Design choices, machine learning, and the cross-section of stock returns

Latest version here

Minghui Chen Matthias X. Hanauer Tobias Kalsbach

December 2025

Abstract

We fit over one thousand machine learning models for predicting stock returns, systematically varying design choices across algorithm, target variable, feature selection, and training methodology. Our findings demonstrate that the nonstandard error in portfolio returns arising from these design choices exceeds the standard error by 59%. Furthermore, we observe a substantial variation in model performance, with monthly mean top-minus-bottom returns ranging from 0.13% to 1.98%. These findings underscore the critical impact of design choices on machine learning predictions, and we offer recommendations for model design. Finally, we identify the conditions under which non-linear models outperform linear models.

Keywords: Machine learning, stock returns, design choices, nonstandard errors, portfolio performance

JEL Classifications: C52, C55, G11, G17

All authors are at TUM School of Management, Technical University of Munich. Minghui Chen can be contacted at minghui.chen@tum.de. Matthias X. Hanauer is also at Robeco and can be contacted at matthias.hanauer@tum.de. Tobias Kalsbach is also at PwC Strategy& and can be contacted at tobias.kalsbach@tum.de. The views expressed in this paper are solely those of the authors and not necessarily shared by their respective institutions. We thank Tobias Hoogteijling, Christoph Kaserer, Harald Lohre, Albert Menkveld, Markus Pelger, Amar Soebhag, Marc Stam, Florian Weigert, Adam Zaremba, Terri van der Zwan, and participants at the 2025 Financial Management Association (FMA) European Conference, the 2025 Conference on Asset Pricing and Machine Learning hosted by the University of Gothenburg, the doctoral seminar at the 30th Annual Meeting of the German Finance Association (DGF), and seminars at Robeco and the Technical University of Munich for valuable discussions and feedback.

1 Introduction

Machine learning (ML) models for predicting stock returns have gained substantial popularity in both academic studies and industry practice in recent years. For example, Freyberger et al. (2020), Gu et al. (2020), and Chen et al. (2024) study return predictability for the United States, while Rasekhschaffe and Jones (2019) and Tobek and Hronec (2021) focus on developed markets, and Hanauer and Kalsbach (2023) examine emerging markets. Overall, these studies show that models allowing for non-linearities and interactions—such as neural networks and tree-based models—yield superior out-of-sample (OOS) returns compared to linear models.

However, as a still developing field, existing studies vary considerably across several key design choices. For instance, when predicting future stock returns, Gu et al. (2020) and Freyberger et al. (2020) employ the excess return over the risk-free rate as the target variable, whereas Tobek and Hronec (2021) and Hanauer and Kalsbach (2023) use the stock outperformance relative to the market. While these studies use a continuous target variable, Rasekhschaffe and Jones (2019) advocate for predicting categories, such as distinguishing outperformers from underperformers. Additionally, Gu et al. (2020), Tobek and Hronec (2021), and Hanauer and Kalsbach (2023) implement an expanding window to train their models, while Freyberger et al. (2020) and Rasekhschaffe and Jones (2019) use rolling windows. This variety in design choices underscores the divergence of the research framework in machine learning for stock return predictions, making it challenging to compare performance across studies.

In this study, we analyze the variation in seven key design choices in machine learning studies and document the resulting differences. These design choices include (1) algorithm, (2) target variable, (3) target transformation, (4) post-publication treatment, (5) feature selection, (6) training window, and (7) training sample. To assess the importance of these choices, we examine all possible combinations, resulting in a total of 1,056 machine learning models. Each model is trained on a common set of features for the U.S. stock market, and we evaluate their out-of-sample performance using top-minus-bottom decile portfolios. Thereby, we document the economic relevance of design choices and provide recommendations for model design.

The main findings of our study can be summarized as follows: First, we document substantial variation in top-minus-bottom decile returns across different machine learning models. For example, monthly mean returns range from 0.13% to 1.98%, with corresponding annualized Sharpe ratios ranging from 0.08 to 1.82. Thereby, we find that among the investigated design choices, the post-publication treatment, training window, target transformation, algorithm, and target variable are the most influential. By contrast, feature selection and training sample have little impact.

Second, we find that the variation in returns stemming from these design choices, i.e., the nonstandard error, is approximately 1.59 times larger than the standard error from the statistical bootstrapping process. This ratio is comparable to or even exceeds findings in related studies on nonstandard errors, such as 1.60 in Menkveld et al. (2024), 1.06 in Soebhag et al. (2024), 1.10 in Walter et al. (2024), and 1.55 in Fieberg et al. (2024). Moreover, when we use alternative portfolio construction methodologies or focus on Sharpe ratios instead of returns, we still observe that nonstandard errors exceed standard errors, with ratios ranging from 1.52 to 2.09. We also do not find evidence that any single design choice predominantly drives the high nonstandard error. When holding one specific design choice constant, nonstandard errors always exceed standard errors, with ratios ranging from 1.04 when using only published features to 1.85 when using a continuous target variable. Overall, these results suggest that the variation resulting from the different design choices introduces additional sizeable uncertainty beyond the traditional statistical sampling process. Consequently, design choices in financial machine learning are of substantial importance.

Third, we identify the most influential design choices and provide recommendations for design choices based on prediction goals and economic effects. For example, the recommended target variable depends on the prediction goals. If the aim is to forecast higher relative raw returns, as common in cross-sectional stock return studies, the outperformance relative to the market is more suitable than the excess return over the risk-free rate. Conversely, if the goal is to achieve high market-risk-adjusted returns, CAPM beta-adjusted returns are preferable, as the feature importance shows that this target effectively captures the low-risk effect.

Fourth, we identify conditions under which non-linear models outperform linear models. We find non-linear ML models significantly outperform linear OLS models when using stock outperformance relative to the market as the target variable, employing continuous target returns, or adopting expanding training windows.

Finally, we document a strong momentum effect across ML design choices. By going long past winning strategies and short losing strategies, the machine learning momentum strategy generates positive and statistically significant returns. This finding underscores the persistence of model performance and highlights the role of variation in design choices in shaping the profitability of machine learning models.

Our study makes four key contributions to the existing literature. First, we contribute to the body of research that employs machine learning models for stock return prediction (Rasekhschaffe and Jones, 2019; Freyberger et al., 2020; Gu et al., 2020; Tobek and Hronec, 2021; Azevedo et al., 2023; Cakici et al., 2023; Hanauer and Kalsbach, 2023; Howard, 2024). These studies typically apply specific design choices when tuning machine learning models. However, these choices often vary across studies. In contrast, we assess the importance of seven key design choices by systematically analyzing all possible combinations. Consequently, we validate the replicability of machine learning predictions and provide deeper insights into the performance variations resulting from these different choices. Our study is also related to Bali et al. (2024), who analyze the stock-level variation of 100 machine learning predictions as a predictive signal for the cross-section of stock returns. Unlike our study, they focus on the variation itself as a predictive signal. Furthermore, the 100 different return predictions are derived solely from random forest models that randomly select 76 features from a total set of 153 features, without varying other design choices.

Second, we contribute to the literature on nonstandard errors in finance. Menkveld et al. (2024) introduce the concept of nonstandard errors and apply it to six hypotheses using EuroStoxx 50 index futures data. Soebhag et al. (2024) and Walter et al. (2024) apply this concept to stock portfolio sorts and factor constructions, while Fieberg et al. (2024) focus on cryptocurrency portfolio sorts. These studies find that nonstandard errors are often more prominent than standard errors. We confirm that design choices play a crucial role in machine learning-based return predictions, with nonstandard errors of similar or even higher magnitude.

Finally, we contribute to studies that provide guidelines for finance research. For instance, Ince and Porter (2006) offer guidelines for handling international stock market data, Harvey et al. (2016) propose a higher hurdle for testing the significance of potential factors, and Hou et al. (2020) recommend methods for mitigating the impact of small stocks in portfolio sorts. By offering guidance on design choices for machine learning-based stock return predictions, we help reduce uncertainties in model design and enhance the interpretability of prediction results.

In an independent and contemporaneously written working paper, Lalwani et al. (2024) also investigate the role of methodological choices on the performance of machine learning strategies. However, in contrast to our study, they primarily focus on the effect of different sample filters, such as size, price, age, and industries, and differences in evaluation choices, such as the number of quantiles or portfolio weighting. More importantly, they do not investigate the effects of design choices related to target, target transformation, post-publication treatment, and feature preselection. The reported ratio of 1.99 of the nonstandard error with the standard error for gross value-weighted returns is slightly higher than in our study.

Our study has important implications for machine learning research in finance. A deeper understanding of the critical design choices is essential for optimizing machine learning models and thereby improving their reliability and effectiveness in predicting stock returns. By address-ing variations in research settings, researchers can demonstrate the robustness of their findings. Furthermore, our study underscores the need for researchers to carefully motivate their design choices.

The remainder of this study is structured as follows: In Section 2, we describe our data, data sources, the seven identified research design choices, and portfolio performance measurements. In Section 3, we examine the impact of these seven choices and compare standard errors with nonstandard errors. In Section 4, we identify the most influential design choices and provide guidance on the selection of design choices. In Section 5, we analyze two applications of design choices, including the conditions under which non-linear models outperform linear models, and the development of a machine learning momentum strategy to validate the performance persistence in design choices. Section 6 summarizes our findings.

2 Data and methodology

2.1 Data

Our analysis is based on U.S. common shares (codes 10 and 11) that are listed on NYSE, NYSE MKT (formerly AMEX), or NASDAQ. We retrieve monthly market data from the Center for Research in Security Prices (CRSP) and merge it with return predictors from the Open Source Asset Pricing (OSAP, March 2022 version) library of Chen and Zimmermann (2022). We select the 163 clear and 44 likely predictors as the lagged features in the machine learning algorithms.[1] Our sample period ranges from January 1957 to December 2021.

To mitigate the impact of small and illiquid stocks, we exclude microcaps, which are defined as stocks with a market capitalization below the 20th percentile of NYSE market capitalization (cf., Hou et al., 2020). After filtering, we have 1,632,495 monthly observations and a monthly average of 2,093 stocks.

Finally, we follow Gu et al. (2020), Avramov et al. (2023), Leippold et al. (2022), Hanauer and Kalsbach (2023), and Howard (2024) in preprocessing the features. More specifically, we cross-sectionally rank the stocks each month by each characteristic into the [-1, 1] interval. In case of missing feature values, we set the value to 0.[2]

[1]Moreover, we incorporate delisting returns and compute the short-term reversal, price, and size characteristics following the code by Chen and Zimmermann (2022).

[2]We do not treat the data preprocessing as a key machine learning design choice but follow the most common choice. By adhering to this common preprocessing procedure, we limit computational efforts as, e.g., numerous ways exist to handle missing values (cf., Bryzgalova et al., 2024; Freyberger et al., 2024). Therefore, the variation in the resulting returns of machine learning portfolios might be even higher when considering different data preprocessing choices.

2.2 Research design choices

When predicting stock returns using machine learning algorithms, researchers and practitioners face a number of important methodological choices. We identify such variations in design choices in several published machine learning studies, all of which predict the cross-section of stock returns. More specifically, these studies include Gu et al. (2020), Freyberger et al. (2020), Avramov et al. (2023), and Howard (2024) for U.S. market, Rasekhschaffe and Jones (2019) and Tobek and Hronec (2021) for global developed markets, Hanauer and Kalsbach (2023) for emerging markets, and Leippold et al. (2022) for the Chinese market. In total, we identify variations in seven common research design choices across these studies, and we categorize them into four main types regarding the algorithm, target, feature, and training process. Table 1 summarizes the specific design choices of these studies.

[Table 1 about here.]

Note that we focus on design choices regarding the setup of the model training and do not consider differences with respect to the sample period, the set of features, or the portfolio construction to evaluate the predictions.[3] Figure 1 illustrates the seven research design choices. In the remainder of this section, we explain each choice in more detail.

[Figure 1 about here.]

2.2.1 Algorithm (N=11)

The investigated studies use a variety of algorithms to predict stock returns, covering both linear prediction models and models that allow for non-linearities and interactions. Linear models comprise simple linear models such as ordinary least squares (OLS), penalized linear models such as least absolute shrinkage and selection operator (LASSO) or elastic net (ENET), or linear predictions that reduce dimensions such as principal component regression (PCR). Models that allow for non-linearities and interactions include tree-based methods such as random forest (RF) and gradient-boosted regression trees (GB), and neural networks (NN).

Covering all available algorithms is beyond the scope of this study. Therefore, we select eleven representative machine learning models, including two linear models with ordinary least squares (OLS) and elastic net (ENET) regression, two tree-based models with random forest (RF) and gradient-boosted regression trees (GB), five neural network models with one to five hidden layers (NN1, NN2, NN3, NN4, NN5), one forecast combination model of all neural

[3] Soebhag et al. (2024) and Walter et al. (2024) investigate design choices regarding portfolio sorts and factor construction.

networks (ENS NN), and one forecast combination model of all non-linear machine learning algorithms (ENS ML).[4] The forecast combination is the simple average across all the monthly predictions from each constituent model. We provide a more detailed description of the models in Appendix A.

Hyperparameter tuning is also a crucial step in the training of a machine learning model. Hyperparameters are external configuration settings that are not learned from the data but rather set before the training process begins. Most of the machine learning papers employ hyperparameter tuning. By fine-tuning the hyperparameters, we can enhance the model's predictive accuracy. Note that different algorithms have their own set of hyperparameters, and we cover a broad range of hyperparameters commonly used in previous studies, detailed in Appendix Table B.1. We train our models once a year in December and then use the tuned model to make out-of-sample predictions for each of the following twelve months based on the latest monthly data.[5]

2.2.2 Target variable (N=3)

There are three main choices of target variable in the literature: the stock excess return over the risk-free rate (RET-RF), the stock outperformance over the market return (RET-MKT), and the risk-adjusted return (RET-RISK).

Using the excess return over the risk-free rate is the most common choice (cf., Freyberger et al., 2020; Gu et al., 2020; Avramov et al., 2023; Leippold et al., 2022). While the excess return over the risk-free rate is the dependent variable in asset pricing models, it seems a surprising choice when one focuses on the "cross-section of stock returns," i.e., the stock returns relative to other stocks and not the premium of a stock over the risk-free rate.

Therefore, Tobek and Hronec (2021) and Hanauer and Kalsbach (2023) use the abnormal return relative to the market return as the return target. They argue that removing this common market element increases the signal-to-noise ratio. In this paper, we compute the abnormal stock return as the monthly stock return minus the value-weighted return of all stocks in our sample.

Finally, Rasekhschaffe and Jones (2019) argue that most investors want to outperform net of risk, so a natural approach is to define performance categories for risk-adjusted return. To adjust for risk, various methodologies can be employed, including volatility-adjusted returns or alphas from an appropriate risk model, such as the capital asset pricing model (CAPM),

[4]ENS NN takes the simple average forecasts from NN1 to NN5. ENS ML takes the average of ENS NN, RF, and GB.

[5]The effectiveness of our process is demonstrated in Appendix Figure B.1, where we tune models based on the most common choices and observe results similar to Gu et al. (2020), Tobek and Hronec (2021), or Hanauer and Kalsbach (2023).

the Carhart (1997) four-factor model, or the Fama and French (2015) five-factor model. In our analysis, we apply the CAPM-adjusted return as a representative measure, which involves deducting the product of the stock's beta (item Beta in Chen and Zimmermann, 2022) and the market excess return from the stock excess return, where excess return refers to the total return over the risk-free rate.

All target variables represent one-month-ahead returns. We do not investigate the prediction horizon as a separate design choice, as all of the above-mentioned studies focus on predicting one-month forward returns. Furthermore, Blitz et al. (2023) investigate the effects of training on longer prediction horizons in detail.

