Predicting Analyst Performance: A Machine Learning Fund-of-Analysts Framework

Matéo Molinaro - Thibault Charbonnier

08 January 2026

Contents

1 Abstract

Inspired by Alpha in Analysts (2025), this paper investigates whether the cross-sectional and time-series heterogeneity in sell-side analyst skill can be exploited using modern machine learning techniques. We treat each analyst as a portfolio manager and construct analyst-level portfolios based on their 12-month target price recommendations, translating implied returns into systematic long–short positions. The objective is to predict one-month-ahead 12-months cumulative returns of these analyst portfolios using a rich set of analyst and portfolio-level characteristics.

We extend the original framework along three key dimensions. First, we expand the information set from six to twenty-four predictive features capturing analyst behavior, portfolio composition, and historical performance. Second, we move from rolling ordinary least squares to an expanding walk-forward cross-validation framework, allowing for realistic real-time prediction and model selection. Third, we compare linear and non-linear models by evaluating 8 forecasting approaches: OLS benchmarks with

Ridge, Lasso, and Elastic Net regularization, as well as Random Forests, XGBoost, LightGBM, and multilayer perceptrons.

Our empirical analysis covers 1,474 unique analysts and 105,866 analyst-month observations. We find that, although machine learning provides only modest improvements in point forecasting accuracy, it delivers large gains in cross-sectional ranking ability, a key input for analyst- selection and long–short investment strategies. However, for now, these gains difficulty translate into economically meaningful improvements in a dynamic "fund-of-analysts" allocation strategy that weights analysts based on predicted performance.

2 Data

The sample consists of monthly observations from January 1999 to December 2024. Equity return and price data are obtained from the Center for Research in Security Prices (CRSP) and include all ordinary common shares listed on the NYSE, AMEX, and NASDAQ. Analyst target price data are drawn from the Institutional Brokers' Estimate System (I/B/E/S). All datasets are accessed via the Wharton Research Data Services (WRDS) platform. The CRSP–I/B/E/S linkage follows standard procedures as documented by WRDS [1]. Our code is available on GitHub [2].

3 Methodology

3.1 Entities, timeline and notation

We follow the setup of *Alpha in Analysts* and adapt it to a monthly rebalancing grid. We worked with the following entities :

- Let $m \in \{1, \dots, M\}$ rebalancing months and let t_m denote the *last NYSE trading day* of calendar month m. As we worked only with US stocks for this study we decided to use US last trading day of month to gather ending month data for the rebalancing.
- We denote by $i \in \{1, \dots, I\}$ an analyst
- We denote by $k \in \{1, \dots, N\}$ a stock.
- At each month m, analyst i covers a (time-varying) set of stocks $\mathcal{S}_{i,m} \subseteq \{1, \dots, N\}$.
- Let P_k^m be the observed stock price at date t_m
- Let $\widehat{P}_{i,k}^{m+12}$ the 12-month ahead target price issued by analyst i (for stocks $k \in \mathcal{S}_{i,m}$).

3.2 Analyst-implied self-financing long–short books

For each covered stock $k \in \mathcal{S}_{i,m}$, the target price implies a 12-month return forecast

$$\widehat{R}_{i,k}^{m+12} = \frac{\widehat{P}_{i,k}^{m+12} - P_k^m}{P_k^m}. \tag{1}$$

We split the covered universe into positive and negative-forecast buckets:

$$\mathcal{K}_{i,m}^{+} = \{k \in \mathcal{S}_{i,m} : \widehat{R}_{i,k}^{m+12} > 0\}, \qquad \mathcal{K}_{i,m}^{-} = \{k \in \mathcal{S}_{i,m} : \widehat{R}_{i,k}^{m+12} < 0\}. \tag{2}$$

A stock is included in the coverage set $\mathcal{K}_{i,t}$ if and only if the analyst issues a valid target price and a corresponding market price is available in CRSP. Given the near-complete coverage of CRSP prices, this provides a reliable proxy for analyst coverage.

The analyst-implied book is constructed as a self-financing long–short portfolio whose weights are proportional to the magnitude of the implied return forecasts.

[1] `https://wrds-www.wharton.upenn.edu/pages/wrds-research/applications/python-replications/linking-ibes-and-crsp-data-python/`

[2] `https://github.com/thibault-charbonnier/analysts-alpha-ml-signals`

For each $k \in \mathcal{K}_{i,m}^{+}$, define the long-bucket weights:

$$w_{i,k,m}^{+} = \frac{\widehat{R}_{i,k}^{m+12}}{\sum_{j \in \mathcal{K}_{i,m}^{+}} \widehat{R}_{i,j}^{m+12}}. \tag{3}$$

For each $k \in \mathcal{K}_{i,m}^{-}$, define the short-bucket weights:

$$w_{i,k,m}^{-} = \frac{-\widehat{R}_{i,k}^{m+12}}{\sum_{j \in \mathcal{K}_{i,m}^{-}} -\widehat{R}_{i,j}^{m+12}}. \tag{4}$$

Whenever both $\mathcal{K}_{i,m}^{+}$ and $\mathcal{K}_{i,m}^{-}$ are non-empty, these satisfy $\sum_{k \in \mathcal{K}_{i,m}^{+}} w_{i,k,m}^{+} = 1$ and $\sum_{k \in \mathcal{K}_{i,m}^{-}} w_{i,k,m}^{-} = 1$, so the long–short book is self-financing. The resulting stock-level book weights are

$$w_{i,k,m} = \begin{cases} w_{i,k,m}^{+} & \text{if } k \in \mathcal{K}_{i,m}^{+}, \\ -\, w_{i,k,m}^{-} & \text{if } k \in \mathcal{K}_{i,m}^{-}, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

(*Remark:* if one bucket is empty, the portfolio is no longer self-financing and becomes net long or net short. With a sufficiently large dataset of analysts it should reasonably not happen.)

3.3 Portfolio Rebalancing and Holding Period

Analysts do not update target prices on a fixed monthly schedule. Updates are irregular and stock-dependent, reflecting quarterly information releases or firm-specific events. Consequently, analyst portfolios cannot be assumed to rebalance at uniform or synchronized intervals. To address this issue, when an analyst issues a 12-month target price at time t, we let the associated portfolio weight drift for up to 12 months. If the analyst revises the target price for the same stock before the 12-month horizon expires, the new recommendation overrides the previous one and the portfolio is rebalanced accordingly. If no update occurs within 12 months, the stock is considered out of coverage and its weight is set to nan. On could also argue that a 6 month old target price doesn't translate as much confidence in the position than a 1 week old one. That's why we also used a exponential decay of the implied return. This approach is consistent with the economic interpretation of a 12-month price target and allows sufficient time for the recommendation to materialize. Transaction costs of 10 basis points are applied at each rebalancing event. A pseudocode for the rebalancing algorithm is provided in the appendix: 6.3

3.4 Realized returns and analyst P&L

Let P_k^{m+12} be the stock price observed 12 months after m (between t_m and t_{m+12}). The realized 12-month return is

$$R_k^{m+12} = \frac{P_k^{m+12} - P_k^{m}}{P_k^{m}}. \tag{6}$$

The P&L of analyst i's implied long–short book formed at month m and marked to month $m+12$ is

$$\mathrm{PnL}_i^{m+12} = \sum_{k \in \mathcal{K}_{i,m}^{+}} w_{i,k,m}^{+}\, R_k^{m+12} - \sum_{k \in \mathcal{K}_{i,m}^{-}} w_{i,k,m}^{-}\, R_k^{m+12}. \tag{7}$$

3.5 Fund-of-analysts: allocating across analyst books

Assume an investor has access to all analysts and views each analyst as a portfolio manager. One of the main idea of the reference paper is to consider all analysts' implied portfolios as the "asset universe" and then invest a fraction of wealth in an analyst i at month m denoted c_i^m. From now we will call this approach the *meta-portfolio* or the *fund-of-analysts portfolio.*

The investor of this meta-portfolio will then earn :

$$\mathrm{PnL}^{m+12} = \sum_{i=1}^{I} c_i^m\, \mathrm{PnL}_i^{m+12}. \tag{8}$$

Meta-portfolio problem formulation. Starting from here, the problem becomes a classical asset allocation framework where we want to allocate weights on the assets (analysts or to be more exact the analysts' books) which we think they will perform in the future. The fundamental idea of the paper is to model analysts performance and transcript these views into meta-portfolio weights.

Predicting analyst performance. Analyst performance is modeled as a function of analyst characteristics A_i given the information set $\mathcal{F}_m$ available up to month m:

$$\mathrm{PnL}_i^{m+12} = g(A_i \mid \mathcal{F}_m)\,, \qquad \widehat{\mathrm{PnL}}_i^{\,m+12} \approx g(A_i \mid \mathcal{F}_m)\,. \tag{9}$$

Creating the meta-portfolio. Define the sets of analysts with positive/negative predicted performance:

$$\mathcal{I}_m^+ = \left\{ i : \widehat{\mathrm{PnL}}_i^{\,m+12} > 0 \right\}, \qquad \mathcal{I}_m^- = \left\{ i : \widehat{\mathrm{PnL}}_i^{\,m+12} < 0 \right\}. \tag{10}$$

The investor allocates long weights to analysts in $\mathcal{I}_m^+$ and short weights to those in $\mathcal{I}_m^-$:

$$c_{i,m}^+ = \frac{\widehat{\mathrm{PnL}}_i^{\,m+12}}{\sum_{j\in\mathcal{I}_m^+} \widehat{\mathrm{PnL}}_j^{\,m+12}} \quad (i \in \mathcal{I}_m^+), \qquad c_{i,m}^- = \frac{-\widehat{\mathrm{PnL}}_i^{\,m+12}}{\sum_{j\in\mathcal{I}_m^-} -\widehat{\mathrm{PnL}}_j^{\,m+12}} \quad (i \in \mathcal{I}_m^-), \tag{11}$$

so that $\sum_{i\in\mathcal{I}_m^+} c_{i,m}^+ = 1$ and $\sum_{i\in\mathcal{I}_m^-} c_{i,m}^- = 1$.

