Learning Factors for Stock Market Returns Prediction: A Comparative Study of Linear Filter Reduction and Nonlinear Tree-Based Models

Your Name
MAFS5440

December 5, 2025

Abstract

This report presents a comparative study of two approaches to the “Learning factors for stock market returns prediction” challenge proposed by Qube Research & Technologies on the ENS Challenge Data platform. The task is to predict next-day cross-sectional stock returns using only past returns, under a non-standard evaluation metric: the mean cosine similarity between predicted and realised cross-sections.

The first approach performs a model reduction of a ten-factor linear model into a single temporal filter, which is optimised directly for the cosine-similarity metric via regularised Nesterov-accelerated stochastic gradient descent, followed by a factor reconstruction step that yields ten interpretable factors for submission. This method achieves a mean cosine similarity of about **0.0784**, compared with **0.053–0.054** for simple baselines.

The second approach uses nonlinear tree-based ensemble models—XGBoost, Random Forest, and Bagging—to capture complex interactions in the lagged-return space and then extracts latent factors from fitted models in a rolling-window forecasting scheme. While these models slightly improve mean squared error, their test cosine similarities remain around **0.02** or below, clearly worse than the linear filter.

The emphasis of this report is on scientific analysis rather than on raw leaderboard performance: we discuss the geometry of the cosine metric, the effect of regularisation, and the bias–variance trade-offs between linear and nonlinear approaches.

1 Challenge Description and Problem Formulation

1.1 Data and Constraints

In the QRT 2022 challenge, the training dataset contains the returns of $N = 50$ stocks observed over $T \approx 754$ trading days. For each date t, the cross-sectional return vector is denoted by $R_t \in \mathbb{R}^N$. The challenge imposes:

- a fixed number of factors $F = 10$;

- a fixed temporal window depth $D = 250$;
- the requirement that predictions be expressed as linear combinations of these factors.

1.2 Evaluation Metric: Mean Cosine Similarity

The official metric is the mean cosine similarity:

$$\mathcal{M} = \frac{1}{T_{\text{test}}} \sum_{t \in \text{test}} \frac{\hat{R}_t^\top R_t}{\|\hat{R}_t\|_2 \, \|R_t\|_2}, \tag{1}$$

where T_{test} is the number of evaluation dates.

Key properties:

- **Scale invariance**: only directions matter;
- **Sensitivity to signs and ranking** in the cross-section;
- **Non-convexity** w.r.t. model parameters.

As a consequence, a model with low MSE can still have poor cosine similarity; conversely, a model with moderate MSE may perform well on the official metric if it gets the directions mostly correct.

2 Baselines and Sanity Checks

Before building more sophisticated models, we construct simple baselines to calibrate the difficulty of the task.

2.1 Random and Linear Baselines

We consider the following baselines:

Random search
random linear filters are generated and the best one is selected on a validation set. This yields a mean cosine similarity of about **0.0539**.

Lasso regression
a sparse linear model is fitted by minimising MSE, using lagged returns as predictors. This baseline attains a cosine similarity of roughly **0.05**.

Unregularised SGD
direct optimisation of the cosine similarity using plain stochastic gradient descent without any smoothness regularisation achieves around **0.054**.

All three baselines achieve a small but positive cosine similarity, indicating non-trivial directional information, but they remain well below the performance of the regularised linear filter introduced in the next section.

These baselines serve two purposes: they provide reference numerical levels, and they highlight that naive MSE-driven or unregularised optimisation is insufficient given the noisy and high-dimensional nature of the data.

3 Approach I: Linear Filter Reduction

3.1 From Ten Factors to a Single Temporal Filter

The original factor model writes the prediction as

$$\hat{R}_{t+1} = \sum_{i=1}^{F} \beta_i F_{i,t}, \quad F = 10, \tag{2}$$

where each factor $F_{i,t}$ is a linear combination of the past D days of returns. Let $X_t \in \mathbb{R}^{D\times N}$ denote the matrix stacking the D lagged cross-sections. Then

$$\hat{R}_{t+1} = W X_t, \quad W \in \mathbb{R}^{N\times D}.$$

Imposing a rank-one structure,

$$W = u\, w^\top, \tag{3}$$

with $u \in \mathbb{R}^N$ a cross-sectional vector and $w \in \mathbb{R}^D$ a temporal filter, we obtain

$$\hat{R}_{t+1} = u\, w^\top X_t.$$

Since the cosine metric is scale-invariant, u can be absorbed into a normalisation and we focus on learning w.

3.2 Regularised Cosine-Similarity Objective

The empirical mean cosine similarity on a training set $\mathcal{T}$ is

$$\mathcal{M}(w) = \frac{1}{|\mathcal{T}|} \sum_{t\in\mathcal{T}} \frac{\hat{y}_t(w)^\top y_t}{\|\hat{y}_t(w)\|_2 \, \|y_t\|_2}.$$

To avoid overfitting, we penalise finite differences of w:

$$J_2(w) = \lambda_2 \sum_{j=1}^{D-1} (w_{j+1} - w_j)^2, \tag{4}$$

$$J_1(w) = \lambda_1 \sum_{j=1}^{D-1} |w_{j+1} - w_j|. \tag{5}$$

3.3 Nesterov-Accelerated Stochastic Optimisation

We optimise $\mathcal{L}(w)$ using mini-batch stochastic gradient descent with Nesterov momentum:

$$v_{k+1} = \mu v_k + \nabla \mathcal{L}(w_k + \mu v_k), \tag{6}$$

$$w_{k+1} = w_k + \eta_k v_{k+1}, \tag{7}$$

where η_k is a learning rate and μ a momentum parameter.