2.2.3 Target transformation (N=2)

Machine learning algorithms can forecast both continuous and discrete variables. For instance, Gu et al. (2020) employ a regression scheme to predict stock returns directly. In contrast, Rasekhschaffe and Jones (2019) suggest predicting categories, such as outperformers versus underperformers for a specific target, and finally, getting the probability of outperforming the market for each stock.

In this context, we define two types of transformation for target variables: the raw continuous return without any transformation (RAW) and the transformation to a dummy variable (DUMMY), which is 1 if the target (RET-RF, RET-MKT, or RET-RISK) is greater than zero, and is 0 otherwise. These two schemes yield different target variables and provide distinct market information.

2.2.4 Post publication (N=2)

McLean and Pontiff (2016) document a strong post-publication effect in the U.S. market. They find that the variable returns decline by 26% out-of-sample and 58% post-publication. Most machine learning papers do not filter variables based on publication date. In contrast, Tobek and Hronec (2021) allow only published anomalies to enter predictions in each year. Therefore, we also consider the inclusion and exclusion of the post-publication effect as one research design choice.

We obtain the anomaly's publication year from Chen and Zimmermann (2022). When training the machine learning models at the end of year t, we incorporate only those characteristics that were published prior to the end of year t. In other words, we exclude all the unpublished characteristics to account for the post-publication effect.

2.2.5 Feature selection (N=2)

Jointly using all available features from Chen and Zimmermann (2022) might lead to problems in the model training in case the features are highly correlated.[6] Freyberger et al. (2020) also argue that it is crucial to investigate which characteristics can provide independent incremental information for the cross-section of expected returns and apply a LASSO-based feature selection in a first step. Incorporating such a pre-selection, on the one hand, may enhance the signal-to-noise ratio and achieve better results. On the other hand, the non-selected features may still contain (non-linear) information about future returns or useful features for interactions.

In our study, we consider two research design choices: using all available features or performing a feature pre-selection via a LASSO regression. The LASSO method introduces a penalty term and gives a weight of zero to less important features, thus effectively discarding irrelevant features and mitigating overfitting. We then only utilize the pre-selected features to predict the one-month-ahead returns.

2.2.6 Training window (N=2)

It is a common practice to split the data into training/validation/test subsamples. The two main approaches for defining the training window and sample splitting are the rolling scheme and the expanding scheme.

The expanding scheme is the most widely used. For instance, Gu et al. (2020), Tobek and Hronec (2021), Leippold et al. (2022), and Hanauer and Kalsbach (2023) apply this scheme. The main idea is to use all available data to predict future returns. In this study, we follow Gu et al. (2020) and apply an expanding window with a minimum of 30 years of data. In our case, the first training data comprises the first 18 years, spanning from January 1957 to December 1974. The initial validation data covers the next 12 years, from January 1975 to December 1986. Consequently, our first out-of-sample prediction is made for January 1987. Thereafter, we recursively increase the training sample by 1 year, keep the length of the validation sample constant, and make the OOS prediction over the subsequent year.

On the one hand, a rolling scheme results in more adaptive models that incorporate only more recent data that might be more representative of the current environment. Rolling windows are used, for instance, by Rasekhschaffe and Jones (2019) and Freyberger et al. (2020). Avramov et al. (2023) further apply various training windows. Following Freyberger et al. (2020), we apply a rolling window of 10-years. To have the same split as in the initial expanding window, the

[6] Cochrane (2011) coined the term “factor zoo” and asks for a consolidation. To this end, Swade et al. (2024) show that a set of more than 150 factors can be spanned by 15 factors.

first 6 years are used for training, while the subsequent 4 years are used for validation. The rolling window holds the length of both the training and validation sample fixed. To ensure comparability of prediction results, our first OOS prediction is also made for January 1987.[7]

2.2.7 Training sample (N=2)

Sometimes, researchers may select different training samples. For example, Avramov et al. (2023) use the full sample, non-microcaps, and other subsamples to make predictions. Howard (2024) finds that training separate models for different market capitalization groups results in better performance, especially for value-weighted portfolios. In this study, we consider two different choices regarding training samples. In the first scenario, the training and prediction samples are the same. The models are trained on all-but-micro stocks and predict the same set of stocks. In the second scenario, we train the models on all stocks (micro and non-micro stocks) but only predict the subset of all-but-micro stocks. On the one hand, the large training sample has more observations and might provide more information. On the other hand, if microcaps have structurally different return drivers, adding these to the training might do more harm than good.

2.3 Methodology

Training the machine learning models that reflect all possible combinations of choices results in $11 \times 3 \times 2 \times 2 \times 2 \times 2 \times 2 = 1,056$ models, in total. For each model and month, we obtain the one-month-ahead prediction for each stock from January 1987 to December 2021, a total of 35 years or 420 months. Next, we allocate stocks into decile portfolios based on these predictions and follow the recommendation of Hou et al. (2020) by using NYSE breakpoints and computing value-weighted portfolio returns. Finally, we get a long-short return time series for each model by taking a long position in the top decile stocks with the highest predictions and a short position in the bottom decile stocks with the lowest predictions. Based on these long-short returns, we further derive the metrics to compare the standard error versus the nonstandard error.

The traditional standard error is the sampling error of the population. Following Soebhag et al. (2024) and Fieberg et al. (2024), we estimate the standard errors by employing a block bootstrapping approach for each given set of construction choices. We split the 420 months into 210 adjacent pairs of months. In each simulation, we draw a random sample of 210 pairs with replacements to generate a simulated times series of long-short portfolio returns. We repeat this

[7]There are also some papers using a fixed scheme by training and validating the model only once, primarily due to capacity in computational resources, such as Jiang et al. (2023).

simulation process 10,000 times for each individual model. Subsequently, the standard error is calculated as the standard deviation of returns obtained from block bootstrapping. This provides us with a measure of the uncertainty associated with a given model's performance, considering the given set of research design choices.

On the other hand, the nonstandard error is defined as the standard deviation of the generated portfolio returns across the 1,056 possible construction models. According to Menkveld et al. (2024), the nonstandard error represents the uncertainty in the variation across choices made by researchers in the evidence-generating process, and it adds an additional layer of error to the analysis.

By comparing these two metrics, we can determine whether the nonstandard error exceeds the standard error. This comparison is crucial as it demonstrates that ML-predicted returns are not solely dependent on input characteristics but also influenced by the research design choices.

3 Empirical results

3.1 Overview of machine learning portfolios

We first investigate the performance dispersion of the different machine learning strategies resulting from different design choices. Figure 2 shows the cumulative performance of the 1,056 long-short portfolios. Each line represents the performance of one specific set of research design choices.

[Figure 2 about here.]

The figure shows that the variation in design choices leads to a substantial variation in returns. A hypothetical \$1 investment in 1987 leads to a final wealth ranging from \$0.94 (annual compounded return of -0.17%) to \$2,652 (annual compounded return of 24.48%) in 2021. The best model in our sample is characterized by the following design choices: Algorithm (ENS ML), Target (RET-MKT, RAW), Feature (No Post Publication, No Feature Selection), and Training (Expanding Window, ExMicro Training Sample). On the other hand, the worst-performing model is associated with the following design choices: Algorithm (RF), Target (RET-CAPM, RAW), Feature (Post Publication, Feature Selection), and Training (Rolling Window, All Training Sample). The details of the top- and bottom-performing models are documented in Appendix Table B.2. Apart from that, we also observe that all machine learning models perform worse in recent years, particularly after 2004, which aligns with the findings of Blitz et al. (2023).

We proceed to analyze the impact of research design choices based on the value-weighted long-short portfolio returns over the entire 420-month period. We focus on the value-weighted return to mitigate the bias toward small stocks. The portfolio returns are arranged in ascending order and visualized in Figure 3.

[Figure 3 about here.]

We find that machine learning models exhibit large variations in portfolio returns within our set of possible construction methods. Depending on how we train the models, the monthly returns vary between 0.13% to 1.98%. Besides, 80% of the models fall within the range of 0.56% (10th percentile) to 1.47% (90th percentile). The remaining models generate either very high or very low returns, so the graph exhibits steeper slopes at both ends. This substantial dispersion among the models signifies the importance of carefully considering the research design choices as they significantly impact the performance outcomes.

3.2 Decision choice impact

In this subsection, we examine the impact of each individual research design choice on portfolio returns. We aim to assess the effects in more detail both within and across seven decision choices. Figure 4 shows the portfolio returns in a box plot with the mean, median, first quartile, third quartile, minimum, and maximum values.[8]

[Figure 4 about here.]

The algorithm choice contains eleven alternatives, comprising linear methods (OLS, ENET), tree-based methods (RF and GB), neural networks with one to five hidden layers (NN1-NN5), as well as an ensemble of all neural networks (ENS NN) and an ensemble of all non-linear ML methods (ENS ML). The results show that the composite methods exhibit higher mean and median portfolio returns than the other nine individual algorithms. While our primary focus is not to compare individual algorithms, we notice that the neural networks (NN) display better performance, whereas random forest (RF), on average, performs the worst.

The superior performance of the ensemble model underscores the efficacy of forecast combination techniques, as they help reduce noise and provide more accurate information by focusing on the relationships that are robust to different algorithms. This supports the notion that forecast combination methods can outperform even the best individual forecasts, as discussed

[8]We also exmine the impact of each individual research design choice on portfolio Sharpe ratios in Appendix Figure B.3 that reveals consistent patterns.

in Timmermann (2006) in general and shown in machine learning studies for stock return prediction in Rasekhschaffe and Jones (2019) and Hanauer and Kalsbach (2023). Therefore, we suggest using the forecast combination in practical applications.

Among the three choices in the target variable, the stock outperformance relative to the market (RET-MKT) exhibits the highest portfolio returns. In contrast, the most popular design choice, the excess return over the risk-free rate (RET-RF), performs worst. This suggests that for models aiming to predict relative returns, a target that measures returns relative to the market can help increase the signal-to-noise ratio. More specifically, the average portfolio returns of RET-MKT is approximately 13% higher than that of RET-RF. This indicates that stock outperformance relative to the market might potentially provide a more effective framework than the commonly used excess return approach.

Regarding the target transformation, the continuous target exhibits an average return that is 17% higher than that of the discrete target. It seems that the dummy target variable, which only classifies stocks as outperformers or underperformers, may lose some valuable information, thereby compromising prediction results in some combination of design choices.

When considering the included features, the post-publication has the strongest effect within our set of choices. Including all the characteristics at all times results in a mean return of 1.28% per month, while excluding the unpublished characteristics during each training period leads to a considerably lower return of 0.75%. In other words, when not considering the post-publication effect, the portfolio performance is 71% higher. This finding aligns with McLean and Pontiff (2016) and underscores the importance of accounting for the post-publication effect in practice.

The choice of feature pre-selection, by contrast, has little impact on the average portfolio performance. When no pre-selection is applied, we observe a mean return of 1.03%, while implementing the LASSO method to eliminate the irrelevant features results in almost the same return of 1.01%. Therefore, the predictors employed in our study, which have already been validated by prior academic research, are deemed sufficiently effective, and no further feature pre-selection is needed.

When tuning to the training process of the machine learning algorithm, the choice of training window has a notable impact. The expanding training window contains more historical information, and it has a higher average return of 1.12%. In contrast, the rolling method with a short training window may not capture the whole business cycle, and it has a lower return of 0.92%. The expanding window is 22% higher. Therefore, the training window also comes out as one of the more important design choices.

Finally, we discuss the design choice regarding the training sample, which seems to have little impact. When training the model on a large dataset with all firms and using this trained model to predict the all-but-micro stocks, we observe almost no increase in the average portfolio returns. Therefore, it seems that two effects, having more information vs. group-specific information, seem to balance out.[9]

All the design choices we examined demonstrate the expected impact on the portfolio returns, albeit with varying magnitudes. When comparing different categories of design choices, we find that post-publication has the most significant impact on the portfolio returns, followed by training window, and target transformation. The feature preselection and training sample shows the smallest differences among the design choices we explored.

Additionally, we provide supplementary information in Table 2, which reports the summary statistics of the seven decision choices. Apart from the monthly portfolio returns, it also documents the monthly standard deviation, annualized Sharpe ratios, standard error, nonstandard error, and the ratio of nonstandard error to standard error in each decision group. These statistics reveal substantial variation in portfolio performance within each category of design choices. These findings highlight the importance of considering each design choice carefully, as they can have substantial effects on the performance and variations of the machine learning models.

[Table 2 about here.]

3.3 Nonstandard error versus standard error

Next, we compute the nonstandard error and compare it with the estimated standard error. The standard error is based on the traditional statistical perspective, and it measures the uncertainty in the sampling approach to estimate the population parameters. In our analysis, the standard error is equal to the standard deviation of the long-short portfolio returns derived from block bootstrapping. By contrast, the nonstandard error captures the uncertainty in research design choices. It is equal to the standard deviation of the portfolio returns of different models. We estimate the nonstandard error either within each of the seven categories of design choices or for all 1,056 combinations. By examining both the standard and nonstandard errors, we gain a comprehensive understanding of the sources of uncertainty in machine learning predictions. This allows us to assess the overall impact of research design choices on the variability of the resulting portfolio returns.

[9]It is important to note that we focus solely on the liquid universe of stocks to ensure the results are not biased towards micro stocks. However, if we construct the long-short portfolios using all stocks, we can obtain significantly higher returns than the long-short portfolios comprising all-but-micro stocks. This can be attributed to the fact that micro stocks tend to have higher returns, which can drive up the overall return of the long-short portfolio. This finding is also consistent with Avramov et al. (2023), who compare four different samples, including full, non-micro, credit rating, and non-downgrade samples.

We first compare the standard error and nonstandard error for each category of research design choices in Table 2. The standard error remains quite stable across different research design choices, with values around 0.23. Moreover, regardless of the decision choices considered, the nonstandard error consistently surpasses the standard error. When holding one design choice fixed, the highest NSE-to-SE ratio is 1.85 when using a continuous target variable (Target Transformation = Raw), while the lowest NSE-to-SE ratio is 1.04 when using published features (Post Publication = YesPost). This indicates that the variation resulting from the different design choices introduces additional sizeable uncertainty beyond the traditional statistical sampling process.

Then, we compare the overall standard error with the overall nonstandard error in Figure 5. We define the overall nonstandard error as the standard deviation of mean returns across 1,056 specifications, and we obtain a value of 0.36. The overall standard error is derived from the block bootstrapping approach. The mean standard error is equal to 0.23, with a maximum value of 0.35 and a minimum value of 0.15. Therefore, the nonstandard error is 1.59 times the mean standard error and even exceeds the maximum value of the standard error.[10]

[Figure 5 about here.]

This finding suggests that the nonstandard error resulting from design choices for machine learning predictions is sizable relative to the standard errors, and the ratio is comparable or higher to the ratio reported in other studies. For example, Menkveld et al. (2024) observe a ratio of 1.60 regarding researcher discretion in finance experiments, while Soebhag et al. (2024) report a ratio of 1.06, and Walter et al. (2024) observe a ratio of 1.10 in the context of constructing asset pricing factors. These comparisons suggest that the variations in machine learning design choices can be even more pronounced than those of other tasks in finance research. The researchers and practitioners have a very high degree of discretion in building machine learning models.

