The Expected Returns on Machine-Learning Strategies

Vitor Azevedo[a,*], Christopher Hoegner[b], Mihail Velikov[c]

[a] *School of Business and Economics, RPTU Kaiserslautern-Landau, Gottlieb-Daimler-Straße 42, 67663 Kaiserslautern, Germany*

[b] *Department of Financial Management and Capital Markets, TUM School of Management, Technical University of Munich, Arcisstr. 21, 80333 Munich, Germany*

[c] *Smeal College of Business, Penn State University, State College, PA 16802, U.S.*

Abstract

This study assesses the expected returns of machine learning-based anomaly trading strategies, accounting for transaction costs, post-publication decay, and the post-decimalization era of high liquidity. Contrary to claims in prior literature, more sophisticated machine learning strategies are profitable, earning net out-of-sample monthly returns of up to 1.42%, despite having turnover rates exceeding 50% and selecting some difficult-to-arbitrage stocks. A trading strategy that employs a long short-term memory model to combine anomaly characteristics yields a six-factor generalized (net) alpha of 1.20% (t-stat of 3.46). While prevalent cost-mitigation techniques reduce turnover and costs, they do not improve net anomaly performance. Overall, we document return predictability from deep-learning models that cannot be explained by common risk factors or limits to arbitrage.

Keywords: Stock market anomalies; machine learning models; return prediction; transaction costs; asset pricing models.

JEL classifications: G11, G12, G14, C45, C58.

*Corresponding author.

Email addresses: vitor.azevedo@rptu.de (Vitor Azevedo), christopher.hoegner@tum.de (Christopher Hoegner), velikov@psu.edu (Mihail Velikov)

1. Introduction

A growing body of literature in finance documents the remarkable ability of machine learning techniques to enhance predictability in the cross-section of stock returns.[1] Studies that use these techniques to extract expected return signals routinely report annualized Sharpe ratios on trading strategies employing these signals in excess of 1.0, with extreme examples exceeding 2.0 (e.g., Freyberger et al., 2020; Chen et al., 2023; Cong et al., 2020), performance that corresponds to about five times the historical market Sharpe ratio of 0.43, estimated over the entire CRSP sample from 1925-2021.

Despite this impressive paper performance, the extent to which these strategies can be implemented in practice remains an ongoing debate. Avramov et al. (2022) argue that trading strategies based on machine learning models extract profitability from difficult-to-arbitrage stocks and during high limits-to-arbitrage market states, and their performance deteriorates in the presence of economic constraints because of high turnover. Blitz et al. (2023) advocate using longer prediction horizons to train the machine learning models and show that those can improve performance even after accounting for 25-basis-points-per-trade transaction costs. Jensen et al. (2022) go further and develop a framework that integrates trading-cost-aware portfolio optimization with machine learning.

Complicating the matters, simply accounting for transaction costs still does not answer the question of what the expected returns on machine learning strategies are and whether we can expect to see similar performance in the future. This question is more subtle, because many anomalies were not discovered for significant periods of the samples in which they are typically used for backtesting machine learning strategies. Thus, even if we make the optimistic assumption that the machine learning techniques were available for investors, the anomaly signals were not. Moreover, individual anomaly performance deteriorates post-publication (Mclean and Pontiff, 2016) and has further deteriorated in the more recent sample post-decimalization due to the new era of investment and trading technology (Chordia et al., 2014).

Chen and Velikov (2023) show that the profitability of the average anomaly virtually disappears after accounting for three distinct effects: 1) transaction costs, 2) post-publication decay, and 3) the effect of decimalization on liquidity. In their data, the strongest anomalies net expected returns of around 10 basis points per month, and even simple strategies that combine

[1] See, e.g., Gu et al. (2020); Leippold et al. (2022); Hanauer and Kalsbach (2023); Azevedo and Hoegner (2023); Chen et al. (2023); Azevedo et al. (2023); Cakici et al. (2023).

the entire anomaly zoo yield around 20 basis points per month in expectation. Whether their conclusions extend to more sophisticated machine learning techniques, however, is an empirical question.

In this study, we aim to fill this gap in the literature and estimate the expected returns on machine-learning anomaly strategies accounting for the three effects noted above. To this end, we estimate return signals by combining up to 320 *published* anomalies from the Chen and Zimmermann (2022) dataset using nine different machine learning techniques: Ordinary Least Square with Huber Loss Function (OLS-HUBER), an Elastic Net (ENET), which combines a Least Absolute Shrinkage and Selection Operator (LASSO) and ridge regression, Feedforward Neural Network (FFNN) with two to five hidden layers (FFNN2, FFNN3, FFNN4, and FFNN5), two variations on Long Short-Term Memory (LSTM) with one and two hidden layers (LSTM1 and LSTM2), and an ensemble model. We construct trading strategies based on these signals and study their post-2005 performance net of the high-frequency effective bid-ask spread estimates from Chen and Velikov (2023).

We find that machine learning combination signals perform well based on common out-of-sample regression metrics, even during the more recent era of high liquidity, and only including anomalies after their publication dates. For example, the out-of-sample R^2's vary from 0.05% for OLS-HUBER to 0.76% for LSTM1. The feedforward neural network-based models rank in between, with R^2's ranging from 0.19% to 0.36%. Similarly, decile-sorted, value-weighted trading strategies based on the machine learning combination signals earn significant returns of up to 1.63% per month and Fama and French (2018) six-factor model alphas of up to 1.40% per month before costs in the post-2005 period. The average returns are also accompanied by economically large Sharpe ratios, ranging from 0.32 (OLS-HUBER) to 1.11 (LSTM1).

Due to the improved liquidity in the post-2005 period, the trading costs associated with these strategies are on the order of 20-25 basis points per month. As a result, contrary to the insights of Avramov et al. (2022), most machine learning combination strategies continue to deliver significant returns net of costs despite one-sided turnover ranging between 60% and 70%. While the Chen and Velikov (2023) effective spread estimator renders the average returns to a few of the strategies insignificant (OLS-HUBER, FFNN2, FFN5), the rest of the signals continue to generate sizable and statistically significant average returns ranging from 0.64% per month (t-statistic of 2.43) for ENET to 1.42% per month (t-statistic of 3.99) for LSTM1.

The machine learning signals continue to generate significant net performance even after

accounting for their exposure to the Fama and French (2018) six-factor model factors, gauged using Novy-Marx and Velikov (2016) generalized alphas, which measure the extent to which a test asset improves the ex-post mean-variance efficient portfolio, accounting for the costs of trading both the asset and the explanatory factors. The generalized alphas of all machine learning strategies are positive, with five out of nine being also statistically significantly different from zero. This indicates that an investor with access to the net returns of the six Fama and French (2018) factors would have benefited from investing in the net machine learning strategies.

