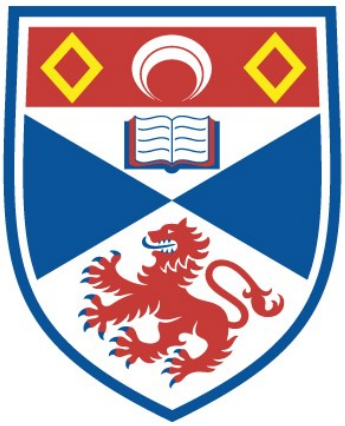
Financial market predictability with artificial intelligence and machine learning techniques

Lazaros Zografopoulos

A thesis submitted for the degree of PhD
at the
University of St Andrews



2025

Identifier to use to cite or link to this thesis:

DOI: https://doi.org/10.17630/sta/1211

ABSTRACT

This thesis explores the intersection of financial markets and machine learning, focusing on financial return predictability and the imputation of missing values in financial datasets. The research aims to enhance our understanding of financial market dynamics through the lens of interpretable machine learning models. Specifically, the thesis employs advanced machine learning techniques to predict financial returns and address missing data issues, which are common but often overlooked in financial literature.

The first chapter uses an interpretable machine learning model, LassoNet, to forecast U.S. industry portfolio returns. LassoNet combines a regularization mechanism with a neural network architecture to enforce covariate sparsity. The findings show that LassoNet outperforms linear and non-linear models in forecasting accuracy, with valuation ratios and individual and cross-industry lagged returns being the most critical covariates. The model's forecasts enable the construction of profitable industry ETF portfolios that outperform benchmarks in annualized returns, Sharpe ratios, and alpha values.

The second chapter focuses on imputing missing hedge fund return data using a deep learning model, the bidirectional recurrent imputation network for time series (BRITS). BRITS is compared to other imputation methods like the cross-sectional mean and matrix completion. The results indicate that BRITS significantly enhances forecasting accuracy and economic performance of predictive models when used to impute the missing values in the data. The imputed data leads to lower out-of-sample errors and higher investment returns, demonstrating BRITS' superiority in handling missing values.

In the third chapter, the state-of-the-art neural network architecture TabNet is utilized to forecast the directional movements of excess returns in industry portfolios. TabNet surpasses other models in classification accuracy and highlights the importance of valuation ratios and lagged returns in its predictions. The model effectively captures seasonal effects and cross-industry economic links and attains the highest annualized returns and positive Sharpe ratios in trading applications.

I dedicate this thesis to my mother, Anta Ptochopoulou,

my brother, Dr. Ioannis Zografopoulos,

and to my late father, Fotis Zografopoulos, who never saw this adventure

ACKNOWLEDGEMENTS

First and foremost, I would like to express my deepest gratitude to my supervisor Dr. Ioannis Psaradellis. Your guidance and unwavering support have been invaluable. Your expertise and constructive comments have greatly enhanced the quality of my work. I am also thankful to Dr. Maria Chiara Iannino for her insightful feedback and support throughout my Ph.D. studies. I would like to thank the School of Economics and Finance for fostering a supportive and dynamic research environment.

I am also grateful to Sofoklis Achillopoulos Foundation for providing me with a scholarship for the third and fourth years of the Ph.D. programme. This scholarship has been crucial in completing my studies and producing high-quality research. I would also like to thank the Clelia Hajiioannou Foundation for generously offering me a scholarship during my second year of studies.

Many thanks to my colleagues and fellow students in the School of Economics and Finance, with whom I enjoyed my time in St Andrews and had interesting discussions. Ciarán Mac Domhnaill, a special thanks to you for all your support during these years, which is deeply appreciated.

Finally, I am grateful to my family for encouraging me to pursue the Ph.D. programme and for their unconditional support during this challenging journey. My brother, Dr. Ioannis Zografopoulos, has inspired me to pursue this degree, and his achievements and resilience serve as the greatest source of motivation to continue improving my skills and realize my full potential as a researcher and financial industry professional. Last but not least, I would like to thank my childhood friend Dimitris Zotos for his support and ability to make me smile even during the most challenging periods.

Candidate's declaration

I, Lazaros Zografopoulos, do hereby certify that this thesis, submitted for the degree of PhD, which is approximately 40,000 words in length, has been written by me, and that it is the record of work carried out by me, or principally by myself in collaboration with others as acknowledged, and that it has not been submitted in any previous application for any degree. I confirm that any appendices included in my thesis contain only material permitted by the 'Assessment of Postgraduate Research Students' policy.

I was admitted as a research student at the University of St Andrews in August 2020.

I received funding from an organisation or institution and have acknowledged the funder(s) in the full text of my thesis.

Date Signature of candidate

21/10/2024

Supervisor's declaration

I hereby certify that the candidate has fulfilled the conditions of the Resolution and Regulations appropriate for the degree of PhD in the University of St Andrews and that the candidate is qualified to submit this thesis in application for that degree. I confirm that any appendices included in the thesis contain only material permitted by the 'Assessment of Postgraduate Research Students' policy.

Date Signature of supervisor

21/10/2024

Permission for publication

In submitting this thesis to the University of St Andrews we understand that we are giving permission for it to be made available for use in accordance with the regulations of the University Library for the time being in force, subject to any copyright vested in the work not being affected thereby. We also understand, unless exempt by an award of an embargo as requested below, that the title and the abstract will be published, and that a copy of the work may be made and supplied to any bona fide library or research worker, that this thesis will be electronically accessible for personal or research use and that the library has the right to migrate this thesis into new electronic forms as required to ensure continued access to the thesis.

I, Lazaros Zografopoulos, confirm that my thesis does not contain any third-party material that requires copyright clearance.

The following is an agreed request by candidate and supervisor regarding the publication of this thesis:

Printed copy

No embargo on print copy.

Electronic copy

No embargo on electronic copy.

Date Signature of candidate

21/10/2024

Date Signature of supervisor

21/10/2024

Underpinning Research Data or Digital Outputs

Candidate's declaration

I, Lazaros Zografopoulos, hereby certify that no requirements to deposit original research data or digital outputs apply to this thesis and that, where appropriate, secondary data used have been referenced in the full text of my thesis.

Date Signature of candidate

21/10/2024

FUNDING

This thesis was generously supported by the Clelia Hajiioannou Foundation Scholarship for Postgraduate Studies (academic year 2021-2022), and Sofoklis Achillopoulos Foundation Scholarship for Postgraduate Studies (academic years 2022-2023, and 2023-2024).

PUBLICATIONS

The first chapter of my Ph.D. thesis has been published to the European Journal of Operational Research [doi: 10.1016/j.ejor.2024.08.032].

Table of Contents

LIST OF TABLES

LIST OF FIGURES

INTRODUCTION

In recent years, machine learning has become a transformative force in the financial industry, revolutionizing areas such as quantitative analysis, investment management, customer servicing, and fraud prevention. According to a 2022 survey by the Bank of England, over 72% of financial services firms report using or developing machine learning applications, highlighting the pervasive integration of this technology. The ability of machine learning algorithms to process vast amounts of data efficiently and extract meaningful patterns is well suited to the data-rich environments in which financial institutions operate. Expectedly, major market players (see J.P. Morgan, Goldman Sachs, Fidelity, CITADEL) have incorporated machine learning into their day-to-day operations to optimize their processes. The widespread integration of machine learning in algorithmic trading, investment management (e.g., Fidelity's robo-advisors), automated customer service systems (e.g., chatbots), and fraud detection proves that these innovative algorithms have revolutionized how financial services are delivered.

While the impact of machine learning on the financial industry is profound, it is essential, from a financial literature perspective, to assess whether these techniques can advance our understanding of financial markets and improve our ability to predict asset price fluctuations. Traditional econometric models, though well-established, often fail to capture the non-linear dependencies and complex patterns inherent in financial data. The difficulty of predicting financial returns has been a long-standing focus in financial literature. Timmerman and Granger (2004) argued that the Efficient Market Hypothesis (Fama, 1970) presents a significant challenge for forecasters, as its most basic interpretation suggests that financial returns cannot be predicted. Nonetheless, several financial studies provide evidence that challenges the notion of market efficiency. For instance, Jacobs (2015) provides an overview of 100 market anomalies, and Asness et al. (2013) report consistent value and momentum return premia across eight diverse markets and asset classes. Indeed, Psaradellis et al. (2018) state that since the 1960s, academics and practitioners have claimed that predictable patterns exist in financial assets' returns, which sparked a fruitful debate. However, modeling financial returns and extracting predictable patterns from data exhibiting non-linearities is challenging for linear models.

Rasekhschaffe and Jones (2019) support that machine learning techniques may provide a better approach than linear models in predicting asset prices. They define machine learning as an umbrella term for methods that allow machines to uncover patterns without explicit programming instructions, and thus, modelers can supply a variety of factors that might help in forecasting future returns and use machine learning algorithms to learn which factors are the most informative. Gu et al. (2020) further validate the above claims and indicate that machine learning models offer an improved description of

the expected return behavior relative to traditional forecasting methods. This thesis seeks to contribute to the growing body of literature on the intersection of machine learning and finance by investigating the predictability of financial markets through interpretable machine learning models. Until recently, machine learning techniques were used for financial applications as "black boxes" that offer little to no insight into how predictions are made (see Feng et al., 2018; Bartram et al., 2020; Gu et al., 2020; Bussmann et al., 2021). The research is motivated by the need for models that improve prediction accuracy and offer transparency, enabling academic researchers and financial professionals to understand and trust the models generating the predictions. Furthermore, this thesis seeks to leverage the flexibility of machine learning techniques for tasks such as missing value imputation, an area often overlooked in financial literature. Many studies either fail to address missing data or handle it superficially, resulting in biased estimates and flawed inferences (Freyberger et al., 2021).

The primary contribution of this thesis is threefold. First, it evaluates whether machine learning offers a promising alternative to traditional models in forecasting financial returns and making informed investment decisions. Second, it emphasizes the importance of interpretability in these models. Interpretable modeling frameworks are essential for forecasting financial returns, high-stakes decisions, and investment management. From a financial literature standpoint, it is crucial to investigate which variables are the most informative and best explain asset price movements. Model transparency is not optional but necessary to advance our understanding of financial markets. At the same time, the financial industry requires the adoption of high-performing yet interpretable models. Interpretable models highlight influential variables, helping portfolio managers make informed decisions aligned with their expertise and ensuring strategies follow the intended investment philosophy. Unlike many black-box machine learning models that sacrifice transparency for accuracy, this work ensures that the models used are both interpretable and high-performing. Third, the thesis explores applications beyond return forecasting, such as missing value imputation and directional return forecasting, areas that are crucial but underexplored in financial research. Missing values in financial datasets affect both model fit and post-estimation inference, making effective imputation essential for asset pricing and predictive tasks. Our research highlights the advantages of machine learning methods for missing value imputation. Additionally, we examine the performance of machine learning models in predicting directional movements of financial returns. Directional forecasting, which focuses on classifying the sign of the next period's return, is an alternative to predicting return levels. Since machine learning models excel at classification tasks (LeCun et al., 2015), we investigate whether their classification accuracy holds in directional forecasting and whether these forecasts are profitable within a trading strategy.

The methodological approach adopted in this thesis spans several cutting-edge machine learning frameworks. The first chapter explores the prediction of U.S. industry portfolio returns using an interpretable deep learning framework. Specifically, we use the LassoNet neural network method of Lemhadri et al. (2021) to forecast U.S. industry portfolio returns by extracting information from a large-scale dataset of multiple predictors. Our objective is to uncover the factors that lead our forecasts based on the Shapley additive global importance (SAGE) interpretability method (see Covert et al., 2020; Covert & Lee, 2021). We leverage the two state-of-art approaches in a two-step process. First, we apply the LassoNet model, which imposes sparsity and allows only a selected subset of a large pool of covariate variables to drive its forecasts endogenously, thus providing more accurate predictions. On that level, we provide evidence that jointly optimizing for input variable selection and minimizing the loss function, as LassoNet performs it, generates far superior accuracy compared to using a linear model to select a subset of the covariates as a first step, and then input this subset into a neural network as a second independent step. Hence, unlike other neural network architectures, LassoNet can capture data non-linearities without sacrificing interpretability for performance. Second, to shed light on the individual predictability importance of each covariate selected by the LassoNet and make our model transparent, we feed our post-LassoNet findings to SAGE. The SAGE method uses a cooperative game theoretic framework to identify the predictors contributing most to reducing out-of-sample (OOS) LassoNet's error. Therefore, SAGE reveals which covariates are the most influential in increasing the predictability of the forecasting model. We apply the above two-step approach in forecasting and trading 10 U.S. industry portfolio returns based on 88 predictors over the 2000 – 2019 OOS period. We compare our results with those obtained by linear (i.e., linear regression) and several non-linear machine learning models (i.e., Group Lasso, Elastic-Net, and two neural network specifications). Finally, we assess the economic significance of LassoNet forecasts by forming long-short portfolios of corresponding industry ETFs based on the forecasted returns on industry portfolios.

The main findings reveal that LassoNet attains superior OOS performance against all benchmark techniques employed and, therefore, exhibits superior capacity in predicting industry portfolios' returns. Importantly, LassoNet achieves the smallest forecasting error compared to other powerful machine learning models, such as the XGBoost and standard neural networks. Multiple statistical tests also validate those findings. The SAGE method reveals the importance of valuation ratios and individual and cross-industry lagged industry returns to our predictions against other profitability, liquidity, and efficiency ratios covariates. The covariates belonging to these categories of predictors present the highest SAGE values and have higher explanatory power on industry portfolios' returns. Thus, we provide new evidence that valuation ratios and individual and cross-industry lagged industry returns are significant determinants of expected returns of stocks and portfolios of stocks. Regarding the

economic significance of the proposed model, the portfolio of ETFs constructed on LassoNet's industry predictions outperforms several benchmarks, attains statistically significant alphas, and its profitability survives transaction costs. Our modeling framework can guide portfolio construction and investment decisions, particularly when deciding asset allocations in specific industries. Policymakers and central banks can also benefit from the flexibility of our proposed framework, which can be used to effectively forecast any economic variable, such as inflation, unemployment, and exchange rates. The model's transparent nature enhances the reliability of predictions and reveals the most pivotal variables for a given forecast, which can assist in making informed policy design decisions.