The realized informed fund-of-analysts P&L is then

$$\mathrm{PnL}_{m+12}^{\mathrm{Informed}} = \sum_{i\in\mathcal{I}_m^+} c_{i,m}^+ \,\mathrm{PnL}_i^{m+12} - \sum_{i\in\mathcal{I}_m^-} c_{i,m}^- \,\mathrm{PnL}_i^{m+12}. \tag{12}$$

Prediction Transformation. In practice, one could also consider a transformation of the predicted PnL to define the meta-portfolio weights. In our study we used a double sigmoid transformation to emphasize strong convictions both in long and short positions and also to flatten the weights for low predicted performance.

We consider the following function:

$$f(x) = \begin{cases} 0, & |x| \le d, \\ \sigma\big(k_+(x-d)\big) - \frac{1}{2}, & x > d, \\ -\Big(\sigma\big(k_-(-x-d)\big) - \frac{1}{2}\Big), & x < -d, \end{cases}$$

where :

- d = dead zone, where all prediction are converted to 0 (e.g. 5
- k_+ is a constant representing the steepness of the positive PnL mapping (e.g. 12.0)
- k_- is a constant representing the steepness of the negative PnL mapping (e.g. 12.0)

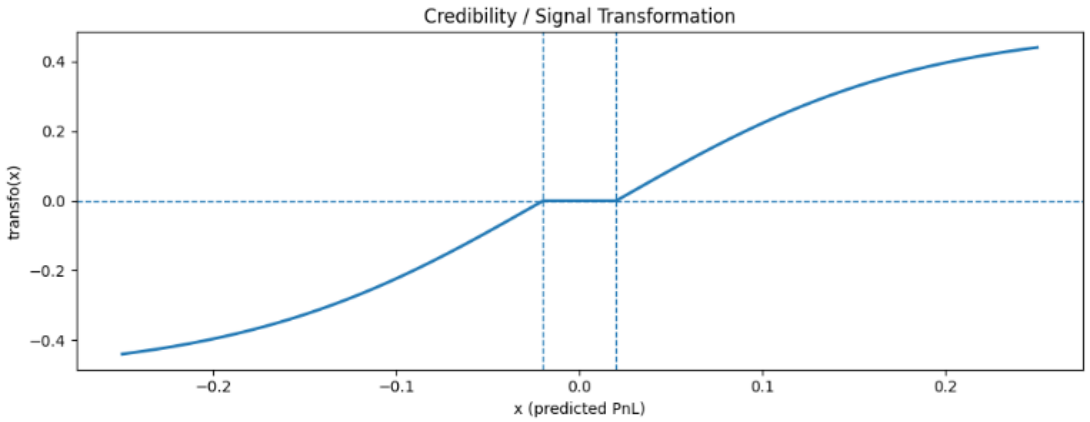

Figure 1: Transformation function applied to the predicted PnL.

3.6 From the paper's setup to our supervised learning dataset

The above construction yields a panel dataset indexed by (i, m):

- **Label:** $y_{i,m} := \mathrm{PnL}_i^{m+12}$ (one-year-ahead P&L of analyst i's implied book).
- **Features:** $X_{i,m}$ computed using information available up to month m.

Our ML models learn $X_{i,m} \mapsto \widehat{\mathrm{PnL}}_i^{\,m+12}$ and we then map predictions to analyst weights via Eq. (11) to build a tradable meta-portfolio.

3.7 Feature Construction

Using the analyst-level portfolio return series, we construct a set of performance and risk features for each analyst. For each base metric, we compute (i) the raw value, (ii) the cross-sectional (ascending: higher value, higher percentile) percentile across analysts, and (iii) two lookback horizons of 6 and 12 months. As a result, each base metric generates four distinct features.

The cumulative return over horizon h is defined as:

$$\mathrm{CumRet}_{i,t}^{(h)} = \prod_{s=t-h+1}^{t} (1 + R_{i,s}) - 1. \tag{13}$$

The mean return is given by:

$$\mathrm{Mean}_{i,t}^{(h)} = \frac{1}{h} \sum_{s=t-h+1}^{t} R_{i,s}. \tag{14}$$

Return volatility is computed as:

$$\mathrm{Vol}_{i,t}^{(h)} = \sqrt{\frac{1}{h-1} \sum_{s=t-h+1}^{t} \left(R_{i,s} - \mathrm{Mean}_{i,t}^{(h)}\right)^2}. \tag{15}$$

The Sharpe ratio, assuming a zero risk-free rate, is defined as:

$$\mathrm{Sharpe}_{i,t}^{(h)} = \frac{\mathrm{Mean}_{i,t}^{(h)}}{\mathrm{Vol}_{i,t}^{(h)}}. \tag{16}$$

The Sortino ratio is defined as:

$$\mathrm{Sortino}_{i,t}^{(h)} = \frac{\mathrm{Mean}_{i,t}^{(h)}}{\sqrt{\frac{1}{h} \sum_{s=t-h+1}^{t} \min(R_{i,s}, 0)^2}}. \tag{17}$$

Finally, analyst coverage is measured as:

$$\mathrm{Coverage}_{i,t}^{(h)} = \frac{1}{h} \sum_{s=t-h+1}^{t} |\mathcal{K}_{i,s}|. \tag{18}$$

Table 1: Analyst portfolio-based features

Feature	**Description**
Cumulative Return	Portfolio cumulative return over the lookback window
Mean Return	Average monthly portfolio return
Volatility	Standard deviation of portfolio returns
Sharpe Ratio	Mean return divided by volatility (risk-free rate = 0)
Sortino Ratio	Mean return divided by downside volatility
Coverage	Average number of stocks covered by the analyst

You can find below an extract of the dataframe feeding the different models. The dataframe is cropped for visual reasons but an image with all the columns is made available in the appendix 6.1.

	date	analyst_id	perf_12m_pct	perf_12m	perf_6m_pct	...	coverage_12m	coverage_12m_pct	coverage_6m	coverage_6m_pct	y
0	2002-01-31	16	0.658537	0.046179	0.501122	...	1.750000	0.971030	2.000000	0.968180	-0.186680
1	2002-01-31	51	0.825203	0.299013	0.433807	...	1.083333	0.946984	0.833333	0.897129	-0.032894
2	2002-01-31	118	0.491870	-0.133531	0.394914	...	2.833333	0.990153	3.166667	0.988340	-0.078027
3	2002-01-31	127	0.544715	-0.070777	0.521316	...	1.083333	0.946984	1.000000	0.919983	-0.042041
4	2002-01-31	177	0.282520	-0.389486	0.287958	...	1.500000	0.963827	1.666667	0.957245	0.657964
...	...	...	...	...	...	...	...	...	...	...	...
105861	2024-12-31	115583	0.743711	0.189328	0.547358	...	1.916667	0.888008	2.333333	0.901741	-0.514230
105862	2024-12-31	122739	0.271488	-0.136930	0.322228	...	7.083333	0.989065	8.000000	0.991190	0.143030
105863	2024-12-31	124345	0.410901	-0.049043	0.944788	...	4.500000	0.960717	4.833333	0.963205	-0.299557
105864	2024-12-31	124805	0.515199	0.018456	0.788196	...	3.916667	0.947243	4.333333	0.954706	-0.747329
105865	2024-12-31	136072	0.438679	-0.031299	0.600666	...	3.750000	0.943304	0.500000	0.815869	0.143030

Figure 2: Illustration of dataframe used to feed the different models.

3.8 Machine Learning methodology

To generate out-of-sample (OOS) one-month-ahead forecasts of twelve-month cumulative returns for individual analysts' portfolios, we adopt an expanding walk-forward cross-validation framework, illustrated in Figure 3. The figure is provided for illustrative purposes [3] only, the dates shown do not correspond to the actual sample.

Feature construction and target variable computation require an initial minimum of 24 months of historical data. After this initialization period, an additional 24 months of data are required to obtain sufficiently populated feature vectors and corresponding target returns. Hyperparameter 6.4 selection is then conducted using a validation window of 12 months, followed by a one-month test period used to evaluate out-of-sample performance. Under this design, the first out-of-sample forecast is produced for February 28, 2005.

This expanding-window procedure ensures that model estimation relies exclusively on information available at the time of prediction and allows the training sample to grow over time, closely mimicking a realistic investment setting. The walk-forward setting allows not to break the dynamic of the returns time series.

A pseudo code for the expanding walk-forward cross-validation framework is provided in the appendix 1 and the corresponding Python implementation is available on GitHub.

