LEAP: A Leakage-Free Evolutionary Alpha Pipeline for NLP-Driven DJIA Prediction

Miguel P. Cunha[1(✉)], Bruno L. Dalmazo[1], Viviane Leite Dias de Mattos[1], Richard F. Pinto[1], Diego Renan Bruno[2(✉)], Eduardo N. Borges[1], Giancarlo Lucca[1], Fabian C. Cardoso[3], and Rafael A. Berri[1]

[1] Federal University of Rio Grande (FURG), Rio Grande, Brazil
{miguel.pinho,dalmazo,vivianemattos,richard_pinto,eduardoborges,giancarlo.lucca,rafaelberri}@furg.br
[2] São Paulo State University (UNESP), São José do Rio Preto, Brazil
diego.bruno@unesp.br
[3] University of Rio Verde (UniRV), Rio Verde, Brazil
fabian@unirv.edu.br

Abstract. Financial market prediction utilizing Natural Language Processing (NLP) is notoriously challenging due to low signal-to-noise ratios, high dimensionality, and the persistent flaw of evaluating trading models using symmetric statistical metrics like Accuracy or Area Under Receiver Operating Characteristic Curve (AUC-ROC). To deal with these questions, we propose in this paper a rigorous data-leakage-free machine learning pipeline to forecast the daily directional movement of the Dow Jones Industrial Average (DJIA). Our architecture integrates FinBERT-derived sentiment embeddings with technical indicators and temporal lags. To combat overfitting, we employ Recursive Feature Elimination with Cross-Validation (RFECV) for dimensionality reduction and a Genetic Algorithm (GA) to optimize hyperparameters by explicitly penalizing the generalization gap. By applying Cost-Sensitive Learning to prioritize high-volatility "fat-tail" events, we contrast a Gradient Boosting ensemble with a Multi-Layer Perceptron (MLP) via an out-of-sample financial backtest. These findings point to a statistical-financial paradox: despite the Gradient Boosting's better discrimination (Test AUC 0.5041, Alpha +1.69 p.p.), the MLP statistically "fails" (Test AUC 0.4812) yet takes advantage of market asymmetries to generate a +13.27 p.p. Alpha. These findings indicate that symmetric evaluation metrics are not enough for assessing trading viability, pointing to the need for magnitude-aware training.

Keywords: Algorithmic Trading · Natural Language Processing · Genetic Algorithms · Feature Selection · Financial Time Series

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027
O. Gervasi et al. (Eds.): ICCSA 2026, LNCS 16769, pp. 71–87, 2027.
https://doi.org/10.1007/978-3-032-30494-0_5

1 Introduction

The prediction of financial market movements is a known difficult problem in computational finance [14]. Financial time series are inherently noisy, non-stationary, and heavily influenced by stochastic external events and human behavioral biases. In recent years, the integration of Natural Language Processing (NLP) into algorithmic trading has brought relevant gains to the field. By extracting sentiment signals from financial news, models can quantify market psychology before it is fully priced into historical data [16]. However, the combination of textual embeddings with traditional technical indicators raises the dimensionality of the feature space, increasing the risk of model overfitting, a known limitation in predictive finance [2].

A recurring issue in the applied Machine Learning literature for financial forecasting is the reliance on symmetric statistical metrics, such as Accuracy or AUC-ROC, as the main criteria for model evaluation [10]. These metrics tend to assume that all misclassifications carry the same cost. In financial markets, this assumption is not valid due to the presence of "fat tails" (high-magnitude, low-frequency price movements). A model can fail to predict the market direction on several low-volatility days, degrading its AUC, but still correctly anticipate a market crash or a strong rally, while producing relevant trading gains [13].

To address these challenges, this paper proposes LEAP (Leakage-Free Evolutionary Alpha Pipeline), a data-leakage-free optimization pipeline for predicting the daily directional movement of the Dow Jones Industrial Average (DJIA). Our approach combines domain-specific NLP sentiment (extracted via FinBERT) with classical technical indicators [4,15] and multi-scale temporal lags. To deal with high dimensionality and avoid noise memorization, we employ Recursive Feature Elimination with Cross-Validation (RFECV) combined with an Evolutionary Optimization framework (Genetic Algorithms) [1]. The genetic algorithm's fitness function was built to penalize the generalization gap between training and validation sets, pushing the search toward more generalizable models.

We also apply a Cost-Sensitive Learning approach that uses sample weights based on return magnitude, guiding the models to focus on high-volatility days. Two learning architectures are compared: a Gradient Boosting ensemble and a Multi-Layer Perceptron (MLP) neural network. The evaluation goes beyond standard statistical metrics by using an out-of-sample financial backtest with a dynamic decision threshold.

The primary contributions of this work are three-fold:

1. **A Leakage-Free Predictive Pipeline:** We introduce a highly disciplined chronological framework that integrates NLP sentiment and temporal data while strictly preventing forward-looking bias during feature selection and normalization.
2. **Evolutionary Overfitting Control:** Evolutionary Overfitting Control is applied to the Genetic Algorithm for optimizing the objective function and successfully selecting hyperparameters in highly stochastic environments.

3. **Empirical Demonstration of the Statistical-Financial Paradox:** We reveal that symmetric metrics are insufficient for trading viability. Our results show that while the Gradient Boosting model maintains statistical stability (AUC >0.50) yielding a $+$**1.69 p.p. Alpha**, the MLP architecture, despite a seemingly failing statistical metric (AUC <0.50), effectively captures market asymmetries, delivering a massive $+$**13.27 p.p. Alpha**.