In the second chapter, we focus on the issue of missing values and use a deep learning model that considers both time-series and cross-sectional properties to impute missing entries. We apply the BRITS neural network architecture (Cao et al., 2018) to recover missing entries of hedge fund returns and a large pool of hedge fund predictors. Hedge fund datasets possess a large number of missing values and are, therefore, the ideal data setting to assess the efficacy of the adopted imputation model. Naturally, the issue of missing values is a common problem affecting most financial datasets. Four fundamental properties distinguish BRITS from other techniques that can be used to recover missing observations. First, BRITS imputes a missing value for month t by accounting for both the past as well as future values of the time series. Second, BRITS uses information from the entire cross-section when imputing missing values for each hedge fund. Third, a novel component of the neural network architecture enables BRITS to account for the decaying importance of past information since recent past observations should have more effect on future values than those of the far past. Fourth, since BRITS is a neural network architecture, it can effectively model data characterized by non-linearities, as is often the case with financial datasets. We examine 3,800 hedge funds from January 1994 to November 2021 to conduct our analysis. Our dataset consists of time series for the hedge fund returns, while we also construct a large dataset of 23 fund predictors. To assess imputation performance, we develop a simulation study in which we artificially drop 10% and 20% of observed data. We then measure the error between the observed values, their imputations generated by BRITS, and the benchmarks to evaluate each method's imputation fidelity. As benchmark techniques for the simulation study, we include the cross-sectional mean, the time series mean, and a matrix completion method, singular value thresholding for nuclear norm minimization. We also include a forecasting and trading application to quantify the utility of imputing missing values. We train several machine-learning models on the fully recovered datasets generated by BRITS and the second-best performing imputation technique. The forecasts are then used to create a trading strategy in which we form long, equally-weighted portfolios for the top decile of hedge funds according to the models' predictions.

Our findings highlight that the BRITS method attains the highest imputation performance against all other techniques, including the cross-sectional mean. BRITS achieves the lowest imputation error for the hedge fund returns dataset and most of the predictors' datasets. Furthermore, we observe higher OOS forecasting accuracy across the machine learning models estimated on predictors' datasets imputed by BRITS, contrary to forecasts using cross-sectional imputations of predictors as inputs. We highlight that this result is consistent across all machine learning techniques. The insights we gain from the trading application, again, favor BRITS. In more detail, the forecasts of the machine learning models, which were trained on BRITS imputed datasets, are associated with more profitable trading decisions as measured by higher portfolio annualized returns, Sharpe ratios, and alphas. We can conclude that an effective imputation method, such as BRITS, can provide an information advantage and assist models in producing more accurate and profitable predictions. Academic researchers, financial practitioners, and policymakers can benefit from BRITS effectiveness in recovering missing entries from financial datasets and, therefore, assist their forecasting models in making more accurate and economically meaningful predictions.

In the third chapter, we use TabNet (Arik and Pfister, 2021), a state-of-the-art, interpretable neural network, to predict the directional movements of industries' excess returns. Directional movement prediction is a classification task, focusing on the sign (positive or negative) of the next period's return. TabNet has three key advantages over traditional models like logistic regression: First, it allows for sparse covariate selection and quantifies the importance of each covariate, making it a "white box" model that addresses the typical lack of transparency in neural networks. Second, it is parameter-efficient, focusing learning capacity on the most relevant covariates. Third, TabNet performs non-linear transformations and extracts meaningful data patterns without imposing assumptions on data distribution or structure, making it ideal for financial applications that require interpretable and performance-driven models. We apply our modeling framework to forecast the directional movements for 49 U.S. industry portfolio excess returns for the 2013-2022 OOS period. We utilize a rich set of 127 covariates for our predictive task and evaluate the model's accuracy against other benchmark models using multiple performance metrics and statistical tests. We also construct a trading application to evaluate the profitability of TabNet's predictions. Based on TabNet's predictions, we form the trading positions and shift wealth between the industries' portfolios and a risk-free investment (i.e., the one-month Treasury Bill).

Our main findings reveal that TabNet achieves the highest out-of-sample (OOS) predictive accuracy across all performance metrics compared to other linear and non-linear machine learning models. The statistical tests employed further validate these results. The forecasting application demonstrates that TabNet's classification accuracy is considerably higher than that of the second best-performing model,

logistic regression. Additionally, we establish the profitability of TabNet's directional forecasts through the trading application results. The strategy formed based on TabNet's predictions achieves the highest annualized return, Sharpe ratio, and statistically significant positive alpha values compared to the four- and five-factor models. Notably, the portfolio constructed using TabNet's forecasts also exhibits the lowest volatility and maximum drawdown values. The trading strategy based on the proposed model continues to outperform all other benchmarks, even after accounting for monthly transaction costs. These findings provide financial professionals and portfolio managers with a powerful tool, offering superior predictive accuracy and risk-adjusted performance that can enhance trading strategies. For academic researchers, the results offer a validated framework for further exploring the application of interpretable machine learning models in finance, particularly in areas like directional prediction, which have received less attention in the literature.

In the context of the thesis, interpretability refers to how models explain or provide insights into the relationships between inputs (i.e., covariates) and outputs (i.e., predictions). Compared to traditional econometric and asset pricing models, machine learning approaches like LassoNet-SAGE can identify non-linear patterns and interactions between variables. White-box models (e.g., LassoNet-SAGE, TabNet) maintain a level of transparency by selecting a subset of features, offering insights into what drives predictions. On the other hand, in traditional econometric models, interpretability often comes directly from model coefficients, giving clear causal interpretations. However, traditional approaches may oversimplify complex data patterns by adopting linear model dynamics. For instance, in LassoNet, the deep learning component captures non-linearities, while the covariate selection component retains interpretability by identifying key covariates. Post-estimation methods like SAGE provide additional insights into how different covariates contribute to the prediction. This hybrid approach allows for more flexible and potentially more accurate forecasts than traditional methods, but it also introduces some limitations in interpretability. While SAGE identifies covariate importance, it is a method not directly derived from economic theory and does not provide the same level of direct interpretability as simpler linear models.

This thesis employs a comprehensive set of Python libraries and R packages to analyze the data and implement machine learning models. The research uses Numpy (Harris et al., 2020) and pandas (McKinney, 2010) for data handling and preprocessing purposes. The modeling framework of the first chapter is developed using the lassonet (Lemhadri et al., 2021) and sage-importance (Covert et al., 2020) Python libraries. For the second chapter, we utilize the code associated with the paper that introduced BRITS (Cao et al., 2018), which is available as GitHub repository, and the pypots (Du, 2023) library. The pytorch (Paszke et al., 2019), along with pytorch_tabnet (DreamQuark-ai., 2023) libraries are used to implement TabNet in the third chapter. Several Python libraries, including lightgbm (Ke et

al., 2017), xgboost (Chen & Guestrin, 2016), catboost (Prokhorenkova et al., 2018), tensorflow/keras (Abadi et al., 2016), asgl (Mendez-Civieta et al., 2021) and scikit-learn (Pedregosa et al., 2011) have been being applied for various machine learning tasks and benchmark model implementations. Furthermore, the filling (Fuchs et al., 2021) R package is used to estimate the matrix completion benchmark in the second chapter. To ensure the reproducibility of results, we take the necessary steps to ensure a consistent and verifiable workflow across estimations. The randomness inherent in many machine learning algorithms, such as random initialization of weights, is controlled by using a technique known as fixing the model's random seed. Additionally, the models are serialized using the Python pickle library. Serialization refers to saving a trained model's state (including its parameters, weights, and specification) to disk, enabling it to be reloaded and used afterward without the need for re-estimation. This step further enhances the reproducibility by allowing the exact same model, in the same state, to be reused for further analysis, or comparisons with benchmarks.

While the findings of this research demonstrate the efficacy of interpretable machine learning models such as TabNet, LassoNet, and BRITS in financial forecasting, several promising directions for future research remain. One potential area involves extending the application of these models to a broader set of asset classes, such as bonds, real estate, and alternative investments like cryptocurrencies, to assess their performance across diverse financial instruments. Moreover, further research could also explore integrating machine learning methods with traditional asset pricing frameworks, such as the Capital Asset Pricing Model (CAPM) or multi-factor models, to better understand how these approaches can complement each other and enhance prediction accuracy. Moreover, incorporating unstructured data sources, such as corporate filings, news sentiment, or sector-specific data, may offer new insights and improve forecasting performance. Another avenue is the adaptation of these models for real-time, high-frequency trading environments, where rapid decision-making is crucial. Finally, future work could aim to make these models more computationally efficient, allowing for greater accessibility and usability for smaller financial institutions and individual researchers.

The remainder of the thesis is structured as follows. Chapter 1 presents the methodology and empirical evidence of modeling and forecasting 10 US industry portfolios using the LassoNet-SAGE framework. Chapter 2 describes the application of the BRITS model to impute hedge fund datasets and the forecasting and trading exercise in which we use the imputed datasets to generate OOS predictions and create portfolios. Chapter 3 presents our modeling framework based on TabNet, which we leverage to predict the directional movements and trade 49 US industry portfolios. Finally, Chapter 4 provides the concluding remarks of the thesis.

CHAPTER 1

Industry Return Prediction via Interpretable Deep Learning

1.1. Introduction

There is a significant strand of literature showing that traditional asset pricing factor models do not perform well, or at least as well as initially advertised, in explaining the cross-section of stock returns or stock return predictability (Ferson and Harvey, 1991; Ferson and Korajczyk, 1995; Ferson and Harvey, 1999; Avramov, 2004, Lewellen, et al., 2010). Hence, the return predictability they evinced could result from asset pricing misspecifications. Scientific remedies suggest constructing efficient aggregate portfolios, such as industry portfolios. However, even these solutions still need to improve the explanatory power of linear asset pricing models (see Lewellen et al., 2010). For that reason, recent studies investigate stock return predictability via machine learning techniques better suited to uncover nonlinear patterns and cross-sectional relationships among firm and fund characteristics (see among others, Krauss et al., 2017; Fischer and Krauss, 2018; Gu et al., 2020; DeMiguel et al., 2023). Those techniques can also address issues arising from many irrelevant or highly correlated predictors while minimizing the risk of overfitting, contrary to widely used linear methods. Moreover, sparse literature investigates the aggregate stock return predictability at the industry portfolio level. Most of the studies mainly focus on applying linear methodologies, such as the Ordinary Least Squares (OLS) and Lasso to industry returns, and they often rely on a relatively small number of predictors (see Cohen and Frazzini, 2008; Menzly and Ozbas, 2010; Rapach et al., 2015, 2019). Besides the aggregate market return predictability, the increasing popularity of industry exchange-traded funds (ETFs) and efficient capital allocation across industries make industry-sorted portfolios' predictability an interesting topic for academics and practitioners.

Given the growing applications of machine learning techniques, it has become even more interesting and essential to provide insights into how these methods capture precisely the relationships between predictors and forecasts. Interpretability is an essential tool in empirical asset pricing applications as we can understand which input variables affect the output the most and better identify the problem (see Brigo et al., 2021); likewise, a human who can consistently predict the result of a model and explain the logic behind the result (see, Kim et al., 2016). Existing literature using linear asset pricing models has shown that profitability (see Fama and French, 2015 and Ball et al., 2016), liquidity (see Pastor and Stambaugh, 2003) and industry interdependencies (see Rapach et al., 2015) are some of the most significant factors in determining expected returns of stocks and industry

portfolios of stocks. Thus, it is worth revising those empirical findings under the prism of a nonlinear investigation.

This study uses a deep learning framework to predict industry portfolio returns. Specifically, we apply the LassoNet method of Lemhadri et al. (2021) to forecast U.S. industry portfolio returns by extracting information from a large-scale dataset of multiple predictors. The framework uniquely can unveil the underlying structure between forecasts and predictors in a nonlinear environment. This is achieved by jointly selecting a subset from a large class of input variables and minimizing the objective loss function of a neural network in a mathematically elegant way. As a result, the set of the selected covariates drives the model's forecasts endogenously, thus providing more accurate predictions. In other words, LassoNet captures data nonlinearities via a deep learning mechanism while it performs feature selection due to its Lasso-type component, unlike standard neural network architectures.

However, LassoNet is only partially interpretable because it performs feature selection rather than quantifying the selected features' importance on the model's performance. To assess how much each feature contributes to LassoNet's performance overall and so to uncover the factors leading our forecasts precisely, we use post-LassoNet, the SAGE method of Covert et al. (2020), which is an additive global importance interpretability method relying on Shapley values. We input the features selected by LassoNet to SAGE to shed more light on the importance of each selected covariate on industry portfolio predictability. The advantage of SAGE lies in using a cooperative game theoretic framework to identify the predictors that contribute the most to reducing the out-of-sample (OOS) error or increasing the models' predictability. SAGE achieves this by considering all possible interactions across the dataset of the LassoNet selected predictors in optimizing the model's performance. Such a feature contrasts with other commonly used interpretability methods, such as the partial derivatives or regular SHAP method, which deal with the issue only by identifying the variables causing the most significant variation in the model output or how much each feature contributes to a single prediction.[1] This is the first time LassoNet and SAGE methods have been applied to a financial study, either separately or jointly. Overall, the above two-step approach is employed in forecasting and trading 10 U.S. industry portfolio returns over the 2010 – 2019 period based on a large universe of 88 predictors. In this way, we provide a detailed exercise of LassoNet and SAGE on financial time series forecasting. We compare our results with those obtained by linear (i.e., linear regression, Group Lasso, Elastic-Net) and several nonlinear machine learning models (i.e., XGBoost and two neural network specifications) as proposed

[1] We are aware that other methods, such as deep learning-based Granger causality applications, are well suited for causal relationships in time series data. However, they can indicate only whether a variable helps forecast another. At the same time, SAGE can elucidate how individual components of that variable category contribute to the model's prediction and to lower the LassoNet's loss function.

by the previous literature. Finally, we assess the economic significance of LassoNet forecasts by forming long-short portfolios of corresponding industry ETFs based on the highest-lowest forecasted returns on industry portfolios. In this way we also evaluate LassoNet's ability on capital allocation of aggregate portfolios.