Figure 1: Training and validation of return prediction model

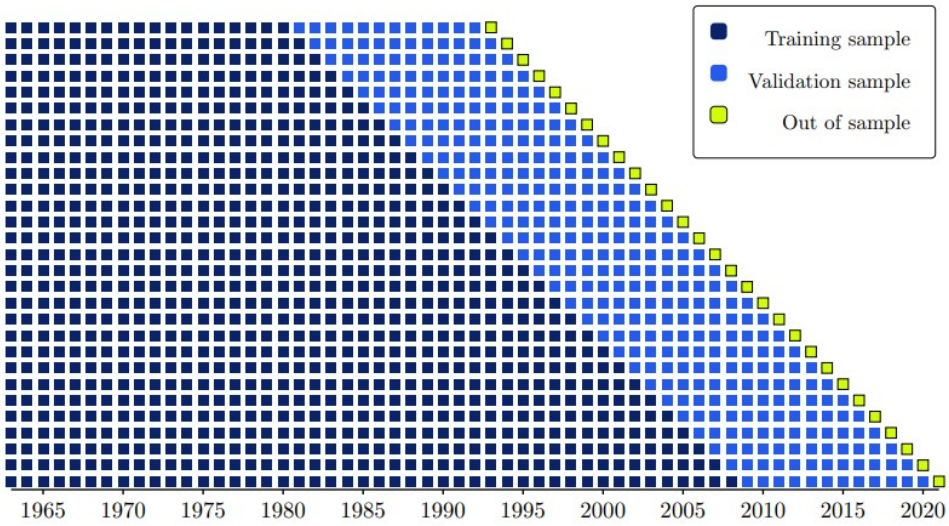

Figure 3: Illustration of the expanding walk-forward cross-validation framework. The figure is for illustrative purposes only and does not reflect the actual sample dates used in the analysis.

[3] https://papers.ssrn.com/sol3/papers.cfm?abstract_id=5851562

4 Results

4.1 Models' results

4.1.1 Linear Models: interpretability

Figure 4 displays the time evolution of the estimated coefficients for each linear model. A first key observation is that the coefficients are clearly *time-varying*, which justifies our expanding-window estimation framework. While some coefficients exhibit relatively stable dynamics over time, others display pronounced variability and occasional jumps, suggesting periods in which their explanatory power changes substantially.

However, due to the large number of features and the overlapping dynamics across coefficients, it is difficult to identify the most relevant predictors directly from these plots. To address this issue, we summarize the coefficient dynamics in Figure 5.

The table reports, for each model, the *proportion of time (in percent)* during which the absolute value of a given coefficient exceeds a threshold of 0.01. This threshold is chosen to filter out economically negligible coefficients and to focus on variables that are meaningfully used by the models.

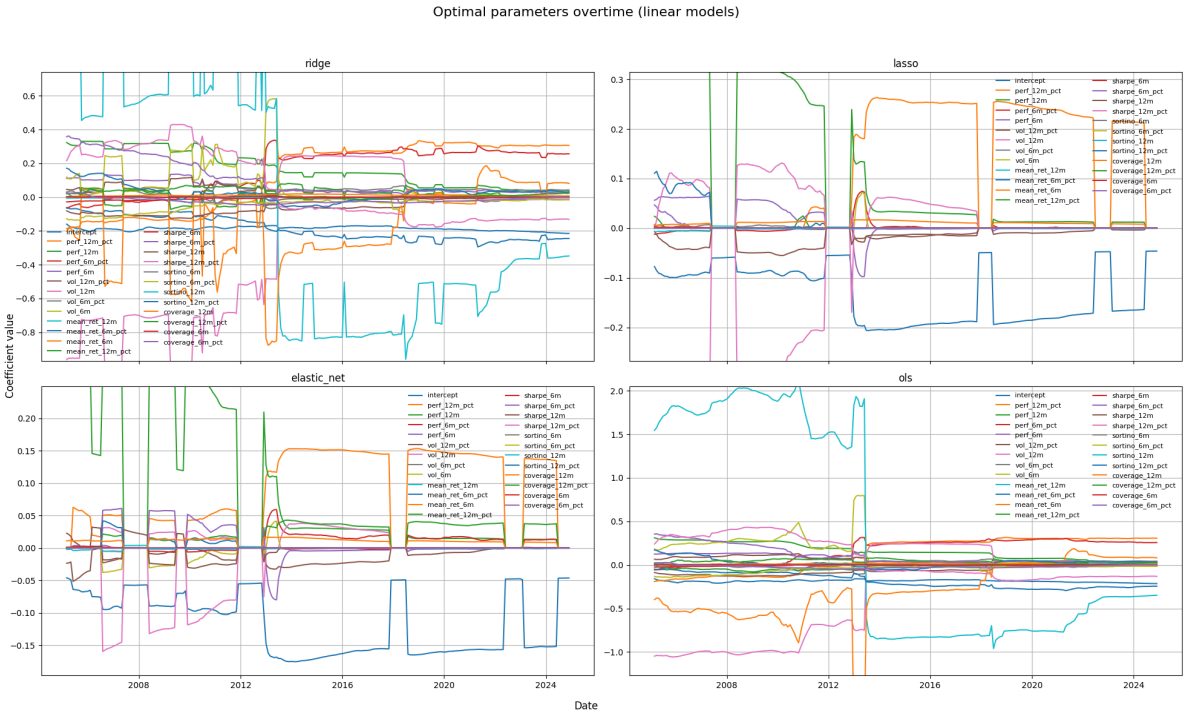

Figure 4: Optimal parameters overtime for linear models.

For the ridge regression, we observe that the variables *sortino_12m*, *sortino_6m*, and *coverage_6m* are never selected, indicating that they do not contribute materially to explaining future performance in this specification. Turning to the lasso model, which explicitly promotes sparsity, we find that the same variables are excluded, along with additional features highlighted in the table. This behavior is consistent with lasso's tendency to retain only the most informative predictors.

Focusing on the most frequently selected variables, we find that *perf_12m*, *perf_12m_pct*, *coverage_12m*, *sharpe_12m*, and *vol_12m* stand out across models. Notably, the simultaneous selection of both percentile-based and non-percentile versions of performance and risk measures suggests that *both time-series and cross-sectional information* play an important role in predicting future returns.

Another interesting result concerns analyst coverage: long-term coverage (*coverage_12m*) appears to be an important predictor, whereas short-term coverage (*coverage_6m*) is consistently irrelevant. This finding is somewhat intuitive, as one might expect that the coverage of the analyst can be informative about its TP credibility. One might think that analysts covering many stocks tend to be less specialized and therefore less informative. We further investigate this hypothesis in the following section.

For the elastic net and OLS models, the set of important variables is broadly consistent with those identified by ridge and lasso, with a few additional predictors entering the models. This overall consistency

across specifications reinforces the robustness of our findings.

Finally, to provide an aggregated measure of feature relevance, we average the selection proportions across all models, as reported in the last columns of Figure 5. This allows us to rank features by their *average proportional importance*. According to this criterion, the five most important variables are the intercept, *perf_12m*, *perf_12m_pct*, *coverage_12m*, and *sharpe_12m_pct*. In contrast, the Sortino-based measures and *coverage_6m* are never selected on average, indicating that they do not carry meaningful information for forecasting in our setting.

	rank	ridge	lasso	elastic_net	ols	mean_models
intercept	1	100.00	100.00	100.00	100.00	100.00
perf_12m	2	100.00	81.09	80.25	100.00	90.34
perf_12m_pct	3	98.32	56.30	76.89	97.48	82.25
coverage_12m	4	67.65	53.36	64.29	64.29	62.40
sharpe_12m_pct	5	100.00	28.99	19.75	100.00	62.18
mean_ret_12m_pct	6	79.83	4.20	71.01	86.97	60.50
perf_6m_pct	7	92.44	3.78	50.84	87.82	58.72
vol_12m	8	68.07	51.26	42.02	68.07	57.36
perf_6m	9	82.77	33.19	24.37	86.13	56.61
vol_12m_pct	10	72.69	24.37	42.02	80.25	54.83
sortino_12m_pct	11	91.18	11.34	16.39	88.66	51.89
sharpe_6m_pct	12	99.58	5.46	0.00	100.00	51.26
sharpe_12m	13	57.98	56.72	31.09	55.46	50.31
sortino_6m_pct	14	100.00	0.00	0.00	100.00	50.00
mean_ret_6m_pct	15	100.00	0.00	0.00	100.00	50.00
mean_ret_12m	16	100.00	0.00	0.00	100.00	50.00
mean_ret_6m	17	97.06	0.00	0.00	100.00	49.26
vol_6m_pct	18	93.28	0.00	0.00	86.97	45.06
coverage_6m_pct	19	76.47	0.00	0.00	79.41	38.97
vol_6m	20	67.23	2.52	15.97	65.13	37.71
coverage_12m_pct	21	54.20	0.00	0.00	54.20	27.10
sharpe_6m	22	28.57	1.68	0.00	28.57	14.70
sortino_6m	23	0.00	0.00	0.00	0.00	0.00
sortino_12m	24	0.00	0.00	0.00	0.00	0.00
coverage_6m	25	0.00	0.00	0.00	0.00	0.00

Figure 5: Proportion (%) of time feature is selected for linear models.

Figure 6 reports the time-series mean of the estimated coefficients for each linear model. Each column corresponds to a model-specific average coefficient, computed over the full out-of-sample period, while the last column (*mean_models*) represents the average coefficient across all linear models. The column (*rank*) (descending) orders the features according to this cross-model average, providing a unified measure of variable importance.

Several clear patterns emerge. First, performance-related variables dominate the ranking. The one-year cumulative performance (*perf_12m*) and its cross-sectional percentile version (*perf_12m_pct*) are ranked first and second, respectively, and display consistently positive coefficients across models. This indicates that analysts with stronger past portfolio performance—both in absolute terms and relative to peers—tend to deliver higher future returns, highlighting the persistence of analyst skill.

In contrast, Sortino-based measures appear systematically less informative, with several Sortino variables ranked near the middle and exhibiting near-zero average coefficients.

Finally, the intercept and volatility-based measures tend to have negative average coefficients and are ranked toward the bottom of the table. In particular, long-term volatility (*vol_12m*) is the least important feature, with a consistently negative sign across models, indicating that higher past volatility is associated with weaker future analyst portfolio performance which can makes us think of the known low volatility anomaly.