The remainder of this paper is organized as follows: Sect. 2 outlines the theoretical foundations, followed by a review of related literature in Sect. 3. Section 4 details the proposed LEAP methodology. Section 5 discusses the empirical results and the financial backtesting paradox. Finally, Sect. 6 concludes the study.

2 Theoretical Background

This study is built around two learning architectures: Gradient Boosting (GB) and the Multi-Layer Perceptron (MLP). GB is a sequential tree-based ensemble that minimizes differentiable loss, known for its good results on noisy tabular data [7]. The MLP is a feed-forward artificial neural network able to handle non-linear interactions without relying on rigid decision boundaries. To optimize these architectures, we employ Genetic Algorithms (GA) [1], an evolutionary search heuristic that searches the hyperparameter space more effectively than grid-based methods by simulating natural selection. In quantitative finance, models are typically assessed using symmetric statistical metrics such as the Area Under the Receiver Operating Characteristic Curve (AUC-ROC). However, financial returns exhibit leptokurtic distributions characterized by "fat tails" [8]. Consequently, symmetric metrics treat all prediction errors the same way, leaving out the economic impact of asymmetric returns. To address this, Cost-Sensitive Learning introduces magnitude-based sample weights into the training phase, making the loss function give more weight to errors on high-volatility days than on low-magnitude ones.

3 Related Work

Financial forecasting puts pressure on the Efficient Market Hypothesis (EMH) [9] by taking advantage of cognitive biases and momentum effects [12]. In recent years, Natural Language Processing (NLP) has become a useful resource for capturing these behavioral shifts. Transformer-based architectures, specifically FinBERT [16], have shown better results than generic models when handling domain-specific financial terms. However, integrating high-frequency news sentiment with technical indicators brings collinearity problems and higher dimensionality to the feature space. To deal with this, previous works point to feature selection techniques such as Recursive Feature Elimination with Cross-Validation (RFECV) [11].

A common problem in applied financial machine learning is the heavy dependence on symmetric metrics as the main measure of model quality [10]. Models

tend to focus too much on frequent, low-magnitude days while overlooking less frequent directional changes that produce real trading gains [13]. While existing literature explores hyperparameter tuning to improve accuracy, traditional methods like Grid Search are computationally costly and likely to overfit in non-stationary settings [3]. This work focuses on filling this gap by introducing a leakage-free pipeline that combines NLP embeddings, Cost-Sensitive Learning, and evolutionary regularization to directly account for the generalization gap and focus on out-of-sample financial performance rather than in-sample symmetric metrics.

4 Methodology

Our methodology follows a sequential, leakage-free pipeline to forecast the DJIA's daily direction. To build this framework, we merge multi-scale temporal lags, classical technical indicators, and NLP-based sentiment signals. Figure 1 details this chronological setup. We execute all feature construction, normalization, and selection strictly within the in-sample data, ensuring no future information leaks prior to the out-of-sample evaluation. Finally, a financial backtest assesses the model's cumulative returns on the held-out test set. This step verifies the system's practical trading potential, giving a more grounded view than standard classification metrics alone.

4.1 Data Inputs

Our study spans August 2008 to July 2016, pairing two public datasets. To capture macro-level psychology rather than direct financial news, we extracted daily geopolitical headlines from the *r/worldnews* board using the *Combined News DJIA* collection. These sentiment signals were linked to standard OHLCV metrics from the *DJIA Historical Prices* dataset. Aligning both sources by their `Date` keys and filtering out incomplete records resulted in a continuous series of 1,989 trading days.

The prediction target is the binary *directional movement* of the DJIA on the following trading day:

$$\text{Label}_t = \begin{cases} 1 & \text{if } R_{t+1} > 0 \quad \text{(market rises)}, \\ 0 & \text{otherwise}, \end{cases} \tag{1}$$

where $R_{t+1} = (C_{t+1} - C_t)/C_t$ is the simple next-day return and C_t denotes the closing price at day t.

To reflect the economic significance of each trading day, a *sample-weighting* scheme is applied. Days whose absolute return exceeds the 75th percentile of the return-magnitude distribution receive a weight of 2.0; all remaining days receive a weight of 1.0. This Pareto-inspired threshold captures the top 25% of moves most likely to exceed transaction costs and generate exploitable alpha [13], while keeping the class distribution well-balanced (≈52% up/48% down), precluding the need for resampling.