Our findings indicate that LassoNet outperforms OOS in all benchmark methodologies employed across the ten industries examined, so it better predicts industry portfolios' returns. For instance, LassoNet achieves the smallest forecasting error compared to other powerful machine learning models, such as the XGBoost and standard neural networks. Those findings are justified by pairwise statistical significance tests for predictability, such as those of Diebold Mariano (1995) and Giacomini and White (2006), as well as tests for statistical inferences of multiple benchmark forecasts at the same time and while accounting for *alpha-level inflation* such as those of Hansen (2005) and Hansen et al. (2011). The SAGE method reveals the importance of valuation ratios and individual and cross-industry lagged industry returns to our predictions against other covariates belonging to profitability, liquidity and efficiency ratios. The covariates belonging to these categories of predictors present the highest SAGE values, and so have higher explanatory power on industry portfolios' returns. Thus, we provide new evidence that valuation ratios, individual and cross-industry lagged industry returns are significant determinants of expected returns of stocks and portfolios of stocks. We improve on recent findings in the linear asset pricing literature, which shows that profitability (see Fama and French, 2015 and Ball et al., 2016) and liquidity (see Pastor and Stambaugh, 2003) are some of the most significant factors. Regarding the economic significance of the proposed model, the portfolio of ETFs constructed on LassoNet's industry predictions outperforms several benchmarks in terms of a battery of performance measures as well as statistically significant alphas. For example, the most profitable long-short ETF portfolio generates a post-transaction costs Sharpe ratio of 2.04 and statistically significant four and five-factor alphas of 20.35% and 20% per annum, respectively.

The remainder of this chapter is structured as follows. Section 1.2 provides a literature review. Section 1.3 presents our methodology and modelling framework. Section 1.4 describes our dataset and experimental design. Section 1.5 covers the main empirical results. Finally, section 1.6 concludes.

1.2. Literature review

Our work is linked to the emerging operations research literature of machine learning methods applications on forecasting equity returns, either as single stocks or industry portfolios of stocks (see among others, Rapach et al., 2015; Krauss et al., 2017; Fischer and Krauss, 2018; Rapach et al., 2019; Gu et al., 2020; Bianchi and McAlinn, 2021). Krauss et al., (2017) use DNNs, gradient-boosted-trees, and random forests to predict the probability of each of the S&P 500 constituents' daily returns to

outperform the market cross-sectionally for 1992 – 2015. The authors construct long-short portfolios based on those predictions while they use different lagged returns of each stock as covariates. The method generating the highest return is an equally weighted machine learning approach ensemble. In a follow-up paper, Fischer and Krauss (2018) assess the predictive power of a state-of-the-art long short-term memory neural network (LSTM) on classifying the same universe of stocks based on the cross-sectional median. The LSTM method outperforms all benchmark models (i.e., random forest, neural network, and logistic regression) in statistical and economic terms. Gu et al. (2020) perform a large-scale comparative analysis of the most popular machine learning algorithms (e.g., penalized regressions, principal component analysis, regression trees, neural networks) in predicting 30 thousand U.S. stocks from March 1957 to December 2016. The study uses 94 stock-level characteristics as predictors in the forecasting experiment of stock returns in a panel data format. It concludes that these models offer an improved description of the expected return behaviour relative to traditional forecast methods. DNNs were the best specification in forecasting and trading tasks, generating the highest R-squared and Sharpe ratios, respectively.

Concerning the industry portfolios' return predictability via machine learning, Rapach et al. (2015) examine the predictability of the adaptive Lasso model of Zou (2006) for shrinkage and optimal variable selection on 30 industry portfolios while using as predictors monthly lagged returns of different industries for the period 1960 – 2014. Their findings report significant evidence of cross-industry returns predictability, with at least four lagged industry returns being significant predictors. These results are also verified by principal component and partial least squares methods by extracting the latent factors of industry returns. In a similar setting and for the same dataset, Rapach et al. (2019) use a Lasso model for dimensionality reduction of lagged industry returns predictors while applying an OLS post-Lasso regression to estimate predictor coefficients and so better forecast industry returns accurately. They also employ a multiple-hypothesis testing framework to assess the statistical significance of the selected predictors. The findings align with those of Rapach et al. (2015) while being statistically significant. The authors also test the economic significance of the above predictability by constructing industry spread portfolios, which generate higher performance than naïve benchmarks. Recently, Bianchi and McAlinn (2021) propose an ensemble of linear predictive regressions for industry portfolio returns, considering the correlation structure of 75 covariates, especially when highly correlated. Even though their proposed method does not belong to the class of nonlinear methods, they compare its performance with that of conventional machine learning techniques. They conclude that financial ratios provide valuable information for forecasting stock returns at the industry and aggregate market levels.

In addition, we relate our study to the recent literature attempting to interpret the machine learning *black box* properties in financial applications. Seminal studies implementing different techniques that assign importance measures to the individual financial covariates include those of Gu et al. (2020) and Kim et al. (2020). Gu et al. (2020) mainly use two standard techniques for feature importance. The first one evaluates the reduction in the R-squared of the predictive regression by setting each covariate to zero while keeping the rest of the predictors unmodified. The second approach assesses the sensitivity of the forecasting model fit to changes in a covariate by measuring the sum squared partial derivatives of the model to each predictor. Such an approach is assumed conventional in machine learning literature (see also Dimopoulos et al., 1995). Recently, Kim et al. (2020) use a DNN to forecast the profitability of retail investors in spread trading while considering different feature groups related to investors (e.g., past performance, preferences in markets and channels, demographics, etc). They employ the information-fusion-based sensitivity analysis (IFBSA) of Delen et al. (2007) as their primary feature importance method to obtain the most informative predictors. IFBSA tests the marginal impact of a predictor on the error of a model without a specific covariate concerning the model, which includes that covariate. At the same time, the procedure is repeated for all covariates.

Our study extends the above literature in four ways. First, we show the ability of state-of-the-art machine learning approaches to better capture nonlinear patterns and interactions in a data-rich environment of covariates and accurately predict aggregate portfolio returns, specifically industry portfolios, than standard machine learning and linear methods. For that purpose, we apply the LassoNet, an interpretable neural network imposing sparsity and allowing only a subset of the covariates to drive its forecasts, even if some covariates are correlated (e.g., valuation ratios). On the same note, we expand the previous literature showing the outperformance of Lasso-type methodologies in predicting industry portfolios' returns (see Rapach et al., 2015; 2019). We prove that applying an interpretable deep learning method based on Lasso, which performs feature selection and *deep learning* in a nonlinear environment, significantly improves forecasting performance. Second, a key distinguishing feature of our work is the focus on model interpretability associated with which covariates are instrumental for the forecasts and crucial for the model's overall predictive performance. We use SAGE, a cooperative game theory approach, to quantify the importance of the LassoNet selected features in explaining industry portfolio returns globally. Third, we assess the ability of LassoNet to allocate capital efficiently across industries and so the economic significance of its forecasts. We construct spread portfolios of industries based on their returns forecasts and show their superiority against buy-and-hold strategies on market benchmarks. Finally, we create and use a large-scale set of 88 predictors, consisting of 10 distinct categories of financial ratios, past returns and

macroeconomic variables, by bringing together and expanding predictors used from the past literature for industry return predictability (e.g., Rapach et al. 2019; Bianchi and McAlinn, 2021). Thus, we expand the universe of the covariates under study compared to the existing literature and offer new insights into their significance in the predictability and profitability of industry returns. Overall, our study can be of great interest to researchers and policymakers to efficiently predict financial market movements and make informed decisions about optimal trading execution and capital allocation.

1.3. Methodology

In this section, we discuss the proposed deep learning framework. In the first subsection, we start with a detailed description of the LassoNet model and its main advantages. In the following subsection, we discuss the model's hyper-parameterization. The last subsection provides a discussion of the game-theoretic framework of Shapley Additive Global importancE (SAGE) (Covert et al., 2020), which is used to assign importance scores to the selected covariates. To adequately assess LassoNet's predictive ability, we also compare its forecasting performance against several benchmark models, which are presented in the Appendix 1.A. The benchmark model set includes linear regression, Group Lasso, Elastic-Net, two Neural Network models, a simple MLP and Lasso combined with an MLP (Lasso-MLP), and XGBoost. The hyperparameters for all models are tuned in-sample (IS). Following the common practice, we employ the early stopping and sensitivity analysis procedure for the neural networks (see Krauss et al., 2017; Fischer and Krauss, 2018; Gu et al., 2020) and cross-validation for all other machine learning models. Using this extensive benchmark model set enables us to compare the LassoNet with models that perform covariate selection (group lasso regression, elastic net regression, Lasso-MLP) as well as models that use the full covariate set for their forecasts (linear regression, MLP, XGBoost). The description of the benchmark models, as well as their optimization hyperparameters, are reported in Appendix 1.A.

We employ several Python libraries for data analysis and machine learning model implementation. Numpy (Harris et al., 2020) and pandas (McKinney, 2010) are used for data preprocessing tasks. The LassoNet-SAGE modeling framework is developed using the lassonet (Lemhadri et al., 2021) and sage-importance (Covert et al., 2020) libraries. The asgl (Mendez-Civieta et al., 2021) library is applied for penalized regression models, while the MLP neural networks and XGBoost are estimated using tensorflow/keras (Abadi et al., 2016) and xgboost (Chen & Guestrin, 2016) Python libraries, respectively. To ensure reproducibility, we fix the random seed to control randomness in the model training process and serialize the trained models using the pickle library, allowing for consistent reuse and analysis of the models.

1.3.1. LassoNet

We define an asset's excess return as the sum of its conditional expected return and the prediction error component. The conditional expected return of an industry portfolio i at time t+1 can be represented as a function of covariates that maximizes the OOS prediction of the realized return, $r_{i,t+1}$, in a nonlinear setting (see also Gu et al., 2020):

$$r_{i,t+1} = \mathbb{E}\left(r_{i,t+1}\right) + e_{i,t+1} = g^{\star}\left(\mathbb{X}_{i,t}\right) + e_{i,t+1} \tag{1.1}$$

where the conditional expected return $g^{\star}(\bullet)$ term represents a nonlinear flexible function that a machine learning model parameterizes, $\mathbb{X}_{i,t}$ is a D-dimensional vector of covariates and $e_{i,t+1}$ is the error term. In our case, we use a balanced panel dataset $\left\{\left(\mathbb{X}_{i,t}, r_{i,t+1}\right)\right\}_{1\le i\le n}$ spanning across the covariate set for the ten industries and the period examined in our study. We denote with $r_{i,t+1}$ the industry returns, the target variable in our forecasting task. To construct the mapping $g^{\star}: \mathbb{X}_{i,t} \mapsto r_{i,t+1}$ for each industry, the LassoNet method extends the traditional linear regularized regression models by simply adding a nonlinear component (see Lemhadri et al., 2021). In essence, the added term is the nonlinear transformation of the input variables as they propagate forward through the layers of a neural network with activation functions. Mathematically, the LassoNet for each industry is formulated as follows:

$$g^{\star} \equiv g_{\theta,W}: \mathbb{X}_{i,t} \mapsto \theta^{T}\mathbb{X}_{i,t} + H_{W}\left(\mathbb{X}_{i,t}\right) \tag{1.2}$$

where $g^{\star}$is a class of residual feed-forward neural networks of arbitrary width and depth.[2] The network is parameterized by weights $\{(\theta, W)\}$, where θ denotes the vector of weights in the residual layer (i.e., skip-connection), and W denotes the vector of weights in the hidden layer of a fully connected feed-forward network H_{w}. Hence, $\theta^{T}\mathbb{X}_{i,t}$ corresponds to the linear component, and $H_{W}(\mathbb{X}_{i,t})$ corresponds to the nonlinear component of the neural network architecture. Following Lemhadri et al. (2021), the objective function for each industry's prediction takes the form of:

$$\min_{\theta,W} L(\theta, W) + \lambda\|\theta\|_{1} \tag{1.3}$$

$$subject\ to\ \left\|W_{p}^{(1)}\right\|_{\infty} \le \mathrm{M}\left|\theta_{p}\right|, \qquad p = 1, 2, \ldots, D \tag{1.4}$$

[2] The exact LassoNet architecture is different for each industry after hyperparameter optimization since we train the model separately for each industry.

where $L(\theta, W)$ is the loss function, λ denotes the feature sparsity penalty parameter, ${W_p}^{(1)}$ indicates the weights for covariate p in the first hidden layer, and M is a hierarchy coefficient.[3] The key features of LassoNet are introducing a penalty term in the loss function, which enforces covariate selection, and the so-called *hierarchy coefficient* M, which controls the relative strength of linear and nonlinear components of the model. The residual and the first hidden layer are jointly optimized.

The main innovation of the model lies in the constraint $\left\|{W_p}^{(1)}\right\|_\infty \leq \mathrm{M}\left|\theta_p\right|$, which indicates that a covariate p is not involved in the feed-forward network (i.e. ${W_p}^{(1)}$=0) if the residual layer weight is zero (i.e., $\theta_p = 0$). In other words, the constraint conditions the level of participation of a covariate p in the nonlinear operations of the model (i.e., first and subsequent layers) based on its relative importance, which is achieved by tying every covariate to the single coefficient, θ, of the linear component (i.e., skip-connection). In this way, the linear component is used to guide feature sparsity in the nonlinear component (i.e., feed-forward neural network), and both components are fitted simultaneously to capture nonlinear patterns in the dataset via the neural network. Moreover, a closer inspection of the objective function reveals that the LassoNet nests the linear Lasso and the standard feed-forward neural network in cases where the *hierarchy coefficient,* M, takes the extreme values of zero and infinity, respectively.

The estimation of LassoNet includes a standard backpropagation process, which is initially applied to all model parameters, and a proximal operator is applied on the input layer's set of weights (i.e., $\left\{\theta, W^{(1)}\right\}$). Specifically, we use gradient descent with Adam optimizer to update the LassoNet set of weights as a first step. A *hierarchical proximal operator* is applied exclusively to the skip-connection weights, θ, and the neural network weights connecting the covariates to the first hidden layer, $W^{(1)}$, as a second step.[4] To fit our dataset optimally, we examine deep neural network architectures with multiple hidden layers to ensure that LassoNet can better capture all possible nonlinearities in our financial dataset. We also implemented the hyperbolic tangent ($\tanh(\bullet)$) as the activation function and the Mean Squared Error (MSE) as the loss function for our LassoNet implementation.[5] The exact hyperparameter tuning is described in the following section.