As a robustness check, we calculate the NSE-to-SE ratios for various portfolio construction design choices. Our baseline model uses decile breakpoints of NYSE stocks and value-weighted portfolio returns. To test alternative specifications, we consider three variations: (1) equal-weighted portfolio returns, (2) quintile sorts, and (3) no NYSE breakpoints. The results, presented in Table 3, indicate that the NSE consistently exceeds the SE, with NSE-to-SE ratios

[10]We also compare the nonstandard error with the standard error based on Sharpe ratios (see also the impact of each individual research design choice on portfolio Sharpe ratios in Appendix Figure B.3). When using the Sharpe ratio as the portfolio performance measurement, the overall nonstandard error is the standard deviation of Sharpe ratios across 1,056 specifications, and we obtain a value of 0.33. The overall standard error is the cross-sectional average of the standard deviations of the Sharpe ratios obtained by block bootstrapping the time-series portfolio returns 10,000 times. The mean standard error is equal to 0.17, with a maximum value of 0.21 and a minimum value of 0.13. Therefore, the nonstandard error is 1.93 times the standard error.

ranging from 1.52 to 2.09. In summary, we confirm the crucial role of research design choices also for alternative portfolio construction methodologies.

[Table 3 about here.]

3.4 Variation in feature importance

Given the substantial variation in portfolio returns due to different design choices, the question arises of whether certain features cause this variation. To investigate this question, we analyze the variation in feature importance based on Shapley (1953) additive explanations (SHAP) values across all models. Figure 6 visualizes the variation in the top 20 most important features via box plots showing the mean, median, first quartile, third quartile, minimum, and maximum feature importance values. The features are ranked by their interquartile range (IQR).

[Figure 6 about here.]

The results show that the interquartile range that is largest for momentum (Mom12m), beta (Beta), short-term reversal (STreversal), trend factor (TrendFactor), and analyst earnings revisions (AnalystRevisions). Interestingly, these features not only show the highest variation but also have the highest average feature importance (cf., also Table 6 in subsection 4.2), underscoring their role as key drivers of portfolio return variations.

4 Relative importance and guidance for design choices

4.1 Significant design choices

Having highlighted the higher nonstandard errors in machine learning for stock return predictions and confirmed that the design choices matter, we next analyze which design choices have significant influence and offer guidance on how to effectively apply machine learning to finance.

To do this, we first examine the differences in average portfolio returns between each category's best design choice and the rest of the models. We conduct a panel regression of monthly long-short portfolio returns on seven dummy variables for design choices. Each dummy variable is set to 1 for the best design choice within each category and 0 for others. The panel regression includes month-fixed effects, with standard errors clustered by month. The result is documented in Table 4.

[Table 4 about here.]

Notably, the post-publication effect has the most substantial influence, with a 0.52% (t-stat of 6.19) higher monthly returns when including still unpublished characteristics in the feature set. This finding is in line with the results in McLean and Pontiff (2016). The second most influential design choice is the training window, where an expanding training window results in a 0.20% (t-stat of 1.90) higher monthly return compared to a rolling window, suggesting that models trained on the full history are more robust despite being less adaptive. Target transformation is next, with continuous targets achieving 0.16% (t-stat 1.38) higher returns on average than dummy targets, although the difference is not statistically significant. Similarly, using stock outperformance relative to the market (RET-MKT) as the target variable outperforms both excess return and risk-adjusted return with about 0.12% per month economically, but not statistically (t-stat of 0.90). Regarding algorithm choices, composite ML algorithms perform best, yielding an average return that is 0.08% (t-stat of 2.17) higher than the average of all other algorithms, confirming that forecasts ensemble can even outperform the best underlying individual forecasts (cf., Timmermann, 2006).

In contrast, feature selection and training sample have minimal impact, resulting in a negligible difference in portfolio returns of only 0.02% each, which are also statistically insignificant (t-stats of 1.06 and 0.42, respectively). Thus, pooling all features that have already been validated by academic research for prediction appears to work well, and feature pre-selection adds little value. Additionally, including micro stocks in the training sample offers minimal benefit for predicting non-micro stocks, as micro stocks may possess inherently different properties and structures, thereby introducing additional noise into the model. This suggests that aligning the training sample with the evaluation sample is sufficient.

Therefore, our analysis highlights the importance of key design choices that impact portfolio returns, including post-publication effects, training window, target transformation, algorithm selection, and target variable. After analyzing the differences in average portfolio returns, we proceed to evaluate the economic impact of some of these significant design choices in more detail.

4.2 Target variable and evaluation metrics

The selection of the target variable in machine learning usually depends on the investment objective. Some investors focus on absolute portfolio returns, while others may prioritize risk-adjusted returns. We have demonstrated that targeting market-adjusted return performs better than the commonly used excess return when applying the raw portfolio return as the evaluation metric. In this subsection, we further analyze the case for maximizing risk-adjusted return, such as the CAPM alpha.

Following our previous procedure, we construct the long-short portfolio by taking a long position in the top decile with the highest predictions and a short position in the bottom decile. Instead of calculating the raw portfolio return, we further adjust it for the market factor to obtain the portfolio's CAPM alpha as the evaluation metric.

The impact of most design choices remains consistent, as illustrated in Appendix Figure B.2. Similar to the previous results, algorithm selection, post-publication adjustment, and training window continue to have a significant influence, while feature selection and training sample do not exhibit a strong effect. However, we find a notable difference for the target category. Specifically, the target variable with RET-CAPM now shows the best performance, and the target transformation has less influence.

The comparison between the two evaluation metrics is shown in Table 5. When using long-short portfolio returns, RET-MKT performs best with a portfolio return of 1.16% per month, higher than RET-RF (1.07%) and RET-CAPM (1.07%). In contrast, for long-short portfolio CAPM alpha, RET-CAPM performs best at 1.35%, followed by RET-MKT (1.21%) and RET-RF (1.12%).

[Table 5 about here.]

To rationalize these findings, we perform a feature importance analysis based on Shapley (1953) additive explanations (SHAP) values (cf., Lundberg and Lee, 2017). The result in Table 6 indicates that short-term reversal (STreversal) and momentum factor (Mom12m) are the most important firm characteristics for the overall models. In contrast, the beta factor (Beta) becomes the most important feature for the models with the target of RET-CAPM. The machine learning algorithm effectively picks up the low beta effect, achieving better performance with this target. Therefore, the design choice of the target variable depends on the evaluation metrics, and it needs to be aligned with the investment objective.

[Table 6 about here.]

In conclusion, the design choice of the target variable critically depends on the evaluation criteria. Besides, the evaluation metrics have a more substantial impact on the choice of target, while the conclusions regarding algorithm, feature, and training are robust across different evaluation metrics. This finding underscores the importance of aligning the design choice of the target variable with the evaluation metrics and the investment objective.

4.3 Summarizing the effect of design choices

In summary, the design choices regarding feature preselection and the inclusion of micro stocks always have minimal impact. In contrast, all the other five investigated design choices are crucial for the prediction performance. Therefore, we give brief guidance on making these important design choices:

First, ensembles of machine learning models generally outperform individual algorithms. Although the selection of specific algorithms can be discretionary, neural networks tend to deliver robust performance. Furthermore, we recommend leveraging ensemble forecasting methods to further improve the model performance.

Second, the choice of the target variable depends on the investment objective. For achieving higher long-short returns, stock outperformance relative to the market (RET-MKT) is superior to often-used excess returns (RET-RF). For achieving higher market-risk-adjusted returns, the CAPM risk-adjusted returns (RET-RISK) are preferable, as they capture the low beta effect according to our feature importance analysis.

Third, continuous targets perform better than dummy targets. Restricting the target variable to a binary classification of outperformers versus underperformers may result in a loss of valuable information when predicting stock returns. Consequently, we recommend using a regression algorithm over a classification algorithm.

Fourth, performance decreases significantly after adjusting for the post-publication effect. Therefore, researchers should be aware that including non-published features in the model training may inflate out-of-sample performance.

Finally, the choice between rolling and expanding windows depends on discretion and assumptions. An expanding window retains the entire historical information and is more suitable for a constant and consistent market environment. Conversely, a rolling window leverages more recent information and is more suitable for more frequent changes in market dynamics. Based on our results, an expanding window is superior, in particular for methods that allow non-linearities and interactions.

5 Design choice effects on non-linearity and performance persistency

After summarizing the significant design choices, this section further examines two empirical applications arising from variation in design choice selection. First, we examine the circumstances under which non-linear models outperform linear models, thereby shedding light on the conditions that make particular design decisions more effective. Second, we investigate whether superior design choices can be identified not only ex post but also during our sample period by analyzing the persistence in the relative performance of machine learning models.

5.1 Conditions for non-linear model outperformance

Most studies (cf., Rasekhschaffe and Jones, 2019; Gu et al., 2020; Hanauer and Kalsbach, 2023) report that models allowing for non-linearities and interactions outperform linear models. However, our analysis reveals that across various design choices, many non-linear models still underperform their linear counterparts. More specifically, we find that even among the 96 composite ML models (ENS ML), 33 models (34%) yield lower portfolio returns than a simple OLS model. This observation motivates us to investigate the conditions under which non-linear models can outperform linear models.[11]

To achieve this, we first compute the difference in long-short portfolio returns between each ENS ML model and its corresponding OLS model with identical design choices. Figure 7 visualizes the density distribution of these return differences within each design choice category.

[Figure 7 about here.]

We document that the composite non-linear model (ENS ML) outperforms the linear model (OLS) under the following conditions: (i) the target variable is defined as the stock outperformance over the market (Target Variable = RET-MKT), (ii) the target variable is a continuous return (Target Transformation = Raw), or (iii) an expanding training window (Training Window = Expanding) is used.

More specifically, for the different target variables, RET-MKT delivers the highest average outperformance, yielding 0.22% per month, followed by RET-RF at 0.09% and RET-CAPM at -0.05%. The difference between RET-MKT and RET-RF (RET-CAPM) is 0.13% (0.27%) with a significant t-stat of 2.06 (5.00). Additionally, non-linear models with continuous target variables are more likely to outperform linear models, with continuous targets averaging an outperformance of 0.14% compared to 0.03% for dummy targets. The 0.11% difference is marginally significant at the 10% level with a t-stat of 1.82. Furthermore, expanding windows, with an average outperformance of 0.16%, prove more beneficial than rolling windows, which average 0.01%, in enhancing the performance of non-linear models. The difference between the expanding and rolling window is 0.15%, which is highly significant at the 1% level (t-stat of 4.12).

[11] Dudda and Hornuf (2024) also advocate that studies should report under which conditions machine learning models can outperform traditional statistical models.

In contrast, post-publication adjustments, feature selection, and training sample size have minimal impact on the outperformance of non-linear models. Across these design choices, the non-linear and linear models, on average, achieve the same portfolio returns, and the differences are not statistically significant. Although the post-publication adjustment significantly affects absolute portfolio returns, these adjustments have little impact on the relative performance of non-linear versus linear models when the post-publication setting is held constant.

These findings indicate that more complex machine learning models require larger training datasets with longer training windows to robustly capture non-linearities and interactions in the data.[12] Furthermore, deducting the common market component appears to improve the signal-to-noise ratio, while classifying stocks merely as outperformers or underperformers may discard valuable information.

5.2 Performance persistence in design choices

We have documented that certain design choices yield significantly better performance than others. However, the findings are only ex-post observations. To investigate whether an investor could also have identified superior design choices in real time, we examine the performance persistence of the different machine learning strategies.[13] More specifically, we investigate the performance of an ML design choice momentum strategy.[14]

This ML design choice momentum strategy goes long in past winning machine learning strategies and short in past losing strategies. To implement this, we rank machine learning models based on their cumulative long-short portfolio returns over the previous one-, three-, five-, or ten-year windows. The ML design choices momentum strategy takes a long position in the top-decile models with the highest past performance and a short position in the bottom-decile models with the lowest performance. We then evaluate the subsequent one-month returns of the long-short portfolio to assess the persistence of model performance. The results are reported in Table 7.

[12]While a linear model estimates a single parameter per predictor, non-linear models involve a rapidly expanding number of parameters even with a moderate number of predictors (cf. Gu et al., 2020; Hanauer et al., 2022). Consequently, using an expanding window will arguably improve the observations-to-parameters ratio.

[13]We also thought about making the design choices a separate hyperparameter, but given the different target variables (RET-RF, RET-MKT, and RET-RISK), target transformation (RAW and DUMMY), validation windows (stemming from expanding and rolling training windows), and training samples (full sample and non-microcaps) this is not easily possible.

[14]Momentum effects are widely documented in financial markets. Since the seminal work of Jegadeesh and Titman (1993), the momentum strategy of buying past winners and selling past losers has been established as a robust predictor of returns across asset classes and markets. Subsequent research has identified various forms of momentum, including momentum in international stocks (Rouwenhorst, 1998), industry momentum (Moskowitz and Grinblatt, 1999), and momentum in various asset classes (Asness et al., 2013). In addition, the concept of factor momentum has been introduced, showing that not only individual securities but also systematic factors exhibit performance persistence (e.g., Gupta and Kelly, 2019; Arnott et al., 2023).

[Table 7 about here.]

Our results indicate a clear machine learning momentum pattern: models that perform well in the past tend to continue to outperform in the near future. For instance, models in the top decile based on past one-year performance achieve an average return of 1.39% in the subsequent month, compared to only 0.65% for bottom-decile models. The resulting momentum strategy delivers an average long-short return of 0.74% per month, with a t-statistic of 2.96 and a Sharpe ratio of 0.50.

Furthermore, the machine learning momentum effect remains robust when cumulative past returns are calculated over different horizons, such as three-, five-, and ten-year windows. These findings suggest that the performance persistence of machine learning models is closely linked to their underlying design choices and that the past success of certain models contains predictive information about their future success.

To rule out that the ML design choices strategy is just long ML strategies that use all features (Post Publication - NoPost) and short strategies that use only already published signals (Post Publication - YesPost), we test the ML design choices momentum strategy on the 528 machine learning models using all signals. We report the results Panel B of Table 7 and find that the long–short portfolio returns decline but remain statistically significant for the one-, five-, and ten-year rolling windows. For example, when models are sorted by past one-year performance, the top decile earns an average monthly return of 1.48%, compared with 0.94% for the bottom decile. The return spread of 0.54% is statistically significant at the 5% level, with a t-statistic of 2.32.

In addition, Table 8 reports the proportion of design choices represented in the long- and short-side portfolios, along with their differences. These proportions align closely with the significance of the design choices discussed earlier in Section 4.3. More specifically, composite models and neural network models are more frequently selected by the long side. Regarding targets, the stock outperformance over the market (RET-MKT) and continuous target variables are more likely chosen by the long side, whereas excess return (RET-RF) and dummy-transformed targets are more common on the short side. Similarly, expanding windows are more prevalent in long portfolios, while rolling windows are more common on the short side. In contrast, feature selection and training sample choices exhibit only minor differences between the long and short portfolios. These patterns are consistent with our earlier analysis of significant design choices and suggest that machine learning momentum delivers consistent returns based on different model specifications.

[Table 8 about here.]

In sum, superior design choices exhibit performance persistence. This persistence is not only observable ex post but can also be identified in real time through ML design choice momentum strategy.

6 Conclusion

Our study underscores the critical impact of design choices in machine learning-based stock return predictions. We observe that existing studies differ substantially in key design choices, including algorithms, target variables, feature treatment, and training processes. This lack of consensus not only results in significant variations in outcomes but also hinders the comparability and replicability of findings. To address these challenges, we present a systematic framework for evaluating and selecting design choices in machine learning for return prediction.

In this study, we analyze 1,056 machine learning models derived from various combinations of research design choices documented in the academic literature. Our findings reveal that these design choices have a substantial impact on return predictions, leading to significant variations in long-short portfolio returns. Specifically, we find that the nonstandard error arising from these choices is 1.59 times higher than the standard error. Thus, these results confirm that design choices matter for machine learning return prediction models.

We identify the most influential design choices that affect portfolio returns, including post-publication treatment, training window, target transformation, algorithm, and target variable. Notably, excluding unpublished features in model training leads to an average decrease of 0.52% in monthly portfolio returns, while using an expanding training window yields a 0.20% higher monthly return than a rolling window. Additionally, models with continuous targets and forecast combinations show superior performance, underscoring the critical importance of these design choices.