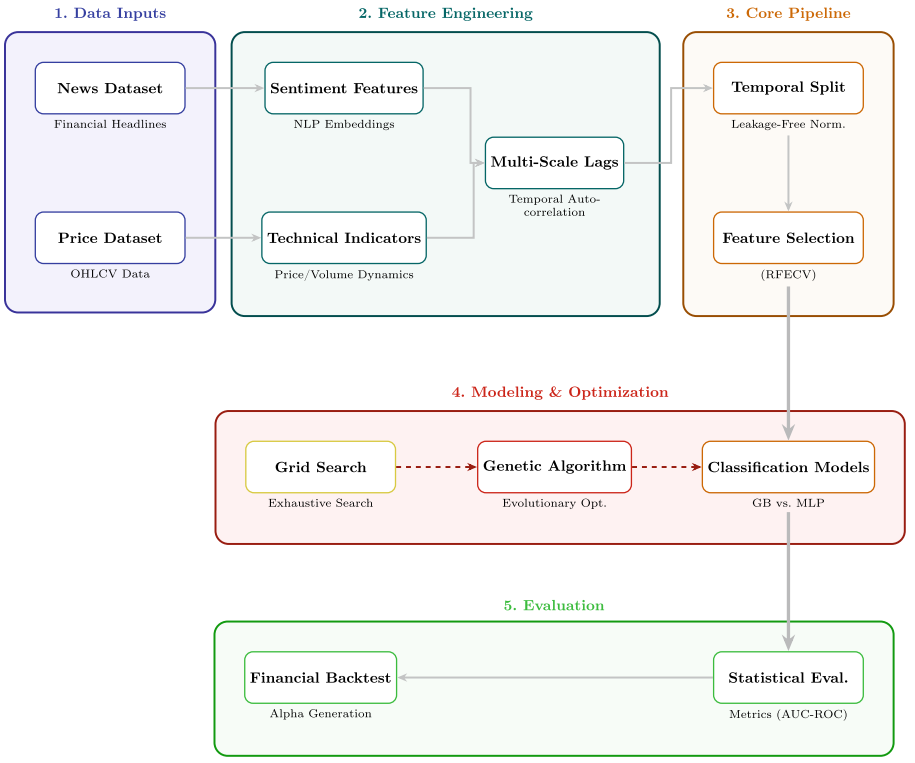

Fig. 1. Overview of our Leakage-Free Evolutionary Alpha pipeline with a two-layered module architecture. The upper layer flows horizontally from raw datasets through feature engineering to the core processing pipeline. A vertical main arrow connects to the lower layer, which integrates model training, comparative hyperparameter optimization, and final statistical and financial performance evaluation.

4.2 Feature Engineering

Raw inputs are transformed into a rich, multi-modal feature space spanning three complementary information sources: NLP-derived sentiment from news headlines, price- and volume-based technical indicators, and multi-scale temporal lags that encode autocorrelation structure. Together they produce **109 candidate features**.

Sentiment Features via FinBERT. Financial news headlines carry a substantial predictive signal but require a domain-adapted language model to properly decode financial vocabulary and tone. The extraction of sentiment signals was performed using the *FinBERT-Tone* model[1], a BERT-based architecture. While more recent large language models (LLMs) and generative architectures

[1] Available at: https://huggingface.co/yiyanghkust/finbert-tone.

have emerged since 2020, we opted for *FinBERT* due to its specialized fine-tuning on financial corpora and its superior computational efficiency. This balance is critical for an evolutionary pipeline, where sentiment extraction must be integrated into an optimization loop without prohibitive latency. Furthermore, as our dataset spans 2008 to 2016, *FinBERT* effectively captures the language patterns of this period without requiring the excessive world knowledge of larger generative models. This resolves the ambiguity of terms such as "liability" or "share", which have neutral connotations in common English but carry strong semantic weight in a market context.

Given a headline H, the model processes the token sequence and outputs logits for three classes: Positive (z_{pos}), Negative (z_{neg}), and Neutral (z_{neu}). We compute the likelihood of each class by passing the model's output through a Softmax layer. This step gives us the following probabilities:

$$P(c) = \frac{e^{z_c}}{\sum_{k \in \{pos,neg,neu\}} e^{z_k}} \tag{2}$$

Working with three separate dimensions per news item complicates the feature space. Therefore, we compress these outputs into a *Compound Score* (S). This metric maps the headline's mood onto a $[-1, 1]$ scale by taking the difference between the positive and negative likelihoods:

$$S = P(pos) - P(neg) \tag{3}$$

The original dataset provides 25 headlines per day ($H_1, \ldots, H_{25}$). To convert this text corpus into a structured format suitable for supervised classification models, we ran 26 statistical transformations grouped into four categories: Central Tendency, Distribution, Raw Probabilities, and Temporal Dynamics. One contribution of this work is the *Weighted Sentiment*, which gives lower weights to headlines based on their position in the daily ranking, reflecting the idea that investors tend to pay more attention to top-ranked news. The weight w_i of headline i is given by:

$$w_i = e^{-0.3 \cdot (i-1)} \tag{4}$$

Table 1 presents a comprehensive description of all extracted features. All temporal features are computed strictly in chronological order, ensuring no future information contaminates the feature matrix.

Technical Indicators. To complement the textual signal with price-based market dynamics, we compute **39 technical indicators** across five categories, with all price-derived quantities expressed as normalised ratios to ensure stationarity and cross-period comparability:

– **Trend** (13): Close-to-SMA ratios for windows $\{5, 20, 50, 100\}$; Close-to-EMA ratios for windows $\{12, 26\}$; SMA crossover ratios (5/20, 20/50, 50/100); binary signals for price above SMA_{20}, SMA_{50}, SMA_{100}, and a fast golden cross ($\text{SMA}_{20} > \text{SMA}_{50}$).