[3] Penalization of the weights is only required for the neurons' first hidden layer because of the feed-forward architecture of the network, while the λ penalty term in the objective function operates in the same way as in the standard linear Lasso model.

[4] The detailed optimization steps can also be found in Lemhadri et al. (2021).

[5] This is $L(\theta_i, W_i) = \frac{1}{n}\sum_{\tau=t+1}^{t+n}(r_{i\tau} - \hat{r}_{i\tau})^2$

1.3.2. LassoNet's hyperparameters

In this section, we present the LassoNet's hyperparameters optimization process. We optimize LassoNet by using early stopping and sensitivity analysis in the I.S. To conduct early stopping, we use a validation dataset, which is constructed by splitting the I.S. into training and validation sub-samples. We use the first 18 years (i.e., 1985 – 2002) as the training dataset and the last seven years (i.e., 2003 – 2009) as the validation dataset (see also Granger, 1993; Dunis et al., 2011; Gu et al., 2020). Sensitivity analysis in conjunction with early stopping results in the optimal combination of hyperparameters, achieving the lowest mean squared error in the validation sub-sample. We keep this network as the optimal LassoNet architecture for generating the OOS forecasts. More specifically, we optimize the number of hidden layers, hidden neurons, and hyperparameters M and λ with early stopping and sensitivity analysis.

Regarding LassoNet's nonlinear component specification, we follow Gu et al. (2020) and decide on the optimal LassoNet specification from a fixed candidate model set. We avoid shallow neural network architectures with a single hidden layer in the candidate model set because of possible nonlinearities in our dataset (i.e., a vast set of covariates). It has also been shown that deeper architectures with multiple hidden layers outperform shallower ones with a single hidden layer due to the higher-order nonlinear interactions between the covariates (see Mhaskar et al., 2016). However, given that industry portfolios' monthly data frequency limits the number of samples available for the model's estimation, we do not explore architectures with more than three hidden layers to avoid model overfitting. Similar to other studies, we explore architectures with a higher number of neurons in the first layer(s) followed by a layer(s) with a smaller number of neurons (see Gu et al., 2020). Based on the above reasoning, we explore two LassoNet architectures with two hidden layers (i.e., (16 4), (16, 8)) and two architectures with three hidden Layers (i.e., (16, 8, 4), (16, 16, 4)). To determine the number of neurons for the first hidden layer, we follow again the relevant literature (see Gu et al., 2020; Filippou et al., 2022) and a common rule of thumb (i.e., the number of neurons equals the square root of the number of covariates).

For the *hierarchy coefficient* M, we investigate four values (i.e., 0.005, 0.05, 0.5, 5) and choose the optimal one via early stopping and sensitivity analysis procedures. To obtain the optimal value of $the\ \lambda$ hyperparameter, we follow Lemhadri et al. (2021) and adopt a heuristic mechanism based on a sequence of λ values. The initial value of λ equals the generated MSE of a conventional MLP estimated with the selected covariates on the validation data. The exact MLP architecture is defined above (e.g., 16, 4). The initial value of λ is used to run the LassoNet algorithm as described in Section 1.3.1. Then, the estimation of the following λ hyperparameters is given based on a *regularization path multiplier*

of 1.05. For instance, the algorithm is re-estimated every time the λ is increased based on that multiplier. The procedure continues until the λ reaches a value that imposes a regularization powerful enough that the LassoNet selects no covariate. Table 1.1 presents the different hyperparameter configurations of the LassoNet model. We examine all possible combinations of 4 different hidden layer architectures and four different *M* values for 16 candidate configurations.

Table 1.1. Hyperparameter search space for the LassoNet model.

This table reports the hyperparameter search space for the LassoNet model. We use the validation dataset to choose the set that generates the lowest mean squared error metric.

Architectures	**[(16, 4), (16, 8), (16, 8, 4), (16, 16, 4)]**
M parameter	[0.005, 0.05,0.5, 5]
Regularization Path multiplier	[1.05]

For LassoNet's iterative estimation algorithm, we keep the number of training iterations, known as epochs, at 200. We also use a batch size of 72 observations. To retain the temporal ordering of the data, we enforce that batch construction follows the time sequence of the observations and that the data are not shuffled before feeding them to LassoNet's training algorithm. Finally, we use 50 model training iterations for the early stopping mechanism to avoid model overfitting. The final optimized LassoNet provides transparency regarding which covariates drive its forecasts. Moreover, it provides an input variable selection mechanism to handle high-dimensional asset pricing datasets and regularize a large pool of covariates. However, even though LassoNet can effectively perform covariate selection, the model lacks the ability to determine the importance of each individual covariate.

1.3.3. Estimating covariates' importance

For many years, the financial literature criticised standard machine learning approaches for their black-box properties, or in other words, not providing the importance of financial and economic factors on target forecasts as economic and asset pricing theory would expect. We apply an extra step on top of LassoNet's feature selection property to identify the covariates that drive our model's forecasts. As described above, the optimized LassoNet selects specific covariates through its penalized regression component in nonlinear environments. However, it does not perform feature importance in ranking covariates based on the extent to which the model depends on each of them overall. For that reason, we employ a cooperative game-theoretic method based on Shapley values to measure the importance of each selected covariate. The SAGE method of Covert et al. (2020) provides the Shapley values, which quantify the ranking in the importance of each covariate or group of covariates. To avoid confusion,

we call the Shapley values estimated by SAGE as *SAGE* values. The significant contribution of SAGE, contrary to the commonly used SHapley Additive exPlanations (SHAP) method of (Lundberg and Lee, 2017), is that it represents a global interpretability method which provides feature importance by considering the behaviour of the model across the whole dataset and not how much each feature contributes to an individual prediction (i.e., every month). Also, the approach fundamentally differs from other interpretability methods (e.g., partial derivatives), which only measure importance in a way that finds which covariates cause the most considerable variation in the model output. Influencing the model output does not necessarily indicate a covariate as informative, so it is more insightful to identify which covariates drive the forecasts and, at the same time, consider whether these covariates enhance the model's predictive accuracy.

In our study, the SAGE method assigns credit to the covariates based on their contribution to lowering the LassoNet's loss OOS (i.e., $L\big(\ \mathbb{E}\big[g^{\star}\big(\mathbb{X}_{i,t}\big)\big]\ , r_{i,t+1}\ \big)$). This contrasts with the SHAP method, which only assigns each feature a value representing whether it pushes the prediction higher or lower. Additionally, we acknowledge that covariates contribute different information when inputted into a model together with other covariates versus being in isolation (see Covert et al., 2020). To properly account for variable interaction effects and synergies, we consider all possible subsets of our selected covariate set and then measure the degree of increase of the model's error without a specific covariate. This process is repeated by focusing on a different covariate each time. Implementing this method involves constructing a cooperative game v_f that represents the model's overall performance and is defined as follows:

$$v_f(S) = -\mathbb{E}\left[\, L\big(\ \mathbb{E}\left[\, g^{\star}\big(\mathbb{X}_{i,t}\big)\middle|\, \mathbb{X}_{i,t}^{S}\,\right], r_{i,t+1}\ \big)\right] \tag{1.5}$$

where $g^{\star}$ represents the optimized LassoNet, S indexes a subset of the total number of covariates (i.e.,$S \leq D$), v_f quantitatively represents the model's performance given the subset $\mathbb{X}_{i,t}^{\mathrm{S}}$ of covariates. The minus sign in front of the loss indicates that a lower loss increases the value of $v_f(S)$.

According to the above framework, a specific Shapley value, $\varphi_p(v_f)$, is attributed to each covariate p to quantify the contribution to lowering the model's prediction error. Only the covariates with positive values $\varphi_p(v_f) > 0$ are essential for the forecast and instrumental in increasing the model's statistical accuracy and improving its forecasting performance. The estimated Shapley values represent importance scores for the corresponding covariates when they are estimated for cooperative games in the form of $v_f(S)$. Finally, we identify covariate importance on an aggregated category level. To arrive at this estimation, we sum the SAGE values for all covariates that were selected by the LassoNet and belong to the same category of variables.

1.4. Data and experimental design

We forecast 10 U.S. industry portfolio returns as given by Kenneth French's website for the period 1985 to 2019 in a rolling-window format, using January 2010 to December 2019 as the OOS. Specifically, we refit all models yearly and use IS data to produce forecasts for the following year (i.e., 12 months) (see Gu et al., 2020; Chen et al., 2023; DeMiguel et al., 2023). The forecasting horizon is one month ahead. The ten industry sectors we examine in this study are: Consumer Durables (DURBL), Energy (ENRGY), High-Technology (HITEC), Health (HLTH), Manufacturing (MANUF), Consumer Nondurables (NODUR), Shops (SHOPS), Telecommunications (TELCM), Utilities (UTILS), and the other remaining industry sectors merged (OTHER)[6]. Accordingly, the industry returns correspond to the value-weighted average of their constituent stocks.

For our prediction task, we construct a set consisting of 88 covariates. To construct our dataset, we use the Compustat database from the Wharton Research Data Services (WRDS) platform and precisely 63 industry financial ratios, which belong to capitalization, efficiency, financial soundness, solvency, liquidity, profitability, and valuation categories. Capitalization ratios measure the debt component of a firm's total capital structure; efficiency ratios capture the effectiveness of the firm's usage of assets and liability; financial soundness and solvency ratios capture the firm's ability to meet long-term obligations; liquidity ratios measure a firm's ability to meet its short-term obligations; profitability ratios measure the ability of a firm to generate profit; valuation ratios estimate the attractiveness of a firm's stock. To aggregate financial ratios at the industry level, we take the median value from the companies belonging to the specific industry. The covariates set is on a monthly frequency. In case of only quarterly or annual data for some ratios, the most recently available observation item is carried forward to fill each month. All observations are lagged by two months to avoid look-ahead bias issues and ensure that the information was publicly announced at a given timestamp in our dataset. Following the findings of Rapach et al. (2015), which provide evidence of industry interdependencies and cross-industry return predictability, we also decided to include other industries' lagged returns to extend our dataset of covariates further. Thus, for each U.S. industry portfolio, we also include up to 12-month excess lagged returns and the 1-month value-weighted lagged returns of all other nine industries as extra covariates.[7]. Our dataset also includes four macroeconomic variables downloaded from the FRED database, namely the Chicago Fed National Financial Conditions Index (NFCI), Chicago Fed National Activity Index (CFNAI), Chicago Fed National Activity Index: Production and Income

[6]The industry definitions are available on Kenneth French's website: https://mba.tuck.dartmouth.edu/pages/faculty/ken.french/Data_Library/det_10_ind_port.html.

[7] We use the 3-Month Treasury Bill to calculate excess returns. The data were downloaded from the FRED database: https://fred.stlouisfed.org/

(PANDI), and the Consumer Price Index (CPI). The NFCI index captures U.S. financial conditions in money, debt, and equity markets and the traditional and *shadow* banking systems. The PANDI index provides information regarding the national economy's expansion with respect to its historical trend rate of growth. The CFNAI index captures overall economic activity and the related inflationary pressure, while the CPI is used as an inflation index. In the trading application, we use ETFs prices from the CRSP Stocks and Mutual Funds datasets. We select all traded index-funds identified by share code 73, with style stated as *Equity*, *Domestic* and *Sectorial*. Then, we keep the ETFs which include in their name the industry classification closest to the Fama-French industries and with the longest time series.[8]

Regarding the experimental design, our full sample ranges from January 1985 until December 2019. We use 2010 – 2019 as the OOS period and the previous 25 years of monthly data (i.e., 300 months/ data points) as I.S. in a rolling-window structure. Our partition corresponds to the 70%-30% split commonly employed in the related literature (see also Harvey and Liu, 2015). Figure 1.1 presents the separation of the full sample in IS and OOS periods.

Figure 1.1 In-sample and out-of-sample partition of monthly observations.

We also generate forecasts and evaluate LassoNet's predictive ability and covariates importance on four OOS subperiods (i.e., 2000-2006, 2007-2009, 2010-2014, 2015-2019) for robustness purposes. This way, we examine how the model's performance varies across time and business cycles. The relevant results of the subperiod analysis are available in the Appendix 1.C and confirm the superiority of LassoNet in predicting industry portfolios across the four different subperiods.

[8] We present all the employed covariates for our predictive task, their corresponding categories, and the ETF details used for our trading simulation in Appendix 1.B. We have also trained LassoNet with a broader set of 98 covariates, including more macroeconomic and volatility predictors than those followed by the relevant literature. More specifically, we added in the covariates set the 3-Month Treasury Bill, the Implied Volatility Index (VIX), the dividend yield of the S&P Global index, the momentum factor, three commodities (silver, gold, crude oil), and three exchange rates (EUR/USD, GPB/USD, YEN/USD). We find that the LassoNet's performance remains the same by including additional covariates, so they did not comprise additional information beyond the initial set of 88 covariates. The relevant results are available upon request.

1.5. Empirical results

1.5.1 Forecasting accuracy

We conduct a forecasting evaluation of the LassoNet and benchmark models' forecasts by computing the root mean squared error (RMSE) and the mean absolute error (MAE) metrics. Table 1.2 presents the OOS error metrics across the ten industries from 2010 to 2019.

The results indicate that the LassoNet consistently outperforms all the other benchmark models across most industries by generating the lowest prediction error metrics under the RSME and MAE criteria. Consequently, the optimization process in LassoNet is superior to simply optimizing a standard regularization linear model or feed-forward neural networks. The outperformance stems from LassoNet's algorithm, which jointly optimizes linear and nonlinear components, enabling LassoNet to retain the advantages of both components without retaining any limitations. We show that covariate selection enhances a model's forecasting ability OOS, which can provide additional reasoning on why LassoNet outperforms a standard neural network trained on the complete set of covariates. Table 1.2 also reports that the XGBoost, Group Lasso and Elastic Net are the second, third and fourth-best models, with the remaining models (OLS, MLP and Lasso-MLP) showing far worse performance. For instance, XGBoost generates better forecasting performance than LassoNet for *utilities* industry portfolios. Table 1.2 also presents the spread between the minimum and maximum values of the error metrics across all industries. A wider spread denotes that a specific model performs well for certain industries and poorly for others, while less variation for the error metrics represents consistent performance across industries, which is desirable.