However, the construction of the machine learning strategies discussed so far largely ignores trading costs. Avramov et al. (2022) apply economic constraints, such as excluding the bottom 20% of stocks by market capitalization or filtering out the 30% stocks with the highest previous transaction costs. Similarly, Novy-Marx and Velikov (2016) and Novy-Marx and Velikov (2019) explore transaction cost mitigation techniques such as increasing the holding period to up to four months or introducing trading hysteresis through a Buy-Hold-Spread (BHS) (i.e., entering a position for a top/bottom decile but exiting it only if they fall out of the top-/bottom-quintile). Applying these mitigation techniques results in a significant reduction in turnover and transaction costs. However, the corresponding reduction in gross average returns makes up for it in most cases. Increasing the holding period to two months is the only technique leading to average net performance improvements across all machine learning strategies. However, even in that case, that improvement is marginal at only five basis points.

Our paper contributes to a fast growing literature on using machine learning in asset pricing settings. Many papers demonstrate the impressive predictive power of machine learning signals in the cross-section of U.S. stock returns. For example, Freyberger et al. (2020) use adaptive group LASSO, Gu et al. (2020) survey and apply many machine learning techniques including elastic net, dimension reduction techniques (PCR and PLS), trees, and neural networks. Cong et al. (2020) apply a deep reinforcement learning model. Simon et al. (2022) use neural networks to optimize portfolio weights as a function of firm characteristics. Chen et al. (2023) apply both feedforward and recurrent neural networks with long short-term memory.

Our main contribution relative to all these studies is our focus on the expected returns of machine learning strategies through careful treatment of trading costs, post-publication decay, and the staleness of historical data. While most of the studies cited above attempt to address the issue of implementability, they do so indirectly through economic constraints in the spirit of Avramov et al. (2022) such as size or turnover cutoffs or using crude trading costs measures.

For example, Freyberger et al. (2020) use the Brandt et al. (2009) trading costs imputation based on size and Blitz et al. (2023) use a flat 25 basis points per trade assumption. The Chen and Velikov (2023) effective bid-ask spread measure we employ presents a more realistic estimate of trading costs post-decimalization since it is based on high-frequency TAQ data. To the best of our knowledge, we are also the first to estimate the Novy-Marx and Velikov (2016) generalized alphas for machine learning strategies. Integrating these generalized alphas into machine learning approaches enables a more precise comparison and understanding of risk-adjusted net returns. This method directly addresses the frequently overlooked impact of trading costs in asset pricing models, a factor often neglected in studies that use gross return asset pricing models to explain anomalies in net returns.

Furthermore, our study demonstrates that machine learning strategies can be profitable, even in the recent era of high liquidity and when using only discovered anomalies. This is in stark contrast to the conclusions in Avramov et al. (2022). While it is true that machine learning strategies concentrate on historically difficult-to-arbitrage stocks, value-weighting stocks in the portfolios and the sharp decline in trading costs over the past couple of decades combine to result in significant profits for these strategies.

Our study is also related to recent papers that use machine learning techniques to construct improved factor models (Feng et al., 2023), show that technical analysis works, though its profitability decreases through time (Brogaard and Zareei, 2023), explain the post-earnings announcement drift (Hansen and Siggaard, 2023; Meursault et al., 2023), measure firm complexity (Loughran and McDonald, 2023), and uncover sparsity and heterogeneity in firm-level return predictability (Evgeniou et al., 2023).

Finally, we also add to the debate on the virtue of complexity. Kelly et al. (2023) establish the rationale for using machine learning to model expected returns and theoretically show that "complex" models should outperform "simple" models. Consistent with their findings, our strongest results are obtained using the most sophisticated machine learning models - the LSTM. All of our machine learning strategies are stronger compared to the ones in Chen and Velikov (2023), who find that using simpler combination techniques results in measly expected returns for strategies based on sorts of individual stock returns.

2. Methodology and Data

This section describes our data and methodology.

2.1. Data Sources, Samples and Pre-Processing

Our anomaly data come from Chen and Zimmermann (2022), who provide the most comprehensive dataset of replicated anomalies.[2] We use the March 2022 version of their signals. Motivated by Kelly et al. (2023), we download all 320 anomalies to ensure we provide our machine learning signals with the largest set of characteristics possible.

We follow common practice and include only common equity stocks (CRSP share code 10 or 11). Our sample is from March 1957 to December 2021, totaling ~3.4mn stock-month observations over nearly 65 years. The anomaly signals have varying ranges of values over which they are defined, making it more difficult for neural networks to estimate suitable parameters during training (Singh and Singh, 2020). Consequently, we follow the current literature by percent-ranking all anomaly features into the same range [-1;1] (Kelly et al., 2019; Freyberger et al., 2020; Gu et al., 2020). Missing values are replaced with 0.

We enrich the anomaly dataset with further relevant data points following Gu et al. (2020) and include eight key macroeconomic predictors of Welch and Goyal (2008), namely dividend-price ratio, earnings-price ratio, book-to-market ratio, net equity expansion, treasury-bill rate, term-spread, default spread, and stock variance. The objective is to inform our models about the current macroeconomic situation and make them capable of setting it into context for anomaly returns. Furthermore, we incorporate the 49 Fama and French (1997) industry classification indicators publicly available on the Kenneth R. French data library[3] into the feature set. We use one-hot encoding to ensure that our studied models do not suffer multi-collinearity issues, leading to 48 additional features. Next-month returns, market capitalization, and further metadata are obtained from the Center for Research in Security Prices (CRSP). This leads to $320 + 8 + 48 = 376$ features per observation for our models.

Figure 1 reports the number of input features we use over time. For the first year of our asset pricing tests, 2005, our models only use the 137 anomalies from Chen and Zimmermann (2022) with publication dates up to 2004, resulting in $137 + 48 + 8 = 193$ features. The Chen and Zimmermann (2022) database ends in 2016, at which point our list of features reaches its maximum of 376.

[Figure 1 about here.]

[2]For a detailed description of the methodology and anomaly composition, as well as the corresponding code and dataset, see their website: https://www.openassetpricing.com

[3]For more information, see https://mba.tuck.dartmouth.edu

To train our machine learning models and tune hyperparameters without any data snooping or look-ahead bias, we follow standard machine learning practice and split the overall dataset into three subsets: a training-, validation-, and testing-set. To allow our model to learn from new information and adapt to the non-stationarity characteristics of stock return time series, we follow the latest literature using an expanding window approach for the training data (Gu et al., 2020; Azevedo and Hoegner, 2023). We re-train the models annually to include new data while keeping a fixed-length moving validation set of six years and a one-year out-of-sample test set. For example, for the out-of-sample year 2005, we use all available stock-month observations from Mar 1957 to Dec 1998 to train the model. We then tune hyperparameters based on the six-year validation set from Jan 1999 to Dec 2004 to ensure the temporal ordering of the observations in the training process. The out-of-sample test set, which we use to evaluate our models regarding machine learning metrics and long-short portfolio performance, contains predictions for January 2005 to December 2021 (i.e., each new year, we move this approach one year ahead, extending the training set).