Table 1. Comprehensive Description of NLP Sentiment Features

Feature (Code)	Calculation Description	Academic Justification
1. Central Tendency and Weighting		
sent_mean_compound	Arithmetic mean of S scores.	Daily sentiment baseline (equal weighting).
sent_median_compound	Median of S scores.	Robustness against outliers and isolated extreme headlines.
sent_weighted_mean	$\frac{\sum(S_i \cdot w_i)}{\sum w_i}$	Simulates attention bias (top headlines carry more weight).
sent_weighted_std	Standard deviation weighted by w_i.	Measures opinion divergence in major news.
sent_std_compound	Simple standard deviation.	Indicator of ambiguity and noise in the news flow.
sent_min_compound	Lowest S value of the day.	Tail risk detector (Panic/FUD).
sent_max_compound	Highest S value of the day.	Euphoria detector (FOMO).
2. Raw Probabilities		
sent_mean_pos	Mean of $P(pos)$ probabilities.	Isolates the pure optimism signal.
sent_mean_neg	Mean of $P(neg)$ probabilities.	Isolates the fear signal (Loss Aversion).
sent_mean_neu	Mean of $P(neu)$ probabilities.	Indicates the absence of fundamental drivers.
3. Distribution and Shape		
sent_pct_positive	% of news with $S > 0.1$.	Market breadth (Generalized optimism).
sent_pct_negative	% of news with $S < -0.1$.	Breadth of pessimism.
sent_pct_neutral	% of news between -0.1 and 0.1.	Market inertia and indecision.
sent_dominance	S value with the highest absolute $\|S\|$.	Identifies if the day is driven by a strong single event.
sent_consistency	$1/(1+\sigma_s)$.	Signal confidence (Consensus among sources).
sent_skewness	Skewness of the S distribution.	Detects hidden risks on apparently neutral days.
sent_range_norm	$S_{max} - S_{min}$.	Polarization and narrative conflict.
4. Temporal Dynamics (Time Series)		
sent_mean_change	$S_t - S_{t-1}$ (1st Derivative).	Captures the change in expectations (Delta).
sent_mean_change_pct	$(S_t - S_{t-1})/\|S_{t-1}\|$	Percentage change in mood (Relative sensitivity).
sent_acceleration	$\Delta S_t - \Delta S_{t-1}$ (2nd Derivative).	Strength of trend reversal.
sent_momentum	Rolling sum of S (3 d).	Recent cumulative psychological effect.
sent_volatility_5d	Rolling standard deviation (5 d).	Market uncertainty (Weekly).
sent_volatility_10d	Rolling standard deviation (10 d).	Market uncertainty (Bi-weekly).
sent_vs_sma5	$S_t - \text{SMA}_5(S)$.	Mean Reversion indicator (Short-term).
sent_vs_sma10	$S_t - \text{SMA}_{10}(S)$.	Mean Reversion indicator (Medium-term).

- **Momentum** (11): RSI_{14} with overbought/oversold flags; MACD, signal line, and histogram normalised by closing price with a bullish crossover flag; ROC_{12}; Stochastic Oscillator (K, D, bullish flag).
- **Volatility** (7): Bollinger Band upper and lower boundaries normalised by closing price; band width; relative position within the band $\in [0, 1]$; above-upper and below-lower breakout flags; ATR_{14} normalised by closing price.
- **Volume** (4): OBV percentage change; 20-day volume ratio; high-volume binary flag (ratio > 1.5); MFI_{14}.
- **Trend strength** (4): ADX_{14}, positive DI, negative DI, and a strong-trend flag (ADX > 25).

Multi-scale Lag Features. To explicitly encode short- and medium-term autocorrelation structure [12], we augment the feature matrix with lagged OHLCV values and lagged daily returns across three temporal scales:

- **Ultra-short term** ($\ell \in \{1, 2\}$ days): captures mean-reversion signals immediately following large price moves.

- **Short term** ($\ell \in \{3, 4, 5\}$ days): encodes weekly momentum consistent with documented serial autocorrelation in equity returns.
- **Medium term** ($\ell \in \{10, 20\}$ days): reflects bi-weekly and monthly market cycles.

Applying the seven lag values to the five OHLCV series and to the daily return series yields **42 lag features**. Rows containing NaN values introduced by lagging are removed before any model fitting, preserving strict chronological integrity.

4.3 Core Pipeline

Temporal Split and Leakage-Free Normalization. Financial time series exhibit strong serial dependence, making standard random K-Fold cross-validation inappropriate [6]. We adopt a strict chronological partitioning of the 1,989 aligned trading days:

- **Training set (70%)**: used exclusively for model fitting, scaler calibration, and RFECV feature selection.
- **Validation set (15%)**: used for model selection and hyperparameter tuning.
- **Test set (15%)**: held out until final evaluation; never exposed to any fitting procedure, corresponding to the period from May 2015 to June 2016.

Normalization is performed using a `StandardScaler` fitted *exclusively* on the training partition. The learned statistics (μ_{train}, σ_{train}) are then applied to transform the validation and test sets:

$$x' = \frac{x - \mu_{\text{train}}}{\sigma_{\text{train}}}. \tag{5}$$

This design simulates a realistic deployment scenario in which future price statistics are unknown at prediction time, thereby eliminating a common but subtle source of data leakage [13].

Feature Selection via RFECV. With 109 candidate features, irrelevant and redundant inputs risk degrading generalisation through the curse of dimensionality. We apply Recursive Feature Elimination with Cross-Validation (RFECV) [11] exclusively within the training set, using a 5-fold time-series-aware scheme that preserves chronological order across folds.

Three estimators serve as wrapper models: Logistic Regression, Random Forest, and Gradient Boosting. Each wrapper eliminates features one at a time (step = 1) and selects the minimal subset that maximises 5-fold cross-validated accuracy on the training partition. The wrapper with the best validation-set accuracy sets the final feature subset shared across all classifiers, so that performance differences come from the learners themselves and not from differences in the input data. To run a fair comparison against the tree-based ensemble, the architecture goes through hyperparameter tuning via Genetic Algorithms

(GA). Unlike Grid Search, which is computationally costly and tends to overfit, the evolutionary approach uses natural selection to explore the hyperparameter search space. To reduce the risk of overfitting in financial time series, the GA relies on a penalized fitness function. Instead of only maximizing the cross-validated validation metric, the objective function penalizes the Generalization Gap, which is the difference between the training performance and the validation performance. Here, $\boldsymbol{\theta}$ stands for the hyperparameter vector of a given individual in the population. The fitness function $F(\boldsymbol{\theta})$ is defined as:

$$F(\boldsymbol{\theta}) = \mathrm{AUC}_{\mathrm{val}}(\boldsymbol{\theta}) - \lambda \cdot \max\left(0, \mathrm{AUC}_{\mathrm{train}}(\boldsymbol{\theta}) - \mathrm{AUC}_{\mathrm{val}}(\boldsymbol{\theta})\right). \tag{6}$$