We report such an error metric range in panel B of the same table for that purpose. The range is calculated by the difference between the highest and the lowest values of each model's generated error (i.e., RMSE, MAE) across all industries. Again, LassoNet presents the lowest dispersion across error metrics' minimum and maximum values. Therefore, we can infer that the LassoNet has the most consistent performance across the ten industries, with its forecasting ability not being industry or data-dependent.

Table 1.2. OOS statistical performance for the LassoNet and the employed benchmark models.

The table reports the OOS statistical performance over the 2010 – 2019 period. For each industry, we compare the performance of the LassoNet model against the employed benchmarks (i.e., OLS-Regression, Group-Lasso, Elastic-Net, MLP-NN, Lasso-ANN-MLP, and XGBOOST). We report the root mean squared error (RMSE) and the mean absolute Error (MAE) as error metrics. In Panel B, we also present the range for the RMSE and MAE metrics, which are calculated by the difference between each error metric's highest and lowest values. The lowest values are reported in bold.

Panel *A*: *Loss Criteria*	**LASSONET**	**OLS**	**GROUP LASSO**	**ELASTIC NET**	**MLP**	**LASSO-MLP**	**XGBOOST**		**LASSONET**	**OLS**	**GROUP LASSO**	**ELASTIC NET**	**MLP**	**LASSO - MLP**	**XGBOOST**
	DURBL								**ENRGY**						
RMSE	**0.0618**	0.0821	0.0630	0.0640	0.2369	0.0658	0.0674		**0.0534**	0.1060	0.0808	0.0879	0.2300	0.1581	0.0589
MAE	**0.0467**	0.0630	0.0508	0.0527	0.1610	0.0489	0.0500		**0.0421**	0.0816	0.0643	0.0692	0.1692	0.1252	0.0474
	HLTH								**NODUR**						
RMSE	**0.0356**	0.1240	0.0415	0.0409	0.1692	0.0780	0.0439		**0.0328**	0.0640	0.0424	0.0406	0.1327	0.1115	0.0361
MAE	**0.0281**	0.0869	0.0348	0.0341	0.1257	0.0576	0.0330		**0.0259**	0.0482	0.0330	0.0316	0.1100	0.0884	0.0287
	TELCM								**UTILS**						
RMSE	**0.0353**	0.1280	0.0420	0.0390	0.1392	0.1502	0.0410		0.0364	0.0841	0.0512	0.0587	0.1996	0.1643	**0.0331**
MAE	**0.0287**	0.0778	0.0320	0.0312	0.1159	0.1102	0.0337		0.0287	0.0681	0.0404	0.0463	0.1308	0.1360	**0.0263**
	MANUF								**HITEC**						
RMSE	**0.0348**	0.0700	0.0410	0.0438	0.1538	0.1291	0.0502		**0.0341**	0.1785	0.0762	0.0786	0.1782	0.1051	0.0467
MAE	**0.0269**	0.0557	0.0325	0.0336	0.1223	0.1030	0.0379		**0.0269**	0.1421	0.0618	0.0631	0.1465	0.1376	0.0370
	SHOPS								**OTHER**						
RMSE	**0.0334**	0.0935	0.0566	0.0529	0.2673	0.2569	0.0400		**0.0384**	0.0849	0.0480	0.0511	0.1819	0.1726	0.0464
MAE	**0.0269**	0.0713	0.0422	0.0401	0.1756	0.1637	0.0312		**0.0277**	0.0609	0.0381	0.0400	0.1270	0.1303	0.0359

Panel B: *Error Metric Range*	**LASSONET**	**OLS**	**GROUP LASSO**	**ELASTIC NET**	**MLP**	**LASSO - MLP**	**XGBOOST**
RMSE Range	**0.0290**	0.1710	0.0398	0.0489	0.1346	0.1911	0.0343
MAE Range	**0.0208**	0.0938	0.0323	0.0380	0.0656	0.1148	0.0237

To examine the statistical significance of the performance of the LassoNet, we first perform the Diebold and Mariano (1995) (D.M.) test and the Giacomini and White (2006) (G.W.) test based on the forecasts' squared error loss functions.[9] We use these tests to compare the OOS performance of the LassoNet against each one-off implemented benchmark. The D.M. tests the null that two forecasts have equal predictive ability based on the difference of the loss functions of two forecasts against the alternative that the loss differential is different from zero. A negative and significant *t*-statistic rejects the null hypothesis, and it reports the superiority of LassoNet against the benchmark (i.e., lower loss).

Table 1.3 reports the generated *t*-statistics from the D.M. test and their corresponding *p*-values in parenthesis for all industries under study. D.M. test results show that the LassoNet has superior predictive ability against most benchmarks and across industries (i.e., corresponding *p*-values are below the significance thresholds, and *t*-statistics are negative). There are only two industries, namely *durables* and *utilities*, in which LassoNet shows equivalent performance with the XGBoost, Group Lasso and Elastic Net and XGBoost models, respectively. For the remaining industries, LassoNet generates significantly better performance.

Table 1.3. Diebold Mariano test results for the LassoNet against benchmark models.

The table displays the *t*-statistics and *p*-values of the D.M. (1995) test for LassoNet against each benchmark pairwise across industries for the 2010-2019 OOS period. The null hypothesis tests that LassoNet and the benchmark forecast have equal predictive ability. Bold *p*-values and *t*-statistics indicate that we reject the null hypothesis of the two forecasts' equivalence and show the superiority of LassoNet against the benchmark.

	D.M. test: *t-statistic (p*-value)									
	DURBL	**ENRGY**	**HITEC**	**HLTH**	**MANUF**	**NODUR**	**OTHER**	**SHOPS**	**TELCM**	**UTILS**
OLS	**-2.84 (0.003)**	**-5.06 (0.000)**	**-7.17 (0.000)**	**-4.09 (0.000)**	**-6.03 (0.000)**	**-5.00 (0.000)**	**-4.56 (0.000)**	**-5.92 (0.000)**	**-3.90 (0.000)**	**-6.40 (0.000)**
GROUP LASSO	-0.38 (0.752)	**-4.00 (0.000)**	**-6.25 (0.000)**	**-2.89 (0.002)**	**-1.76 (0.089)**	**-3.00 (0.002)**	**-2.16 (0.024)**	**-3.71 (0.000)**	**-2.49 (0.012)**	**-3.54 (0.000)**
ELASTIC NET	-0.64 (0.583)	**-4.41 (0.000)**	**-6.32 (0.000)**	**-2.59 (0.009)**	**-2.49 (0.011)**	**-2.57 (0.012)**	**-2.57 (0.010)**	**-3.54 (0.000)**	**-2.44 (0.014)**	**-4.46 (0.000)**
MLP	**-5.03 (0.000)**	**-6.02 (0.000)**	**-8.27 (0.000)**	**-5.40 (0.000)**	**-6.78 (0.000)**	**-8.92 (0.000)**	**-4.75 (0.000)**	**-4.92 (0.000)**	**-8.42 (0.000)**	**-4.28 (0.000)**
LASSO-MLP	**-2.13 (0.037)**	**-6.74 (0.000)**	**-7.04 (0.000)**	**-5.00 (0.000)**	**-8.00 (0.000)**	**-7.04 (0.000)**	**-6.68 (0.000)**	**-3.82 (0.000)**	**-5.66 (0.000)**	**-8.70 (0.000)**
XGBOOST	-1.37 (0.157)	**-2.20 (0.026)**	**-3.91 (0.000)**	**-3.03 (0.000)**	**-3.51 (0.000)**	**-2.28 (0.022)**	**-1.92 (0.081)**	**-2.37 (0.019)**	**-2.81 (0.004)**	1.46 (0.147)

[9] We also implement Giacomini and Rossi's (2010) (GR) fluctuation test, which measures the relative local forecasting performance of the LassoNet model compared to one benchmark over time given changing conditions. The relevant results show the outperformance of LassoNet against the benchmarks and are presented in Appendix 1.D

The G.W. conditional predictive ability test assesses the null of equal predictive ability between two models (i.e., pairwise comparison) when the forecasting model (i.e., LassoNet) may be miss-specified. G.W. test complements D.M. in terms of reflecting the effect of estimation uncertainty and permitting a unified treatment of nested and non-nested models. The latter feature is essential for our experiment as LassoNet includes both Lasso and neural network components. Also, the G.W. test uses available information to predict which forecast will be more accurate for a specific future date (i.e., one month ahead in our case), conditional on given information. This property improves D.M. test which evaluates which forecast was more accurate, on average, in the past. The alternative hypothesis of G.W. suggests which forecast performs better by producing a lower average loss than the competing model.

Table 1.4 presents the *p*-values generated by the G.W. test by performing a pairwise comparison of LassoNet against each benchmark for all industries examined. A *p*-value rejecting the null indicates that LassoNet performs better than the benchmark. Again, our findings show that LassoNet is a better forecaster than all benchmarks for most industry portfolios. Only in the case of DURBL industry, we observe that LassoNet does not generate more accurate predictions.

Table 1.4. Giacomini and White (2006) test results for the LassoNet against benchmark models.

The table reports the Giacomini and White (2006) test *p*-values for the LassoNet against each benchmark across all industries over the 2010 – 2019 OOS period. The null hypothesis is the equal predictive ability between two models when the forecasting model (i.e., LassoNet) may be misspecified. Significant *p*-values indicate the superiority of LassoNet against the benchmark. *, **, *** denote significance at the 10%, 5% and 1% level, respectively.

	G.W. test: *p*-value									
	DURBL	**ENRGY**	**HITEC**	**HLTH**	**MANUF**	**NODUR**	**OTHER**	**SHOPS**	**TELCM**	**UTILS**
OLS	0.024**	0.000***	0.000***	0.000***	0.000***	0.000***	0.000***	0.000***	0.000***	0.000***
GROUP LASSO	0.756	0.000***	0.000***	0.012**	0.056*	0.014**	0.056*	0.016**	0.045**	0.002***
ELASTIC NET	0.829	0.000***	0.000***	0.041**	0.016**	0.041**	0.032**	0.003***	0.056*	0.000***
MLP	0.000***	0.000***	0.000***	0.000***	0.000***	0.000***	0.000***	0.000***	0.000***	0.000***
LASSO-MLP	0.106	0.000***	0.000***	0.000***	0.000***	0.000***	0.000***	0.000***	0.000***	0.000***
XGBOOST	0.161	0.023**	0.002***	0.009***	0.004***	0.081*	0.065*	0.033**	0.029**	0.086*

The LassoNet's predictive performance is further assessed using the unconditional *Superior Predictive Ability* (SPA) procedure of Hansen (2005) and the *Model Confidence Set* (MCS) procedure of Hansen et al. (2011) with a 10% test size. Those tests simultaneously perform an OOS statistical inference of many forecasts while controlling for data snooping (i.e., *alpha-level inflation* problem). The SPA test assesses the relative forecasting performance of LassoNet, as the point of the reference model, against the complete set of benchmark forecasts. The null hypothesis of the SPA test is that the LassoNet forecast is not inferior to the best alternative model's forecast based on a given loss function. Table 1.5 reports that the SPA test generated *p*-values under the RMSE and MAE criteria. The SPA test

results show that the corresponding *p*-values of LassoNet against the benchmarks are high enough (i.e., fail to reject the null hypothesis) to conclude that LassoNet is not inferior to the benchmarks under both the RMSE and MAE criteria and across all industry portfolios.

Table 1.5. SPA test results for the LassoNet and the employed benchmark models.

The table displays the *p*-values for the SPA test of Hansen (2005) over the 2010 – 2019 OOS period. The test provides insights regarding the relative performance of LassoNet against the employed benchmarks (OLS-Regression, Group Lasso, Elastic-Net, MLP, Lasso-MLP, and XGBOOST). Bold *p*-values indicate a failure to reject the null hypothesis that LassoNet is not inferior to the benchmarks.

		2010-2019									
		DURBL	**ENRGY**	**HITEC**	**HLTH**	**MANUF**	**NODUR**	**OTHER**	**SHOPS**	**TELCM**	**UTILS**
P-VALUE											
	MSE	**0.405**	**0.871**	**0.950**	**0.894**	**0.824**	**0.999**	**0.774**	**0.859**	**0.882**	**0.348**
	MAE	**0.427**	**0.815**	**0.990**	**0.998**	**0.838**	**0.990**	**0.716**	**0.991**	**0.895**	**0.361**

Implementing and making robust conclusions about predictive ability with the SPA test is challenging when there is no natural model as a reference point or when more than one model is considered. The MCS procedure of Hansen et al. (2011) addresses such an issue by bypassing point-of-reference models. The MCS aims to find a superior set of models indistinguishable from the best, including the best model. It consists of a sequence of tests which permits the construction of a set of *superior* models, where the null hypothesis of equal predictive ability is not rejected at a certain confidence level. The test requires two procedures: an equivalence test, determining whether models are equal according to their loss and an elimination rule, which dictates which model to eliminate if the equivalence test reveals that two models are not equivalent (i.e., there is one with a larger loss). The output of the MCS is a model set containing the true set of *best* models with a probability weakly larger than $1 - a$, where a is the significance level. Also, if only one best model exists, the test will find it asymptotically. As a rule of thumb, a low (high) *p*-value is associated with a model that is unlikely (likely) to belong to the set of the *best* models. Therefore, *p*-values that exceed the nominal significance levels advocate that the tested model belongs to the confidence set of *best* models (Psaradellis and Sermpinis, 2016; Grønborg et al., 2021). However, The MCS *p*-value is not a statement about the probability that a model is the best.

Table 1.6 presents the relevant results based on a 10% significance level. In particular, the table reports the MCS *p*-values for each model. We present the *p*-values of the models belonging to the confidence set in bold. We observe that LassoNet always belongs to the confidence set of *best* models, while for most of the cases, it is the model with the lowest loss (i.e., *p*-value = 1), except for the case of the *utilities* industry. Interestingly, LassoNet is the only model in the true set for half of the industries

examined (i.e., ENERGY, HITECH, HLTH, SHOPS and TELCM). We can conclude that no models are indistinguishable from the best (LassoNet).