The portfolio construction process follows common practices in anomaly research. We construct decile portfolios based on the models' next-month stock return predictions, calculating the long-short gross excess return on a monthly rebalancing frequency. Transaction costs are estimated using the composite high-frequency Chen and Velikov (2023) effective bid-ask spread estimator and applied in the calculation of the net excess return following Detzel et al. (2023). To ensure an openly accessible and thus replicable construction and testing protocol, we use the methodology and library of Novy-Marx and Velikov (2023). Further construction details can be found in their paper and on their website.[4]

2.2. Applied machine learning algorithms and evaluation methodology

Our choice of machine learning models is motivated by prior literature, which shows that neural networks outperform traditional linear regressions as well as penalized ones such as elastic nets (Gu et al., 2020; Chen et al., 2023; Azevedo and Hoegner, 2023; Azevedo et al., 2023; Avramov et al., 2022). We follow Gu et al. (2020), and include the non-constrained OLS-HUBER, a regularized linear model using ENET, and four feedforward neural networks (FFNNs) with hidden layers ranging from two to five.[5] We extend this core model set with two (one- and two-hidden layers, respectively) recurrent neural networks with long short-term

[4]Documentation available at `http://assayinganomalies.com/`.

[5]A more detailed explanation of the models can be found in the internet appendix of Gu et al. (2020).

memory (LSTMs), which are designed to capture long-term dependencies. Finally, we create an (ENSEMBLE) model by taking the average of all deep-learning models (FFNNs and LSTMs).

Following Chen et al. (2023), we use hyperparameter tuning to find the best set of parameters in each model. To optimize the tuning parameters for each model, we initially implemented a random search strategy. For efficiency and computational practicality, we selected a representative subset of the data, comprising 20% of the total dataset, to fine-tune these parameters. Once established, these parameters were consistently applied throughout the expanding window estimation process. However, this approach was not feasible for neural network models due to their extensive computational demands and the wide variability in their parameter ranges. For the neural network approaches, we apply a fixed set of parameters. Like Gu et al. (2020), we use the geometric rule to derive the specific neuron configuration for our 2-, 3-, 4-, and 5-hidden-layer FFNNs, and similarly for our 1- and 2-hidden layer LSTMs. All neural networks use an ADAM optimizer with a learning rate of 0.01, the mean squared error (MSE) validation metric, and 200 epochs with a batch size of 10,000 observations. We apply dynamic learning rate shrinkage by factor 5 when our validation metrics have not improved for ten epochs of training. We also regularize the models through an early stopping callback, which stops training when validation metrics have not improved for 20 epochs.

We evaluate the actual out-of-sample trading strategy performance using common portfolio metrics, mainly gross and net excess return of the long-short portfolios, their statistical significance, Sharpe ratio, information ratio, turnover ratio, and transaction costs, as well as the R2. Also, we test the model against the most comprehensive factor model to date, the Fama and French (2018) six-factor model (FF6). To the best of our knowledge, we are the first to estimate the Novy-Marx and Velikov (2016) generalized alphas to machine-learning-based strategies in order to evaluate the ability of these strategies to expand the net-of-costs mean-variance efficient frontier based on the six factors alone.

2.3. Applied turnover and cost mitigation techniques

Novy-Marx and Velikov (2016) and Novy-Marx and Velikov (2019) study the impact of cost-mitigation techniques on the profitability of anomaly trading strategies after accounting for trading costs. In addition to their proposed three techniques, we add further variations and filters to test the effect of cost mitigation on the net performance of machine-learning-based strategies.

One major driver to reduce transaction costs is to reduce turnover rate. Most straightfor-

wardly, this can be achieved using an increased holding period/reduced rebalancing frequency for the portfolio construction. Since we train our models based on monthly predictions, we extend the holding period mitigation to 2-, 3-, or 4-months (H2, H3, H4). Additionally, we create quintile- instead of decile portfolios (QUINTILE), hypothesizing that this could reduce turnover while keeping a significant signal-to-noise ratio. As a more complex variation of adapted overall holding period and number of portfolios, we apply the BHS technique outlined by Novy-Marx and Velikov (2016) to enter a position for the top-/bottom-decile but exit a position only if they fall out of the top-/bottom-quintile.

Furthermore, we test multiple stock universe filters and weightings. We follow Fama-French in creating a high-market cap filter that excludes the bottom 20% of stocks by market capitalization for our tradable stock universe (HMCAP20). We hypothesize that those high market capitalization stocks are more liquid, i.e., they should face lower transaction costs. In addition, we follow the approach of Novy-Marx and Velikov (2016) in directly filtering out stocks with high previous transaction costs, i.e., using only stocks in the bottom 70%-percentile transaction costs (LTC 70). As an alternative way to incorporate transaction costs into the portfolio construction process, we weight predictions to 25% with the transaction cost percentile (TCWEIGHTED75).[6] Finally, we create two combination strategies that use (a) both H2 and BHS (COMBO1) and (b) H2, BHS, and TCWEIGHTED75 (COMBO2).

3. Machine learning performance without transaction costs

Table 1 reports the returns to long-short portfolios sorted on the machine learning signals. The portfolios are constructed as value-weighted decile-sorted portfolios using NYSE breakpoints in a 1-month holding period/rebalancing frequency and no trading cost adjustment. The average monthly excess returns to these strategies range from an insignificant 48 basis points per month (t-statistic 1.34) for OLS-HUBER to an impressive 1.63% per month (t-statistic of 4.57) for LSTM1. Five of the nine strategies' average returns have t-statistics above 3, the threshold suggested by Harvey et al. (2016). The out-of-sample R^2's vary from 0.05% for OLS-HUBER to 0.76% for LSTM1. The feedforward neural network-based models rank in between, with R^2's ranging from 0.19% to 0.36%.

[6] Example: a prediction of 1% excess return but with a transaction cost at the 80% percentile will result in $75\% * 1.0\% + 25\% * (1 - 80\%) = 0.8\%$, while the same return prediction with lower transaction cost at the 20% percentile will result in a $75\% * 1.0\% + 25\% * (1 - 20\%) = 0.95\%$ signal for the models.

The Fama and French (2018) six-factor model alphas exhibit similar patterns, with the LSTM1 model-based trading strategy having the highest alpha of 1.40% per month. The Sharpe ratios for these strategies are impressive even when benchmarked against the post-2005 Sharpe ratio of 0.69, where five out of nine machine learning strategies come out on top. Moreover, given that these are hedge strategies that are, for the most market neutral, their information ratios, even to the Fama and French (2018) six-factor model, are sizable, ranging from 0.52-1.01.

[Table 1 about here.]

4. Machine learning performance accounting for transaction costs

Thus far, we estimate the long-short returns of machine learning models without accounting for transaction costs. Avramov et al. (2022) argue that when transaction costs are introduced, most models do not show statistically significant returns because their performance largely depends on small, illiquid, and expensive stocks. We test this explicitly and observe a reduction in average monthly returns and overall financial performance after accounting for the Chen and Velikov (2023) effective bid-ask spread estimate. Figure 2 shows the percentage drop in average returns for the nine machine-learning strategies from introducing trading costs. We can observe that the reduction in performance ranges from 13% to 40%.

[Figure 2 about here.]

