Learning from Missingness: Enhancing imputation with the NN3 architecture

FIN-407: Machine-Learning in Finance

Tallula Graber, Piotr Kleymenov, Noah Louis Truttmann and Rocco Pio Lorenzo Ventruto

2025

Contents

1 Introduction and problem statement

Reliable one-month-ahead return forecasts are essential for asset managers aiming to enhance risk-adjusted performance, optimize portfolio-rebalancing schedules, and uncover which empirical anomalies remain persistent over decades. Daily returns are often too noisy, while quarterly or annual horizons can mask short-term dynamics; the monthly return therefore offers the right balance of signal and stability.

This project build and evaluate a deep-learning (MLP) pipeline that predicts one-month-ahead excess returns of U.S. common stocks, comparing its performance to standard machine-learning models such as Lasso regression over the period from July 31, 1961 to December 31, 2024. By covering multiple market regimes, from the post-Kennedy boom, through the tech bubble, the 2008 crisis, and up to today's low-rate environment, we test model robustness across very different conditions.

The returns we aim to predict come from the CRSP Security-Level Prices database, which provides daily and monthly price and return series for U.S. common stocks and serves as the foundation for our classical technical and return-based features. We augment these with the Compustat North America Fundamentals Quarterly database which provides standardized accounting data from income statements, balance sheets, cash-flow statement for U.S. and Canadian public companies. In addition, we use the well-known Jensen–Kelly–Pedersen factor returns: 153 monthly, zero-cost long–short portfolios that proxy the state of traditional anomaly premia.

The challenge lies in handling the high-dimensional, time-varying nature of financial data while managing issues such as missing values, outliers and non stationary financial markets.

2 Data preprocessing and feature engineering

For the CRSP database, we first removed all rows that lacked an observation for our target variable (the monthly return) and any stocks with only a single data point. Next, we dropped all redundant identifier columns and any string fields that could not be usefully encoded. Finally, by using the Fama French monthly risk free rate, we computed our excess return target variable and constructed a series of price-based technical features: the one-month-lagged return, 3-, 6-, and 12-month momentum, 12-month downside risk, skewness, kurtosis, maximum drawdown and sharp ratio.

Regarding the Compustat dataset, we started by dropping string features that were neither interesting nor variable, one-hot-encoded certain others that might carry specific signals and retained only those features with fewer than 60 % missing values. Next, we focused our analysis exclusively on industrial, active firms reporting under GAAP, and considered only the consolidated data. Moreover, since the features were presented as year-to-date figures, we converted them to quarterly data to facilitate both variable selection and model predictions.

For the JKP dataset, we performed only NaN-imputation cleaning, which will be detailed later.

To link CRSP and Compustat data reliably, we use the CRSP/Compustat Merged (CCM) database. CCM database ensures that each Compustat gvkey is matched to the correct CRSP permno over the appropriate validity period, avoiding mismatches and look-ahead bias. We retained only observations for which both links exist. We focus exclusively on primary and co-primary share classes, one-hot-encode the co-primary share information to capture any relevant signal, and align quarterly data with the corresponding three monthly returns. JKP characteristics, which are indexed only by month, are mapped onto the returns according to their dates.

In all three datasets, we apply a systematic procedure to impute the remaining NaNs after the initial preselection filters. First, for any feature that contains only 0s and NaNs for a given asset, we fill all values with 0. Next, we classify the remaining NaNs into three categories:

- **Bottom Block (BB):** NaNs occurring from the asset's first observation onward.
- **Top Block (TB):** NaNs occurring from a specific date until the asset's last observation.
- **Mid Block (MB):** NaNs occurring between two non-NaN value blocks.

For BB and TB patterns, we replaced NaNs with zeros by propagating forward for BB and backward for TB, up to the first non-zero, non-NaN value. For MB gaps, we imputed the cross-sectional median of the affected feature at that date.

Prior to imputing the NaNs, we constructed a flag matrix of the same dimensions as the original DataFrame. Each entry that originally contained a value is marked with a 0, and each entry that was a NaN and has since been imputed is assigned a 1. The idea is to enable the model to treat these cases differently, to capture missingness as a signal and to help the network handle imputation more precisely.

We perform winsorization and standardization repeatedly on rolling windows, directly within the model and prior to feature selection. This procedure accounts for the non-stationarity of financial markets and prevents look-ahead bias by ensuring that neither winsorization nor standardization relies on future values. Both operations are applied feature-wise on a cross-sectional basis. All continuous variables are winsorized at the 1st and 99th percentiles to limit extreme outliers while preserving the original distributional shape. For standardization, we apply Z-score normalization within each rolling window.