Table 1.6. MCS test results for the LassoNet and the employed benchmark models.

The table reports *p*-values for Hansen et al.'s (2011) MCS procedure over the 2010 -2019 OOS period and each industry at a 10% confidence level. A sequence of significance tests is performed to find model forecasts that are not inferior to others. *P*-values that exceed the nominal significance levels (i.e., 1%, 5%, and 10%) show that the model belongs to the MCS. The models belonging to the confidence set at the 10% significance level are reported in bold.

	MCS test: *p*-value									
	DURBL	**ENRGY**	**HITEC**	**HLTH**	**MANUF**	**NODUR**	**OTHER**	**SHOPS**	**TELCM**	**UTILS**
OLS	0.019	0.000	0.000	0.016	0.008	0.029	**0.142**	0.020	0.003	0.008
GROUP LASSO	**0.814**	0.079	0.000	0.025	**0.108**	**0.193**	**0.142**	0.060	0.056	**0.127**
ELASTICNET	**0.814**	0.079	0.000	0.016	0.098	**0.193**	**0.142**	0.060	0.056	0.014
MLP	0.019	0.000	0.000	0.014	0.000	0.000	0.093	0.020	0.000	0.008
LASSO-MLP	**0.487**	0.000	0.000	0.016	0.000	0.000	0.004	0.020	0.003	0.000
XGBOOST	**0.487**	0.079	0.000	0.016	0.098	**0.193**	**0.142**	0.060	0.056	**1.000**
LASSONET	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**0.485**

1.5.2 Covariates' importance

1.5.2.1 SAGE value estimation

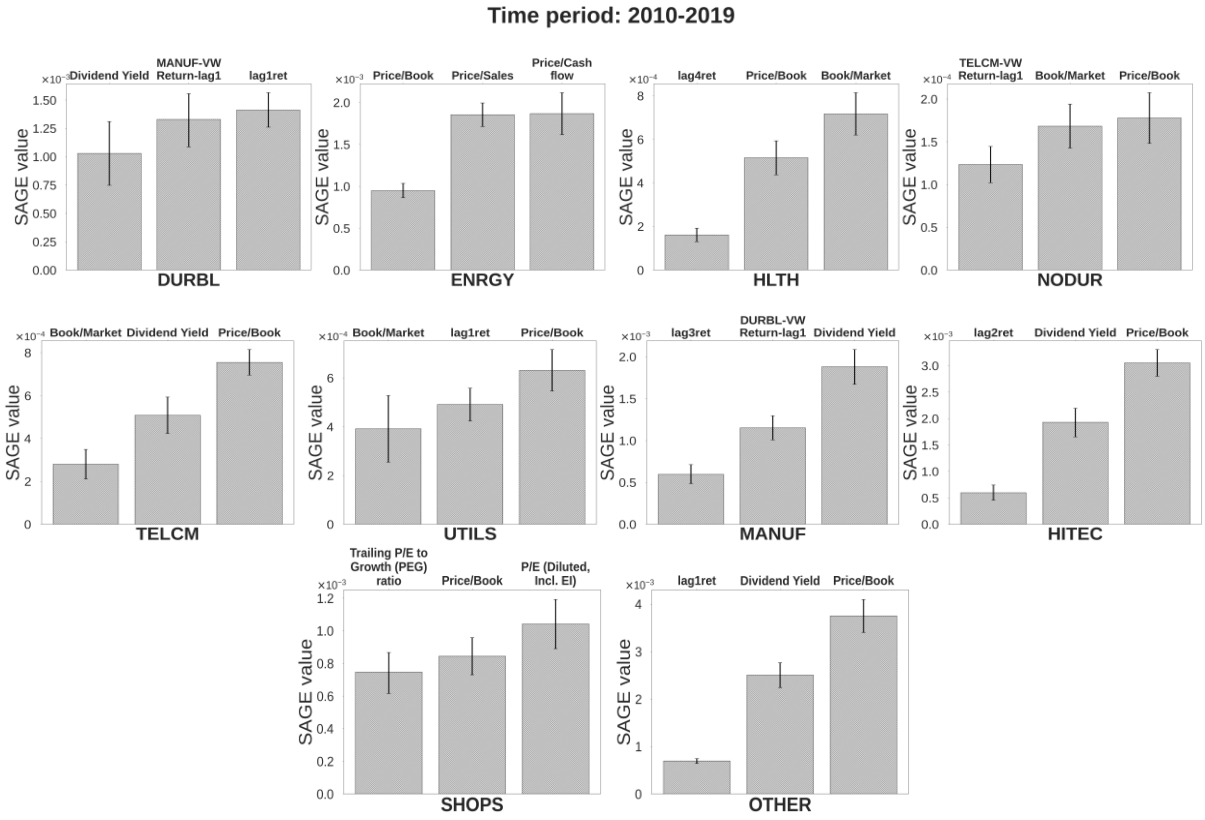
After establishing that the LassoNet outperforms all other benchmark models, it is crucial to investigate the covariates driving its forecasts. Figure 1.2 presents the three covariates with the highest SAGE values OOS for every industry separately, along with 95% confidence intervals around the mean SAGE value of each covariate across the rolling windows.[10]

We arrive at three significant conclusions. First, the valuation ratios, followed by the individual industry's lagged returns and the cross-industry one-month lagged returns, are the most pivotal categories for the model's predictive performance. Second, this result is generally consistent across the different industries. Third, we witness greater variability and less consistency regarding the participation of other covariate categories in the three highest SAGE value positions. The above results are consistent with a body of literature that examines the predictive relationship between valuation ratios and asset returns. Notable examples include the work of Keim and Stambaugh (1986), Fama and French (1988), and Campbell and Shiller (1988). More recent studies outlining that valuation ratios can effectively predict stock returns include the work of Campbell and Yogo (2006) and Campbell and Thompson (2008).

[10] We obtain the OOS SAGE values per covariate by aggregating their values for each rolling window estimation while we calculated the 95% confidence intervals around the mean of SAGE values across all rolling windows.

Figure 1.2. SAGE values bar plots.

The figure displays the three covariates with the highest SAGE values for every industry across the 2010 – 2019 OOS period. We restrict our results to the three covariates with the highest SAGE values to investigate the most significant variables (and the categories they belong to). The bar graphs also include 95% confidence intervals around the mean SAGE value of each covariate.

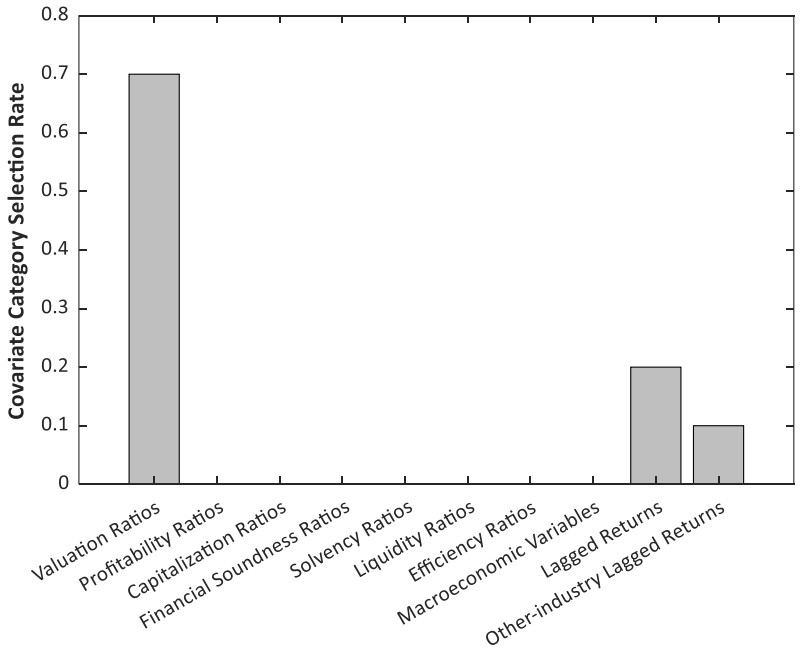


Additionally, our framework reveals significant interdependencies and gradual information diffusion across the industries, given that individual and cross-industry lagged returns are the second and third most crucial covariate categories, respectively. In this direction, Rapach et al. (2015) note that links between industries can be established not just with customer-supplier relationships but even more broadly via technology spillovers and production chain interactions.

We also calculate the selection rate of each variable's category based on the highest SAGE values generated across the ten industries examined and rolling windows. To define a category's selection rate, we consider the appearance of its corresponding covariates within the top three highest SAGE values OOS across the ten industries and window estimations. Then, for each category, we calculate the fraction of the covariates with the highest SAGE values across all industries. Figure 1.3 presents the estimated selection rates for the different covariates' categories.

Figure 1.3. Selection rates for the covariates' categories.

The figure displays the selection rate that each category's covariates appear within the three highest positions regarding their corresponding SAGE values across the 2010 – 2019 OOS period and ten industries.



Consistent with the findings of Figure 1.2, we evidence that the valuation ratios have the highest aggregate presence, as indicated by a selection rate of up to 70%. In second and third place come each industry's lagged returns and *other-industry* lagged returns with selection rates of around 20% and 10%, respectively. While these rates are lower than the valuation ratios, they still report the importance of each industry's lagged and cross-industry lagged returns.

1.5.2.2 Statistical significance of SAGE values

According to the SAGE value of a specific variable, we can measure its positive contribution to the overall LassoNet performance. By summing all variables belonging to a specific category across industries and forecast windows, we can obtain an aggregate measure of a category's overall positive contribution to the model's performance. We employ pairwise hypothesis tests between the covariates' categories to evaluate any differences in positive contribution OOS statistically. We effectively create a set of 10 category aggregate SAGE values. Finally, we employ pairwise two-tailed *t*-tests between the aggregate SAGE of covariate categories. In Table 1.7, we report the *t*-statistics and their corresponding *p*-values of the cross-category hypothesis tests and the corresponding *p*-values controlling for *p-value correction* under the Hommel (1988) criterion, which are reported in brackets. The pairwise tests compare the mean of the column covariate against the mean of every other

covariate category presented in each row. Hence, a positive *t*-statistic with a low *p*-value indicates that the column category has a statistically significant higher positive contribution than the row category.

The findings presented in Table 1.7 quantitatively validate the results of the categories' selection rates, as presented in Figure 1.2. For example, they report that the mean of the valuation ratios against the mean of every other covariate category is positive and statistically significant at the 1% level for almost all cases. In addition, the *t*-statistics and their *p*-values also reveal a favourable and statistically significant outcome for the mean positive contribution of the lagged returns category compared to most of the covariate categories at 1%, except to those *of other-industry* lagged returns and valuation ratios. The results for the *other-industry* lagged returns are similar to the lagged returns category and are at a statistical significance level of 1%. Also, macroeconomic variables categories and financial soundness show positive significance against specific covariates such as solvency, capitalization, liquidity and efficiency ratios.

Controlling for *p-value correction* under the Hommel (1988) criterion, the corresponding *p*-values in brackets reveal a similar picture mainly for valuation ratios, lagged and *other-industry* lagged returns and financial soundness ratios, which retain their positive and statistical significance against the rest of the categories. The above findings can help quantitative fund managers experiment beyond conventional statistical models and effectively guide decisions concerning asset allocations while attaining model transparency via the SAGE. Policymakers can also benefit from such an interpretable learning framework when designing economic policies by forecasting the movements of industry sectors and identifying the most critical covariates governing the underlying price dynamics.

Table 1.7. SAGE values pairwise hypothesis tests between the covariates' categories.

We sum the SAGE values of all covariates belonging to the same category for each of the ten industries over the 2010 – 2019 OOS period and the forecasting windows examined. For all covariate categories, we create a set of aggregate SAGE values. We then employ pairwise *t*-tests between the covariates' categories to evaluate differences in the positive contribution statistically. We present the *t-statistics* and the corresponding *p*-values for the hypothesis tests in parenthesis. We additionally report the corresponding *p*-values under the Hommel (1988) criterion for *p*-value correction in brackets. The *t*-statistics and *p*-values of the column category with a statistically significant higher positive contribution than row one are presented in bold. *, **, *** denote significance at the 10%, 5% and 1% level, respectively.