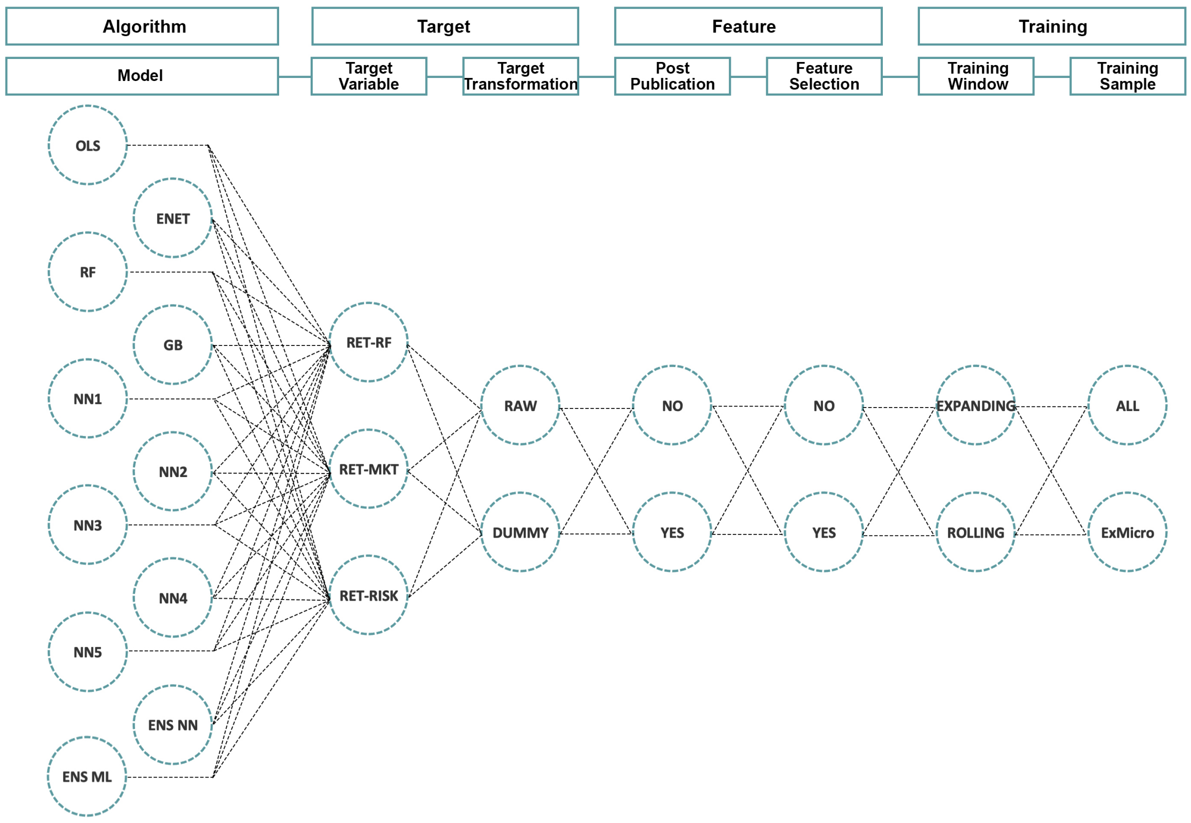
Our analysis provides a nuanced understanding of how design choices affect the performance of machine learning models, and we offer guidance for selecting appropriate choices based on economic effects. For instance, aligning the choice of the target variable with evaluation metrics and investment objectives is crucial for model performance. More specifically, we recommend using the stock outperformance relative to the market as the target variable to achieve higher relative portfolio returns.

Furthermore, we analyze the effects of design choices on the performance of non-linear versus linear models. Our results show that non-linear ML models significantly outperform linear OLS models when using stock outperformance relative to the market as the target variable, when using continuous target returns, or when adopting expanding training windows.

Finally, we document a pronounced machine learning momentum effect, showing that models with superior past performance tend to continue outperforming in the near future, thereby offering a novel application of variation in design choices.

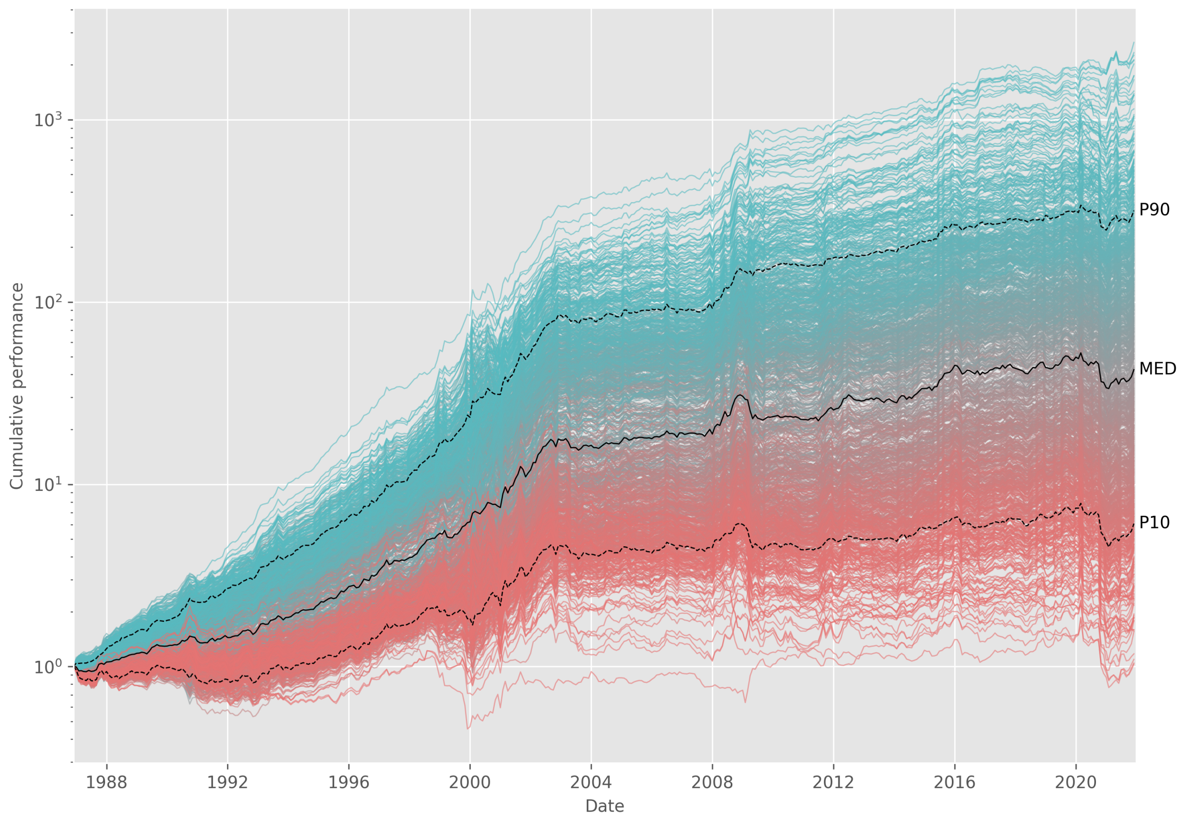
In conclusion, our study highlights the need for careful consideration and motivation in design choices in machine learning. Such decisions are not merely technical details but fundamental determinants of model performance, economic relevance, and the credibility of empirical findings.

Figure 1: Research design choices



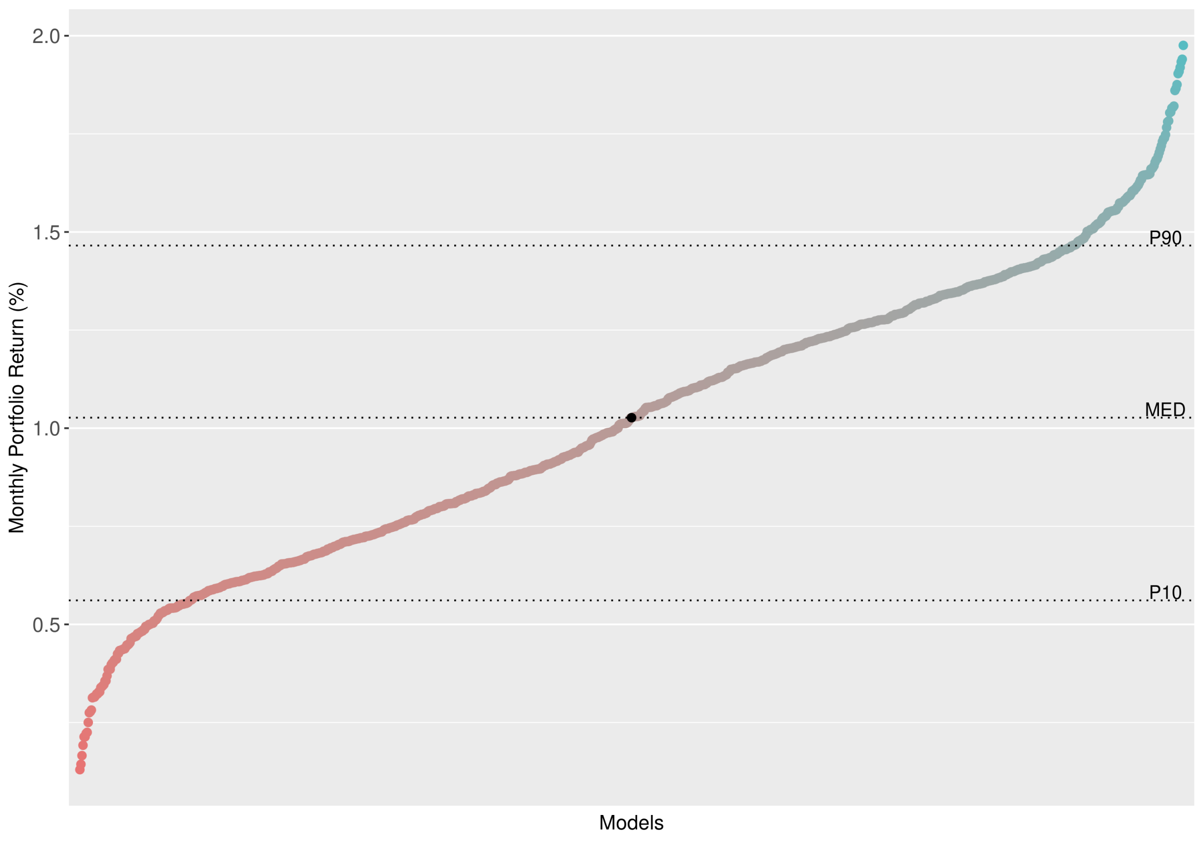
Description: This figure shows the seven key design choices in our analysis. We categorize them into four main types regarding the algorithm, target, feature, and training process, and list them as follows:
(1) Algorithm – Model (OLS, ENET, RF, GB, NN1, NN2, NN3, NN4, NN5, ENS NN, ENS ML),
(2) Target – Target Variable (RET-RF, RET-MKT, RET-RISK),
(3) Target – Target Transformation (RAW, DUMMY),
(4) Feature – Post Publication (NO, YES),
(5) Feature – Feature Selection (NO, YES),
(6) Training – Training Window (EXPANDING, ROLLING),
(7) Training – Training Sample (ALL, EXMICRO).
Each node represents one decision variable, and we have $11 \times 3 \times 2 \times 2 \times 2 \times 2 \times 2 = 1,056$ possible combinations.

Figure 2: Cumulative performance of machine learning portfolios



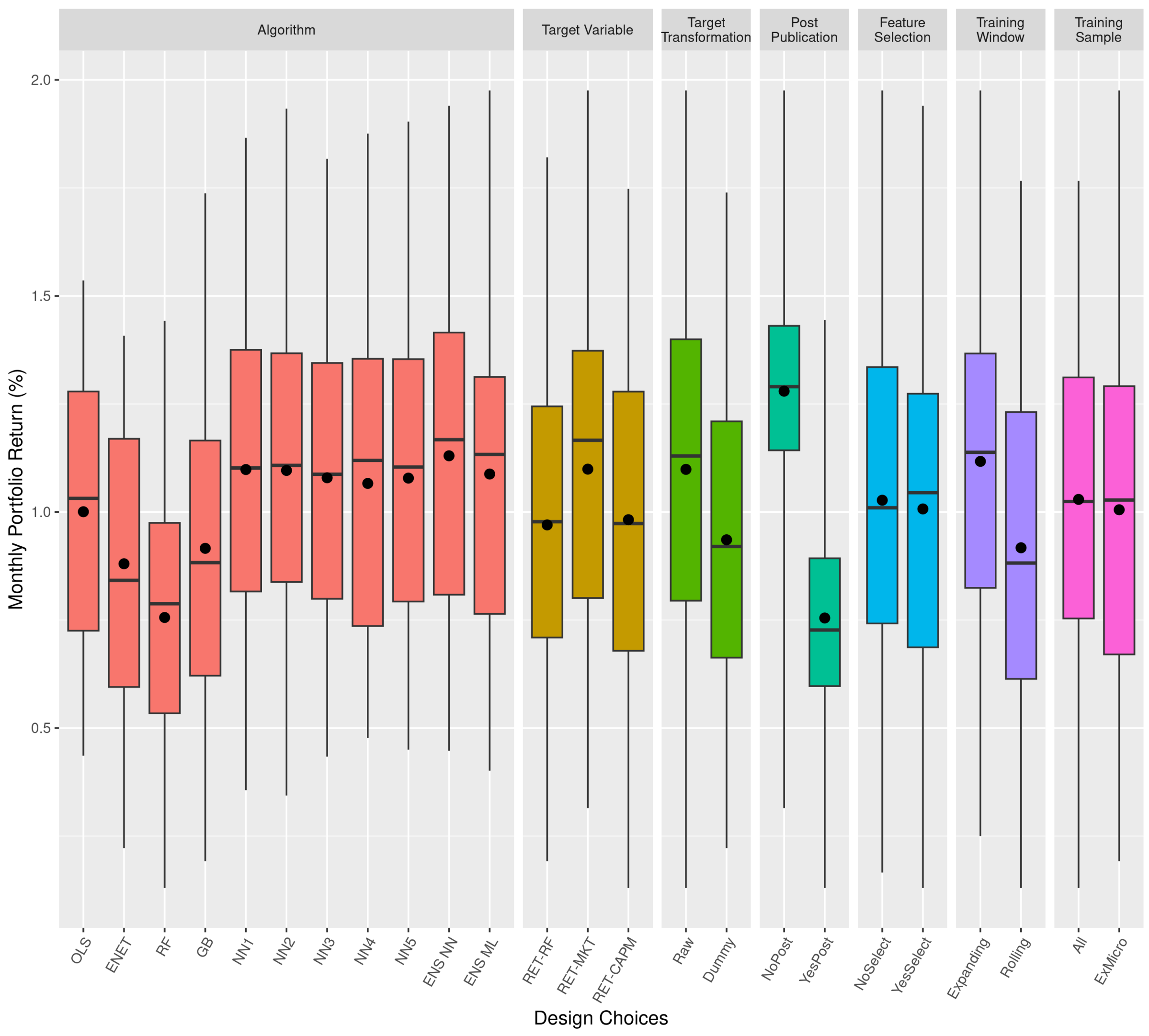
Description: This figure illustrates the cumulative performance of a \$1 initial investment in long-short machine learning portfolios for each possible combination of the research design choices. For each machine learning model and month, we first cross-sectionally sort all stocks based on their one-month-ahead return predictions. We then construct the value-weighted long-short portfolios by going long the top decile and short the bottom decile stocks. The solid black line represents the strategy with the median cumulative performance for each month, and the dashed black lines represent the 10th and 90th percentiles of each month, respectively. The sample period comprises January 1987 to December 2021.