3 Predictive models used

3.a NN3 model architecture

To focus on the most informative variables and suppress spurious signals in our excess-return model, we use a three-stage pipeline that combine Random Forest-based feature selection, hyperparameter optimization, and a neural network. First, we train a Random Forest regressor on the preprocessed training data and compute feature-importance scores, the goal of performing a preselection is to prevent the model from overfitting. To choose the best number of features k (with $20 \leq k \leq 50$), we perform a chronological split of the training set, using the earliest 80 % of observations for fitting and the most recent 20 % for validation, to avoid temporal leakage. This classifies the parameters in order of importance. We then record the validation R^2 for each number of features between 20 and 50 (with a step of 5, so 20, 25, ..., 50); the k with highest out-of-sample R^2 is chosen for downstream modeling. Next, the selected features are used to train a neural network, with a 3-layer architecture of [32, 16, 8] neurons with ReLU activations, batch normalization, and dropout regularization. This architecture was inspired from the NN3 architecture found in Gu et al. (2020)[2]. Within each training window, we grid-search key hyperparameters using the validation set to maximize model performance. The whole process is repeated at each window: 20 years train, 5 years validation and 1 year test. The default hyperparameters used were `N_estimators = 1`, `dropout_rate = 0.2`, `learning_rate = 1e-4`, `batch_size = 256`, `epochs = 100`, `patience = 20` and `weight_decay = 1e-5`.

Below, we describe the differences between each of the three versions of the model.

3.a.i NN3 HR (Heavily Regularized)

This model is the starting point, being exactly the architecture described above, while adding an L1 regularization to the weights of the first layer, namely the layer that goes from the selected

features to the first hidden layer neurons. A grid search is done on the L1 lambda and weight decay, with values `l1_lambda`: {1e-6, 1e-5} and `weight_decay`: {1e-6, 1e-5}. By combining feature selection, batch normalization, dropout, L1 on the input layer, and L2 on all weights, this model is heavily regularized to avoir overfitting at all costs.

3.a.ii NN3 MR (Mildly Regularized)

To mitigate underfitting, which could result from the very regularized NN3 HR model, we wanted to alleviate the regularization. In this architecture, we take the NN3 HR and remove the L1 regularization on the weights of the first layer. We do a grid-search on solely the dropout coefficient, with values `dropout`: {0.05, 0.1, 0.15}. The weight-decay is reset to `weight_decay = 1e-5`.

3.a.iii NN3 MR + F (Mildly Regularized + NaN Flags)

To test the effectiveness of adding the NaN flag matrix to our data set, we tweak NN3 MR model. After the feature selection with Random forest, we add to each selected feature the corresponding flags. This should help the model capture missingness as a signal and help the network handle imputations more precisely. Except the flags, everything else is equal with respect to the NN3 MR model.

3.b Lasso regression

The Lasso is a linear regression equipped with an ℓ_1 penalty. Because the penalty drives many coefficients exactly to zero, the model remains parsimonious even when the feature set is huge, which is common in return prediction problems. That simplicity comes with trade-offs. Lasso is strictly linear: it captures no interactions or polynomial effects. More importantly, the textbook way to pick the hyper-parameter α, cross-validation that minimises mean-squared error, optimizes a statistic that does not necessarily translate into stronger out-of-sample.

To address the disconnect between statistical performance and practical trading outcomes, we sidestepped exhaustive searches like GridSearchCV or the default LassoCV grid. Instead, we fixed a short list of α values (0.0005,0.001,0.0015,0.002,0.0025,0.003,0.0035,0.004,0.0045,0.0055) and tested each on historical data on a rolling window (25 years train / 1 year test). We analyze the out of sample and strategic results for each α and choose the configuration that offers the most compelling balance. Because this Lasso selection is done ex post rather than ex ante, it is not a deployable baseline but rather a ceiling against which to compare our deep-learning approaches on both predictive and strategic metrics.

4 Evaluation methodology and performance metrics

Each neural network is trained using Adam optimization with early stopping based on validation loss, and final predictions are obtained by the Neural network outputs. The choice of the α for the Lasso used as an upper-bound comparison follows a simple rule: we select $\alpha = 0.0025$, since it corresponds to the elbow of the median out-of-sample R^2 curve (c.f. figure 5), beyond which larger α values yield negligible statistical gains, while avoiding the sharp decline in our trading strategy returns observed for $\alpha > 0.0025$ (c.f. figure 6).