	Valuation Ratios	Lagged Returns	Other-industry Lagged Returns	Macroeconomic Variables	Financial Soundness Ratios	Solvency Ratios	Profitability Ratios	Capitalization Ratios	Liquidity Ratios	Efficiency Ratios
Valuation Ratios	-	-	-	-	-	-	-	-	-	-
Lagged Returns	**1.833 (0.086*) [0.086*]**	-	-	-	-	-	-	-	-	-
Other-industry Lagged Returns	**3.118 (0.009***) [0.019**]**	1.549 (0.143) [0.143]	-	-	-	-	-	-	-	-
Macroeconomic Variables	**4.679 (0.001***) [0.003***]**	**3.873 (0.003***) [0.006***]**	**3.454 (0.005***) [0.005***]**	-	-	-	-	-	-	-
Financial Soundness Ratios	**4.718 (0.001***) [0.003***]**	**3.944 (0.003***) [0.006***]**	**3.611 (0.005***) [0.005***]**	0.096 (0.925) [0.925]	-	-	-	-	-	-
Solvency Rations	**5.092 (0.001***) [0.002***]**	**4.534 (0.001***) [0.004***]**	**4.642 (0.001***) [0.005***]**	**2.718 (0.023**) [0.090*]**	**3.906 (0.003***) [0.009***]**	-	-	-	-	-
Profitability Ratios	**4.859 (0.001***) [0.003***]**	**4.162 (0.002***) [0.005***]**	**3.974 (0.003***) [0.005***]**	0.973 (0.346) [0.691]	1.130 (0.273) [0.273]	**-2.326 (0.042**) [0.167]**	-	-	-	-
Capitalization Ratios	**5.082 (0.001***) [0.002***]**	**4.517 (0.001***) [0.004***]**	**4.613 (0.001***) [0.005***]**	**2.631 (0.026**) [0.102]**	**3.749 (0.003***) [0.010**]**	-0.269 (0.791) [0.791]	**2.192 (0.051*) [0.109]**	-	-	-
Liquidity Ratios	**5.066 (0.001***) [0.002***]**	**4.491 (0.001***) [0.004***]**	**4.564 (0.001***) [0.005***]**	**2.483 (0.032**) [0.102]**	**3.460 (0.005***) [0.014**]**	-0.544 (0.594) [0.791]	**1.963 (0.073*) [0.143]**	-0.308 (0.762) [0.762]	-	-
Efficiency Ratios	**5.027 (0.001***) [0.002***]**	**4.431 (0.002***) [0.005***]**	**4.463 (0.002***) [0.005***]**	**2.211 (0.051*) [0.152]**	**3.076 (0.009***) [0.019**]**	-1.483 (0.158) [0.475]	1.571 (0.143) [0.143]	-1.184 (0.253) [0.505]	-0.752 (0.462) [0.462]	-

1.5.3 Trading application

To adequately assess the economic significance of the LassoNet's predictive ability, we apply a trading simulation to use the OOS industry returns' forecasts to form spread portfolios on their corresponding industry ETFs. We construct monthly spread portfolios of industry ETFs based on the best and worst-performing industries according to their forecasts. For example, each month, we sort the industries based on the corresponding LassoNet's return predictions, and then, we form long positions on ETFs for the industries with the highest forecasted returns and short positions on ETFs for the industry with the lowest returns. As ETFs are commonly used for passive investing, with investors opting for a few highly profitable ones, we present three variations of long-short portfolios.. First, the *Max1-Min1* which considers the spread of only the top and bottom forecasted industry returns. Second, the *Max2-Min2* is the return of being long (short) on the two top (bottom) industry portfolios with the highest (lowest) predicted returns and likewise, *Max3-Min3* is the spread of the top and bottom three portfolios, respectively. We consider an expense ratio of 0.25% every time we take a position on each fund.[11] [12]

Table 1.8 presents a battery of performance metrics for each portfolio and several benchmarks. We report the annualized mean return, volatility, annualized Sharpe ratio, maximum drawdown and the annualized alphas of 4-factor (Carhart, 1997) and 5-factor (Fama & French, 2015) models. We also regress the portfolios' returns against the four-factor (Carhart, 1997) and five-factor models (Fama & French, 2015), and explore the presence of positive and statistically significant alphas. We choose to compare the performance of ETF portfolios based on LassoNet forecasts against the corresponding portfolios constructed based on Group Lasso, Elastic Net and XGBoost, which are the following best performers after LassoNet in terms of statistical accuracy. We also use a buy-and-hold strategy on the value-weighted returns of the CRSP index and the S&P 500 index, equally weighted portfolios of the industry portfolio returns, equally weighted portfolios of the selected industry ETFs, and a strategy that shorts the ETFs' returns in the current month to trade their spread on the following as our benchmarks. The constituents of the CRSP indexes directly match those in the industry returns we used to train our machine learning models, offering the most representative benchmark. Moreover, this choice aligns with earlier literature, given the diversification properties of the market return and

[11] According to Morningstar and Vanguard, the average industry ETF expense ratio has been around 0.25% over the past years. We present the results with a more conservative ratio of 0.5% in Appendix 1.E, and we find that the overall picture remains the same. We also present the performance of a trading strategy investing directly in industry portfolios.

[12] We also perform a similar trading exercise on the forecasts of the maximum number of industry portfolios (i.e., 49) available on Kenneth French's website. Since not enough ETFs are available to track the expanded number of industry portfolios (i.e., 49 industries) we implement our strategy directly on industry portfolios. The findings show that LassoNet yields the highest performance and are available in Appendix 1.F.

being free from anomalies specific to individual industries (see Moskowitz and Grinblatt, 1999; Dong et al., 2022).

The ETF portfolios based on LassoNet forecasts generate the highest returns, Sharpe ratios, and positive and statistically significant alphas compared to the rest of the machine learning models and benchmarks. In particular, the *Max2-Min2* portfolio achieves the highest performance, generating a Sharpe ratio 2.04, followed by the *Max1-Min1* portfolio. The same portfolio yields a statistically significant annualized alpha of 20%. Similarly, the maximum drawdown of the LassoNet constructed portfolios is the lowest most of the time. These findings demonstrate that LassoNet's forecasts can effectively generate positive ETF returns not captured by seminal factor models while minimizing the downside risk.

1.6. Conclusion

We apply an interpretable machine learning framework, the LassoNet, to forecast U.S. industry portfolio returns over the 2010 – 2019 period based on a data-rich environment of 88 predictors. We compare the performance of LassoNet with that of a battery of linear (i.e., linear regression, Group Lasso, Elastic-Net) and nonlinear (i.e., XGBoost, neural networks) methods. We quantify the critical determinants of our forecasts by applying the SAGE, global importance interpretability method, on features selected by LassoNet. Finally, we evaluate the economic significance of industry portfolio returns forecasts in a capital allocation strategy. Our findings contribute to the relevant literature in four ways:

First, we find that state-of-the-art interpretable deep learning specifications can capture nonlinear patterns and interactions among our predictors to forecast better industry portfolio returns than linear and nonlinear machine learning approaches. LassoNet reports significantly smaller forecasting errors across most industries examined than other machine learning models, such as the XGBoost and neural networks, as shown by a battery of statistical tests such as the D.M., GR, SPA and MCS tests. Such a performance lies in the specific characteristic of LassoNet to simultaneously perform feature selection and deep learning for forecasting purposes. Second, the application of the SAGE method for global interpretability shows that valuation ratios and individual and cross-industry lagged industry returns generate the highest SAGE values, and so they are critical determinants for industry portfolio forecasts.

Table 1.8. Performance of industry ETF portfolios based on OOS forecasts.

The table demonstrates performance metrics for trading strategies of industry ETFs based on LassoNet's, Group Lasso, Elastic Net and XGBoost forecasts and those of benchmark strategies over the 2010 – 2019 OOS period. The *Max1-Min1*, *Max2-Min2*, and *Max3-Min3* industry ETFs spread portfolios are constructed based on the highest and lowest-performing industries according to their corresponding forecasts while considering a 0.25% expense ratio. We report the annualized mean return and Sharpe ratio, maximum drawdown and annualized alphas. The reported alphas are obtained using the 4-factor (Carhart, 1997) and 5-factor (Fama & French, 2015) models. *, **, *** denote significance at the 10%, 5% and 1% level, respectively.

	Portfolios				**Benchmark Strategies**				
***Panel A :* Max1-Min1**	**LASSONET**	**GROUP LASSO**	**ELASTIC NET**	**XGBOOST**	**VW-CRSP**	**SP500**	**EW-Ind. Port**	**EW-Ind. ETFs**	**1 MAH**
Ann. Mean Return (%)	20.39	18.63	19.60	5.06	12.32	11.46	9.24	10.28	--8.50
Volatility (%)	14.74	15.07	14.65	12.49	12.72	12.46	12.15	12.46	16.53
Ann. Sharpe ratio	1.34	1.20	1.30	0.37	0.93	0.88	0.71	0.78	-0.54
Max Drawdown (%)	20.80	19.80	16.40	33.89	18.52	17.04	17.69	17.86	109.80
Ann. 4-factor alpha (%)	19.41***	18.03***	19.50***	4.86***	-	-	8.31***	9.22***	-12.07***
Ann. 5-factor alpha (%)	19.02***	17.02***	18.71***	4.58***	-	-	8.39***	9.18***	-13.44***
Panel B: Max2-Min2	**LASSONET**	**GROUP LASSO**	**ELASTIC NET**	**XGBOOST**	**VW-CRSP**	**SP500**	**EW-Ind. Port**	**EW-Ind. ETFs**	**1 MAH**
Ann. Mean Return (%)	21.27	16.65	16.02	2.29	12.32	11.46	9.24	10.28	-3.32
Volatility (%)	10.16	10.39	10.26	9.86	12.72	12.46	12.15	12.46	12.80
Ann. Sharpe ratio	2.04	1.55	1.51	0.18	0.93	0.88	0.71	0.78	-0.29
Max Drawdown (%)	6.93	8.18	9.09	18.25	18.52	17.04	17.69	17.86	54.46
Ann. 4-factor alpha (%)	20.35***	16.01***	15.59***	0.14	-	-	8.31***	9.22***	-5.49
Ann. 5-factor alpha (%)	20.00***	16.44***	15.00***	-0.04	-	-	8.39***	9.18***	-7.02*
Panel C: Max3-Min3	**LASSONET**	**GROUP LASSO**	**ELASTIC NET**	**XGBOOST**	**VW-CRSP**	**SP500**	**EW-Ind. Port**	**EW-Ind. ETFs**	**1 MAH**
Ann. Mean Return (%)	19.11	12.07	11.09	0.95	12.32	11.46	9.24	10.28	-5.53
Volatility (%)	8.20	8.35	8.05	7.34	12.72	12.46	12.15	12.46	10.37
Ann. Sharpe ratio	2.26	1.38	1.31	0.06	0.93	0.88	0.71	0.78	-0.58
Max Drawdown (%)	4.70	6.64	10.29	18.18	18.52	17.04	17.69	17.86	65.57
Ann. 4-factor alpha (%)	19.00***	11.81***	10.98***	-1.22	-	-	8.31***	9.22***	-7.25**
Ann. 5-factor alpha (%)	18.77***	11.40***	10.72***	-1.75	-	-	8.39***	9.18***	-8.00**

Such evidence complements the relevant findings of studies using linear asset pricing for return predictability, which mainly reveal profitability and liquidity ratios as essential factors of stock returns. The power of SAGE depends on its cooperative game theory framework, which considers all possible interactions across the dataset of selected predictors in optimizing the model's forecasting performance. Third, after accounting for transaction costs, we prove the economic significance of LassoNet forecasts in constructing more profitable spread portfolios than buy-and-hold strategies on market indices. All LassoNet-constructed portfolios generate the highest Sharpe ratios and positive and statistically significant multifactor alphas.

While the Shapley values and SAGE method offer significant advantages in quantifying covariate importance and handling complex, non-linear interactions in machine learning models, it has limitations when compared to traditional asset pricing models. Traditional models provide clear, interpretable coefficients directly tied to economic theory, allowing for more straightforward causal explanations. In contrast, SAGE focuses on identifying the covariates which have the highest predictive power, but its outputs are less intuitive and harder to interpret in economic terms, making it challenging to align with established financial models. Future research can explore combining LassoNet and SAGE with traditional asset pricing models to develop a hybrid framework that offers high predictive accuracy and further economic interpretability. By integrating the variable selection and non-linearity capture of LassoNet with the economic foundations of models like Fama-French (2015), researchers can gain deeper insights into factor importance while maintaining connections to financial theory. This approach could help bridge the gap between flexible machine learning techniques and traditional asset pricing frameworks.

Overall, we expand the previous studies reporting the ability of Lasso methods to accurately predict industry portfolios' returns by successfully applying a Lasso-based deep learning method for nonlinear environments while constructing a large-scale set of predictors for the same task. For all the above reasons, our study can be of great interest to academics and practitioners in portfolio and asset management industries.

Appendix 1

1.A. Benchmark models

To adequately assess the LassoNet forecasting performance, we utilize an extensive set of benchmark forecasting models.

1.A.1 Extreme Gradient Boosting (XGBoost)

The XGBoost model was introduced in the work of Chen and Guestrin (2016) and extends the boosting algorithm developed by Friedman (2001). Jabeur et al. (2021) state that XGBoost is an ensemble model based on decision trees that uses an optimization process leading to superior performance compared to individual techniques. Nobre and Neves (2019) note that the output of the XGBoost model can be calculated with the formula:

$$\hat{r} = \sum_{k=1}^{K} f_k(\mathrm{X}), \;\; f_k \in \mathcal{F}$$

where f is a function in the functional space $\mathcal{F}$, $\mathcal{F} = \{f(\mathrm{X}) = w_{q(\mathrm{X})}\}$ is the space of the regression trees, q is the structure of each regression tree, w is the leaf weight, f_k is the k-th regression tree, K is the number of regression trees.

The loss function that is optimized to train the model effectively is the following:

$$L = \sum_{t} l(\hat{r}, r) + \sum_{k} \Omega(f_k)$$

where l is the squared error loss function measuring the difference between the predicted return $\hat{r}$ and the realized return r, and Ω is a regularization term. Ω is specified by the following formula:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \| w\|^2$$

where γ is a regularization hyperparameter, T is the number of leaves in each regression tree, and λ is a regularization hyperparameter.

In our XGBoost implementation, we use the XGBoost Python library (Chen & Guestrin, 2016) associated with the original paper and utilize the library's default parameters. Expressly, the number of trees is set to 100, the maximum depth of a tree to 3, the learning rate to 0.1, γ to 0, and λ to 1.

1.A.2 Group Lasso

Yuan and Lin (2006) suggest the following Group Lasso[13] estimator:

$$\vartheta^* = \underset{\vartheta_0, \vartheta}{argmin} \left\| r - \vartheta_0 - \mathrm{X}^{\mathrm{T}}\vartheta \right\|_2^2 + \lambda \sum_{\xi=1}^{K} \sqrt{d_\xi} \left\| \vartheta^\xi \right\|_2$$

where λ is a tuning parameter, K is the number of categories the variables are divided into, the term $\sqrt{d_\xi}$ weights each category according to its size and d_ξ is the size of the ξ category, ϑ^ξ is a sub-vector of coefficients from ϑ with components that correspond to the covariates in ξ category. We use five-fold cross-validation to decide on the optimal value of λ, and following (Gu et al., 2020), our search space is $\{10^{-4}, 10^{-1}\}$[14].