In this formula, $\mathrm{AUC}_{\mathrm{train}}(\boldsymbol{\theta})$ and $\mathrm{AUC}_{\mathrm{val}}(\boldsymbol{\theta})$ represent the average ROC-AUC scores computed during the training and validation stages of the TimeSeriesSplit. The λ parameter controls the strength of this regularization. We used a value of 0.5 for λ throughout our tests. This choice actively filters out candidate models that overfit the historical data while failing to generalize to new data.

4.4 Modeling and Optimization

We evaluate a diverse set of classifiers representing distinct families of inductive bias. This comparative approach ensures that the performance gains derived from the feature engineering pipeline are rigorously validated across fundamentally different learning paradigms:

- **Logistic Regression (LR)**: A linear baseline providing interpretable coefficients and a lower-bound performance reference.
- **Random Forest (RF)**: A bagging ensemble providing variance reduction through feature randomization [5], offering inherent robustness to noise.
- **Gradient Boosting (GB)**: A sequential additive ensemble that minimizes a differentiable loss through boosting. Due to its recognized superiority in modeling noisy tabular data [7], GB is treated as our primary tree-based architecture.
- **Multi-Layer Perceptron (MLP)**: A feed-forward neural network utilized not merely as a comparative baseline, but to explicitly capture highly non-linear, asymmetric interactions between NLP sentiment shocks and technical dynamics that rigid tree-based splits might fail to isolate.

All models are configured with `class_weight='balanced'` to compensate for any residual class imbalance in the training folds. Crucially, during the training phase, the magnitude-based sample-weight vector defined in Eq. 1 is passed to the objective function. This Cost-Sensitive Learning approach directs the gradient descent to penalize misclassifications on economically significant days more heavily.

Evolutionary Hyperparameter Optimization. Financial time series tend to overfit easily, and using default hyperparameters can create a large Generalization Gap between in-sample memorization and out-of-sample performance. To handle this, both core architectures (Gradient Boosting and MLP) go through a careful hyperparameter tuning process. Traditional methods, such as Grid Search, tend to reach overfitted solutions by simply maximizing validation metrics without considering the noise present in financial data. To move past this limitation, we use an Evolutionary Optimization approach via Genetic Algorithms (GA) combined with a chronological *TimeSeriesSplit* cross-validation scheme. The evolutionary process relies on a penalized fitness function that accounts for the generalization gap, pushing the algorithm to drop configurations that fit the training data well but struggle on validation. For the Gradient Boosting model, the GA searches through the regularization space, tuning structural parameters such as maximum tree depth, learning rate, and subsample ratios. For the MLP architecture, the GA looks into the network topology space, tuning hidden layer dimensions, L_2 regularization limits, learning rates, and solver dynamics. This approach helps the final models find a balance between predictive power and generalization needed to produce steady out-of-sample financial gains.

4.5 Evaluation

Accuracy and AUC-ROC measure statistical discrimination but do not directly quantify whether a model generates exploitable profit in a realistic trading setting. We therefore complement all classification metrics with a **Financial Backtest** that translates predictions into cumulative returns, which constitutes the primary evaluation criterion of this work.

The strategy adopts a *Trend Following* logic applied to the optimized models on the held-out test set. At each trading day t, the model outputs a probability $\hat{p}_t = P(\text{Label} = 1 \mid \mathbf{x}_t)$. The resulting market position is determined by a dynamic decision threshold τ:

$$\text{pos}_t = \begin{cases} +1 \text{ (long)} & \text{if } \hat{p}_t > \tau, \\ -1 \text{ (short)} & \text{otherwise.} \end{cases} \tag{7}$$

The daily strategy return is $r_t^s = \text{pos}_t \times R_{t+1}$, and the cumulative capital curve is:

$$W_T = \prod_{t=1}^{T} (1 + r_t^s), \quad W_0 = 1. \tag{8}$$

The benchmark is a passive Buy-and-Hold strategy, $W_T^{\text{BH}} = \prod_{t=1}^{T}(1+R_{t+1})$, and the excess return (alpha) over the test period is reported as $\alpha = W_T - W_T^{\text{BH}}$.

To prevent the strategy from being systematically biased toward long or short positions due to the inherent class probability shift of the evolved models, we do not rely on a fixed 0.50 decision boundary. Instead, we implement a *Dynamic Threshold* mechanism. The decision threshold τ is set to the median prediction

probability of the validation set, keeping market exposure balanced while avoiding look-ahead bias. The strategy works with unit position sizing and leaves transaction costs out, choices that fit the scope of this work, which focuses on measuring the raw predictive alpha of the pipeline.

5 Results and Discussion

In this section, we present the empirical results of the proposed methodology, divided into three fundamental stages: dimensionality reduction via RFECV, hyperparameter optimization guided by a Genetic Algorithm (GA), and finally, the out-of-sample performance evaluation contrasting traditional statistical metrics with actual financial returns (Backtest).

5.1 Feature Selection and Dimensionality Reduction

The integration of natural language processing embeddings (FinBERT) with traditional tabular variables generated an initial vector space of 109 features. Since tree-based models are susceptible to overfitting in high-dimensional spaces with inherent noise, the Recursive Feature Elimination with Cross-Validation (RFECV) method was applied.