4.a Predictive power

We begin by assessing the models ability to predict future stock returns using the coefficient of determination (R^2). For each window, we compute the **in-sample (IS)** R^2, measuring fit on the training data and the **out-of-sample (OOS)** R^2, measuring generalization to unseen

data. This distinction enables us to detect potential overfitting and to compare models' robustness, especially in different periods. The evolution of R^2 across the 38 windows is plotted and compared across model variants in figure 1.

In purple lines, we have added important dates where crashes happen, to see how our models react. The dates are the Black Monday Crash in October 1987, the Dot-Com Bubble Peak in March 2000, the Global Financial Crisis (Lehman fall-out) in September 2008 and the COVID-19 Market Panic in March 2020, that will be marked in all graphics.

Note that the dates reported are the test end dates. The values of the OOS R^2 correspond to the test set that ended at t and started at t-1, while the IS R^2 reported for the same date corresponds to the train set which ended at t-5 and started at t-25 (because the validation set is in between).

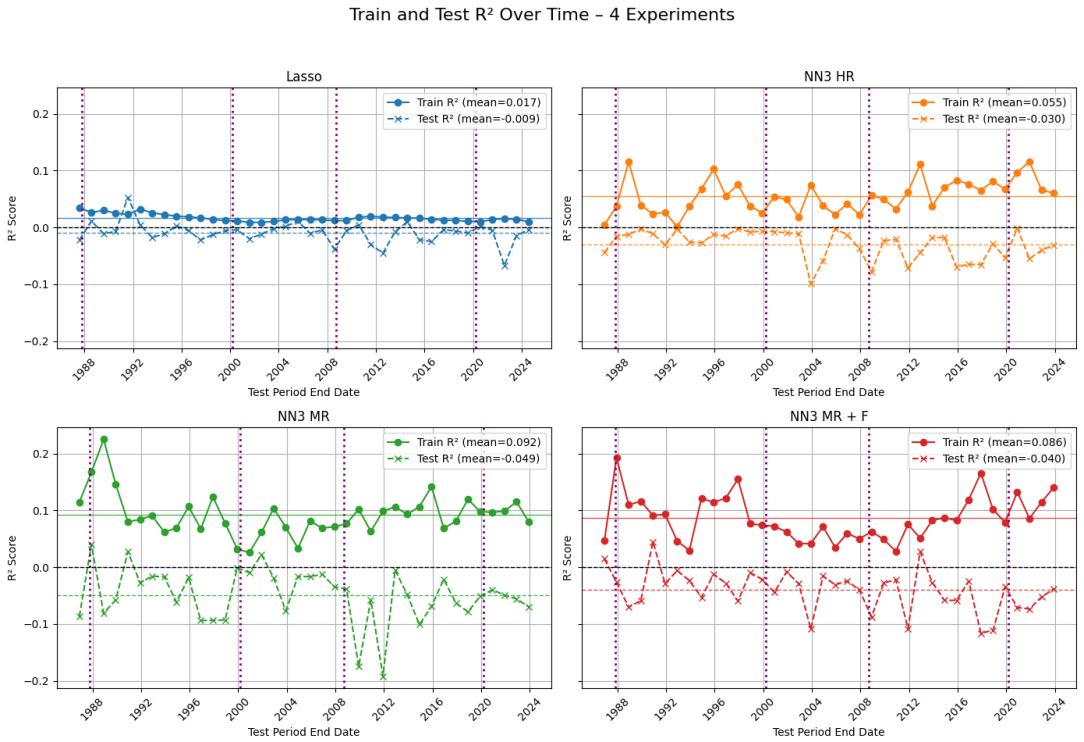

Figure 1: IS and OOS R^2 over time and models

As we can see when comparing the three NN3 models, our hypothesis regarding the highly regularized model NN3 HR seems to be correct. NN3 HR tends to generalize better (mean R^2 OOS = -0.030), but does not learn well on the train data. When we switch to the mildly regularized NN3 MR, we see that the IS R^2 is much higher, while the mean OOS R^2 is lower. This indicates that the less regularized model fits the training data better, but generalizes worse. When we add the flags in NN3 MR + F, we see that the general metrics are very similar to NN3 MR, but the model generalizes slightly better, indicating that the model seems to be able to capture missingness as a signal and handle imputations more precisely. We also note that the OOS R^2 is more stable, with no values larger than -0.1 across windows.

Compared to Lasso, all three NN3 models achieve significantly higher in-sample R^2 (Lasso's mean IS R^2 is only 0.017), demonstrating stronger learning in the training set. However, Lasso clearly generalizes better, with the highest mean out-of-sample R^2 of -0.009, and remarkably stable performance across windows. However, we know that this is in the nature of the lasso regression, since it is a linear tool. We should also not forget that it was chosen ex-ante, and

should only be regarded as an ideal linear upper bound.

