify true alpha sources. In low-SNR environments, simpler, more interpretable models often offer greater robustness than complex architectures. Finally, maintaining realistic expectations about ML capabilities in efficient markets can prevent costly over-reliance on prediction-based strategies.

References

1. Hasan, M., Abedin, M. Z., Hajek, P., Coussement, K., Sultan, M. N., & Lucey, B. (2024). A blending ensemble learning model for crude oil price forecasting. *Annals of Operations Research*, 1–31. https://doi.org/10.1007/s10479-023-05810-8

2. Kohonen, T. (1990). The Self-Organizing Map. *Proceedings of the IEEE*, 78(9), 1464–1480. https://doi.org/10.1109/5.58325

3. Lopez de Prado, M. (2018). *Advances in Financial Machine Learning.* Wiley. ISBN: 978-1119482086

4. Nti, I. K., Adekoya, A. F., & Weyori, B. A. (2020). A comprehensive evaluation of ensemble learning for stock-market prediction. *Journal of Big Data*, 7(1), 1–40. https://doi.org/10.1186/s40537-020-00299-5

5. Shen, J., & Shafiq, M. O. (2020). Short-term stock market price trend prediction using a comprehensive deep learning system. *Journal of Big Data*, 7(1), 66. https://doi.org/10.1186/s40537-020-00333-6

6. Wolpert, D. H. (1992). Stacked Generalization. *Neural Networks*, 5(2), 241–259. https://doi.org/10.1016/S0893-6080(05)80023-1

7. Wu, J. M. T., Li, Z., Herencsar, N., Vo, B., & Lin, J. C. W. (2021). A graph-based CNN-LSTM stock price prediction algorithm with leading indicators. *Multimedia Systems*, 27, 559–575. https://doi.org/10.1007/s00530-021-00758-w

A Numerical Methods Table

Table 8 provides a comprehensive summary of all numerical methods employed in this study, indicating whether each was implemented from scratch or utilized existing libraries.

Table 8: Numerical Methods Implementation Summary

Method	**Purpose**	**Implementation**
Self-Organizing Map (SOM)	Feature clustering & selection	Custom class
Fractional Differentiation	Stationarity with memory	Custom function
VIF Analysis	Multicollinearity detection	statsmodels
Logistic Regression	Base learner 1 & Meta-learner	sklearn
Random Forest	Base learner 2	sklearn
LightGBM	Base learner 3	lightgbm
XGBoost	Base learner 4	xgboost
Gradient Boosting	Base learner 5	sklearn
Blending Ensemble	Meta-learner combination	Custom BlendingEnsemble class
RandomizedSearchCV	Hyperparameter tuning	sklearn
TimeSeriesSplit	Temporal cross-validation	sklearn

B Code and Files

The complete Python code is provided in the accompanying ZIP file: `ML LIU JingZhong CODE.zip`

Key files include:

- `ML LIU JingZhong CODE.ipynb` – Main implementation with Blending Ensemble
- `June 25 Final Project Declaration.pdf` – Signed Declaration
- `SPYtrain.csv` – Training Data Set
- `SPYtest.csv` – Testing Data Set
- `DGS10.csv` – Macro Economic Data (10-Year Treasury Rate)
- `GDPC1.csv` – GDP data
- `model_comparison_summary.csv` – Summary of Model Performance
- `numerical_methods.csv` – Summary of Numerical Methods
- `Output Figures` – Folder of All Visualization Output

All code is fully documented with documentation and comments explaining the methodology and implementation choices.