Table 2 reports the performance metrics for our nine machine-learning-based strategies after accounting for trading costs. We can observe that costs have a significant impact on performance, reducing average returns across the board. The trading costs for the strategies, reported in the last column, range between 19 and 26 basis points per month, rendering the average returns to FFNN2 and FFNN5 insignificant. The two-sided portfolio turnover, reported in the second-to-last column, varies between 120 and 140% per month, classifying the machine-learning strategies as high-turnover anomalies based on the Novy-Marx and Velikov (2016) taxonomy.

Nevertheless, we still observe economically and statistically significant average returns and generalized six-factor model alphas for the majority of the models. Both LSTM strategies continue to exhibit t-statistics on their average returns and alphas in excess of three. The feedforward neural network models perform slightly worse, but their average returns and alphas are still sizable. The clear winner is the LSTM1 strategy, which earns an impressive 1.42% per month (t-stat of 3.99) and a generalized FF 6 alpha of 1.20% (t-stat of 3.46).

[Table 2 about here.]

Figure 3 shows the cumulative *net* performance of the machine learning strategies. It reveals that the outperformance of the LSTM and ENSEMBLE strategies is largely due to their impressive performance during the Great Recession. All other machine learning strategies exhibited a severe drawdown in 2009, which the LSTM strategies completely and the ENSEMBLE strategy to some extent are able to avoid. The LSTM1 strategy also performed better in the aftermath of the COVID-19 pandemic at the end of 2020, while all other strategies suffered losses.

[Figure 3 about here.]

5. Applying turnover and transaction cost mitigation strategies

The machine-learning-based strategies examined thus far were designed without regard for trading costs. Recently, Blitz et al. (2023) show that training the models in a longer time horizon can lead to higher returns. Furthermore, Avramov et al. (2022) show that excluding firms with economic constraints can reduce the significance of machine learning models. In this section, we investigate different mitigation techniques and their impact on the net performance of machine learning-based strategies.

[Figure 4 about here.]

Table 3 and Figure 4 show the impact of the previously outlined mitigation approaches on our four classes of model architectures, namely linear models, FFNNs, LSTMs, and the ensemble model. We show the impact of absolute differences in the net excess return portfolio metrics and generalized FF6 alpha and relative changes in turnover and transaction costs. As the results show, most of the cost-mitigation techniques significantly reduce turnover and, as a result, transaction costs.

[Table 3 about here.]

This decrease in transaction costs, however, is only beneficial if it is not accompanied by a larger reduction in gross returns. As we can observe in Table 3, the average change net excess returns across the nine machine learning models is negative for all but one mitigation technique. This implies that the drop in the gross average returns due to the mitigation techniques more

than compensates for the reduced trading costs. This is likely because our testing sample period, which consists of the last two decades, is marked by higher liquidity and significantly lower trading costs post-decimalization (Chordia et al., 2014; Chen and Velikov, 2023). The only technique that seems to marginally improve the net average returns across the nine machine learning strategies is the two-month holding period. Not surprisingly, the stock universe filters have a smaller impact on turnover but a similar impact on transaction costs, as they aim to reduce the weight of high-cost stocks. However, the change in net excess returns for these methods is similarly negative.

6. Conclusion

Our study assesses the expected returns of machine learning-based trading strategies. We do so by accounting for transaction costs, post-publication decay, and the recent era of high liquidity.

Due to the improved liquidity in this more recent period, the trading costs associated with these strategies are on the order of 20-25 basis points per month. As a result, contrary to the insights of Avramov et al. (2022), most machine learning combination strategies continue to deliver significant returns net of costs despite one-sided turnover ranging between 60% and 70%. While the Chen and Velikov (2023) effective spread estimator renders the average returns to a few of the strategies insignificant (OLS-HUBER, FFNN2, FFN5), the rest of the signals continue to generate sizable and statistically significant average returns ranging from 0.64% per month (t-statistic of 2.43) for ENET to 1.42% per month (t-statistic of 3.99) for LSTM1. We also find that cost-mitigation techniques, while significantly reducing turnover and trading costs, do not lead to improvements in net anomaly performance.

Our findings have significant implications for academic research and practical investment strategy design. As we venture deeper into the age of machine learning and AI, studies such as ours will be crucial in navigating the complex labyrinth of returns, ultimately guiding us toward a better understanding of financial markets.

References

Avramov, D., Cheng, S., and Metzker, L. (2022). Machine learning vs. economic restrictions: Evidence from stock return predictability. *Management Science*.

Azevedo, V. and Hoegner, C. (2023). Enhancing stock market anomalies with machine learning. *Review of Quantitative Finance and Accounting*, 60(1):195–230.

Azevedo, V., Kaiser, S., and Müller, S. (2023). Stock market anomalies and machine learning across the globe. *Journal of Asset Management*, Forthcoming:1–23.

Blitz, D., Hanauer, M. X., Hoogteijling, T., and Howard, C. (2023). The term structure of machine learning alpha. *SSRN Electronic Journal*, pages 1–40.

Brandt, M. W., Santa-Clara, P., and Valkanov, R. (2009). Parametric portfolio policies: Exploiting characteristics in the cross-section of equity returns. *The Review of Financial Studies*, 22(9):3411–3447.

Brogaard, J. and Zareei, A. (2023). Machine learning and the stock market. *Journal of Financial and Quantitative Analysis*, 58(4):1431–1472.

Cakici, N., Fieberg, C., Metko, D., and Zaremba, A. (2023). Machine learning goes global: Cross-sectional return predictability in international stock markets. *Journal of Economic Dynamics and Control*, 155:104725.

Chen, A. Y. and Velikov, M. (2023). Zeroing in on the expected returns of anomalies. *Journal of Financial and Quantitative Analysis*, 58(3):968–1004.

Chen, A. Y. and Zimmermann, T. (2022). Open source cross-sectional asset pricing. *Critical Review of Finance*, 27(2):207–264.

Chen, L., Pelger, M., and Zhu, J. (2023). Deep learning in asset pricing. *Management Science*, (Forthcoming).

Chordia, T., Subrahmanyam, A., and Tong, Q. (2014). Have capital market anomalies attenuated in the recent era of high liquidity and trading activity? *Journal of Accounting and Economics*, 58(1):41–58.

Cong, L., Tang, K., Wang, J., and Zhang, Y. (2020). AlphaPortfolio for Investment and Economically Interpretable AI. *SSRN Electronic Journal*.

Detzel, A., Novy-Marx, R., and Velikov, M. (2023). Model comparison with transaction costs. *The Journal of Finance*, 78(3):1743–1775.

Evgeniou, T., Guecioueur, A., and Prieto, R. (2023). Uncovering sparsity and heterogeneity in firm-level return predictability using machine learning. *Journal of Financial and Quantitative Analysis*, 58(8):3384–3419.

Fama, E. F. and French, K. R. (1997). Industry costs of equity. *Journal of Financial Economics*, 43(2):153–193.

Fama, E. F. and French, K. R. (2018). Choosing factors. *Journal of Financial Economics*, 128(2):234–252.

Feng, G., He, J., Polson, N. G., and Xu, J. (2023). Deep learning in characteristics-sorted factor models. In *Journal of Financial and Quantitative Analysis*. Cambridge University Press.