Regarding the crash dates, we can see that for the 1987 crash, there is little to no impact on the models' out-of-sample R^2. In contrast, both the 2000 dot-com bubble and the 2008 financial crisis coincide with noticeable drops in predictive performance, particularly for NN3 MR and NN3 MR + F. These models, while stronger in-sample, appear more sensitive to sudden market regime changes, as shown by sharper declines in test R^2 during these periods. NN3 HR, although weaker overall, maintains relatively stable performance through these events. Interestingly, the 2020 COVID-19 market shock has a limited effect on all models, with only minor fluctuations in test R^2, suggesting improved robustness in recent years or faster market recovery. This highlights a trade-off between in-sample learning strength and resilience to structural breaks. Interestingly, the Lasso model shows very limited sensitivity to the major market crashes. Its out-of-sample R^2 remains relatively flat across all four crisis dates, including the 2000 dot-com bubble and the 2008 financial crisis. This suggests that, although it has lower predictive power, the Lasso exhibits greater robustness to structural breaks, likely due to its strong regularization. Again however, this tool was chosen ex-ante, and cannot be regarded as a direct comparison with our models.

4.b Strategy evaluation

At the end of every month we form a zero-cost long–short portfolio by ranking all stocks on their predicted returns, buying the top 1 percent and shorting the bottom 1 percent with equal weighting inside each sleeve so the long side totals +0.5 and the short side –0.5. The following month, rather than closing out every position and rebuilding the portfolio from scratch, we simply adjust our existing holdings, avoiding unnecessary transaction costs. All results are stated net of realistic implementation frictions: we apply a transaction cost of 10 bps per 100 % turnover (i.e. cost = 0.001 × turnover), and we reduce the S&P 500 benchmark's return by an annualized TER of 3 bps to simulate a competitive ETF (Exchange Traded Fund) (see [3]).

To evaluate the resulting strategies, we report the following performance metrics: **mean monthly return**, **annualized return**, **annualized volatility**, **Sharpe ratio (SR)**, **Calamar ratio** and **maximum drawdown**. These indicators allow us to quantify the risk-adjusted profitability of each model's predictions when used in practice.

This section synthesizes predictive and strategic insights to evaluate the practical value of each approach.

Table 1: Long–short statistics (top 1%–bottom 1%) net of costs + Market

Model	Mean month.	Ann. return	Ann. vol.	Sharpe	Calamar	Max drawdown
Lasso $\alpha = 0.0025$	0.0076	0.0915	0.1225	0.7466	0.2609	0.3506
NN3 HR	−0.0006	−0.0077	0.1145	−0.0670	−0.0116	0.6622
NN3 MR	0.0020	0.0243	0.1228	0.1982	0.0628	0.3875
NN3 MR + F	0.0024	0.0284	0.1085	0.2619	0.0850	0.3342
Market	0.0088	0.1060	0.1696	0.6250	0.1999	0.5303

Figure 2 plots the value (net of costs) of five portfolios, each starting at \$1: the Lasso strategy with $\alpha = 0.0025$, three neural-network variants (NN3 HR, NN3 MR and NN3 MR+F) long short portfolios at threshold 1% and the passive ETF market benchmark (S&P).

4.b.i General performance

We can clearly see that our 3 models perform very badly compared to the market and the ex-ante Lasso upper bound. Interestingly, looking at table 1, we see that our models' strategies

have lower volatilities than the market, but due to the very low returns, as corroborated in figure 2, the Sharpe ratios are much lower. While the max drawdowns are also lower than the market's, the Calamar ratios are still very low given the important difference in returns. Overall, the portfolio-value trajectory produced by the NN3 strategy operates on a completely different order of magnitude compared to both our ex ante Lasso benchmark and the market.