Figure 3: Average monthly return of machine learning portfolios



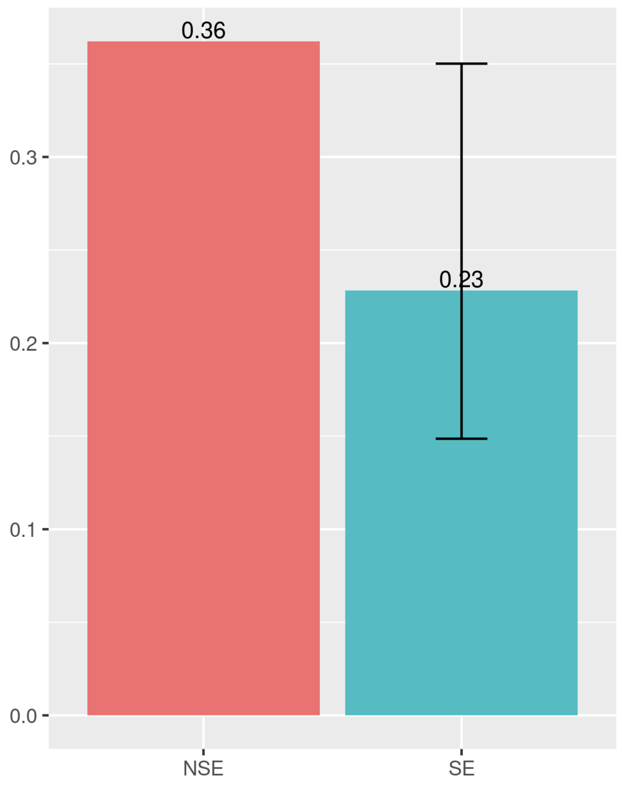
Description: This figure illustrates the average monthly value-weighted returns of 1,056 long-short machine learning portfolios in ascending order. Each dot represents one set of combinations of the research design choices. The highlighted black dot represents the model with the median returns. The upper (lower) dotted line represents the 90th (10th) percentile. The sample period comprises January 1987 to December 2021.

Figure 4: Portfolio returns by research design choices



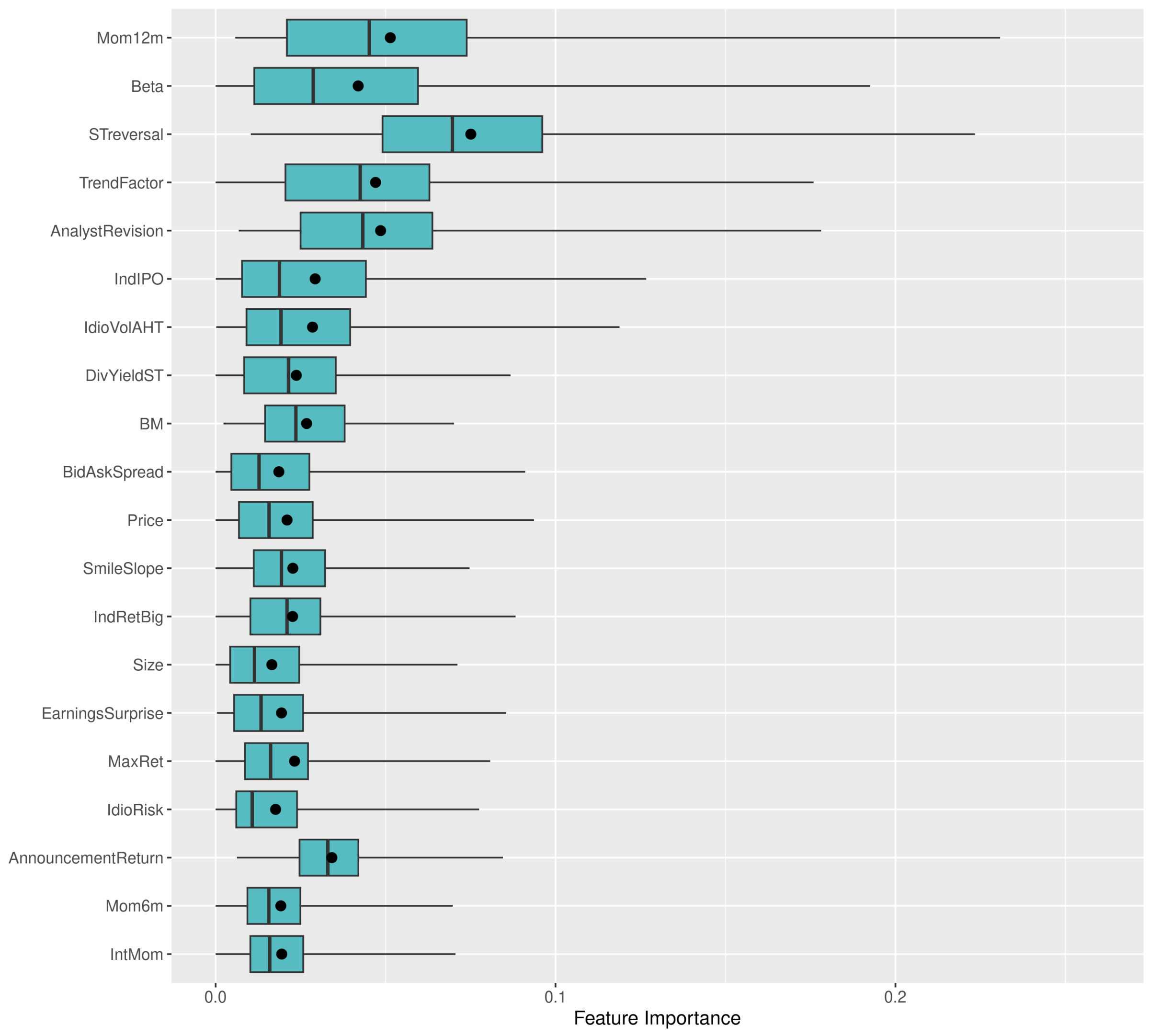
Description: This figure compares the monthly portfolio returns (in %) per design choice in a box plot. We consider seven distinct research design choices: algorithm, target variable, target transformation, post publication, feature selection, training window, and training sample. The middle black lines within each box plot indicate the median of the portfolio returns; the black dot within each box indicates the mean, and the lower and upper hinges correspond to the first and third quartiles. The sample period comprises January 1987 to December 2021.

Figure 5: Nonstandard error and standard error



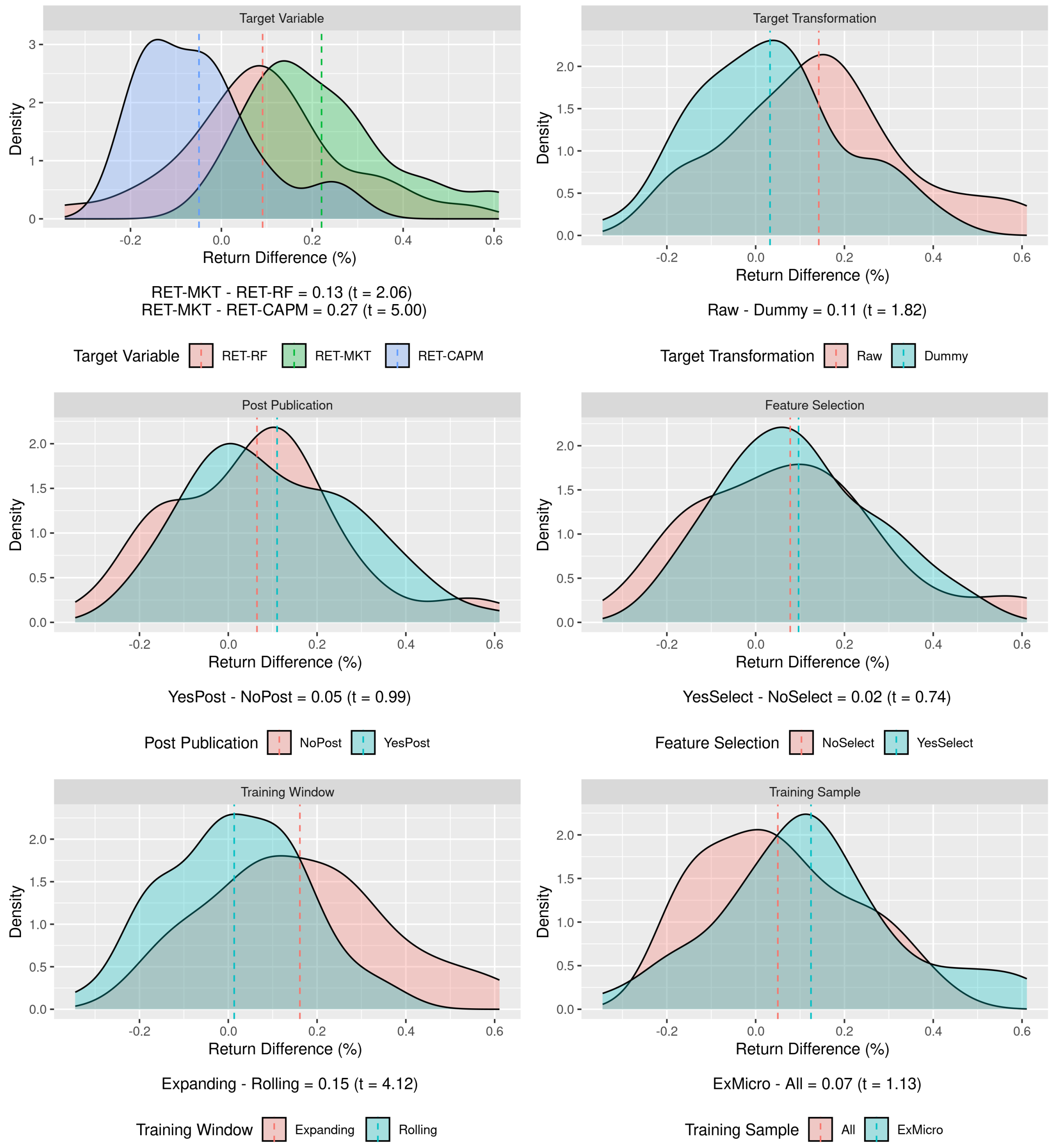
Description: This figure plots the overall nonstandard error (NSE) in the red bar and the overall standard error (SE) in the blue bar. The nonstandard error is calculated as the standard deviation of portfolio returns across the 1,056 machine learning models. The standard error is defined as the standard deviation of the portfolio returns obtained from block-strapping a machine portfolio's long-short returns 10,000 times. We take the average over each set of research design choices. The error line on the dashed bar indicates the minimum and maximum standard error. The standard error represents the traditional statistical uncertainty, and the NSE captures the additional sources of uncertainty arising from design choices. The sample period comprises January 1987 to December 2021.

Figure 6: Variations in feature importance



Description: This figure illustrates the variation in feature importance measured by Shapley Additive Explanations (SHAP) values. To ensure comparability, SHAP values are normalized to sum to one within each model and period. We then compute time-series averages of these normalized SHAP values for each machine learning model. The box plot illustrates the range of SHAP values for each feature across all machine learning models. We rank the features based on their interquartile range (IQR) and show the 20 features with the highest variations. In each box, the central black line represents the median feature importance, while the black dot marks the mean. The lower and upper hinges denote the first and third quartiles, with whiskers extending to the minimum and maximum values. Higher SHAP values indicate greater feature importance.

Figure 7: Return difference between ENS ML and OLS models



Description: This figure illustrates the difference in monthly portfolio returns between the composite non-linear machine models (ENS ML) and linear models (OLS) with the same design choices. We show the density distribution of the return difference within each category of design choices. The dotted line represents the mean difference of each category. Finally, we report the average difference between the design choices, along with Newey and West (1987) adjusted t-statistics with six lags. The analysis period ranges from January 1987 to December 2021.

Table 1: Design choices in published machine learning studies

	Gu et al. (2020)	Freyberger et al. (2020)	Avramov et al. (2023)	Howard (2024)	Rasekhschaffe and Jones (2019)	Tobek and Hronec (2021)	Hanauer and Kalsbach (2023)	Leippold et al. (2022)
Algorithm – Model	OLS, PLS, PCR, ENET, GLM, RF, GBRT, NN1-NN5	OLS, Adapitive Group LASSO	NN3, Adversarial Approach, IPCA, CA	OLS, ENET, RF, GB, NN1-NN5, ENS	AdaBoost, GBRT, NN, SVM	WLS, Panelized WLS, GBRT, RF, NN	OLS, ENET, RF, GB, NN1-NN5, ENS	OLS, PLS, LASSO, ENET, GBRT, RF, VASA, NN1-NN5
Target – Target Variable	RET-RF	RET-RF	RET-RF, RET-RISK	RET-RF, RET-Median	RET-RF, RET-RISK	RET-MKT	RET-MKT	RET-RF
Target – Target Transformation	Raw	Raw	Raw	Raw	Dummy	Raw	Raw	Raw
Feature – Post Publication	NoPost	NoPost	NoPost	NoPost	NoPost	YesPost	NoPost	NoPost
Feature – Feature Selection	NoSelect	YesSelect	NoSelect	NoSelect, YesSelect	NoSelect	NoSelect	NoSelect	NoSelect
Training – Training Window	Expanding	Rolling	Expanding, Rolling	Expanding	Rolling	Expanding	Expanding	Expanding
Training – Training Sample	All	All, ExMicro	All, ExMicro	All, Small, Middle, Large	All	ExMicro	ExMicro	All

Description: The table summarizes the variations in design choices in published machine learning studies predicting the cross-section of stock returns. These studies include Gu et al. (2020), Freyberger et al. (2020), Avramov et al. (2023), and Howard (2024) for U.S. market, Rasekhschaffe and Jones (2019) and Tobek and Hronec (2021) for global developed markets, Hanauer and Kalsbach (2023) for emerging markets, and Leippold et al. (2022) for the Chinese market.

Table 2: Summary statistics

	RET	STD	SR	SE	NSE	NSE/SE
Panel A: Algorithm						
OLS	1.00	4.34	0.82	0.21	0.32	1.51
ENET	0.88	4.58	0.70	0.22	0.33	1.52
RF	0.76	5.24	0.51	0.25	0.32	1.28
GB	0.92	4.87	0.67	0.23	0.35	1.48
NN1	1.10	4.65	0.85	0.23	0.37	1.62
NN2	1.10	4.64	0.84	0.23	0.35	1.54
NN3	1.08	4.63	0.84	0.23	0.35	1.55
NN4	1.07	4.61	0.83	0.22	0.35	1.56
NN5	1.08	4.60	0.84	0.22	0.35	1.55
ENS NN	1.13	4.69	0.86	0.23	0.36	1.60
ENS ML	1.09	5.01	0.79	0.24	0.36	1.49
Panel B: Target						
RET-RF	0.97	4.89	0.72	0.24	0.35	1.46
RET-MKT	1.10	4.18	0.92	0.21	0.37	1.80
RET-CAPM	0.98	5.07	0.69	0.24	0.35	1.48
Panel C: Target Transformation						
Raw	1.10	4.33	0.89	0.21	0.39	1.85
Dummy	0.94	5.10	0.66	0.25	0.32	1.28
Panel D: Post Publication						
NoPost	1.28	4.68	0.98	0.23	0.26	1.14
YesPost	0.75	4.75	0.57	0.23	0.24	1.04
Panel E: Feature Selection						
NoSelect	1.03	4.73	0.78	0.23	0.37	1.60
YesSelect	1.01	4.70	0.77	0.23	0.36	1.58
Panel F: Training Window						
Expanding	1.12	4.60	0.88	0.22	0.34	1.52
Rolling	0.92	4.83	0.68	0.23	0.36	1.53
Panel G: Training Sample						
ExMicro	1.01	4.63	0.79	0.22	0.39	1.74
All	1.03	4.80	0.77	0.23	0.33	1.43
Panel H: All Models						
All Models	1.02	4.71	0.78	0.23	0.36	1.59

Description: This table documents the summary statistics for the value-weighted long-short machine learning portfolios. Columns 1-3 report the mean of the monthly return (RET in %), monthly standard deviation (STD in %), and annualized Sharpe ratio (SR). Columns 4-6 display the standard error (SE), non-standard error (NSE), and the ratio of non-standard error to standard error (NSE/SE), respectively. The machine learning portfolios are categorized into seven distinct research design choices, labeled Panels A through G, with a comprehensive summary of all models provided in Panel H. The analysis period spans from January 1987 to December 2021.