Freyberger, J., Neuhierl, A., and Weber, M. (2020). Dissecting characteristics nonparametrically. *The Review of Financial Studies*, 33(5):2326–2377.

Gu, S., Kelly, B., and Xiu, D. (2020). Empirical asset pricing via machine learning. *The Review of Financial Studies*, 33(5):2223–2273.

Hanauer, M. X. and Kalsbach, T. (2023). Machine learning and the cross-section of emerging market stock returns. *Emerging Markets Review*, 55:101022.

Hansen, J. H. and Siggaard, M. V. (2023). Double machine learning: Explaining the post-earnings announcement drift. *Journal of Financial and Quantitative Analysis*, Forthcoming.

Harvey, C. R., Liu, Y., and Zhu, H. (2016). ... and the cross-section of expected returns. *Review of Financial Studies*, 29(1):5–68.

Jensen, T. I., Kelly, B. T., Malamud, S., and Pedersen, L. H. (2022). Machine learning and the implementable efficient frontier. *SSRN Electronic Journal*, pages 1–67.

Kelly, B., Malamud, S., and Zhou, K. (2023). The virtue of complexity in return prediction. *The Journal of Finance*.

Kelly, B. T., Pruitt, S., and Su, Y. (2019). Characteristics are covariances: A unified model of risk and return. *Journal of Financial Economics*, 134(3):501–524.

Leippold, M., Wang, Q., and Zhou, W. (2022). Machine learning in the Chinese stock market. *Journal of Financial Economics*, 145(2):64–82.

Loughran, T. and McDonald, B. (2023). Measuring firm complexity. *Journal of Financial and Quantitative Analysis*.

Mclean, R. D. and Pontiff, J. (2016). Does academic research destroy stock return predictability? *Journal of Finance*, 71(1):5–32.

Meursault, V., Liang, P. J., Routledge, B. R., and Scanlon, M. M. (2023). PEAD.txt: Post-earnings-announcement drift using text. In *Journal of Financial and Quantitative Analysis*, volume 58, pages 2299–2326. Cambridge University Press.

Novy-Marx, R. and Velikov, M. (2016). A taxonomy of anomalies and their trading costs. *Review of Financial Studies*, 29(1):104–147.

Novy-Marx, R. and Velikov, M. (2019). Comparing cost-mitigation techniques. *Financial Analysts Journal*, 75(1):85–102.

Novy-Marx, R. and Velikov, M. (2023). Assaying anomalies. *SSRN Electronic Journal*, pages 1–36.

Simon, F., Weibels, S., and Zimmermann, T. (2022). Deep parametric portfolio policies. *SSRN Electronic Journal*.

Singh, D. and Singh, B. (2020). Investigating the impact of data normalization on classification performance. *Applied Soft Computing*, 97:105524.

Welch, I. and Goyal, A. (2008). A comprehensive look at the empirical performance of equity premium Prediction. *Review of Financial Studies*, 21(4):1455–1508.

Table 1: Out-of-sample gross performance performance of machine-learning anomaly strategies

Model architecture	Gross monthly excess return in % [t]	Alpha FF6 in % [t]	Sharpe ratio	Information ratio	Avg. # of long stocks	Avg. # of short stocks	R2 (in %)
OLS-HUBER	0.48	0.52	0.32	0.52	934	254	0.0525
	[1.34]	[2.01]					
ENET	0.83	0.71	0.77	0.72	437	570	0.0998
	[3.16]	[2.79]					
FFNN2	0.82	0.76	0.61	0.65	540	709	0.1976
	[2.50]	[2.54]					
FFNN3	1.01	0.88	0.73	0.76	537	739	0.3377
	[3.02]	[2.98]					
FFNN4	0.92	0.80	0.65	0.62	558	721	0.3550
	[2.67]	[2.42]					
FFNN5	0.66	0.60	0.47	0.48	536	749	0.2751
	[1.95]	[1.86]					
LSTM1	1.63	1.40	1.11	1.01	530	570	0.7593
	[4.57]	[3.94]					
LSTM2	1.27	1.12	0.96	0.91	512	576	0.5549
	[3.94]	[3.56]					
ENSEMBLE	1.07	1.04	0.80	0.86	499	662	0.4926
	[3.30]	[3.37]					

The table shows the key portfolio metrics of our models in a no-transaction-cost environment during our out-of-sample period from January 2005 to December 2021. All portfolios are constructed as value-weighted long-short decile portfolios with NYSE breakpoints and a 1-month holding period/rebalancing frequency.

Table 2: Out-of-sample net performance performance of machine-learning anomaly strategies

Model architecture	Net monthly excess return in % [t]	Generalized alpha FF6 [t]	Two-sided turnover in %	Transaction costs in %
OLS-HUBER	0.29	0.38	122.11	0.20
	[0.79]	[1.47]		
ENET	0.64	0.55	139.76	0.19
	[2.43]	[2.20]		
FFNN2	0.57	0.57	129.14	0.24
	[1.75]	[1.90]		
FFNN3	0.75	0.68	128.40	0.26
	[2.25]	[2.25]		
FFNN4	0.66	0.58	127.27	0.26
	[1.92]	[1.78]		
FFNN5	0.40	0.38	130.25	0.26
	[1.17]	[1.20]		
LSTM1	1.42	1.20	129.56	0.22
	[3.99]	[3.46]		
LSTM2	1.06	0.95	129.07	0.21
	[3.30]	[3.07]		
ENSEMBLE	0.83	0.84	129.91	0.24
	[2.56]	[2.74]		

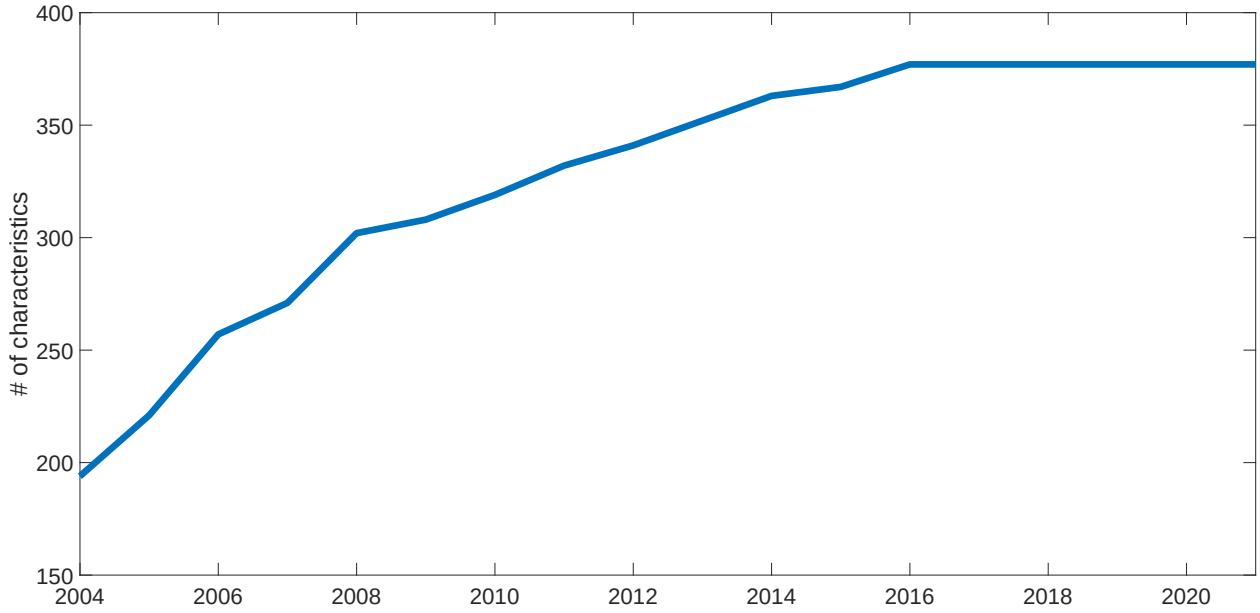
The table shows the key portfolio metrics of our tested models, similar to the previous table, but in an environment with transaction costs. The transaction costs are estimated using the Chen and Velikov (2023) high-frequency combination effective bid-ask spread estimator. All portfolios are constructed as value-weighted long-short decile portfolios with NYSE breakpoints and a 1-month holding period/rebalancing frequency.