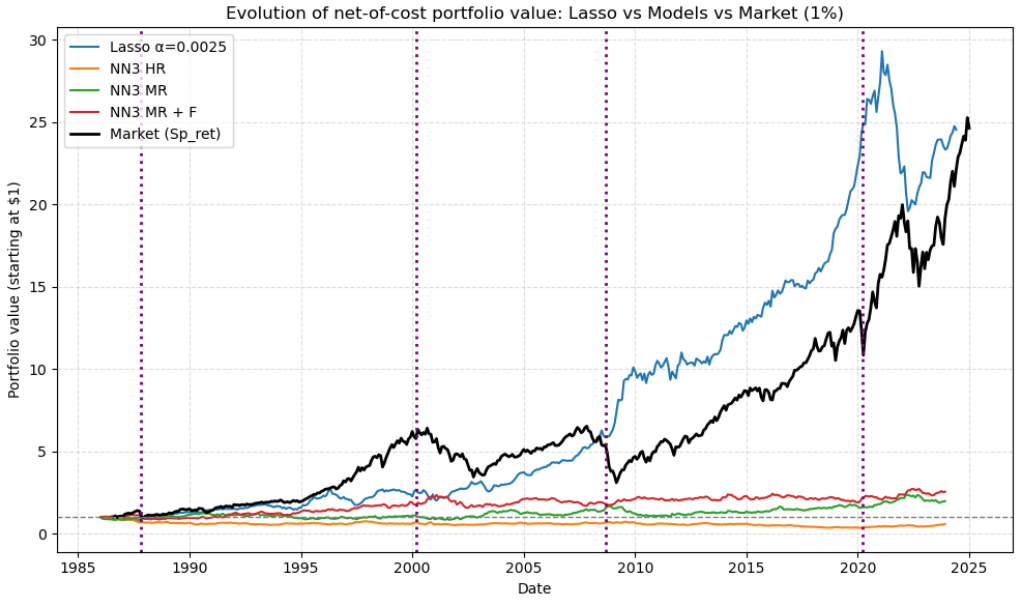

Figure 2: Evolution of net-of-cost portfolio value: Lasso vs. NN3 models vs. Market.

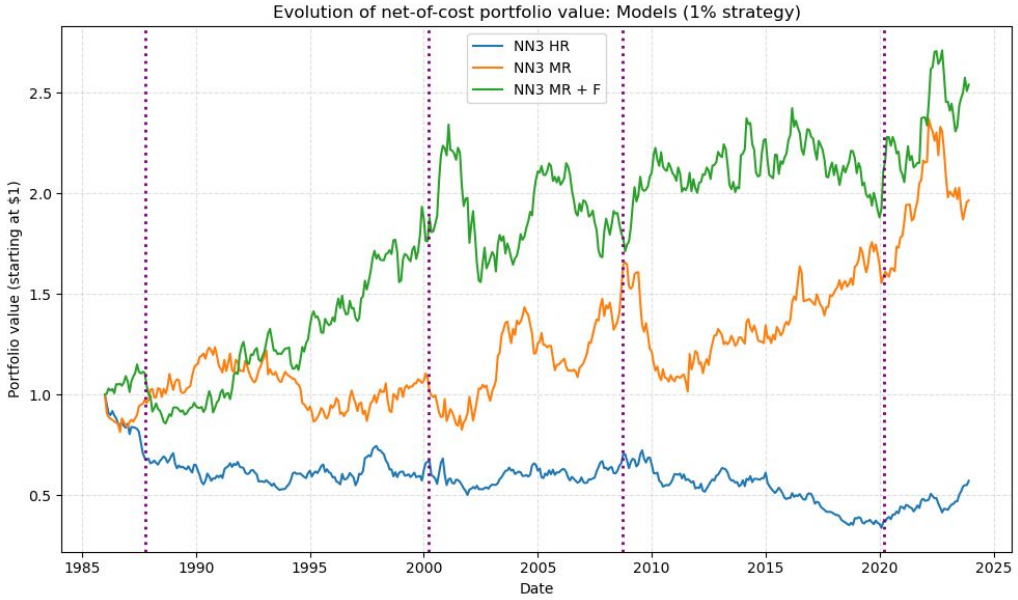

Figure 3: Net-of-cost portfolio value for NN models using a 1% long–short rule.

4.b.ii NN3 models performance

Looking at figure 3, we can compare the performances between our models directly. We see that our best performing model is the NN3 MR + F, closely followed by the NN3 MR. While the NN3 HR was exhibiting the highest average OOS R^2 of the 3 models (cf. figure 1), it is the worst performing one of the three.

Among the three NN3 models, NN3 MR + F achieves the best performance overall. It delivers the highest annualized return (2.84%) and Sharpe ratio (0.2619), while also maintaining the lowest maximum drawdown (33.4%), reflecting superior risk-adjusted performance. NN3 MR follows with slightly lower returns and a similar drawdown profile, but noticeably lower Sharpe and Calmar ratios. In contrast, NN3 HR fails to generate positive returns, posting a negative annual return of -0.77% and the highest drawdown among the three. These results are consistent with the figure 3, where NN3 MR + F outperforms across most of the period, especially after 2008, while NN3 HR stagnates and decays. Looking at performance during the crisis periods (purple lines), we see that our top model(NN3 MR + F), is not only resilient but actually thrives in those environments, exhibiting sharp upticks in returns immediately following the 2000, 2008, and 2020 instabilities.