Table 3: NSE vs SE across different portfolio constructions

	NSE	SE	Max(SE)	Min(SE)	NSE/SE
Baseline	0.36	0.23	0.35	0.15	1.59
EW	0.44	0.21	0.34	0.14	2.09
Quintile sort	0.27	0.18	0.29	0.12	1.52
No NYSE breakpoints	0.40	0.26	0.39	0.17	1.57

Description: This table presents a comparison of NSE to SE ratios across various portfolio construction design choices. The baseline model employs NYSE breakpoints and value-weighted long-short portfolio returns in decile sorts. In contrast, the three alternative specifications are based on: (1) equal-weighted portfolio returns, (2) quintile sorts, and (3) no NYSE breakpoints. The sample period spans from January 1987 to December 2021.

Table 4: Performance and design choices

	Coef.	t-stat.
Feature - Post Publication: No	0.52	6.19
Training - Traning Window: Expanding	0.20	1.90
Target - Target Transformation: Raw	0.16	1.38
Target - Target Variable: RET-MKT	0.12	0.90
Algorithm - Model: ENS ML	0.08	2.17
Feature - Feature Selection: No	0.02	1.06
Training - Training Sample: All	0.02	0.42
Obs.	443,520	
Adj. R^2	0.45	

Description: This table reports the outperformance in portfolio returns of each category's best research design choice compared to the rest of the models. We conduct a panel regression of monthly long-short portfolio returns on seven dummy variables for design choices. Each dummy variable is set to 1 for the best design choice within each category and 0 for others. For instance, in the Algorithm category, the dummy variable is set to 1 for the ENS ML model and 0 for other models. The panel regression includes month fixed effects, with standard errors clustered by month. The sample period comprises January 1987 to December 2021.

Table 5: Target variable and evaluation metrics

Target	RET	CAPM Alpha	SR
RET-RF	1.07	1.12	1.03
	(7.38)	(7.44)	
RET-MKT	1.16	1.21	1.11
	(7.44)	(7.50)	
RET-CAPM	1.07	1.35	0.81
	(5.68)	(7.87)	

Description: This table presents the portfolio-level prediction performance, based on long-short portfolio returns in %, CAPM alpha in %, and annualized Sharpe ratios. Here, we show the design choice for three target variables: RET-RF, RET-MKT, and RET-CAPM, and we focus on models with continuous target variable (Target Transformation = Raw). We first take the average long-short returns across all ML portfolios with the same target variable in each month, then compute the time-series mean and compute Newey and West (1987) adjusted t-statistics using six lags. The sample period comprises January 1987 to December 2021.

Table 6: Feature importance

	Overall	Target		
		RET-RF	RET-MKT	RET-CAPM
STreversal	0.075	0.065	0.081	0.081
Mom12m	0.051	0.047	0.052	0.055
TrendFactor	0.051	0.045	0.065	0.043
AnalystRevision	0.049	0.052	0.053	0.041
Beta	0.043	0.025	0.020	0.085
AnnouncementReturn	0.034	0.032	0.039	0.032
IndIPO	0.029	0.046	0.023	0.019
IdioVolAHT	0.029	0.040	0.022	0.025
BM	0.027	0.024	0.030	0.026
DivYieldST	0.024	0.026	0.022	0.023
MaxRet	0.023	0.029	0.014	0.026
SmileSlope	0.023	0.019	0.027	0.022
IndRetBig	0.023	0.022	0.026	0.020
High52	0.021	0.028	0.017	0.018
Price	0.021	0.023	0.022	0.018
CF	0.020	0.018	0.022	0.021
IntMom	0.019	0.016	0.020	0.023
EarningsSurprise	0.019	0.019	0.021	0.018
MomSeason	0.019	0.018	0.020	0.020
Mom6m	0.019	0.019	0.020	0.018

Description: This table highlights the top 20 most important predictors across all models, as well as specifically for the design choice of three target variables (RET-RF, RET-MKT, and RET-CAPM). The feature importance is determined through Shapley additive explanations (SHAP) values. To ensure comparability, SHAP values are normalized within each model and period to sum to one. Subsequently, we calculate the time-series averages of the SHAP values for each machine learning model. Higher SHAP values in darker backgrounds indicate greater feature importance.

Table 7: ML design choices momentum

	Long	Short	L-S	t(L-S)	SR
Panel A: All models					
Rolling 1Y	1.39	0.65	0.74	2.96	0.50
Rolling 3Y	1.38	0.81	0.57	2.50	0.47
Rolling 5Y	1.42	0.77	0.65	2.89	0.58
Rolling 10Y	1.39	0.72	0.67	2.68	0.59
Panel B: Only models using all signals					
Rolling 1Y	1.48	0.94	0.54	2.32	0.38
Rolling 3Y	1.44	1.14	0.30	1.32	0.25
Rolling 5Y	1.48	1.03	0.45	2.01	0.41
Rolling 10Y	1.42	0.90	0.52	2.24	0.47

Description: This table documents the momentum effects on ML strategies. Each month, all ML strategies are ranked from highest to lowest based on past cumulative performance over rolling 1-, 3-, 5-, and 10-year windows. Panel A uses the full set of 1,056 ML models for ranking, while Panel B restricts the sample to 528 models using all signals. The ML design choices momentum strategy takes long positions in the top-decile models with the highest past cumulative returns and short positions in the bottom-decile models. The returns of the resulting long–short portfolio are then calculated for the following month. The columns report the average monthly returns of the long portfolios (top-decile models), short portfolios (bottom-decile models), the long–short spread (L–S), the Newey and West (1987) adjusted t-statistics using six lags, and the corresponding Sharpe ratios.

Table 8: Design choices selected by the ML design choices momentum strategy

	Long	Short	Diff.
Panel A: Algorithm			
ENS NN	12%	7%	5%
ENS ML	11%	7%	3%
NN1	12%	8%	3%
NN2	11%	8%	3%
NN5	10%	7%	2%
NN3	10%	8%	2%
NN4	10%	8%	2%
OLS	7%	6%	1%
GB	8%	11%	-3%
ENET	6%	12%	-6%
RF	5%	17%	-13%
Panel B: Target			
RET-MKT	42%	28%	14%
RET-CAPM	31%	34%	-2%
RET-RF	26%	38%	-12%
Panel C: Target Transformation			
Raw	70%	47%	23%
Dummy	30%	53%	-23%
Panel D: Post Publication			
NoPost	74%	26%	47%
YesPost	26%	74%	-47%
Panel E: Feature Selection			
NoSelect	53%	51%	2%
YesSelect	47%	49%	-2%
Panel F: Training Window			
Expanding	64%	29%	35%
Rolling	36%	71%	-35%
Panel G: Training Sample			
ExMicro	57%	56%	1%
All	43%	44%	-1%

Description: This table reports the distribution of design choices represented in the long and short portfolios of the ML design choices momentum strategy, along with their differences. The long portfolio consists of the top-decile machine learning strategies ranked by their cumulative past performance, while the short portfolio includes the bottom-decile strategies. The percentages indicate the share of models in each portfolio associated with a given design choice, averaged across all years and over four rolling windows of 1-, 3-, 5-, and 10-year horizons. The design choices within each category are ranked by the difference in proportions between the long and short portfolios. A positive difference reflects a greater relative prevalence in the long portfolio, whereas a negative difference denotes a higher relative prevalence in the short portfolio.

Appendices

A Machine learning methodology

A.1 Simple linear regression

The least complex method in our analysis and most widely used in the context of empirical asset pricing is the simple linear regression model estimated via the ordinary least squares (OLS) method. In the case of the simple linear regression, the conditional expectations $f^*(x)$ can be modeled using the following linear model:

$$f(x_{i,t,c}, \theta) = \theta^T x_{i,t,c}, \tag{1}$$

where θ, $\theta^T = (\theta_1, \theta_2, ..., \theta_p) \in \mathbb{R}^p$, is the column vector of coefficients that can be estimated with OLS by minimizing the loss function:

$$L_{EE}(\theta) = \frac{1}{NT} \sum_{i=1}^{N} \sum_{t=1}^{T} (r_{i,t+1,c}^{abn} - f(x_{i,t,c}, \theta))^2, \tag{2}$$

which is also known as the Mean Squared Error (MSE). The OLS has the big advantage that it does not require any hyperparameter input from the user. Further, by minimizing the loss function L_{MSE} a unique analytical solution can be extracted, which is easy to interpret as the coefficients, θ directly describe how a change in the stock characteristics affects the expected return. Additionally, if the number of observations in the underlying dataset is larger than the number of coefficients that need to be estimated, the OLS yields an efficient and unbiased estimator according to Wooldridge (2001). But if the number of characteristics approaches the number of observations in the dataset, the OLS has issues distinguishing between signal and noise. While the signal is the portion we can understand, model, and predict, noise consists of the unpredictable component of price movements. In the case of a small sample or a large number of characteristics, the OLS starts with over-fitting the coefficients to noise rather than extracting the signal. This is of particular importance in the field of asset pricing, which empirically relies on a low signal-to-noise ratio. This overfitting yields a higher in-sample performance but a poor out-of-sample performance. Further, multicollinearity between the different characteristics can lead to a fallacious interpretation of test statistics as well as misleading coefficients. Lastly, the OLS does not model or evaluate any non-linearities of the characteristics nor any potential interactions between them. Any non-linearity would have to be imputed by the user.

A.2 Regularized regression

To avoid overfitting in the case of empirical asset pricing, the user could increase the training sample, reduce the number of characteristics used to predict future returns, or utilize regularized regression techniques that identify which characteristics are informative and omits those that are not. Classical regularized regression techniques include ridge regression, lasso regression, and elastic net. To limit the number of machine learning methods, we concentrate on the elastic net, which is a combination of ridge and lasso regression. While the different regularized regression models have the same linear functional form as the simple linear regression, they differ with respect to the loss function by adding a penalty term ($\phi_{ENET}(\theta, \lambda, \rho)$) to it:

$$L_{ENET}(\theta, \lambda, \rho) = L_{MSE}(\theta) + \phi_{ENET}(\theta, \lambda, \rho). \tag{3}$$

This penalty term reduces the model's in-sample performance and increases its out-of-sample stability by shrinking the coefficients of noisy characteristics, improving the signal-to-noise ratio. The penalty function of the elastic net is defined as:

$$\phi_{ENET}(\theta, \lambda, \rho) = (1-\rho)\lambda \sum_{j=1}^{P} \mid \theta_j \mid + \frac{1}{2}\rho\lambda \sum_{j=1}^{P} \theta_j^2, \tag{4}$$

where λ ($\lambda \in \mathbb{R}^+$) is the regularization parameter that defines the magnitude of shrinkage and ρ ($\rho \in [0,1]$) which determines the relative weight between the two penalty components of the ridge and lasso regression. In the case of $\lambda = 0$ the regularized regression models yield a simple linear regression model. The coefficients are shrunk towards zero by setting $\lambda > 0$. As these two hyperparameters have to be set by the user, we utilize our validation sample to find the optimal in-sample λ and α in the first run. We determine the optimal θ in the second run using the full training and validation sample.

A.3 Tree-based regression

Tree-based models represent the first non-parametric regression model as their structure is decided by the training data. For our return prediction, we will utilize two tree-based methods: the random forest, as well as the gradient-boosted regression tree. Compared to the linear methods, one advantage of these tree methods is that the user does not have to manually add any potential non-linearities or interactions to the data as the tree methods build these by construction.

Regression trees follow the idea of sequentially partitioning the underlying data into groups that behave similarly to each other based on a selected characteristic with regard to the future return. By sequentially separating the data, the tree "grows" and new "branches" are created each time the data is split into new groups. The tree can grow to a depth of D based on the user input. At each new branch, the characteristic is picked that causes the biggest separation in the data based on an optimized cut-off value. In our case, for each separation, the characteristic is selected that minimizes the MSE. As soon as the data cannot be split into subgroups or the depth D is reached, a "leaf" is created. In asset pricing, the tree yields a return that is clustered by the underlying characteristics.

The following equation describes a tree with a depth of D and K leaves:

$$\begin{aligned} f(x_{i,t,c}, \theta, D, K) &= \sum_{k=1}^{K} \theta_k \mathbb{1}_{\{x_{i,t,c} \in C_k(D)\}} \\ \theta_k &= \frac{1}{N_k} \sum_{x_{i,t,c} \in C_k(D)} r_{i,t+1,c}^{abn}, \end{aligned} \tag{5}$$

where D is the depth of the tree measured as the maximum number of separations following the longest branch, $C_k(D)$ indicates the k-th separation of the characteristics, θ_k is average return within the partition, and $\mathbb{1}_{\{x_{i,t,c} \in C_k(D)\}}$ indicates if $x_{i,t,c}$ is part of $C_k(D)$. Following this methodology, a tree of depth D can capture up to $D - 1$ interactions. To avoid overfitting, the tree must be regularized. We follow two different approaches in our analysis.

The first regularization approach uses bootstrap aggregation, or "bagging," developed by Breiman (2001). In this approach, each of the T trees starts with a share of B bootstrap samples from the data and fits an individual regression tree to the bootstrapped data. Afterward, the forecasts from the individual trees are averaged. This reduces the variation in the prediction and stabilizes the prediction performance. In the case of the random forest, the trees additionally use random subsets R of characteristics to grow the branches. This reduces the impact of certain dominant return characteristics and creates de-correlated trees.

The second regularization approach is "boosting." It starts by training a weak and shallow regression tree on the full training data. In the next step, a second regression tree with the same depth D is trained on the residuals of the first tree. The prediction of these two trees is then averaged while the contribution of the second tree is shrunken by a factor LR (learning rate), $LR \in (0, 1)$ to avoid the model overfitting the residuals. At each new step b, till the model reaches a total of B trees, a new shallow tree is fitted to the residual, which is based on the $b - 1$-th model and added to it with a shrinkage weight of LR.

A.4 Neural networks

Neural networks are another highly flexible but opposed to the regression trees, a parametric model. While these models can have various forms, we focus on the standard structure of a feed-forward neural network. A feed-forward neural network consists of an "input" layer of input characteristics and the intercept, at least one "hidden" layer compromising activation functions, and an "output" layer that aggregates the outcome of the last hidden layer into a return prediction.

A feedforward neural network consists of several subsequent layers l, $l = 0, 1, ..., L$, one input layer ($l = 0$), $L - 1$ hidden layers ($l = 1, 2, ..., L - 1$) and one output layer $l = L$. Each layer l contains n^l nodes. In the case of the input layer, the number of nodes is equal to the number of characteristics, including an intercept, while the output layer contains due to the regression setting one node. In the case of the hidden layer, we consider an architecture of up to five hidden layers while the first hidden layer contains 32 nodes and each additional hidden layer divides the number of nodes by two compared to the previous layer following the geometric pyramid rule according to Masters (1993). This results in the following number of nodes per layer:

$$\begin{aligned}
n^0 &= p + 1, \\
n^1 &= 32, \\
n^l &= \frac{n^{l-1}}{2} \forall l \in \{2, ..., L - 1\}, \\
n^L &= 1.
\end{aligned} \tag{6}$$

Each of the nodes in the hidden layer contains an activation function. In our case we follow Gu et al. (2020) and Leippold et al. (2022) and choose the rectified linear unit defined as:

$$\mathrm{ReLU}(x) = \max(0, x), \tag{7}$$

For model training, we minimize the MSE between predicted and realized returns. In addition, we adopt the Adam optimization algorithm (Kingma and Ba, 2017), early stopping, batch normalization (Ioffe and Szegedy, 2015) and five ensembles with individual seeds (Hansen and Salamon, 1990) when training our models.