Table 3: Average mitigation technique effect across different model architectures

Mitigation technique	relative change in %		absolute average Δ to baseline			
	Two-sided turnover	Trans-action costs	Net excess return in %	[t-stat]	Generalized FF6 alpha in %	[t-stat]
H2	-46.10	-46.78	0.03	0.05	0.24	0.69
H3	-62.57	-63.26	-0.18	-0.54	-0.04	-0.17
H4	-71.28	-71.93	-0.15	-0.29	0.03	0.40
BHS	-33.46	-36.03	-0.08	0.06	-0.01	0.32
QUINTILE	-12.17	-34.16	-0.25	-0.04	-0.23	-0.12
HMCAP20	-1.78	-29.40	-0.10	-0.31	-0.08	-0.20
LTC70	3.04	-53.67	-0.12	-0.48	-0.10	-0.46
TCWEIGHTED75	-1.21	-17.51	-0.04	-0.06	-0.03	-0.01
COMBO1	-72.21	-73.20	-0.26	-0.62	-0.10	-0.11
COMBO2	-72.41	-77.28	-0.32	-0.81	-0.17	-0.30

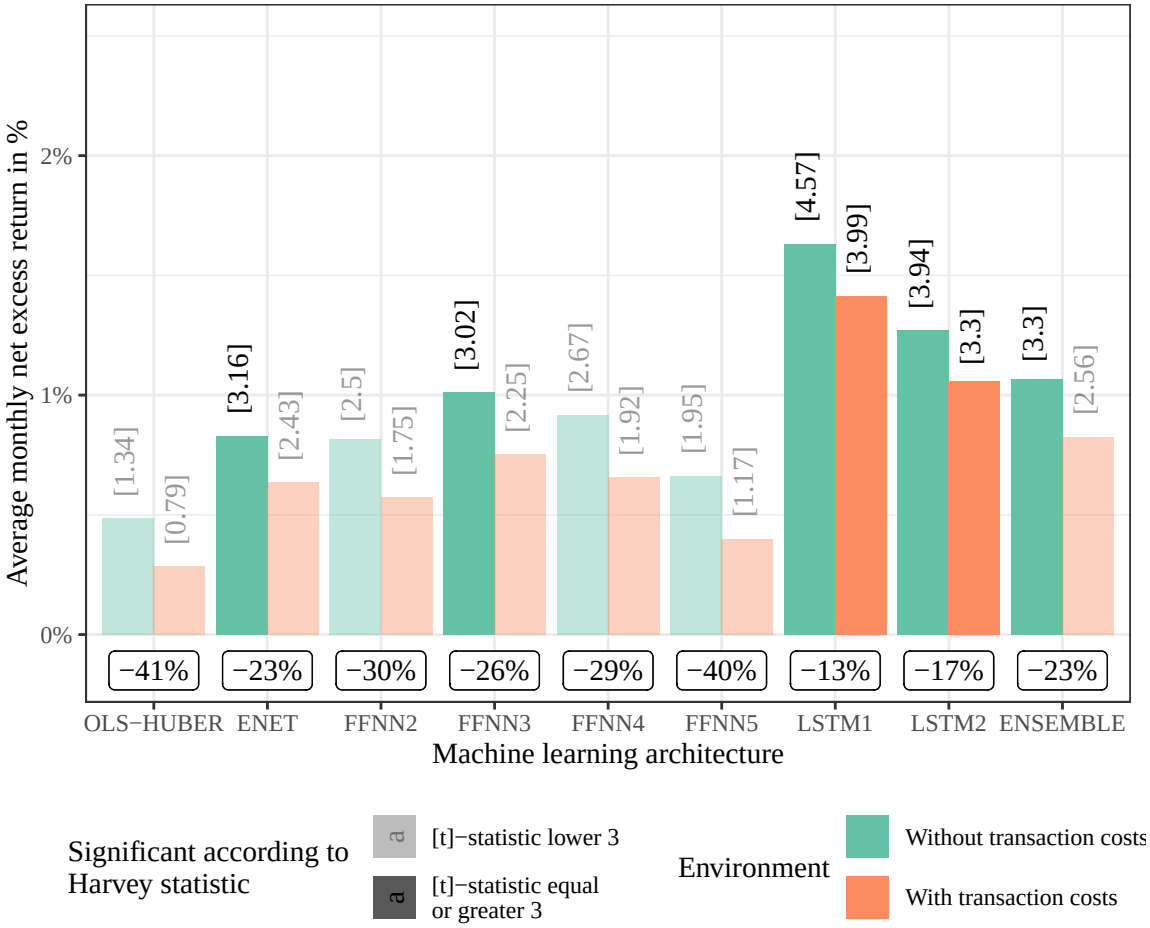
This table shows the average effect on the baseline model of applying each mitigation technique in our out-of-sample period from January 2005 to December 2021. We apply different techniques to our sample: Extended holding period/reduced rebalancing frequency (H2, H3, and H4 for 2-, 3-, and 4-month periods), a BHS, quintile instead of decile portfolios (QUINTILE), a high market cap filter (HMCAP20), a low transaction cost filter (LTC70), and a weighting of next month's predicted returns by estimated transaction costs (TCWEIGHTED75). We report the impact in % change of the respective variable compared to the baseline model without mitigations. Using the improvement in the generalized alpha notation (and thus the minimum variance portfolio under transaction costs) as the key metric, we see that of all the methods, the reduced rebalancing, the BHS, and the signal weighting with transaction costs provide the most meaningful benefits, as well as combinations of these (COMBO1 and COMBO2).