Figure 2 illustrates the selection process for the Gradient Boosting model. The algorithm identified an optimal subset of only 21 features, achieving a Mean CV Accuracy of 0.5180. The green shaded area represents the standard deviation across the cross-validation folds, highlighting that the removal of 88 redundant variables not only reduced computational complexity but also stabilized the model's variance regarding unseen data.

5.2 Evolutionary Optimization of Hyperparameters

With the reduced feature space, the Gradient Boosting model was subjected to optimization via a Genetic Algorithm. To mitigate the memorization of training data (overfitting), the fitness function was modeled as a penalized metric, subtracting the Gap between the Train AUC and Validation AUC from the main metric, as detailed in Sect. 4.

Figure 3 presents the optimization profile. The Early Stopping mechanism was triggered at the 32nd generation, after 25 generations without significant improvements, proving the algorithm's mathematical convergence. The optimal individual achieved a converged fitness of 0.3809, balancing predictive capacity with strong structural regularization (n_estimators=100, max_depth=2, subsample=0.7).

To provide full transparency regarding the evolutionary process, Table 2 details the search space constraints and the configuration of the Genetic Algorithm, alongside the final optimized hyperparameters obtained for the winning Gradient Boosting model. The algorithm successfully identified an optimal configuration characterized by strong structural regularization (e.g., shallow trees with `max_depth`=2 and a conservative `learning_rate`=0.01).

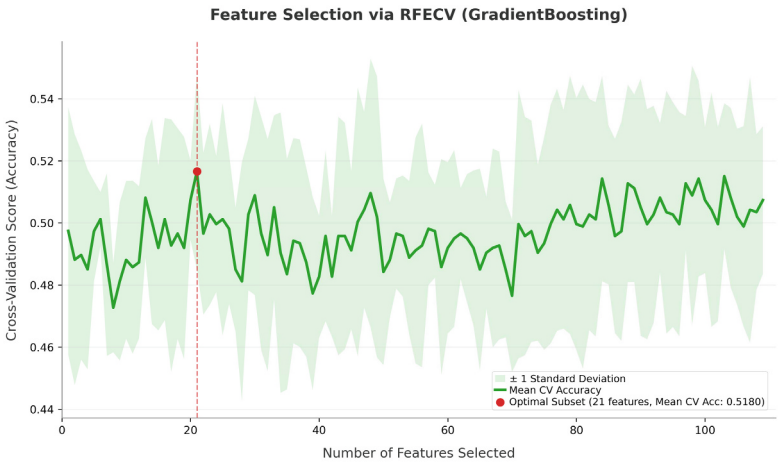

Fig. 2. Feature Selection via RFECV for the Gradient Boosting model. The green line indicates the mean accuracy and the shaded area represents ±1 standard deviation. The red dot highlights the optimal subset of 21 variables. (Color figure online)

Table 2. Genetic Algorithm Configuration, Search Space, and Optimized Parameters for the Gradient Boosting Classifier.

Category	Parameter	Search Space/Value	Optimized Value
GA Settings	Population Size	25	–
	Max Generations	50	–
	Crossover Prob. (*cxpb*)	0.60	–
	Mutation Prob. (*mutpb*)	0.30	–
	Early Stopping Patience	25 generations	–
Search Space (Genes)	`n_estimators`	[100, 150, 200, 250, 300, 350]	100
	`learning_rate`	[0.01, 0.05, 0.08, 0.1]	0.01
	`max_depth`	[2, 3, 4, 5]	2
	`subsample`	[0.6, 0.7, 0.8, 0.9]	0.7
	`min_samples_split`	[5, 10, 15, 20]	20

Comparative Analysis: Exhaustive Grid Search vs. Evolutionary Optimization. To empirically validate the necessity of the penalized evolutionary approach, we benchmarked the Genetic Algorithm against a standard exhaustive Grid Search. The Grid Search evaluated the entire parameter space (1,536 combinations) utilizing a standard TimeSeriesSplit cross-validation, aiming to maximize the ROC-AUC score without any penalty for the generalization gap.

Table 3 presents the stark contrast between the two optimization paradigms.

The exhaustive Grid Search evaluated all 1,536 possible parameter combinations and converged to a highly aggressive configuration (`learning_rate`= 0.1, `max_depth` = 3). This resulted in an exceptional in-sample performance (Train

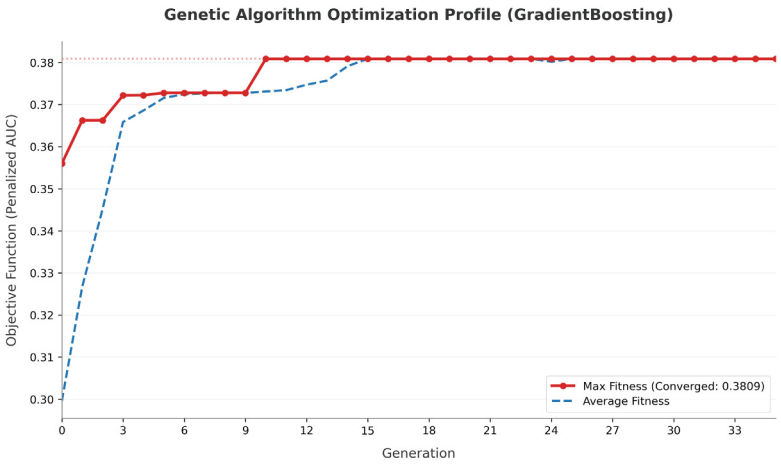

Fig. 3. Genetic Algorithm evolution profile. Absolute convergence occurred before the 40th generation, with the population average line (dashed) matching the global maximum of the penalized objective function.