Overall, the ranking based on the cross-model average coefficients provides a stable and interpretable summary of feature importance. The results emphasize the central role of long-horizon performance and risk-adjusted metrics, while downplaying the relevance of short-term risk measures and coverage intensity.

	rank	ridge	lasso	elastic_net	ols	mean_models
perf_12m	1	0.17	0.10	0.09	0.16	0.13
perf_12m_pct	2	0.13	0.13	0.09	0.12	0.12
mean_ret_12m	3	0.06	0.00	0.00	0.35	0.10
perf_6m_pct	4	0.16	0.00	0.01	0.17	0.08
vol_6m	5	0.07	0.00	-0.00	0.13	0.05
sharpe_6m_pct	6	0.11	0.00	0.00	0.10	0.05
sharpe_12m_pct	7	0.05	0.03	0.00	0.07	0.04
perf_6m	8	0.04	0.01	0.01	0.05	0.03
sortino_12m_pct	9	0.03	0.01	0.00	0.02	0.02
coverage_12m	10	0.01	0.01	0.01	0.01	0.01
coverage_12m_pct	11	0.02	0.00	0.00	0.02	0.01
mean_ret_12m_pct	12	0.02	0.00	0.02	-0.00	0.01
vol_6m_pct	13	0.03	0.00	0.00	0.02	0.01
coverage_6m	14	0.00	0.00	0.00	0.00	0.00
sortino_12m	15	-0.00	-0.00	-0.00	-0.00	0.00
vol_12m_pct	16	0.01	-0.00	-0.01	0.02	0.00
sharpe_6m	17	-0.01	-0.00	-0.00	-0.01	-0.00
sortino_6m	18	0.00	0.00	0.00	0.00	0.00
coverage_6m_pct	19	-0.02	0.00	0.00	-0.02	-0.01
sortino_6m_pct	20	-0.05	0.00	0.00	-0.05	-0.02
sharpe_12m	21	-0.04	-0.02	-0.00	-0.05	-0.03
mean_ret_6m_pct	22	-0.19	0.00	0.00	-0.18	-0.09
mean_ret_6m	23	-0.15	0.00	0.00	-0.30	-0.11
intercept	24	-0.19	-0.13	-0.12	-0.19	-0.16
vol_12m	25	-0.25	-0.08	-0.02	-0.33	-0.17

Figure 6: Time-series mean of betas for linear models.

4.1.2 Cross models comparisons

Figure 7 displays the optimal hyperparameters selected over time for each model within the expanding walk-forward cross-validation framework. At each re-estimation date, hyperparameters are chosen based on out-of-sample validation performance, allowing them to adapt to changing data environments and market conditions.

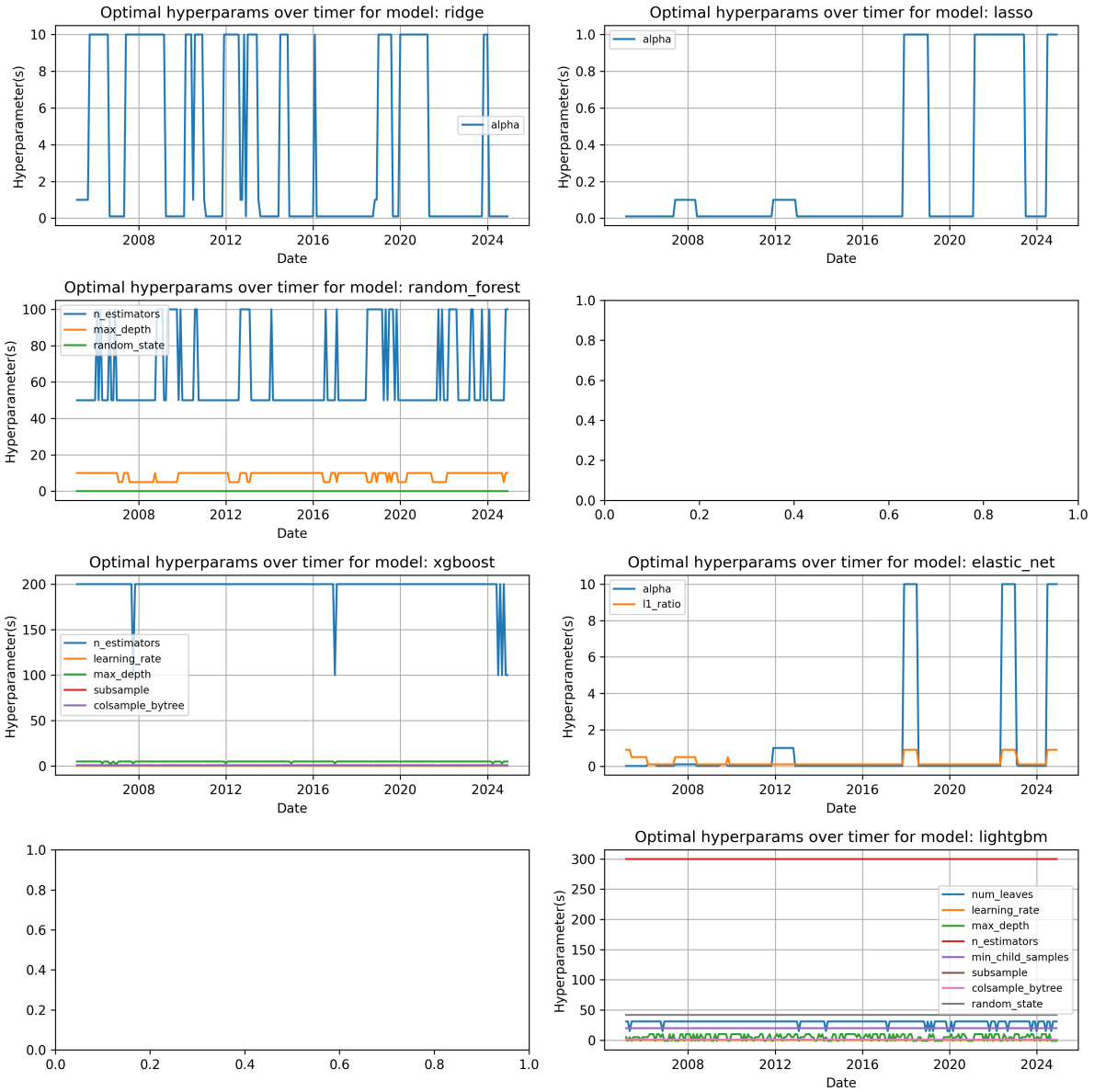

Figure 7: Optimal hyperparameters overtime for all models.

For linear regularized models, we observe substantial time variation. In the Ridge regression, the regularization parameter α switches frequently across the grid, suggesting that the optimal degree of shrinkage is highly regime-dependent. Similarly, the Lasso model alternates (but less frequently, with long periods of persistence) between very small and relatively large values of α, indicating periods where sparsity is either weakly or strongly favored. These dynamics are consistent with the idea that the signal-to-noise ratio of analyst features varies over time.

For the Elastic Net, both the overall regularization strength α and the mixing parameter ℓ_1-ratio exhibit discrete shifts (regimes duration looks like the ones of the Lasso and are quite long/persistent). Periods with a higher ℓ_1-ratio correspond to stronger sparsity, while lower values indicate behavior closer to Ridge regression. This highlights the flexibility of the Elastic Net in interpolating between dense and sparse representations depending on market conditions.

Turning to tree-based models, the Random Forest hyperparameters (especially n_estimators) show less persistent variation (shorter regimes duration) compared to Lasso and Elastic Net.

For XGBoost, most hyperparameters remain stable across time, with occasional discrete changes in the number of estimators and tree depth. Learning rate and subsampling parameters show little variation, indicating that boosting dynamics are relatively robust once calibrated, while overall ensemble size adapts to shifts in data complexity.

The LightGBM model displays different hyperparameter dynamics. Hyperparameters such as the max_depth rapidly change whereas the other hyperparameters remain very stable across time.

Finally, we note that the plot for the MLP model is not reported, as several of its hyperparameters are non-numeric (e.g., network architecture defined by tuples, activation functions), which prevents meaningful visualization on a continuous scale. Similarly, OLS is excluded from this figure as it does not involve hyperparameter tuning.

Overall, this figure illustrates that optimal hyperparameters are far from static and adapt meaningfully over time, reinforcing the importance of dynamic model selection and validation in long-horizon forecasting exercises.

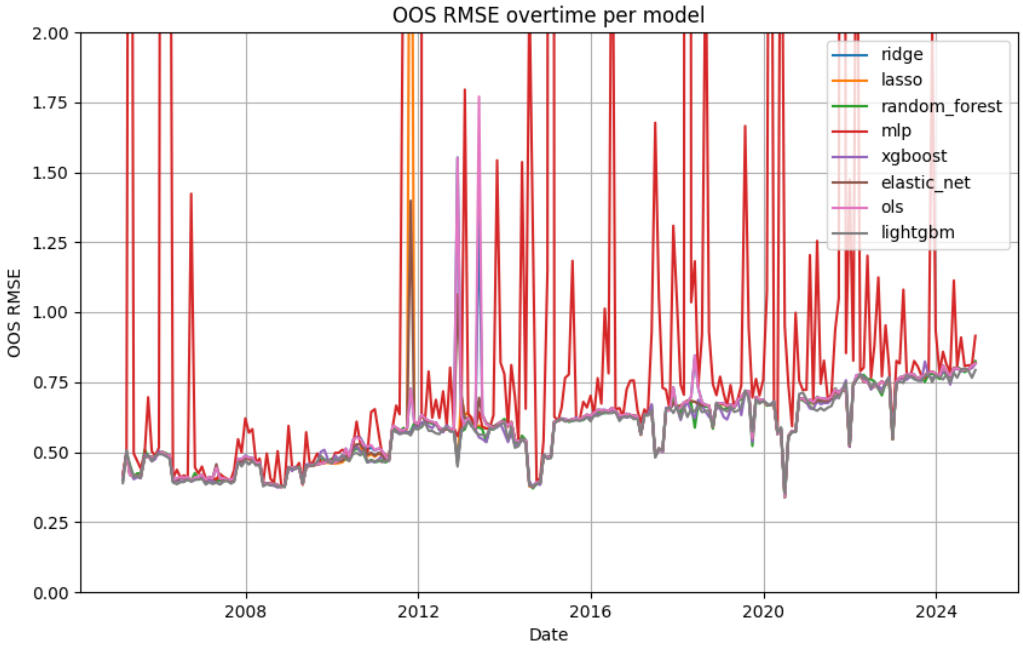

Figure 8: OOS RMSE overtime across models.