Empirically, adding smoothness regularisation and Nesterov acceleration produces:

- smoother, more interpretable filters;
- more stable convergence across random seeds;
- higher out-of-sample cosine similarity than unregularised or purely MSE-based methods.

3.4 Quantitative Results for the Linear Filter

Summary of the Factor Reconstruction Process

The reconstruction of the ten final explanatory factors proceeds in three stages. First, multiple weight vectors are generated under different regularization regimes (L1, L2, and absolute-value smoothness), and each vector induces a candidate signal through linear filtering of lagged returns. These candidates display diverse temporal responses due to the differing structure of their penalties.

Second, from this pool of candidate vectors, a subset of ten is selected to ensure sufficient diversity. Two "anchor" vectors—one obtained under strong L1 regularization and one under strong L2 regularization—are chosen for their low pairwise correlation. Additional vectors are selected greedily according to correlation constraints to form a representative and non-redundant basis for factor construction.

Third, the final ten factors are constructed as linear combinations of these selected signals. The coefficients are obtained through Gram–Schmidt orthogonalization, ensuring orthogonality, followed by a least-squares projection step that aligns the factors with the predictive structure of the optimized reduced model. Each factor is then normalized to unit length.

This reconstruction maintains nearly the same predictive directionality as the optimized single-filter model while satisfying the challenge's requirement of ten interpretable factors.

On the held-out evaluation segment, the best configuration of the regularised linear filter achieves a mean cosine similarity of approximately

$$\mathcal{M}_{\text{filter}} \approx \mathbf{0.0784}.$$

This represents a clear improvement over the baselines discussed earlier:

- random search: $\mathcal{M} \approx 0.0539$,
- Lasso regression: $\mathcal{M} \approx 0.05$,
- unregularised SGD: $\mathcal{M} \approx 0.054$.

In relative terms, the regularised filter yields roughly a 40–50% increase in cosine similarity over these baselines. More importantly, this gain is robust across different splits and random initialisations, indicating that the improvement is not due to overfitting a particular segment.

3.5 Factor Reconstruction

After learning $w^{\star}$, we reconstruct ten factors to satisfy the challenge format. The reconstruction procedure generates a family of filtered signals with different effective horizons, then performs an orthogonal decomposition (e.g. PCA) to extract ten dominant modes. The resulting factors produce forecasts whose mean cosine similarity is numerically indistinguishable from that of the single-filter model, confirming that the reduction is essentially lossless with respect to the challenge metric.

4 Approach II: Nonlinear Tree-Based Factor Models

4.1 Rolling-Window Design

The nonlinear approach uses a rolling-window scheme. For each window:

1. Train an ensemble model on the most recent W_{train} days;
2. Validate on the following W_{val} days;
3. Extract factors from the fitted ensemble;
4. Estimate linear betas and generate forecasts;
5. Slide the window forward by W_{step} days.

Typical values are $W_{\text{train}} = 200$, $W_{\text{val}} = 50$, and $W_{\text{step}} = 50$.

4.2 Model Classes and Factor Extraction

We consider:

XGBoost
gradient-boosted trees; factors derived from leaf-node statistics.

Random Forest
ensemble of decorrelated trees; factors are principal components of tree-level predictions.

Bagging
bootstrap-aggregated trees; factors capture variability across bootstrap models.

4.3 Quantitative Performance of Tree Models

Table 1 summarises performance on a representative test window in terms of MSE and cosine alignment:

Table 1: Performance of nonlinear models (representative window).

Model	Train MSE	Test MSE	Train Align.	Test Align.
XGBoost	0.012131	0.014901	0.040381	0.020606
Bagging	0.012134	0.014906	0.028721	-0.009982
Random Forest	0.012135	0.014909	0.029236	-0.017423

Several observations follow:

- All three models have very similar train and test MSE; XGBoost is marginally best.
- XGBoost also has the highest test cosine similarity, around 0.021, but this is still significantly below the 0.0784 achieved by the regularised linear filter.
- Bagging and Random Forest even yield slightly negative test alignment on this window, indicating mild anti-correlation between their prediction directions and the realised returns.

These results suggest that, in this setting, MSE is a poor proxy for the cosine metric: reducing MSE via more flexible models does not translate into improved directional accuracy.

5 Error Analysis and Robustness

5.1 Temporal Regimes

Splitting the test period into sub-regimes, we find that:

- The linear filter maintains positive cosine similarity in most regimes, with moderate degradation in highly volatile periods.

- Tree-based models show larger fluctuations, including episodes of negative alignment, consistent with their higher variance and misalignment with the metric.

5.2 Cross-Sectional Structure

Inspection of cross-sectional predictions reveals that:

- The linear filter produces smooth tilts across stocks, resembling broad market or sector exposures.
- Tree-based models produce more irregular, spiky patterns, placing large weights on a few names. These bets can reduce MSE when they are correct but hurt cosine similarity when they are wrong, as they flip the direction of the cross-section.

6 Discussion and Conclusion

From a scientific perspective, the main lessons are:

1. **Metric alignment matters**. The linear filter is explicitly optimised for cosine similarity, and this is reflected in its superior score (≈ 0.0784) relative to both linear and nonlinear baselines.
2. **More complex models are not always better**. Tree ensembles slightly improve MSE but fail to match the linear filter on the directional metric; their extra flexibility mainly increases variance.
3. **Regularisation should encode financial structure**. Temporal smoothness of filter weights is both a statistical regulariser and a financially meaningful assumption, ruling out implausibly fast oscillations in signal.

Overall, the regularised linear filter strikes a favourable bias–variance trade-off under the cosine metric, while nonlinear tree-based models provide useful diagnostic insight but do not translate into better directional forecasts in this challenge. Future work could explore hybrids that use nonlinear models to generate candidate factors and linear, metric-aligned filters to aggregate them.