Appendices

B Tables and Figures

Table B.1: Hyperparameters

Models	Hyperparameters
ENET	Regularization (λ) = [0.00001, 0.0001, 0.001, 0.01] L1 ratio (ρ) = [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]
GB	Learning rate (LR) = [0.01, 0.1] Estimators (T) = [100, 200, 300, 400, 500, 600, 700, 800, 900, 1000] Max depth (D) = [1, 2, 3, 4, 5, 6, 7, 8, 9] Subsample (B) = [0.3, 0.5]
RF	Estimators (T) = [100, 300, 500, 1000] Max depth (D) = [1, 2, 3, 4, 5, 6, 7, 8, 9] Max features (R) = [0.01, 0.02, 0.04, 0.08, 0.16, 0.32, 0.64] Subsample (B) = [0.01, 0.02, 0.04, 0.08, 0.16, 0.32, 0.64]
NN	Learning rate (LR) = [0.001,0.01] L1 regularization (l_1) = [0.001, 0.0001, 0.00001] Hidden layers = 1 / 2 / 3 / 4 / 5 Neurons = 32 / 32, 16 / 32, 16, 8 / 32, 16, 8, 4 / 32, 16, 8, 4, 2 Hidden layer activation function = ReLU Output layer activation function = Linear Batch size = 1,000 Epochs = 100 Early stopping patience = 5 Adam params = Default Ensemble = 5

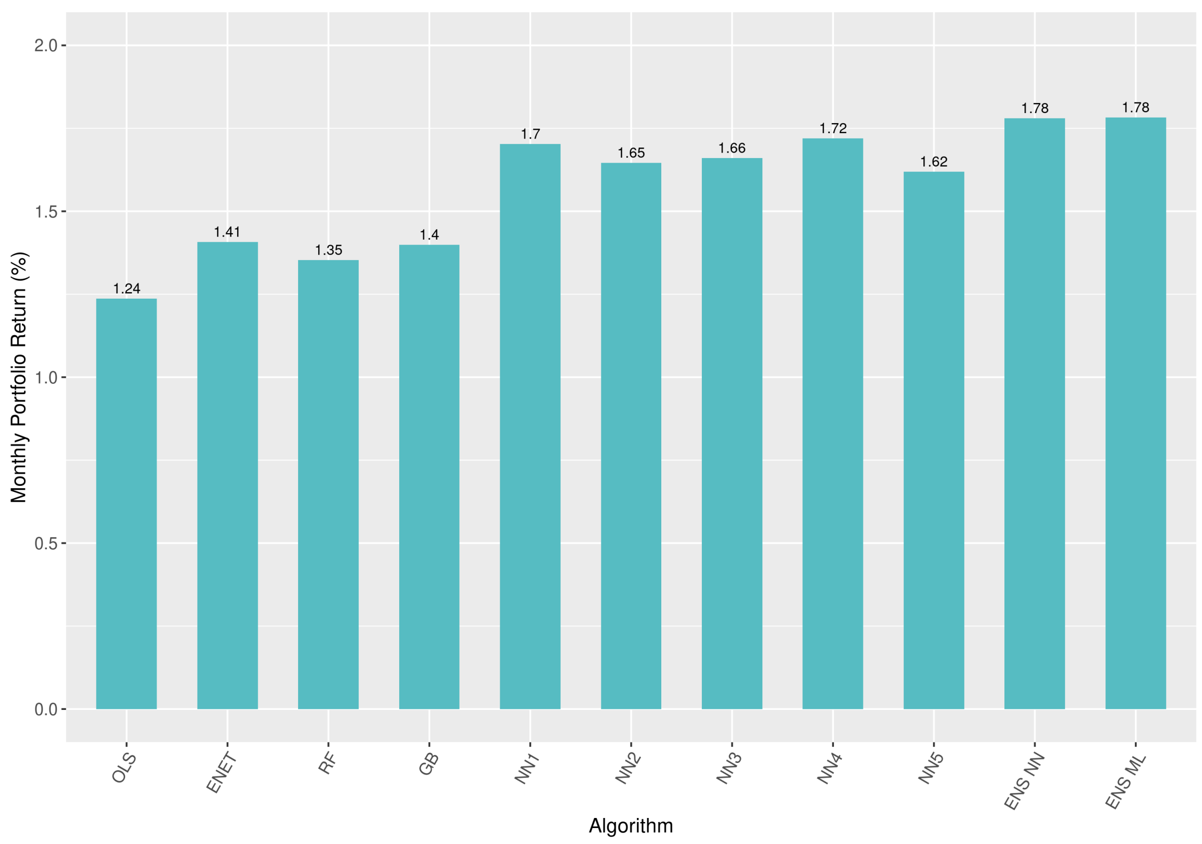
Description: This table documents the hyperparameters for each machine learning model. The hyperparameters are based on the range in Gu et al. (2020), Tobek and Hronec (2021), and Hanauer and Kalsbach (2023).

Table B.2: Top and bottom performance models

	Design choices							Portfolio performance		
Rank	Algorithm	Target Variable	Target Transform	Post Publication	Feature Selection	Training Window	Training Sample	Avg	Std	SR
Panel A: Top 40 models										
1	ENS ML	RET-MKT	Raw	NoPost	NoSelect	Expanding	ExMicro	1.98	4.14	1.65
2	ENS NN	RET-MKT	Raw	NoPost	YesSelect	Expanding	ExMicro	1.94	4.03	1.67
3	NN2	RET-MKT	Raw	NoPost	YesSelect	Expanding	ExMicro	1.93	4.07	1.65
4	ENS ML	RET-MKT	Raw	NoPost	YesSelect	Expanding	ExMicro	1.92	4.39	1.51
5	ENS NN	RET-MKT	Raw	NoPost	NoSelect	Expanding	ExMicro	1.91	3.84	1.72
6	NN5	RET-MKT	Raw	NoPost	NoSelect	Expanding	ExMicro	1.90	3.62	1.82
7	NN4	RET-MKT	Raw	NoPost	YesSelect	Expanding	ExMicro	1.88	4.22	1.54
8	NN1	RET-MKT	Raw	NoPost	YesSelect	Expanding	ExMicro	1.87	3.92	1.65
9	NN5	RET-MKT	Raw	NoPost	YesSelect	Expanding	ExMicro	1.86	4.13	1.56
10	ENS ML	RET-RF	Raw	NoPost	YesSelect	Expanding	ExMicro	1.82	4.31	1.46
11	NN3	RET-MKT	Raw	NoPost	NoSelect	Expanding	ExMicro	1.82	3.88	1.62
12	NN3	RET-MKT	Raw	NoPost	YesSelect	Expanding	ExMicro	1.82	4.03	1.56
13	NN2	RET-MKT	Raw	NoPost	NoSelect	Expanding	ExMicro	1.81	3.92	1.60
14	NN1	RET-MKT	Raw	NoPost	NoSelect	Expanding	ExMicro	1.80	3.83	1.63
15	ENS ML	RET-RF	Raw	NoPost	NoSelect	Expanding	ExMicro	1.78	4.41	1.40
16	ENS NN	RET-RF	Raw	NoPost	NoSelect	Expanding	ExMicro	1.78	3.66	1.68
17	NN1	RET-MKT	Raw	NoPost	NoSelect	Rolling	All	1.77	4.37	1.40
18	NN1	RET-CAPM	Raw	NoPost	NoSelect	Expanding	ExMicro	1.75	4.06	1.49
19	ENS ML	RET-MKT	Dummy	NoPost	NoSelect	Expanding	ExMicro	1.74	4.01	1.50
20	GB	RET-MKT	Raw	NoPost	NoSelect	Expanding	ExMicro	1.74	3.85	1.56
21	ENS ML	RET-CAPM	Raw	NoPost	NoSelect	Expanding	ExMicro	1.73	4.69	1.28
22	NN4	RET-RF	Raw	NoPost	NoSelect	Expanding	ExMicro	1.72	3.61	1.65
23	NN4	RET-MKT	Raw	NoPost	NoSelect	Expanding	ExMicro	1.71	3.89	1.52
24	NN1	RET-RF	Raw	NoPost	NoSelect	Expanding	ExMicro	1.70	3.85	1.53
25	ENS NN	RET-CAPM	Raw	NoPost	YesSelect	Expanding	ExMicro	1.69	4.08	1.44
26	NN3	RET-RF	Raw	NoPost	YesSelect	Expanding	ExMicro	1.69	3.79	1.54
27	ENS NN	RET-CAPM	Raw	NoPost	NoSelect	Expanding	ExMicro	1.68	4.08	1.43
28	GB	RET-MKT	Dummy	NoPost	NoSelect	Expanding	All	1.68	4.01	1.45
29	GB	RET-MKT	Dummy	NoPost	NoSelect	Expanding	ExMicro	1.67	4.01	1.44
30	NN5	RET-CAPM	Raw	NoPost	YesSelect	Expanding	ExMicro	1.66	4.12	1.40
31	NN1	RET-RF	Raw	NoPost	YesSelect	Rolling	All	1.66	4.61	1.25
32	NN3	RET-RF	Raw	NoPost	NoSelect	Expanding	ExMicro	1.66	3.56	1.62
33	ENS ML	RET-MKT	Dummy	NoPost	NoSelect	Expanding	All	1.65	4.24	1.35
34	NN1	RET-CAPM	Raw	NoPost	YesSelect	Rolling	All	1.65	4.93	1.16
35	ENS ML	RET-CAPM	Raw	NoPost	YesSelect	Expanding	ExMicro	1.65	4.80	1.19
36	NN2	RET-CAPM	Raw	NoPost	NoSelect	Expanding	ExMicro	1.65	4.08	1.40
37	NN2	RET-RF	Raw	NoPost	NoSelect	Expanding	ExMicro	1.65	3.64	1.57
38	NN2	RET-CAPM	Raw	NoPost	YesSelect	Expanding	ExMicro	1.65	4.19	1.36
39	ENS NN	RET-MKT	Raw	NoPost	YesSelect	Expanding	All	1.64	4.24	1.34
40	NN3	RET-CAPM	Raw	NoPost	YesSelect	Expanding	ExMicro	1.64	4.13	1.38
Panel B: Bottom 40 models										
1017	NN3	RET-RF	Dummy	YesPost	NoSelect	Rolling	ExMicro	0.43	4.79	0.31
1018	ENS ML	RET-RF	Dummy	YesPost	NoSelect	Rolling	ExMicro	0.43	6.20	0.24
1019	GB	RET-RF	Dummy	YesPost	NoSelect	Rolling	ExMicro	0.43	5.70	0.26
1020	ENS ML	RET-CAPM	Dummy	YesPost	NoSelect	Rolling	ExMicro	0.42	6.04	0.24
1021	GB	RET-RF	Raw	YesPost	NoSelect	Rolling	All	0.41	4.57	0.31
1022	RF	RET-RF	Raw	YesPost	YesSelect	Rolling	ExMicro	0.41	4.41	0.32
1023	ENET	RET-RF	Dummy	YesPost	YesSelect	Rolling	ExMicro	0.41	5.56	0.25
1024	RF	RET-CAPM	Dummy	YesPost	YesSelect	Rolling	ExMicro	0.40	6.15	0.23
1025	ENS ML	RET-RF	Dummy	YesPost	YesSelect	Rolling	ExMicro	0.40	5.97	0.23
1026	ENET	RET-CAPM	Dummy	YesPost	NoSelect	Expanding	ExMicro	0.40	5.76	0.24
1027	RF	RET-RF	Raw	NoPost	YesSelect	Rolling	All	0.39	4.69	0.28
1028	RF	RET-RF	Raw	YesPost	YesSelect	Rolling	All	0.39	4.69	0.28
1029	RF	RET-CAPM	Raw	NoPost	NoSelect	Rolling	All	0.39	5.33	0.25
1030	GB	RET-RF	Dummy	YesPost	YesSelect	Rolling	ExMicro	0.37	5.72	0.22
1031	NN1	RET-RF	Dummy	YesPost	NoSelect	Rolling	ExMicro	0.36	5.73	0.22
1032	RF	RET-RF	Dummy	YesPost	YesSelect	Rolling	ExMicro	0.36	6.24	0.20
1033	RF	RET-CAPM	Dummy	YesPost	NoSelect	Rolling	ExMicro	0.35	6.25	0.19
1034	NN2	RET-MKT	Dummy	YesPost	YesSelect	Rolling	ExMicro	0.34	4.56	0.26
1035	RF	RET-CAPM	Raw	YesPost	YesSelect	Expanding	All	0.34	4.50	0.26
1036	GB	RET-RF	Raw	YesPost	YesSelect	Rolling	ExMicro	0.34	4.33	0.27
1037	ENET	RET-RF	Dummy	YesPost	NoSelect	Rolling	ExMicro	0.33	6.10	0.19
1038	RF	RET-MKT	Raw	YesPost	YesSelect	Rolling	ExMicro	0.33	4.63	0.25
1039	ENET	RET-MKT	Dummy	YesPost	NoSelect	Rolling	ExMicro	0.32	4.25	0.26
1040	ENET	RET-RF	Dummy	YesPost	NoSelect	Expanding	ExMicro	0.32	5.70	0.20
1041	RF	RET-MKT	Raw	YesPost	NoSelect	Rolling	ExMicro	0.32	4.82	0.23
1042	RF	RET-MKT	Raw	NoPost	NoSelect	Rolling	All	0.31	4.72	0.23
1043	RF	RET-MKT	Raw	YesPost	NoSelect	Rolling	All	0.31	4.72	0.23
1044	RF	RET-RF	Dummy	YesPost	NoSelect	Rolling	ExMicro	0.31	6.37	0.17
1045	RF	RET-RF	Raw	YesPost	NoSelect	Rolling	ExMicro	0.28	4.43	0.22
1046	RF	RET-CAPM	Raw	YesPost	NoSelect	Rolling	ExMicro	0.28	5.11	0.19
1047	RF	RET-CAPM	Raw	YesPost	YesSelect	Rolling	ExMicro	0.27	5.19	0.18
1048	RF	RET-CAPM	Raw	YesPost	NoSelect	Expanding	All	0.25	4.49	0.19
1049	ENET	RET-CAPM	Dummy	YesPost	NoSelect	Rolling	ExMicro	0.22	5.79	0.13
1050	ENET	RET-CAPM	Dummy	YesPost	YesSelect	Rolling	ExMicro	0.22	6.02	0.13
1051	RF	RET-RF	Raw	NoPost	NoSelect	Rolling	All	0.21	4.59	0.16
1052	RF	RET-RF	Raw	YesPost	NoSelect	Rolling	All	0.21	4.59	0.16
1053	GB	RET-RF	Raw	YesPost	NoSelect	Rolling	ExMicro	0.19	4.09	0.16
1054	RF	RET-CAPM	Raw	YesPost	NoSelect	Rolling	All	0.17	5.33	0.11
1055	RF	RET-CAPM	Raw	NoPost	YesSelect	Rolling	All	0.14	5.09	0.10
1056	RF	RET-CAPM	Raw	YesPost	YesSelect	Rolling	All	0.13	5.33	0.08