Figure 8 reports the time-series evolution of the out-of-sample (OOS) root mean squared error (RMSE) for all models considered in our study.

A first striking observation concerns the Multilayer Perceptron (MLP). The MLP exhibits substantially higher and more volatile OOS RMSE values compared to all other models. During training, the optimization procedure frequently failed to converge, likely reflecting an unfavorable balance between model complexity and effective sample size. In particular, the dimensionality of the hidden layers may appears too large relative to the available cross-sectional information at each re-estimation date. As a consequence, the MLP delivers highly unstable forecasts and poor predictive performance, as reflected by the magnitude of its OOS RMSE.

A second noteworthy result is that the OOS RMSE dynamics are broadly similar across most models, indicating that forecasting difficulty varies over time in a largely model-independent manner. However, we also observe periods of pronounced dispersion between linear models—most notably OLS, and to a lesser extent Ridge, Lasso, and Elastic Net—and the non-linear models. During such episodes, non-linear methods tend to exhibit smoother RMSE trajectories and lower sensitivity to adverse market conditions, suggesting greater robustness to regime shifts or non-linearities in analyst performance.

Figure 9 summarizes model performance by ranking the methods according to their time-series average OOS RMSE. While LightGBM, Random Forest, and XGBoost consistently outperform the linear

benchmarks (with the exception of the poorly calibrated MLP), the magnitude of the improvement remains modest. This suggests that, although non-linear models capture additional structure in the data, the incremental predictive gains are limited. A formal assessment of statistical significance would require the computation of Diebold–Mariano test statistics, which we leave for future work.

	rank	mean_rmse
lightgbm	1	0.57
random_forest	2	0.58
xgboost	3	0.58
elastic_net	4	0.59
ridge	5	0.60
lasso	6	0.60
ols	7	0.60
mlp	8	162.57

Figure 9: Time-series mean RMSE across models.

Figure 10 reports the evolution of the rolling 12-months Information Coefficient (IC), defined as the Spearman rank correlation between predicted values and the realized target variable. The IC is a central performance metric in asset pricing and portfolio construction, particularly for long–short (L/S) and long-only (L/O) strategies based on cross-sectional sorting. In such settings, portfolio returns depend primarily on the relative ranking of assets—in our case, analysts—rather than on the absolute magnitude of the forecasts.

A key finding of our study emerges when contrasting IC-based performance with RMSE-based evaluation. While models appear broadly similar in terms of average RMSE, their ability to rank analysts differs substantially. As shown in Figure 10, non-linear models consistently and persistently outperform linear specifications in terms of IC. This result indicates that, when trained on panel data incorporating both time-series and cross-sectional features, machine learning models are able to effectively learn relative performance rankings across analysts.

In contrast, linear models exhibit markedly weaker ranking ability, with IC values an order of magnitude smaller on average. This highlights an important limitation of linear forecasting approaches in cross-sectional investment settings: even when predictive accuracy in a mean-squared-error sense is comparable, linear models fail to capture the complex, non-linear interactions required to accurately rank analysts by future performance.

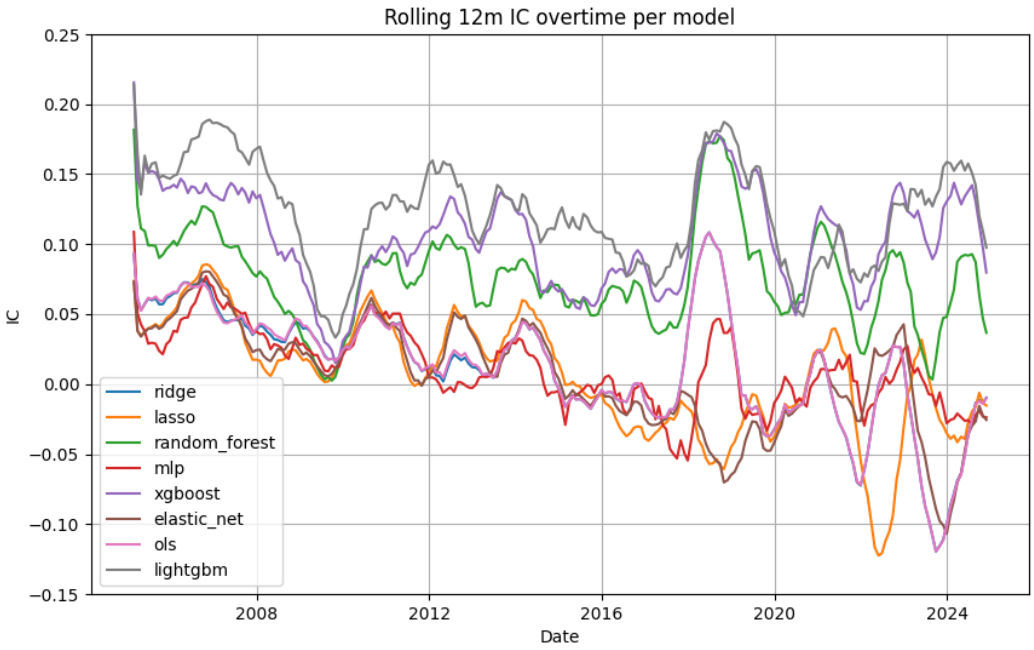

Figure 10: Rolling 12-month IC overtime across all models.

Figure 11 presents a ranking of the models based on their time-series mean Information Coefficient (IC). LightGBM clearly emerges as the best-performing model, exhibiting an ability to rank analysts that is approximately twelve times stronger than that of the OLS benchmark. XGBoost and Random Forest also perform remarkably well, substantially outperforming all linear specifications.

These results reinforce the conclusion that non-linear models are markedly superior at extracting cross-sectional ranking information from analyst-level panel data, while linear models remain severely limited in their capacity to discriminate between high- and low-performing analysts.

	rank	mean_ic
lightgbm	1	12.11
xgboost	2	10.22
random_forest	3	7.40
ols	4	1.08
ridge	5	1.07
mlp	6	1.03
lasso	7	0.40
elastic_net	8	0.38

Figure 11: Time-series mean IC across all models.

4.2 Financial performance: backtests results

We evaluate only **out-of-sample** performance using a walk-forward (expanding window) protocol. Starting in 2005 and ending in late 2024, each month-end rebalancing date t is treated as an out-of-sample decision point. This emulates a live strategy from 2005. At each t:

1. Build analyst implied books $\{w_{a,\cdot,t}\}_{a\in\mathcal{A}_t}$ from target prices available at t.

2. Compute features $\{X_{a,t}\}_{a\in\mathcal{A}_t}$ using information up to t.
3. Generate predictions $\hat{y}_{a,t}$.
4. Construct the meta-portfolio $w^{\text{meta}}_{\cdot,t}$
5. Realize the next-month portfolio return over $[t, t{+}1]$ (end-of-month to end-of-month):

$$R^{\text{strat}}_{t\to t+1} = \sum_i w^{\text{meta}}_{i,t}\, r_{i,t\to t+1}, \qquad r_{i,t\to t+1} = \frac{P_{i,t+1}}{P_{i,t}} - 1. \tag{19}$$

We repeat this procedure for N strategies, where each strategy corresponds to a distinct machine learning model (we trained 8 models in total), yielding 8 out-of-sample meta-portfolios and their associated performance time series.

Benchmarks. To interpret performance, we compare model-driven meta-portfolios against two natural benchmarks.

- Benchmark 1: Aggregated book. At each date t, we aggregate all analyst books without discrimination:

$$w^{\text{agg}}_{i,t} \propto \sum_{a\in\mathcal{A}_t} w_{a,i,t}, \tag{20}$$

  and then normalize exposure to match the strategy's investment profile. This benchmark corresponds to the neutral hypothesis that **all analysts are equally informative** and that there is no value in learning credibility weights.

- Benchmark 2: Equal-weight analysts. This benchmark equalizes analysts first, then aggregates:

$$\tilde{w}_{a,i,t} = \frac{w_{a,i,t}}{\sum_j |w_{a,j,t}|}, \qquad w^{\text{ew}}_{i,t} = \frac{1}{|\mathcal{A}_t|}\sum_{a\in\mathcal{A}_t} \tilde{w}_{a,i,t}, \tag{21}$$

  Unlike the aggregated-book benchmark, this construction ensures each analyst contributes equally regardless of book size or coverage breadth.

Backtest Results (2006–2024)

Scope and models. We report out-of-sample portfolio performance over the full available period, from **February 2006** to **December 2024**, for the following strategies: `Random Forest`, `Ridge`, `Lasso`, `XGBoost`, `LightGBM`, `OLS`, and two benchmarks (`Benchmark` and `Equal-Weighted`).