Table 3. Performance comparison between Exhaustive Grid Search and the Evolutionary Genetic Algorithm for the Gradient Boosting classifier.

Metric/Parameter	Grid Search (Exhaustive)	Genetic Algorithm
Optimized Hyperparameters		
`n_estimators`	100	100
`learning_rate`	0.10	0.01
`max_depth`	3	2
`subsample`	0.6	0.7
`min_samples_split`	20	20
Statistical Performance		
Train AUC (In-Sample)	**0.9117**	0.7118
Validation AUC (In-Sample)	**0.9088**	0.5219
Test AUC (Out-of-Sample)	0.4716	**0.5041**

AUC > 0.91), creating a severe illusion of predictability. However, when exposed to the out-of-sample test set, the model's Test AUC dropped to 0.4716, falling below random guess level. This points to a known problem when applying traditional hyperparameter tuning methods to financial time series: the optimization algorithm ends up memorizing market noise. The Genetic Algorithm, guided by the gap-penalized fitness function (Eq. 6), leaned toward a more conservative setup (`learning_rate`=0.01, `max_depth`= 2). Instead of covering the full search space, the GA reached convergence through early stopping at generation 32. By accepting a lower in-sample accuracy (Train AUC 0.7118), the GA kept variance

under control and avoided picking up stochastic noise. As a result, the model was able to generalize to unseen test data, reaching a Test AUC of 0.5041 and keeping a positive predictive edge over the random walk baseline.

5.3 Financial Backtest and the Statistical-Financial Paradox

Accuracy and AUC-ROC measure statistical discrimination but fail to quantify the economic viability of a model, as they implicitly assume symmetric misclassification costs. To address this, we executed an out-of-sample financial backtest evaluating cumulative returns over the test set, contrasting the optimized Gradient Boosting with an identically evolved Multi-Layer Perceptron (MLP) neural network.

A critical mechanism implemented in our backtesting framework is the *Dynamic Threshold*. Financial sentiment probabilities rarely oscillate symmetrically around the theoretical 0.50 boundary. To prevent the strategy from being systematically biased toward long or short positions due to the inherent class probability shift of the evolved models, the decision threshold was dynamically calibrated as the median probability of the validation set predictions. For the Gradient Boosting model, the dynamic threshold was set at 0.5113, and for the MLP, at 0.5187.

Figures 4 and 5 illustrate the equity curves of the proposed strategies benchmarked against a passive Buy & Hold approach over the 279-day test period.

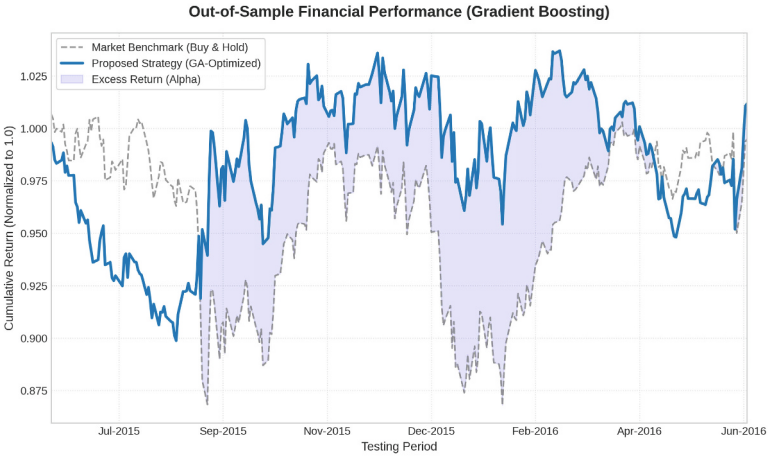

Fig. 4. Out-of-Sample equity curve for the GA-Optimized Gradient Boosting strategy. The model exhibited a highly conservative trading pattern, resulting in a marginal positive Alpha.

The Gradient Boosting model (Test AUC = 0.5041) exhibited a highly conservative behavioral profile. While it successfully navigated the market downturns

without catastrophic losses, its strict variance control prevented the algorithm from fully exploiting high-magnitude volatility events. Operating 189 d on the long side and 90 d on the short side, the strategy yielded a final return of 1.18%, translating to a modest, albeit positive, Alpha of **1.69 p.p.** against the market baseline (–0.51%).

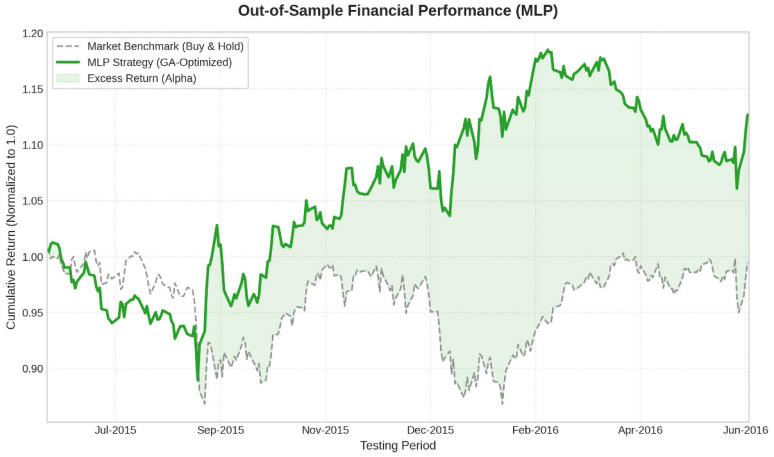

Fig. 5. Out-of-Sample equity curve for the GA-Optimized Multi-Layer Perceptron. Despite poor statistical performance, the neural network effectively captured high-magnitude asymmetric market shifts.