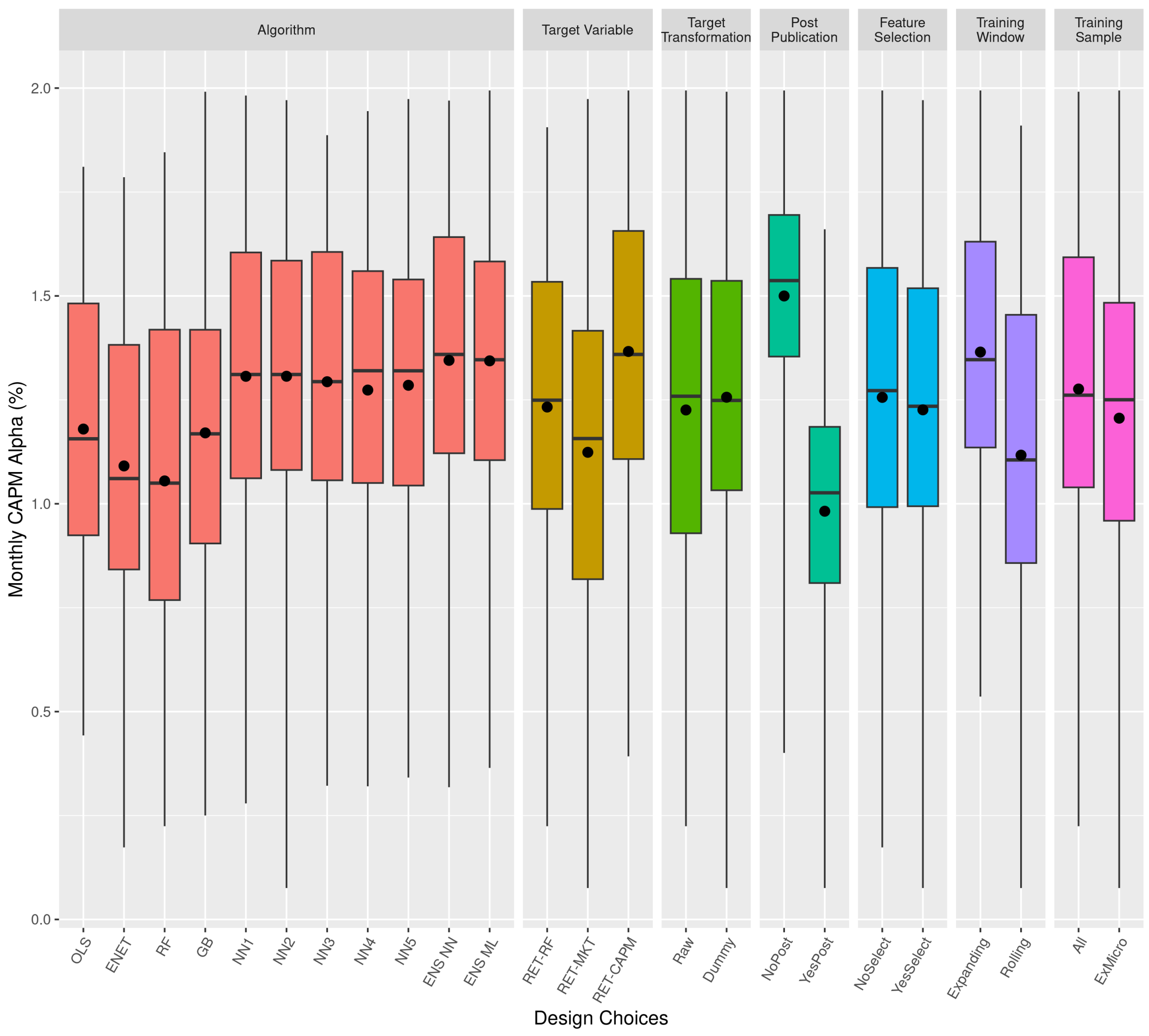
Description: This table documents the top 40 and bottom 40 machine learning models based on average monthly returns. Column 1 displays the rank. Columns 2-8 report the set of design choices. Columns 9-13 report the average monthly return (in %), monthly standard deviation (in %), and annualized Sharpe ratio.

Figure B.1: Baseline model performance



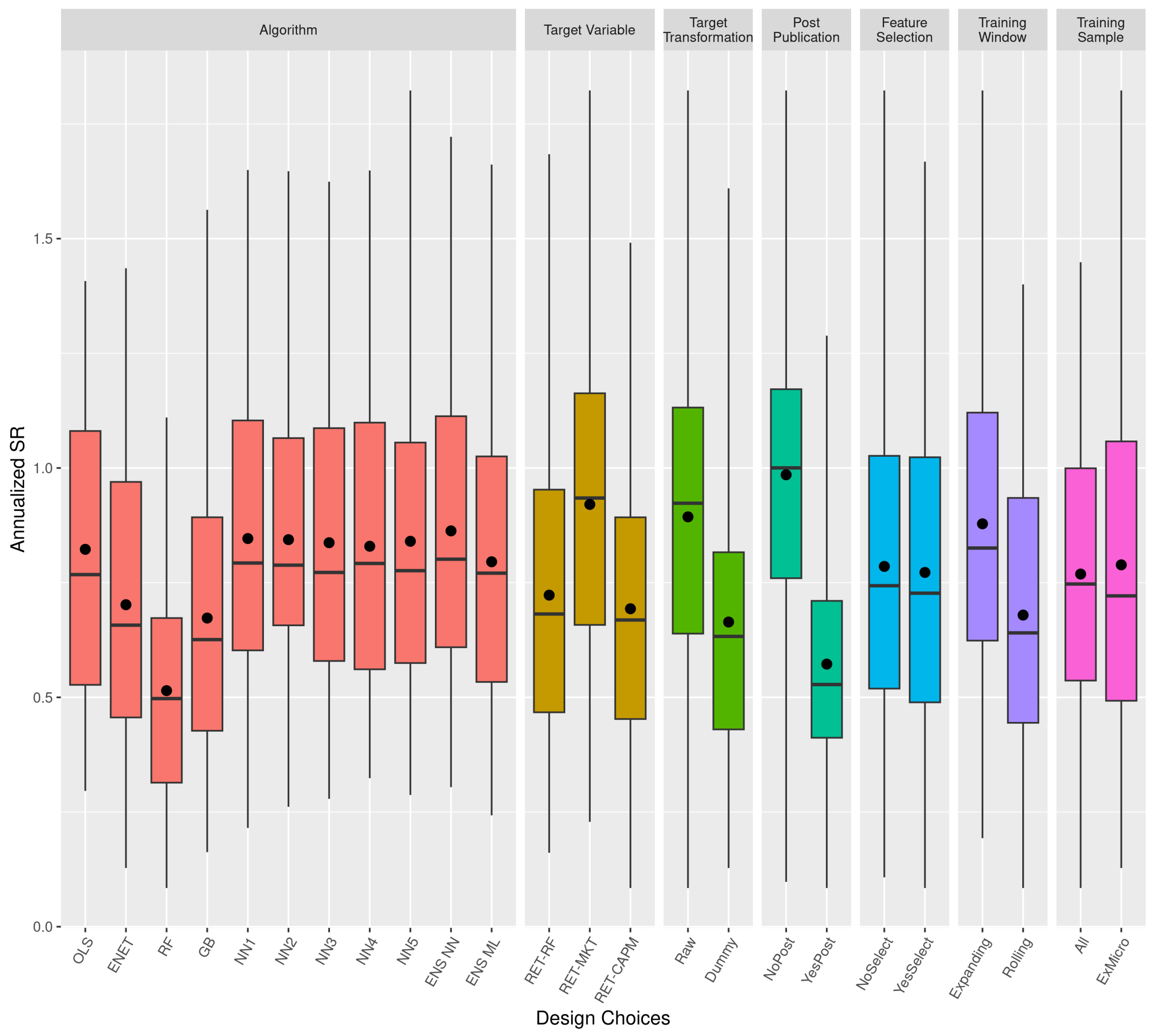
Description: This figure illustrates the performance of the baseline model, which employs the most common design choice of Target (RET-RF, RAW), Feature (No Post Publication, No Feature Selection), and Training (Expanding Window, ExMicro Training Sample). The results align closely with those reported by Gu et al. (2020), Tobek and Hronec (2021), and Hanauer and Kalsbach (2023). The analysis period comprises January 1987 to December 2021.

Figure B.2: Portfolio CAPM alpha by research design choices



Description: This figure compares the portfolio CAPM alpha per design choice in box plot. We consider seven distinct research design choices: algorithm, target variable, target transformation, post publication, feature selection, training window, and training sample. The middle black lines within each box indicate the median CAPM alpha. The black dot within each box indicates the mean CAPM alpha. The lower and upper hinges correspond to the first and third quartiles. The sample period comprises January 1987 to December 2021.

Figure B.3: Portfolio SR by research design choices



Description: This figure compares the portfolio Sharpe ratio per design choice in box plot. We consider seven distinct research design choices: algorithm, target variable, target transformation, post publication, feature selection, training window, and training sample. The middle black lines within each box indicate the median Sharpe ratio. The black dot within each box indicates the mean Sharpe ratio. The lower and upper hinges correspond to the first and third quartiles. The sample period comprises January 1987 to December 2021.

References

Arnott, R.D., Kalesnik, V., Linnainmaa, J.T., 2023. Factor momentum. The Review of Financial Studies 36, 3034–3070.

Asness, C.S., Moskowitz, T.J., Pedersen, L.H., 2013. Value and momentum everywhere. The Journal of Finance 68, 929–985.

Avramov, D., Cheng, S., Metzker, L., 2023. Machine learning vs. economic restrictions: Evidence from stock return predictability. Management Science 69, 2587–2619.

Azevedo, V., Kaiser, G.S., Mueller, S., 2023. Stock market anomalies and machine learning across the globe. Journal of Asset Management 24, 419–441.

Bali, T.G., Kelly, B.T., Mörke, M., Rahman, J., 2024. Machine forecast disagreement. NBER Working Paper no. 31583 .

Blitz, D., Hanauer, M.X., Hoogteijling, T., Howard, C., 2023. The term structure of machine learning alpha. Journal of Financial Data Science 5, 40–65.

Breiman, L., 2001. Random forests. Machine Learning 45, 5–32.

Bryzgalova, S., Lerner, S., Lettau, M., Pelger, M., 2024. Missing financial data. The Review of Financial Studies 38, 803–882.

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

Carhart, M.M., 1997. On persistence in mutual fund performance. The Journal of Finance 52, 57–82.

Chen, A.Y., Zimmermann, T., 2022. Open source cross-sectional asset pricing. Critical Finance Review 27, 207–264.

Chen, L., Pelger, M., Zhu, J., 2024. Deep learning in asset pricing. Management Science 70, 714–750.

Cochrane, J.H., 2011. Presidential address: Discount rates. The Journal of Finance 66, 1047–1108.

Dudda, T.L., Hornuf, L., 2024. The perks and perils of machine learning in business and economic research. SSRN Working Paper no. 4981802 .

Fama, E.F., French, K.R., 2015. A five-factor asset pricing model. Journal of Financial Economics 116, 1–22.

Fieberg, C., Günther, S., Poddig, T., Zaremba, A., 2024. Non-standard errors in the cryptocurrency world. International Review of Financial Analysis 92, 103106.

Freyberger, J., Hoeppner, B., Neuhierl, A., Weber, M., 2024. Missing data in asset pricing panels. The Review of Financial Studies 38, 760–802.

Freyberger, J., Neuhierl, A., Weber, M., 2020. Dissecting Characteristics Nonparametrically. The Review of Financial Studies 33, 2326–2377.

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

Gupta, T., Kelly, B., 2019. Factor momentum everywhere. The Journal of Portfolio Management 45, 13–36.

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

Hanauer, M.X., Kononova, M., Rapp, M.S., 2022. Boosting agnostic fundamental analysis: Using machine learning to identify mispricing in European stock markets. Finance Research Letters 48, 102856.

Hansen, L., Salamon, P., 1990. Neural network ensembles. IEEE Transactions on Pattern Analysis and Machine Intelligence 12, 993–1001.

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

Hou, K., Xue, C., Zhang, L., 2020. Replicating anomalies. The Review of Financial Studies 33, 2019–2133.

Howard, C., 2024. Choices matter when training machine learning models for return prediction. Financial Analysts Journal 80, 81–107.

Ince, O.S., Porter, R.B., 2006. Individual equity return data from Thomson Datastream: Handle with care! Journal of Financial Research 29, 463–479.

Ioffe, S., Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. eprint arXiv:1502.03167 , 1–11.

Jegadeesh, N., Titman, S., 1993. Returns to buying winners and selling losers: Implications for stock market efficiency. The Journal of Finance 48, 65–91.

Jiang, J., Kelly, B., Xiu, D., 2023. (re-)imag(in)ing price trends. The Journal of Finance 78, 3193–3249.

Kingma, D.P., Ba, J., 2017. Adam: A method for stochastic optimization. eprint arXiv:1412.6980 , 1–15.

Lalwani, V., Meshram, V., Jindal, V., 2024. Empirical asset pricing via machine learning: The role of methodological choices. SSRN Working Paper no. 4837337 .

Leippold, M., Wang, Q., Zhou, W., 2022. Machine learning in the chinese stock market. Journal of Financial Economics 145, 64–82.

Lundberg, S.M., Lee, S.I., 2017. A unified approach to interpreting model predictions. Advances in Neural Information Processing Systems 30, 4768–4777.

Masters, T., 1993. Practical neural network recipies in C++. Academic Press Inc. .

McLean, R.D., Pontiff, J., 2016. Does academic research destroy stock return predictability? The Journal of Finance 71, 5–32.

Menkveld, A.J., Dreber, A., Holzmeister, F., Huber, J., Johanneson, M., Kirchler, M., Razen, M., Weitzel, U., ..., 2024. Nonstandard errors. The Journal of Finance 79, 2339–2390.

Moskowitz, T.J., Grinblatt, M., 1999. Do industries explain momentum? The Journal of Finance 54, 1249–1290.

Newey, W.K., West, K.D., 1987. A simple, positive semi-definite, heteroskedasticity and autocorrelation consistent covariance matrix. Econometrica 55, 703–708.

Rasekhschaffe, K.C., Jones, R.C., 2019. Machine learning for stock selection. Financial Analysts Journal 75, 70–88.

Rouwenhorst, K.G., 1998. International momentum strategies. The Journal of Finance 53, 267–284.

Shapley, L.S., 1953. A value for n-person games, in: Kuhn, H.W., Tucker, A.W. (Eds.), Contributions to the Theory of Games, Volume II. Princeton University Press, Princeton. number 28 in Annals of Mathematics Studies. chapter 17, pp. 307–318.

Soebhag, A., Van Vliet, B., Verwijmeren, P., 2024. Non-standard errors in asset pricing: Mind your sorts. Journal of Empirical Finance 78, 101517.

Swade, A., Hanauer, M.X., Lohre, H., Blitz, D., 2024. Factor zoo (.zip). Journal of Portfolio Management 50, 11–31.

Timmermann, A., 2006. Forecast combinations, in: Elliott, G., Granger, C., Timmermann, A. (Eds.), Handbook of Economic Forecasting. Elsevier. volume 1. chapter 4, pp. 135–196.

Tobek, O., Hronec, M., 2021. Does it pay to follow anomalies research? machine learning approach with international evidence. Journal of Financial Markets 56, 100588.

Walter, D., Weber, R., Weiss, P., 2024. Methodological uncertainty in portfolio sorts. SSRN Working Paper no. 4164117 .

Wooldridge, J.M., 2001. Applications of generalized method of moments estimation. Journal of Economic Perspectives 15, 87–100.