4.c Joint analysis and insights

By combining our Strategy and Predictive Power evaluations we can see, from section 4.a, that the model with best generalization to unseen data was NN3 HR, exhibiting the highest OOS R^2 of our models. However, looking at the strategy performance, we clearly see that it is the worst of the three. The best strategy actually comes from the NN3 MR + F, which did not generalized the best. An other factor may need to be taken into account, wich is directional accuracy: as shown in Figure 8, the NN3 MR model achieves close to 54% directional accuracy, meaning that even if it misses the exact return magnitude, it often predicts the correct sign, sufficient to drive strong portfolio performance. On the other hand, NN3 HR, which performs worst as a strategy, has a lower directional accuracy of just 47%, supporting the idea that R^2 alone may not capture the predictive qualities most relevant for portfolio construction. Notably, the NN3 MR + F model also appears to perform especially well during crisis periods, as previously discussed.

5 Conclusion

Our research demonstrates that deep learning models for financial return prediction face fundamental challenges that limit their practical applicability, despite methodological innovations. The implementation of a NaN flags improved NN3 MR + F performance compared to its non-flagged counterpart, indicating that the model seems to capture missingness as a signal and handles imputation more precisely. However, our findings reveal a critical trade-off between regularization and performance: while the heavily regularized NN3 HR model generalizes better, it severely underperforms in learning capacity, whereas the mildly regularized NN3 MR seems to achieves superior directional prediction accuracy of 53% on extreme predictions. Most concerning for practical implementation, all NN3 models' strategies substantially underperform relative to the ideal ex-ante Lasso baseline and most importantly the market benchmarks, primarily due to high turnover costs. Nevertheless, our analysis of larger model architectures suggests potential for improved directional accuracy, although computational constraints prevented full exploration. To address these fundamental limitations, we propose prioritizing the development of classification-based approaches specifically optimized for directional prediction rather than continuous return forecasting, combined with transaction cost optimization techniques and selective trading strategies focused on the most confident predictions. While we implemented the ensemble method in to our code, we unfortunately did not have the computational resources to use them. Future research should explore this and transformer based architectures, all while acknowledging that the path to commercially viable algorithmic trading strategies requires both methodological advances and richer feature environments than currently available in public datasets.

References

[1] Vitor Azevedo, Christopher Hoegner, and Mihail Velikov. *The Expected Returns on Machine-Learning Strategies*. Available at SSRN: https://ssrn.com/abstract=4702406. Nov. 2023. URL: http://dx.doi.org/10.2139/ssrn.4702406.

[2] Shihao Gu, Bryan Kelly, and Dacheng Xiu. "Empirical Asset Pricing via Machine Learning". In: *The Review of Financial Studies* 33.5 (Feb. 2020), pp. 2223–2273. ISSN: 0893-9454. DOI: 10.1093/rfs/hhaa009. eprint: https://academic.oup.com/rfs/article-pdf/33/5/2223/33209812/hhaa009.pdf. URL: https://doi.org/10.1093/rfs/hhaa009.

[3] Vanguard. *Vanguard S&P 500 ETF (VOO)*. https://investor.vanguard.com/investment-products/etfs/profile/voo. Accessed: 2025-06-03. 2024.

A Appendix

Table 2: Lasso long–short statistics (top 1%–bottom 1%) by α, net of costs

α	Mean monthly	Annualized return	Annualized vol	Sharpe	Max drawdown
0.0005	0.0153	0.1835	0.1808	1.0148	0.4846
0.0010	0.0151	0.1812	0.1659	1.0920	0.3836
0.0015	0.0105	0.1264	0.1280	0.9874	0.2423
0.0020	0.0081	0.0972	0.1158	0.8400	0.3012
0.0025	0.0076	0.0915	0.1225	0.7466	0.3506
0.0030	0.0088	0.1051	0.1335	0.7878	0.3675
0.0035	0.0015	0.0175	0.1506	0.1162	0.8103
0.0040	-0.0020	-0.0241	0.1530	-0.1578	0.8337
0.0045	-0.0028	-0.0341	0.1121	-0.3040	0.7880
0.0055	-0.0045	-0.0535	0.0885	-0.6052	0.8899