Figure 1: Number of characteristics used over time



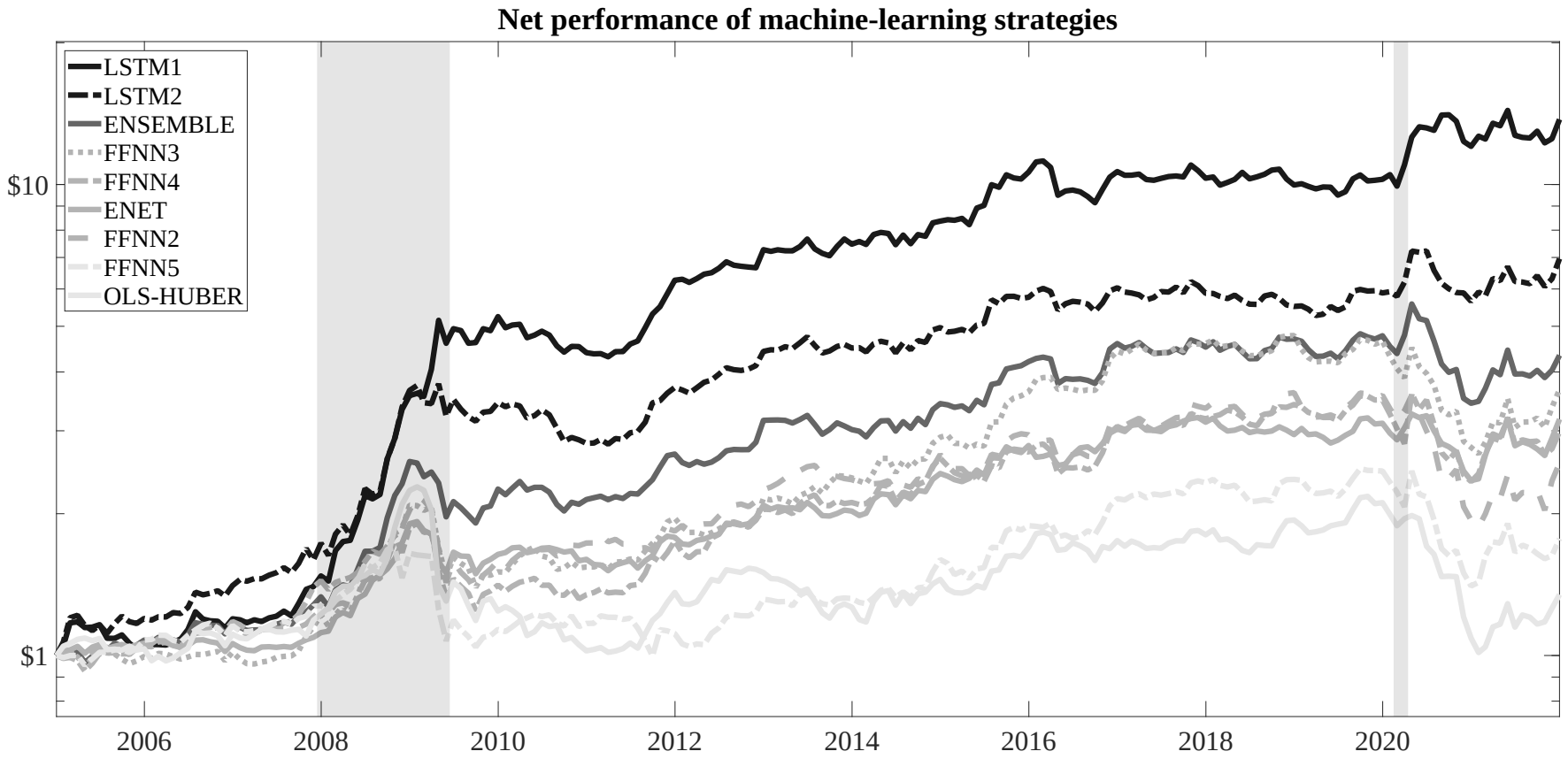
The figure plots the number of characteristics used in the construction of our machine-learning strategies over time.

Figure 2: Baseline performance for model returns with and without transaction costs



The figure describes the monthly performance of our baseline models gross and net of trading costs in the out-of-sample period from January 2005 to December 2021. The transaction costs are estimated using the Chen and Velikov (2023) high-frequency combination effective bid-ask spread estimator. The bars show the average monthly gross and net excess returns of the value-weighted long-short decile portfolios based on NYSE breakpoints. The *t*-statistic is in brackets. The labels below the columns show the respective relative drop in return in % due to the introduction of transaction costs.

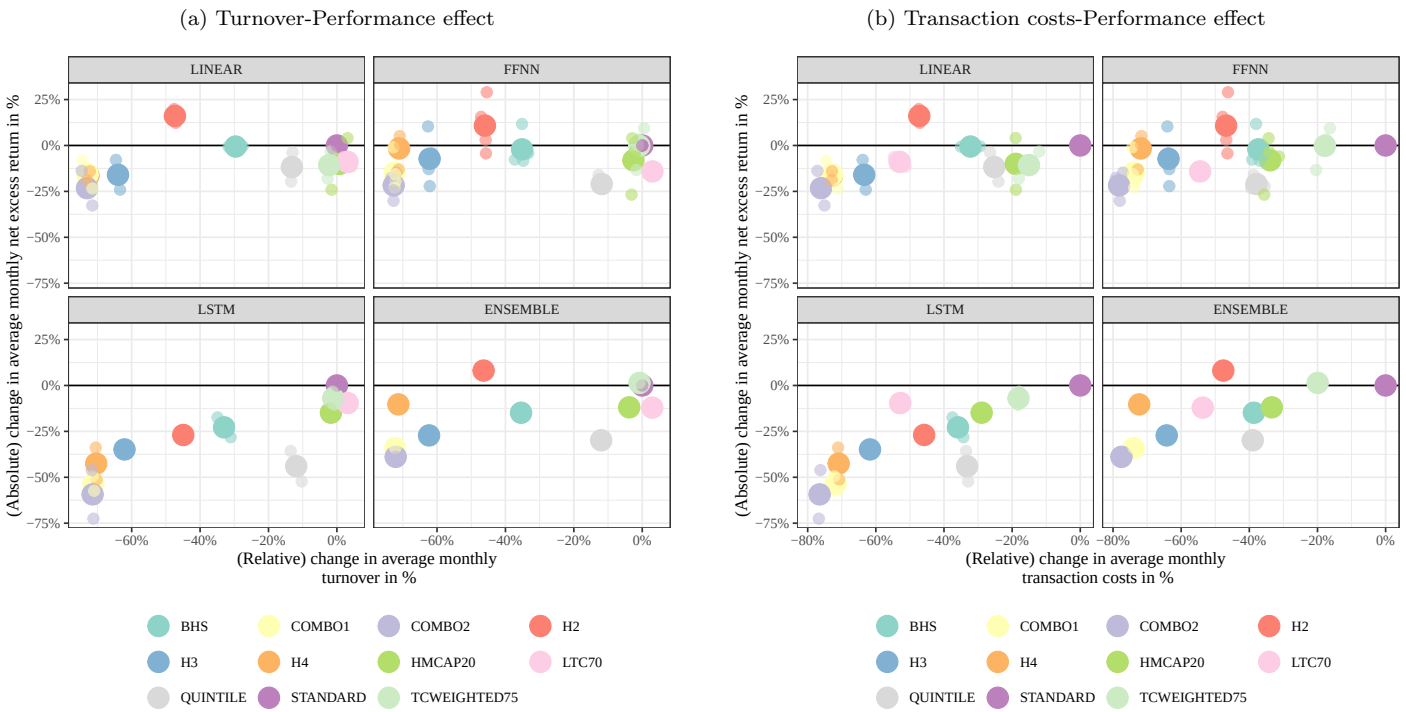
Figure 3: Net performance of machine-learning strategies



The figure describes the monthly performance of machine-learning anomaly strategies net of trading costs in the out-of-sample period from January 2005 to December 2021. The transaction costs are estimated using the Chen and Velikov (2023) high-frequency combination effective bid-ask spread estimator. The bars show the average monthly gross and net excess returns of the value-weighted long-short decile portfolios based on NYSE breakpoints.