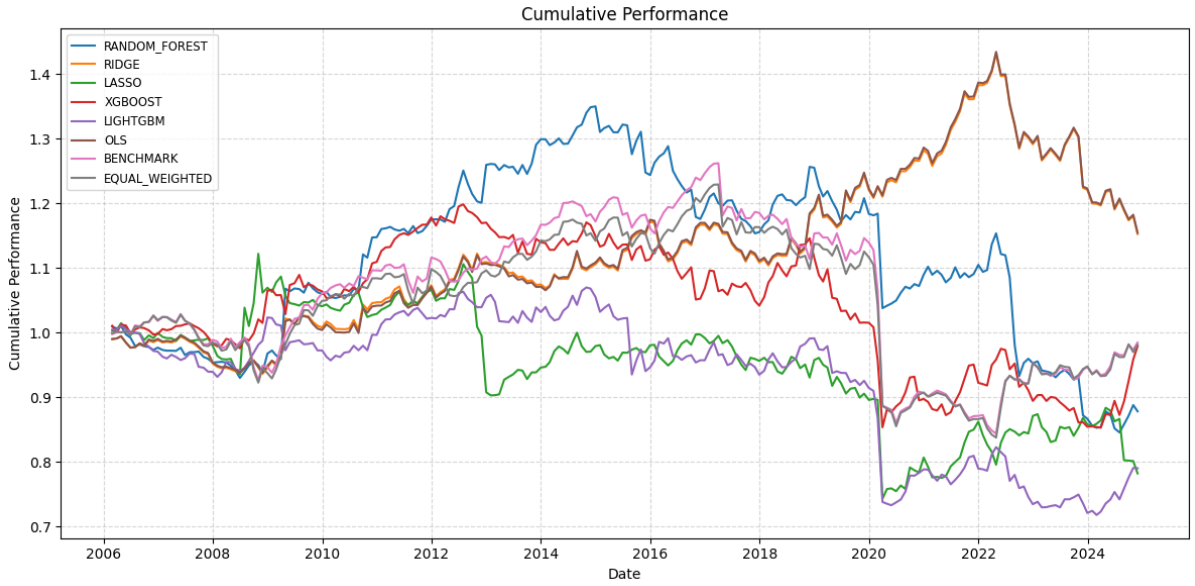

Figure 12: Cumulative performance, February 2006 – December 2024. Strategies: Random Forest, Ridge, Lasso, XGBoost, LightGBM, OLS, Benchmark, Equal-Weighted.

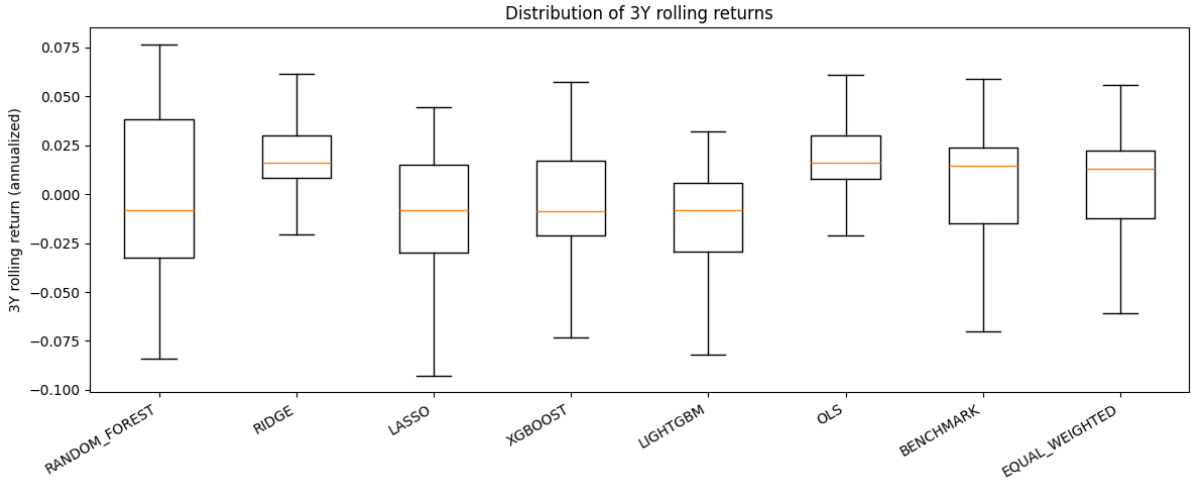

Figure 13: Distribution of **3-year rolling returns** (overlapping windows), February 2006 – December 2024.

Table 2: Performance summary (Full period: Feb 2006 – Dec 2024).

	Random Forest	Ridge	Lasso	XGBoost	LightGBM	OLS	Benchmark	Equal-Weighted
Gross return (%)	-12.18	15.27	-21.78	-2.10	-20.99	15.46	-1.53	-1.87
Annual return (%)	-0.69	0.76	-1.30	-0.11	-1.24	0.77	-0.08	-0.10
Annual volatility (%)	6.49	4.53	7.35	5.61	5.97	4.48	5.92	5.71
Risk-adjusted return	-0.11	0.17	-0.18	-0.02	-0.21	0.17	-0.01	-0.02
Max drawdown (%)	-37.37	-19.52	-33.78	-28.84	-32.92	-19.52	-33.16	-31.87

Pre-COVID Results (2006–2020)

Motivation for a pre-COVID cut. Given the extreme behavior around 2020 and the risk of data/implementation artifacts, we replicate the analysis on a pre-COVID sub-sample (from **February 2006** up to **December 2020**). This helps isolate the "normal-regime" behavior of the strategies.

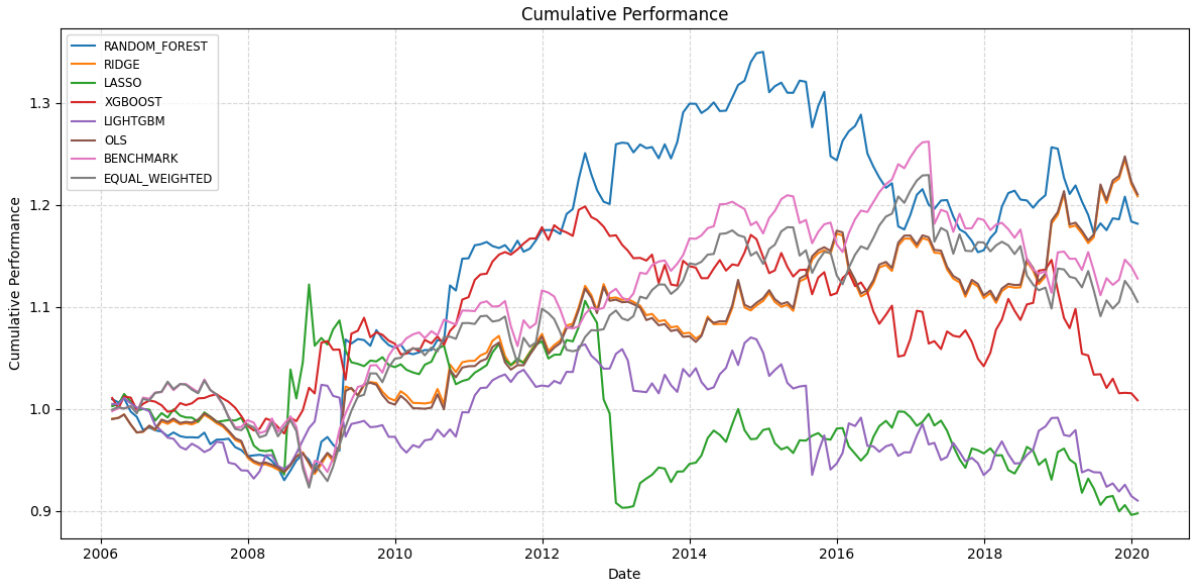

Figure 14: Cumulative performance, February 2006 – February 2020 (pre-COVID).

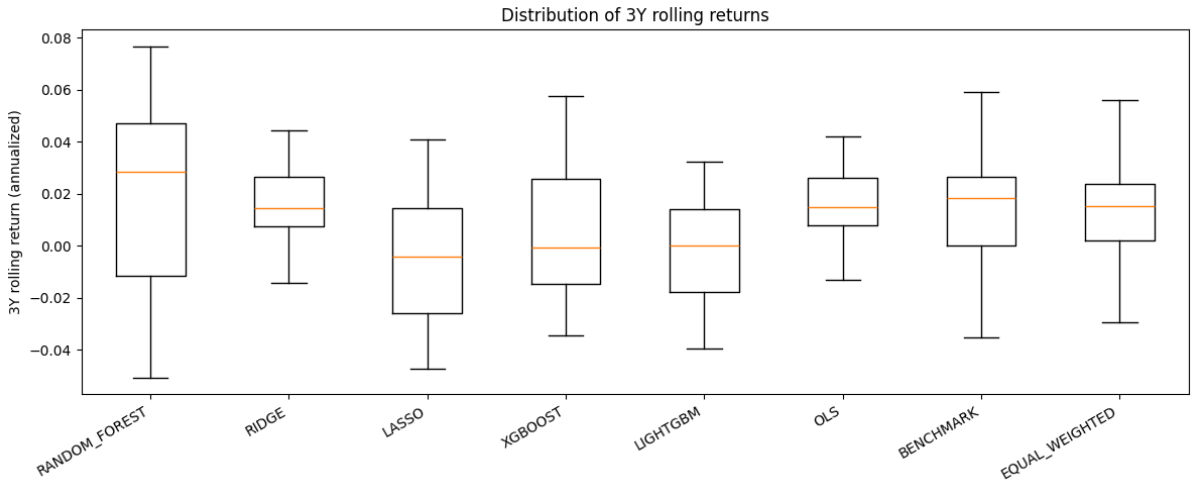

Figure 15: Distribution of 3-year rolling returns, February 2006 – February 2020 (pre-COVID).