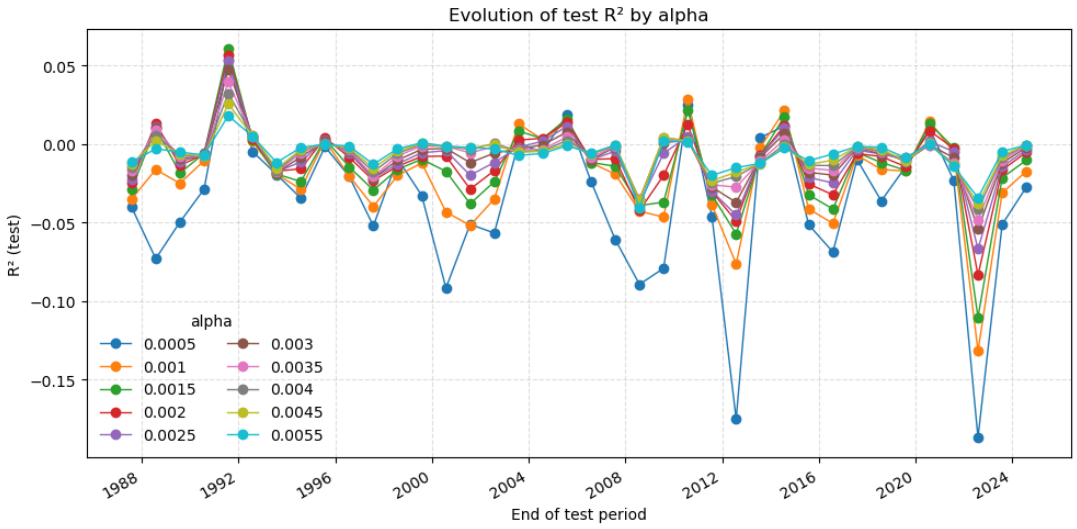

Figure 4: Evolution of Lasso test R^2 by alpha

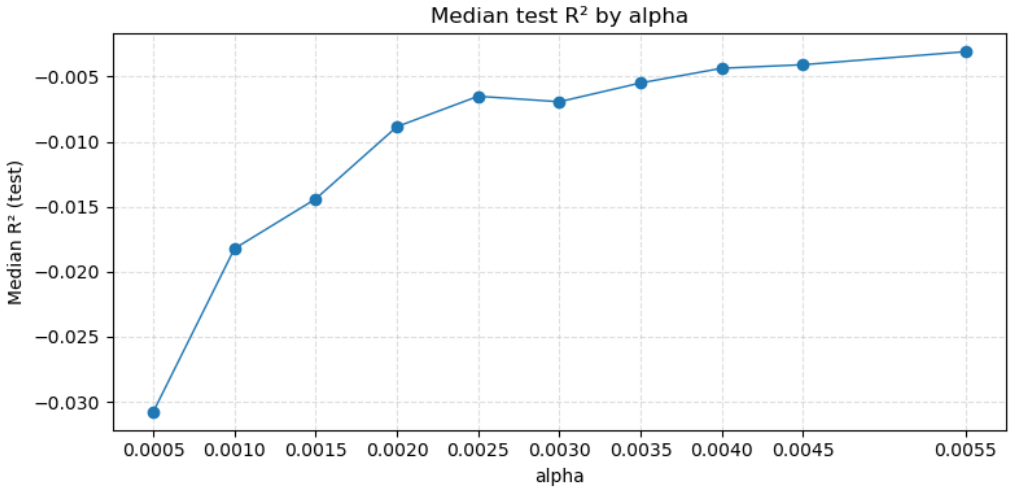

Figure 5: Lasso median test R^2 by α.