.

Figure 4: Net return impact of different turnover and cost mitigation techniques on portfolio performance



The table illustrates the turnover and transaction cost relations to the respective change in net excess return in the out-of-sample period from Jan 2005 to Dec 2021. All changes are in % compared to the baseline model without mitigation techniques in a transaction cost environment.

Appendix A.

Elastic Net Regression

The Elastic Net is a linear regression model that combines the L1 and L2 regularization of the Lasso and Ridge regression methods. This approach is beneficial when dealing with highly correlated independent variables.

Mathematical Formulation

The objective function of the Elastic Net is:

$$\text{Minimize} \left(\sum_{i=1}^{n} (y_i - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \left(\alpha \sum_{j=1}^{p} |\beta_j| + \frac{1-\alpha}{2} \sum_{j=1}^{p} \beta_j^2 \right) \right) \tag{A.1}$$

where y_i represents the response variable, x_{ij} are the predictors, β_j are the coefficients, λ is the regularization parameter, and α is the mixing parameter between Lasso and Ridge penalties.

Overview of Feedforward Neural Networks

Architecture of FFNN with Two Hidden Layers

In this study, FFNNs with two to five hidden layers are used. The architecture of an FFNN with two hidden layers is as follows:

- Input Layer: Receives input features (e.g., stock market anomalies, macroeconomic predictors, and industry dummies).
- First Hidden Layer: Applies a non-linear transformation to the inputs using the ReLU activation function.
- Second Hidden Layer: Further processes the data from the first hidden layer, also using ReLU.
- Output Layer: Produces the final output (e.g., asset return predictions), typically using a linear activation function.

Mathematical Formulation with ReLU Activation

Consider an FFNN with two hidden layers. The mathematical formulation using the ReLU activation function is:

First Hidden Layer:

$$H_1 = ReLU(W_1 \cdot x + b_1) \tag{A.2}$$

Second Hidden Layer:

$$H_2 = ReLU(W_2 \cdot H_1 + b_2) \tag{A.3}$$

Output Layer:

$$\hat{y} = f(W_3 \cdot H_2 + b_3) \tag{A.4}$$

where:

- x is the input vector.
- W_1, W_2, W_3 are the weight matrices for the first hidden layer, second hidden layer, and output layer, respectively.
- b_1, b_2, b_3 are bias vectors for each corresponding layer.
- $ReLU(\cdot)$ is the Rectified Linear Unit activation function, defined as $ReLU(z) = max(0, z)$.
- $f(\cdot)$ is typically a linear activation function for regression tasks.

ReLU Activation Function

The Rectified Linear Unit (ReLU) activation function is defined as:

$$ReLU(z) = max(0, z) \tag{A.5}$$

ReLU introduces non-linearity in the model, enabling the network to learn complex patterns. It is computationally efficient and helps mitigate the vanishing gradient problem.

Backpropagation and Initialization of Weights and Biases

Backpropagation is a fundamental algorithm in supervised learning used for training Feed-forward Neural Networks (FFNNs). The core idea of backpropagation is to adjust the network's weights and biases to minimize the error between the predicted and actual outputs. Mathematically, this involves computing the gradient of the loss function with respect to each weight and bias in the network.

Let's consider a network with L layers, each with weights $W^{(l)}$ and biases $b^{(l)}$ for layer l. The process of backpropagation can be described as follows:

1. Forward Pass: Compute the output of the network for a given input. For layer l, the output $H^{(l)}$ is given by:

$$H^{(l)} = \sigma(W^{(l)} \cdot H^{(l-1)} + b^{(l)}) \tag{A.6}$$

where σ is the activation function and $H^{(0)}$ is the input to the network.

2. Compute Loss: Calculate the loss (error) $\mathcal{L}$ using a loss function which compares the predicted output $\hat{y}$ and the actual output y. For regression tasks, a common loss function is Mean Squared Error (MSE):

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 \tag{A.7}$$

3. Backward Pass: Compute the gradient of the loss function with respect to each weight and bias. For a weight $W_{ij}^{(l)}$ in layer l, the gradient is:

$$\frac{\partial \mathcal{L}}{\partial W_{ij}^{(l)}} = \frac{\partial \mathcal{L}}{\partial H_i^{(l)}} \cdot \frac{\partial H_i^{(l)}}{\partial W_{ij}^{(l)}} \tag{A.8}$$

4. Update Weights and Biases: Adjust the weights and biases in the direction that minimizes the loss. This is typically done using gradient descent:

$$W_{ij}^{(l)} = W_{ij}^{(l)} - \eta \frac{\partial \mathcal{L}}{\partial W_{ij}^{(l)}} \tag{A.9}$$

$$b_i^{(l)} = b_i^{(l)} - \eta \frac{\partial \mathcal{L}}{\partial b_i^{(l)}} \tag{A.10}$$

where η is the learning rate.

A common approach to initializing weights and biases is to use small random values for weights and zeros for biases. This breaks the symmetry in the learning process and allows the network to learn effectively. For example, weights can be initialized from a normal distribution with a mean of 0 and a small standard deviation, e.g., 0.01.

Long Short-Term Memory (LSTM) Networks

LSTMs are a special kind of Recurrent Neural Network (RNN) capable of learning long-term dependencies in sequential data, which is particularly useful in financial time series forecasting.

Core Components

An LSTM unit includes several gates to control the flow of information:

- Forget Gate: Decides what information to discard from the cell state.
- Input Gate: Updates the cell state with new information.

- Output Gate: Determines the next hidden state.

Mathematical Formulation of LSTM Cell

The operations inside an LSTM cell can be formulated as follows:

$$\text{Forget Gate:} \quad f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{A.11}$$

$$\text{Input Gate:} \quad i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{A.12}$$

$$\text{Cell State Update:} \quad \tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{A.13}$$

$$\text{Final Cell State:} \quad C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{A.14}$$

$$\text{Output Gate:} \quad o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{A.15}$$

$$\text{Output:} \quad h_t = o_t * \tanh(C_t) \tag{A.16}$$

Where f_t, i_t, and o_t are the activations of the forget, input, and output gates, respectively; C_t is the cell state; h_t is the hidden state; W and b are the weights and biases for each gate; and x_t is the input at time t.