Table 3: Performance summary (Pre-COVID: Feb 2006 – February 2020).

	Random Forest	Ridge	Lasso	XGBoost	LightGBM	OLS	Benchmark	Equal-Weighted
Gross return (%)	18.16	20.85	-10.25	0.84	-9.00	21.04	12.79	10.50
Annual return (%)	1.20	1.36	-0.77	0.06	-0.67	1.37	0.86	0.72
Annual volatility (%)	5.30	4.15	6.08	4.37	4.64	4.08	4.14	4.10
Risk-adjusted return	0.23	0.33	-0.13	0.01	-0.14	0.34	0.21	0.17
Max drawdown (%)	-14.56	-5.87	-20.16	-15.87	-14.96	-5.81	-11.94	-11.27

Interpretation and discussion. The financial backtest highlights several important (and somewhat unexpected) patterns. We summarize the main takeaways below.

- **Predictive quality (IC) does not systematically translate into trading performance.** Although we evaluate model quality during training using rank-based metrics such as the Information Coefficient (Spearman correlation between predicted and realized forward analyst P&L), higher IC does not necessarily produce higher realized portfolio returns. In particular, models that appear superior in validation (e.g., tree ensembles and boosting methods) may fail to dominate in the backtested strategy. This suggests that:
  - the portfolio mapping from predictions to positions (signal transforms, normalization, long/short budgeting) may distort or compress cross-sectional differences;
  - trading performance is influenced by *tails*, concentration, and stability rather than global rank correlation;
- **Strong non-stationarity and regime dependence.** Strategy performance is highly sensitive to the sub-period. Some models perform exceptionally well during the first part of the sample (notably models with the best IC in that regime, such as `Random Forest` and `XGBoost`), but later stop working and sometimes underperform materially. This instability suggests that the relationship between analyst features and forward performance is not stable over time, and maybe that:
  - the analyst universe itself changes (coverage, behavior, incentives);
  - market regimes shift;
  - the rolling training affect the performance of the strategy
- **Simple linear models appear the most robust over long horizons.** Over the full sample, the model that performs most consistently is the simplest one: a linear specification (`OLS`). Beyond raw returns, its profile is characterized by:
  - **very low drawdowns** relative to other approaches,

- **low realized volatility**,
- and therefore strong **risk-adjusted performance**.

This is striking given that richer models can achieve higher IC at training time. A plausible interpretation is that linear models may capture a stable, low-dimensional component of the signal while avoiding regime-specific overfitting. In addition, `Ridge` was tuned to behave similarly to `OLS`, reinforcing the view that a strongly regularized linear structure can be hard to beat in this setting.

- **Major caveats: implementation complexity and methodological degrees of freedom.** Reproducing an end-to-end pipeline of this type is technically demanding: it combines target price ingestion, book construction, feature engineering, walk-forward training, prediction storage, long/short aggregation, and performance attribution. Given time constraints, it is difficult to guarantee that every step is perfectly validated. Therefore, the results must be interpreted with caution:
  - **Potential implementation errors.** Subtle bugs (date alignment, sign conventions, as-of joins, missing data handling) can significantly affect realized returns and may remain hard to detect without tests.
  - **Methodological choices.** Several details are under-specified in the reference paper and require implementation choices (calendar conventions, end-of-month definitions, normalization schemes, analyst filtering rules). Alternative but reasonable choices can materially change outcomes.
  - **Computational constraints.** The pipeline is computationally intensive (large panel, rolling windows, multiple models). This made harder the robustness checks and debuging process.

Overall conclusion. Even if the financial performance is not uniformly strong across models and regimes, the exercise remains informative: it demonstrates how analyst target prices can be transformed into tradable sub-portfolios and how machine learning signals can be used to weight analysts in a meta-portfolio framework. The variability of results, the apparent disconnect between IC and realized returns, and the sensitivity to implementation details all indicate that further work is required both to strengthen validation and to improve robustness before drawing strong economic conclusions.

5 Conclusion

5.1 Summary

This paper revisits the question of whether sell-side analyst heterogeneity can be systematically exploited for investment purposes, building on the framework of *Alpha in Analysts* and extending it through modern machine learning techniques. By treating each analyst as a portfolio manager and translating target price recommendations into self-financing long–short books, we construct a rich panel dataset of analyst-level returns and characteristics spanning more than two decades.

Our main contribution is threefold. First, we substantially enrich the information set by expanding the feature space to include twenty-four analyst and portfolio-level predictors capturing both time-series dynamics and cross-sectional positioning. Second, we adopt an expanding walk-forward cross-validation framework that closely mimics a real-time investment setting and allows both model parameters and hyperparameters to evolve over time. Third, we conduct a systematic comparison between linear models and a range of non-linear machine learning approaches, including tree-based ensembles and neural networks.

From a forecasting perspective, our results reveal an important distinction between point prediction accuracy and ranking ability. While out-of-sample RMSEs are broadly similar across most models, suggesting limited gains in terms of mean-squared-error minimization, non-linear models dramatically outperform linear specifications in terms of Information Coefficient. In particular, LightGBM, XGBoost, and Random Forests exhibit a persistent and large advantage in their ability to rank analysts by future performance. This finding highlights that machine learning models are particularly effective at extracting cross-sectional structure from analyst-level panel data, even when improvements in traditional error metrics appear modest.

However, when translating predictive signals into tradable fund-of-analysts strategies, the picture becomes more nuanced. Despite their superior IC performance, non-linear models do not consistently dominate in realized portfolio returns. Strategy performance is highly regime-dependent, and simple

linear models, especially OLS and strongly regularized variants, exhibit the most robust behavior over long horizons, characterized by lower volatility and drawdowns. This apparent disconnect between ranking quality and realized profitability underscores the complexity of mapping predictive signals into portfolio allocations and cautions against relying solely on in-sample or validation metrics.

Overall, our findings suggest that analyst heterogeneity is both persistent and predictable, but that extracting economically meaningful alpha from this predictability remains challenging. Machine learning models clearly add value in terms of cross-sectional discrimination, yet robustness, stability, and implementation choices play a decisive role in determining realized performance.

5.2 Limits and extensions

Several limitations of our study point to promising directions for future research. First, the mapping from analyst-level predictions to meta-portfolio weights is necessarily ad hoc. While we adopt a transparent long–short allocation scheme based on predicted performance, alternative signal transformations, concentration controls, or risk-aware weighting schemes may better translate ranking ability into realized returns. In particular, explicitly accounting for prediction uncertainty or tail behavior could improve robustness.

Second, our analysis abstracts from several important real-world frictions. Transaction costs are modeled parsimoniously, and we do not explicitly incorporate liquidity constraints, capacity limits, or short-selling frictions at the analyst-book level. Given that analyst portfolios can be highly concentrated and time-varying, these constraints may materially affect implementability and performance.

Third, the strong non-stationarity observed in both predictive relationships and strategy performance suggests that regime dependence plays a central role. Future work could explore regime-switching models, adaptive feature selection, or meta-learning approaches to dynamically adjust model complexity and signal usage over time. Similarly, combining linear and non-linear models in ensemble or hierarchical frameworks may help reconcile robustness with cross-sectional ranking power.

Finally, the complexity of the end-to-end pipeline, spanning data construction, portfolio formation, feature engineering, model training, and backtesting, introduces scope for implementation risk. While we make our code publicly available, further robustness checks, alternative design choices, and replication efforts are essential before drawing strong economic conclusions.

In summary, this paper provides evidence that machine learning techniques can meaningfully enhance the extraction of information from analyst behavior, particularly in terms of relative ranking. At the same time, it highlights the gap between predictive sophistication and investable performance, emphasizing the need for careful modeling, conservative implementation, and extensive validation in applied asset management settings.

References

[1] Alvaro Cartea, Qi Jin (2025). Alpha in Analysts.

[2] Mikheil Esakia and Felix Goltz (2025). What Drives the Performance of Machine Learning Factor Strategies?