Conversely, the MLP presented a statistical metric indicative of failure (Test AUC = 0.4812). However, its financial simulation told a different story. Through evolutionary optimization, the GA identified a topology of 64–32 units, utilizing the `sgd` solver with a learning rate of 0.001 and an L_2 penalty (α) of 0.0001. Unbound by the rigid splits of decision trees, this architecture was able to capture non-linearities and asymmetric market anomalies dictated by the *FinBERT* sentiment signals. By identifying and trading on "fat-tail" events, the MLP generated a final return of 12.76%, reaching a $+$**13.27 p.p.** Alpha.

These empirical results highlight a critical challenge in computational finance: symmetric statistical metrics, such as AUC-ROC, can be insufficient on their own to attest to the economic viability of trading algorithms. As demonstrated by our findings, the practical success of NLP-driven models relies not necessarily on forecasting the majority of low-volatility days, but rather on capturing market asymmetries and utilizing dynamic probability thresholds to exploit extreme stochastic divergences.

6 Conclusions

This study tackled the problem of predicting the DJIA under noisy market conditions, where symmetric evaluation metrics fall short. To address this, we proposed LEAP, a leakage-free pipeline that brings together FinBERT sentiment, technical indicators, and temporal lags. We reduced overfitting through RFECV and a Genetic Algorithm guided by a generalization-penalized fitness function. Our results brought up an interesting statistical-financial paradox: while Gradient Boosting showed better discrimination (Test AUC 0.5041, Alpha +1.69 p.p.), the Multi-Layer Perceptron (MLP) statistically "failed" (Test AUC 0.4812) yet took advantage of market asymmetries via cost-sensitive learning to produce a **+13.27 p.p.** Alpha. These findings suggest that relying solely on symmetric metrics (e.g., AUC-ROC) can mislead the evaluation of trading algorithms, pointing to the need for magnitude-aware training. Future work could move toward a continuous Cost-Sensitive Learning framework, scaling instance weights by their volatility Z-score ($w_t = 1 + \alpha \frac{|R_t| - \mu_{|R|}}{\sigma_{|R|}}$). Bringing in intra-day data and sequence-to-sequence architectures may help improve alpha extraction from high-frequency news flows. Additionally, ablation studies isolating the individual contribution of each pipeline component, such as the sentiment features, the cost-sensitive weights, and the evolutionary optimization, would further strengthen the understanding of the framework and are left as a direction for future work.

Acknowledgment. The authors would like to thank FAPERGS (24/2551-0001396-2, 23/2551-0000773-8), CNPq (307416/2025-9), FAPERGS/CNPq (23/2551-0000126-8) and the Fesurv-University of Rio Verde.

References

1. Allen, F., Karjalainen, R.: Using genetic algorithms to find technical trading rules. J. Financ. Econ. **51**(2), 245–271 (1999)
2. Bailey, D.H., Borwein, J.M., Lopez de Prado, M., Zhu, Q.J.: Pseudo-mathematics and financial charlatanism: the effects of backtest overfitting. Notices AMS **61**(5), 458–471 (2014)
3. Bergstra, J., Bengio, Y.: Random search for hyper-parameter optimization. J. Mach. Learn. Res. **13**, 281–305 (2012)
4. Bragança, H., Pinto, R., Dalmazo, B., Lucca, G., Borges, E., Mattos, V.L.D., Berri, R.: Enhancing stock market predictions: the role of feature selection techniques in financial modeling. In: Gervasi, O., et al. (eds.) ICCSA 2025. LNCS, vol. 15649. Springer, Cham (2025). https://doi.org/10.1007/978-3-031-96997-3_24
5. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001)
6. Cerqueira, V., Torgo, L., Mozetič, I.: Evaluating time series forecasting models: an empirical study on performance estimation methods. Mach. Learn. **109**(11), 1997–2028 (2020)
7. Chen, T., Guestrin, C.: Xgboost: a scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794 (2016)

8. Cont, R.: Empirical properties of asset returns: stylized facts and statistical issues. Quantitative Finance **1**(2), 223–236 (2001)
9. Fama, E.F.: Efficient capital markets: a review of theory and empirical work. J. Financ. **25**(2), 383–417 (1970)
10. Gu, S., Kelly, B., Xiu, D.: Empirical asset pricing via machine learning. J. Financ. **75**(1), 2223–2273 (2020)
11. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. Mach. Learn. **46**(1–3), 389–422 (2002)
12. Jegadeesh, N.: Evidence of predictable behavior of security returns. J. Financ. **45**(3), 881–898 (1990)
13. Lopez de Prado, M.: Advances in Financial Machine Learning. John Wiley & Sons (2018)
14. Sezer, O.B., Gudelek, M.U., Ozbayoglu, A.M.: Financial time series forecasting with deep learning: a systematic literature review: 2005–2019. Appl. Soft Comput. **90**, 106181 (2020)
15. Souza, A.V.Set al.: Predictive analysis with technical indicators and features selection for futures contracts trading. In: Gervasi, O., et al. (eds.) ICCSA 2025. LNCS, vol. 15649. Springer, Cham (2025). https://doi.org/10.1007/978-3-031-96997-3_22
16. Yang, Y., Uy, M.C.S., Huang, A.: Finbert: a large language model for extracting information from financial text. Contemp. Account. Res. **37**(2), 900–922 (2020)