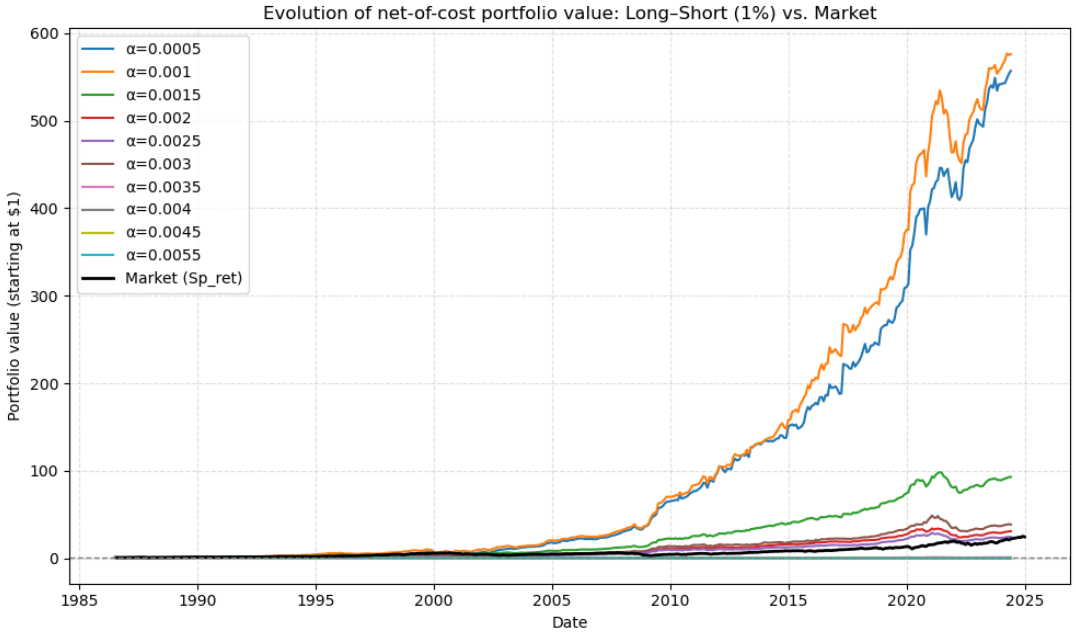

Figure 6: Evolution of net-of-cost portfolio value: Lasso long–short (1%) vs. Market.

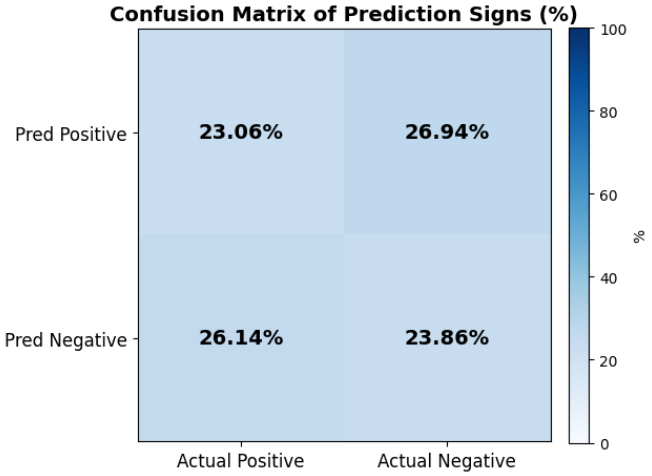

Figure 7: Confusion matrix for the NN3 HR model on top and bottom 1% of predictions

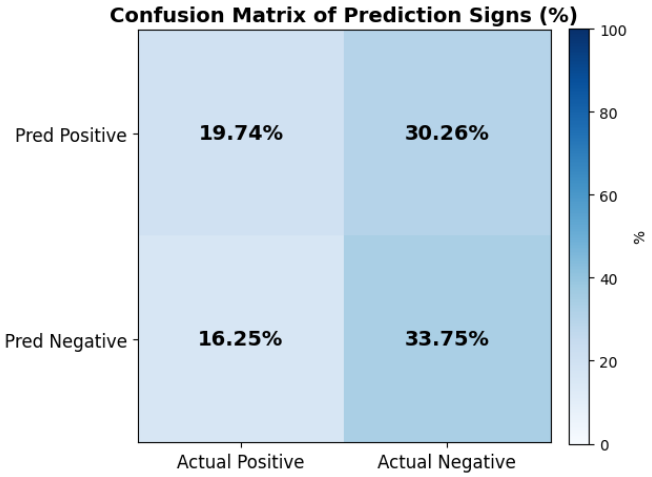

Figure 8: Confusion matrix for the NN3 MR model on top and bottom 1% of predictions

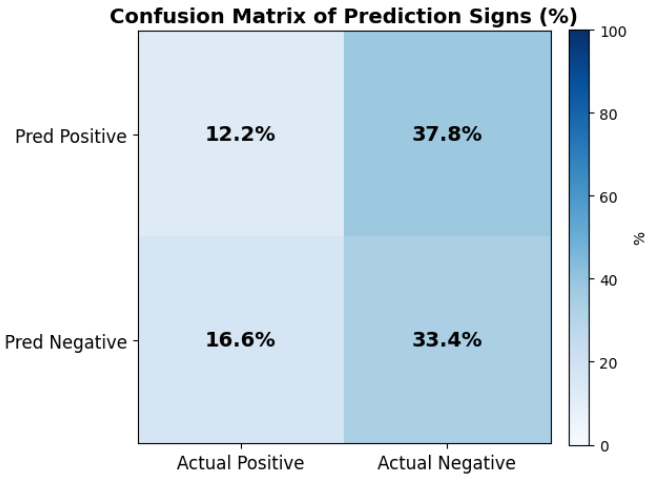

Figure 9: Confusion matrix for the NN3 MR + F model on top and bottom 1% of predictions