6 Appendix

6.1 DataFrame illustration

	date	analyst_id	perf_12m_pct	perf_12m	perf_6m_pct	perf_6m	vol_12m_pct	vol_12m	vol_6m_pct	vol_6m	mean_ret_12m	mean_ret_6m_pct	mean_ret_6m	mean_ret_12m_pct	sharpe_6m	sharpe_6m_pct	sharpe_12m	sharpe_12m_pct	sortino_6m	sortino_6m_pct	sortino_12m	sortino_12m_pct	coverage_12m	coverage_12m_pct	coverage_6m	coverage_6m_pct	y
0	2002-01-31	16	0.658537	0.045179	0.501122	-0.092576	0.585366	0.178469	0.301421	0.100515	0.016791	0.452506	-0.010039	0.666667	-0.343971	0.423336	0.325924	0.630081	-0.928295	0.409968	1.075152	0.721545	1.750000	0.971030	2.000000	0.968180	-0.186680
1	2002-01-31	51	0.825203	0.299013	0.433807	-0.139965	0.691057	0.211838	0.335826	0.111382	0.039085	0.389678	-0.019864	0.833333	-0.617798	0.332835	0.639136	0.768293	-1.793427	0.272508	3.185160	0.892276	1.083333	0.946984	0.833333	0.897129	-0.032694
2	2002-01-31	118	0.491870	-0.133531	0.394914	-0.177794	0.207317	0.099761	0.337322	0.112050	-0.007261	0.348042	-0.026990	0.406504	-0.834413	0.282723	-0.252149	0.380081	-1.712710	0.283762	-0.389870	0.384146	2.833333	0.990153	3.166667	0.988340	-0.078027
3	2002-01-31	127	0.544715	-0.070777	0.521316	-0.069421	0.134146	0.065122	0.272251	0.094724	-0.002860	0.461481	-0.008277	0.441057	-0.302697	0.437547	-0.116388	0.420732	-0.571528	0.477492	-0.255964	0.422764	1.083333	0.946984	1.000000	0.919663	-0.042041
4	2002-01-31	177	0.282520	-0.309486	0.287958	-0.281141	0.603659	0.164784	0.707554	0.225496	-0.022437	0.327599	-0.029827	0.269618	-0.458206	0.382947	-0.420630	0.329268	-0.683941	0.459807	-0.561365	0.349393	1.500000	0.963827	1.666667	0.957245	0.657964
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
105961	2024-12-31	115583	0.743711	0.189328	0.547398	0.014091	0.406281	0.077721	0.409345	0.073924	0.017160	0.534507	0.004629	0.722222	0.216909	0.535935	0.765719	0.736373	0.833235	0.608961	3.131181	0.853095	1.916667	0.886008	2.333333	0.901741	-0.514250
105962	2024-12-31	122739	0.271486	-0.136930	0.322228	-0.088802	0.529350	0.088464	0.712394	0.119118	-0.008524	0.342604	-0.003936	0.269723	-0.276454	0.364103	-0.337641	0.281447	-0.416385	0.444559	-0.523482	0.291186	7.083333	0.989065	8.000000	0.991190	0.145030
105963	2024-12-31	124345	0.410901	-0.049043	0.944788	0.358421	0.884696	0.168145	0.798191	0.143022	0.008559	0.505077	0.060188	0.573375	1.457794	0.851975	0.176334	0.456807	6.959408	0.899948	0.367518	0.500525	4.500000	0.960717	4.833333	0.963205	-0.299507
105964	2024-12-31	124805	0.515199	0.018456	0.788196	0.153576	0.332265	0.069401	0.546310	0.092215	0.003620	0.778677	0.027484	0.489518	1.032450	0.766302	0.180704	0.497379	3.617302	0.820010	0.525276	0.535677	3.916667	0.947243	4.333333	0.954706	-0.747329
105965	2024-12-31	136072	0.439679	-0.031299	0.600666	0.039593	0.263103	0.061522	0.039981	0.026608	-0.000894	0.562113	0.006782	0.399895	0.882977	0.734888	-0.050018	0.395178	1.270963	0.657040	-0.078043	0.398216	3.750000	0.943304	5.500000	0.916869	0.143030

Figure 16: Illustration of dataframe feeding the models.

6.2 Expanding walk-forward cross-validation framework pseudo-code

Algorithm summarizes the expanding walk-forward cross-validation procedure used to generate out-of-sample forecasts, select hyperparameters, and estimate time-varying model parameters. The procedure ensures that all predictions are formed using information available at the time of forecasting and closely mimics a realistic investment setting.

Algorithm 1 Expanding Walk-Forward Cross-Validation with Model Selection

Require: Panel data $\mathcal{D}$ with features X, target y, dates t
Require: Set of models $\mathcal{M}$ and hyperparameter grids $\mathcal{H}_m$
Require: Minimum history $T_{\min}$, validation window T_{val}, forecast horizon h
Initialize OOS predictions $\texttt{OOS\_PRED}[m][t]$ for all $m \in \mathcal{M}$
Initialize OOS true values $\texttt{OOS\_TRUE}[t]$
Initialize validation scores $\texttt{Score}[t, m]$
Initialize hyperparameters $\texttt{Hyperparams}[t, m]$
Initialize coefficient storage $\texttt{Params}[t, m]$ for linear models only
Compute ordered date grid $\{t_1, \ldots, t_T\}$
Set starting index $t_0 = T_{\min} + T_{\text{val}} + h$
for $t = t_0$ **to** $T - h$ **do**
    Define training end date $t_{\text{train}} = t_{t-T_{\text{val}}-h}$
    Define validation end date $t_{\text{val}} = t_{t-h}$
    Construct training set $\mathcal{D}_{\text{train}} = \{t \le t_{\text{train}}\}$
    Construct validation set $\mathcal{D}_{\text{val}} = \{t_{\text{train}} < t \le t_{\text{val}}\}$
    for each model $m \in \mathcal{M}$ **do**
        Initialize best score $S^* \leftarrow -\infty$
        Initialize best hyperparameters $\theta^* \leftarrow \emptyset$
        for each hyperparameter configuration $\theta \in \mathcal{H}_m$ **(parallel) do**
            Fit model $m(\theta)$ on $\mathcal{D}_{\text{train}}$
            Compute validation loss across dates in $\mathcal{D}_{\text{val}}$
            Aggregate validation score $S(\theta)$
        end for
        Select $\theta^* = \arg\max_\theta S(\theta)$
        Store $\texttt{Score}[t, m] \leftarrow S(\theta^*)$
        Store $\texttt{Hyperparams}[t, m] \leftarrow \theta^*$
        Refit model $m(\theta^*)$ on $\{t \le t_{\text{val}}\}$
        Predict OOS outcomes at date t
        Store $\texttt{OOS\_PRED}[m][t]$
        Store $\texttt{OOS\_TRUE}[t]$
        if m is linear **then**
            Store intercept and coefficients in $\texttt{Params}[t, m]$
        end if
    end for
end for
return OOS_PRED, OOS_TRUE, Score, Params

6.3 Analyst portfolio rebalancing pseudo-code

Algorithm 2 Irregular Portfolio Rebalancing with Drift Constraint

Require:
- Observed target weights $W_{t,k}$ for assets $k = 1, \ldots, K$ at dates $t = 1, \ldots, T$
- Asset returns $R_{t,k}$
- Maximum drift horizon D (in months)

Ensure:
- Rebalanced portfolio weights $\widetilde{W}_{t,k}$
- Portfolio turnover TO_t

Initialize months since last update $M_{0,k} \leftarrow 0$ for all k

Normalize initial weights:

$$\widetilde{W}_{0,k} \leftarrow \frac{W_{0,k}}{\sum_j |W_{0,j}|}$$

Initial turnover:

$$\mathrm{TO}_0 \leftarrow \sum_k |\widetilde{W}_{0,k}|$$

for $t = 1$ to T **do**

Drift previous portfolio:

$$W_{t,k}^{\mathrm{drift}} \leftarrow \widetilde{W}_{t-1,k}(1 + R_{t,k})$$

Update months since last signal:

$$M_{t,k} \leftarrow M_{t-1,k} + 1$$

if $W_{t,k}$ is observed **then**

$M_{t,k} \leftarrow 0$

end if

Initialize raw weights $W_{t,k}^{\mathrm{raw}} \leftarrow \mathrm{NaN}$

Case 1: New signal observed

if $W_{t,k}$ is observed **then**

$W_{t,k}^{\mathrm{raw}} \leftarrow W_{t,k}$

end if

Case 2: Drift allowed

if $W_{t,k}$ missing **and** $M_{t,k} \leq D$ **then**

$W_{t,k}^{\mathrm{raw}} \leftarrow W_{t,k}^{\mathrm{drift}}$

end if

Normalize portfolio (gross exposure):

$$G_t \leftarrow \sum_k |W_{t,k}^{\mathrm{raw}}|$$

if $G_t > 0$ **then**

$\widetilde{W}_{t,k} \leftarrow W_{t,k}^{\mathrm{raw}} / G_t$

Turnover:

$$\mathrm{TO}_t \leftarrow \sum_k \left| \widetilde{W}_{t,k} - \widetilde{W}_{t-1,k} \right|$$

else

$\widetilde{W}_{t,k} \leftarrow 0$

$\mathrm{TO}_t \leftarrow \sum_k |\widetilde{W}_{t-1,k}|$

end if

end for

6.4 Hyperparameter Grids

Table 4 reports the hyperparameter search grids used for each model in the expanding walk-forward cross-validation procedure.

Table 4: Hyperparameter grids for all machine learning models

Model	**Hyperparameter grid**
Lasso	$\alpha \in \{0.001, 0.01, 0.1, 1.0\}$
Ridge	$\alpha \in \{0.01, 0.1, 1.0, 10.0\}$
Elastic Net	$\alpha \in \{0.01, 0.1, 1.0, 10.0\}$; $l1\_ratio \in \{0.1, 0.5, 0.9\}$
Random Forest	$n\_estimators \in \{100, 200, 300\}$; $max\_depth \in \{\text{None}, 10, 20\}$; $min\_samples\_split \in \{2, 5\}$
XGBoost	$n\_estimators \in \{100, 200, 300\}$; $learning\_rate \in \{0.05, 0.1\}$; $max\_depth \in \{3, 5\}$; $subsample \in \{0.8, 1.0\}$
LightGBM	$num\_leaves \in \{15, 31\}$; $learning\_rate \in \{0.01, 0.05, 0.1\}$; $max\_depth \in \{-1, 5, 10\}$; $n\_estimators = 300$; $min\_child\_samples = 20$; $subsample = 0.8$; $colsample\_bytree = 0.8$; $random\_state = 42$
MLP Neural Network	$hidden\_layer\_sizes \in \{(8,), (8, 8)\}$; $activation \in \{\text{relu}, \text{tanh}\}$; $\alpha \in \{10^{-4}, 10^{-3}\}$