1.A.3 Elastic-Net

The Elastic-Net model was introduced by Zou and Hastie (2005) and admits the following mathematical formulation:

$$\vartheta^* = \underset{\vartheta_0, \vartheta}{argmin} \left\| r - \vartheta_0 - \mathrm{X}^{\mathrm{T}}\vartheta + \right\|_2^2 + \lambda \left[\frac{1}{2}(1-\rho)\|\vartheta\|_2^2 + \rho\|\vartheta\|_1 \right]$$

where, $\rho \, \epsilon \, [0,1]$ is a tuning parameter, which we set to 0.5 following Gu et al. (2020). When $\rho = 0$, we retrieve exactly the Ridge penalty, and when $\rho = 1$, we retrieve the Lasso. To decide on the optimal value of the tuning parameter λ, we used five-fold cross-validation while examining 100 different values of λ. The maximum value for λ was determined to be the lowest one at which all covariate coefficients are forced to be zero[15].

1.A.4 Neural Networks (ANN-MLP)

Fan et al. (2021) indicated that artificial neural networks (ANN) use a composition of a series of non-linear functions to model non-linearity. In mathematical notation, they take the following form:

$$H^{(h)} = f^{(h)} \circ f^{(h-1)} \circ f^{(h-2)} \circ \ldots \circ f^{(1)}(\mathrm{X})$$

where ○ illustrates the composition of two functions, $and \; h$ is the number of hidden layers.

---

[13] For the implementation of the Group Lasso model, we utilized the asgl python library (Civieta et al., 2021)

[14] The step size is 0.1.

[15] Implementing the Elastic Net model is based on the glmnet R package (Friedman et al., 2010). For more information regarding its estimation, we direct the reader to the official reference manual: https://cran.r-project.org/web/packages/glmnet/glmnet.pdf

By letting $H^0 \triangleq x$, we can define recursively $H^{(l)} = f^l(H^{(l-1)})$ for all $l = 1, \ldots, L$ (Fan et al., 2021). To retrieve the so-called MLP architecture, we define the output of its hidden layer with the tanh activation function as follows:

$$H^{(l)} = f^{(l)}\left(H^{(l-1)}\right) \triangleq \tanh\left(W^{(l)}H^{(l-1)} + b^{(l)}\right)$$

where, $W^{(l)}$ is the weight matrix fully connecting the previous layer with every neuron in the l-th layer, $b^{(l)}$ is the bias (intercept) term. Finally, the output of the final hidden layer, $H^{(L)}$, and the corresponding observed value from the training dataset are used to estimate the loss function we minimize.

1.A.5 Lasso and ANN-MLP (Lasso-ANN-MLP)

As an additional benchmark, we leverage a two-step algorithm. First, we employ a Lasso linear model (Tibshirani, 1996) without an intercept term. We tune the λ parameter using 5-fold cross-validation and the search space $\{10^{-4}, 10^{-1}\}$[16] with a step size of 0.1. Post-estimation, we track the non-zero coefficients and the corresponding covariates. Second, we feed only those non-zero covariates to an ANN-MLP architecture, which generates the final forecasts.

1.A.5.1 Hyperparameter optimization for the ANN benchmarks

To tune the hyperparameters for the ANN-MLP and Lasso-ANN-MLP benchmarks, we construct a validation dataset and leverage the early stopping and sensitivity analysis procedure, as described in Section 1.3.2 of the paper. Table 1.A.1 reports the explored hyperparameter settings for ANN-MLP and Lasso-ANN-MLP.

Table 1.A.1. Hyperparameter search space for ANN-MLP and Lasso-ANN-MLP benchmark models.

This table reports the hyperparameter search space for ANN-MLP and Lasso-ANN-MLP architectures. To optimize for the hyperparameters, we use a validation dataset -as in the case of the LassoNet- and choose the set which provides the lowest mean squared error in a sensitivity analysis.

Neural Network Architectures	[(16, 4) (16, 8) (16, 8, 4) (16, 16, 4)]
Batch size	[72]
Epochs	[200]
Early stopping	[50 epochs]

[16] For industry portfolio *OTHER*, we slightly increase the search space of λ because the original search space was not producing any non-zero coefficients for the covariates.

Due to the inferior performance of the shallow neural networks, we include only architectures with more than one hidden layer as candidate models for the benchmarks. Mirroring the LassoNet's hyperparameter search space, we also use a batch size of 72 observations and 200 epochs for the benchmarks, and we fix the early stopping mechanism at the 50 epochs level.[17].

1. B. Covariates categories and ETFs details

Table 1.B.1 presents a detailed description of covariates used for predicting industry returns and to which category they belong. The entire universe consists of 88 covariates, being part of ten different categories of variables (i.e., valuation, profitability, capitalization, financial soundness, solvency, liquidity, efficiency, macroeconomic, past returns, and *other-industry* past return).

Table 1.B.1. Covariates' categories.

The table reports all the employed covariates for our predictive task and their corresponding categories.

Covariate	Covariate Category
Dividend Payout Ratio	Valuation
Trailing P/E to Growth (PEG) ratio	Valuation
Book/Market	Valuation
Shillers Cyclically Adjusted P/E Ratio	Valuation
Dividend Yield	Valuation
Enterprise Value Multiple	Valuation
Price/Cash flow	Valuation
P/E (Diluted, Excl. EI)	Valuation
P/E (Diluted, Incl. EI)	Valuation
Price/Sales	Valuation
Price/Book	Valuation
Effective Tax Rate	Profitability
Gross Profit/Total Assets	Profitability
After-tax Return on Average Common Equity	Profitability
After-tax Return on Total Stockholders Equity	Profitability
After-tax Return on Invested Capital	Profitability
Gross Profit Margin	Profitability
Net Profit Margin	Profitability
Operating Profit Margin After Depreciation	Profitability
Operating Profit Margin Before Depreciation	Profitability
Pre-tax Return on Total Earning Assets	Profitability
Pre-tax return on Net Operating Assets	Profitability
Pre-tax Profit Margin	Profitability
Return on Assets	Profitability
Return on Capital Employed	Profitability
Return on Equity	Profitability
Capitalization Ratio	Capitalization

[17] For the implementation of the neural network benchmarks, we utilized the asgl (Civieta et al., 2021), Keras (Chollet, 2016), and TensorFlow (Abadi et al., 2016) Python libraries.

Common Equity/Invested Capital	Capitalization
Long-term Debt/Invested Capital	Capitalization
Total Debt/Invested Capital	Capitalization
Inventory/Current Assets	Financial Soundness
Receivables/Current Assets	Financial Soundness
Free Cash Flow/Operating Cash Flow	Financial Soundness
Operating CF/Current Liabilities	Financial Soundness
Cash Flow/Total Debt	Financial Soundness
Cash Balance/Total Liabilities	Financial Soundness
Cash Flow Margin	Financial Soundness
Short-Term Debt/Total Debt	Financial Soundness
Profit Before Depreciation/Current Liabilities	Financial Soundness
Current Liabilities/Total Liabilities	Financial Soundness
Total Debt/EBITDA	Financial Soundness
Long-term Debt/Book Equity	Financial Soundness
Interest/Average Long-term Debt	Financial Soundness
Interest/Average Total Debt	Financial Soundness
Long-term Debt/Total Liabilities	Financial Soundness
Total Liabilities/Total Tangible Assets	Financial Soundness
Total Debt/Equity	Solvency
Total Debt/Total Assets	Solvency
Total Debt (Liabilities)/Total Assets	Solvency
Total Debt/Capital	Solvency
After-tax Interest Coverage	Solvency
Interest Coverage Ratio	Solvency
Cash Conversion Cycle (Days)	Liquidity
Cash Ratio	Liquidity
Current Ratio	Liquidity
Quick Ratio (Acid Test)	Liquidity
Asset Turnover	Efficiency
Inventory Turnover	Efficiency
Payables Turnover	Efficiency
Receivables Turnover	Efficiency
Sales/Stockholders Equity	Efficiency
Sales/Invested Capital	Efficiency
Sales/Working Capital	Efficiency
NFCI	Macroeconomic Variables
CFNAI	Macroeconomic Variables
PANDI	Macroeconomic Variables
CPI	Macroeconomic Variables
lag1ret	1-Month Lagged Return
lag2ret	2-Month Lagged Return
lag3ret	3-Month Lagged Return
lag4ret	3-Month Lagged Return
lag5ret	5-Month Lagged Return
lag6ret	6-Month Lagged Return

lag7ret	7-Month Lagged Return
lag8ret	8-Month Lagged Return
lag9ret	9-Month Lagged Return
lag10ret	10-Month Lagged Return
lag11ret	11-Month Lagged Return
lag12ret	12-Month Lagged Return
DURBL-VW Return-lag1	1-Month O*ther-Industry* Lagged Return
ENRGY-NW Return-lag1	1-Month O*ther-Industry* Lagged Return
HITEC-VW Return-lag1	1-Month O*ther-Industry* Lagged Return
HLTH-VW Return-lag1	1-Month O*ther-Industry* Lagged Return
MANUF-VW Return-lag1	1-Month O*ther-Industry* Lagged Return
NODUR-VW Return-lag1	1-Month O*ther-Industry* Lagged Return
OTHER-VW Return-lag1	1-Month O*ther-Industry* Lagged Return
SHOPS-VW Return-lag1	1-Month O*ther-Industry* Lagged Return
TELCM-VW Return-lag1	1-Month O*ther-Industry* Lagged Return
UTILS-VW Return-lag1	1-Month O*ther-Industry* Lagged Return

Table 1.B.2 presents the details of ETFs collected from the CRSP database as representatives of each Fama and French industry portfolio examined. We match CRSP Stocks and CRSP Mutual funds datasets, and we obtain the prices series of all traded index-funds with share code 73 and stated as *Equity, Domestic* and *Sectorial*. Then, we keep the industry ETFs which report in their name the keyword *industry* closest to the Fama and French industries, and with the most extended available data. In cases where the Fama and French industries corresponds to more than one ETF (i.e., industries OTHERS and SHOPS), we consider the equally weighted portfolio of selected ETF returns. The details include the name of the ETF, the corresponding vendor, and the industry each ETF tracks.

1.C. Subperiods analysis

1.C.1 Forecasting accuracy

For robustness, we assess the LassoNet's predictive ability on four OOS subperiods (i.e., 2000-2006, 2007-2009, 2010-2014, 2015-2019).

Table 1.B.2. ETFs details.

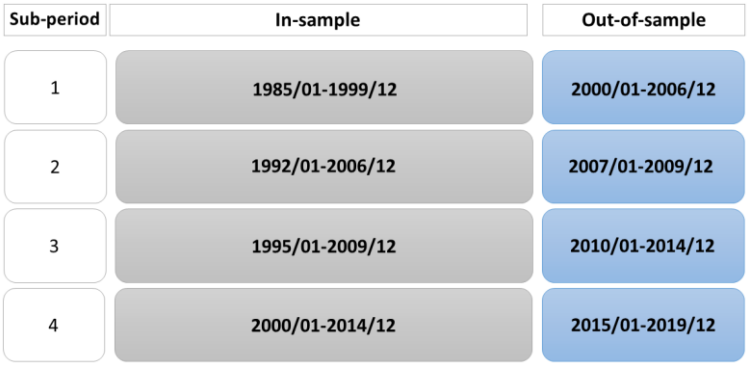
Table 1.B.2 reports the ETF details used for our trading simulation. We matched the Fama and French industries with ETFs from the CRSP database to select the industry funds for our trading exercise. To do so, we use the keyword "industry" to identify the ETFs tracking each industry's returns, and then we choose the ETFs with the most extended available data.

Fund Name	**Industry**
Vanguard World Funds: Vanguard Telecommunication Services Index Fund; ETF Shares	TELCM
Vanguard World Funds: Vanguard Financials Index Fund ETF Shares	OTHER (Financials)
PowerShares Exchange-Traded Fund Trust: PowerShares Dynamic Building & Construction Portfolio	OTHER (Construction)
Vanguard World Funds: Vanguard Consumer Staples Index Fund; ETF Shares	NODUR
Vanguard World Funds: Vanguard Healthcare Index Fund; VIPERs Share Class	HLTH
Vanguard World Funds: Vanguard Industrials Index Fund; ETF Shares	MANUF
First Trust Exchange-Traded AlphaDEX Fund: First Trust Industrials/Producer Durables AlphaDEX Fund	DURBL
StreetTRACKS Series Trust: SPDR Metals & Mining ETF	OTHER (Mining)
Vanguard World Funds: Vanguard Energy Index Fund; ETF Shares	ENRGY
Vanguard Specialized Funds: Vanguard REIT Index Fund; ETF Shares	OTHER (REIT)
iShares Trust: iShares S&P Global Consumer Discretionary Index Fund	SHOPS (Discretionary)
PowerShares Exchange-Traded Fund Trust: PowerShares Dynamic Leisure & Entertainment Portfolio	OTHER (Leisure)
SPDR Series Trust: SPDR S&P Retail ETF	SHOPS (Retail)
iShares Trust: iShares S&P Global Technology Sector Index Fund	HITEC
Vanguard World Funds: Vanguard Utilities Index Fund; ETF Shares	UTILS

Again, the forecasting horizon is one month ahead for each OOS subperiod, while the in-sample (IS) is the previous 15 years of data per industry (i.e., 180 months/data points). This way, we examine how the model's performance varies across time and business cycles. For instance, we isolated the 2007-2009 years to investigate any differentiation in the model's performance during the global financial crisis. On the contrary, according to the NBER dates of the US business cycle expansions and contractions, the preceding years (i.e., 2000 to 2006) are mainly characterized by economic expansion. Economic expansion is also prevalent throughout the 2010-2019 decade without major recession

events. We, therefore, divide 2010-2019 into two equal-length OOS periods to investigate notable differences in the model's performance across the decade. We optimize the hyperparameters of LassoNet following the early stopping procedure presented in Section 1.3.2. Figure 1.C.1 presents the separation of IS and OOS periods.

Figure 1.C.1. In-sample and out-of-sample partition of monthly observations.



Panel A of Table 1.C.1 presents the OOS Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) metrics for the ten industries, which are averaged across the four subperiods (i.e., 2000 – 2006, 2007 – 2009, 2010 – 2014, 2015 – 2019) for practical reasons. The findings again demonstrate that the LassoNet consistently outperforms all the other industry benchmark models by achieving the lowest prediction error metrics. Consistent with the full OOS results of the main paper, the XGBOOST outperforms all the other benchmark models across industries, while the Group Lasso is the third-best model under both RMSE and MAE. In addition, the LassoNet achieves the lowest discrepancy for all error metrics' minimum and maximum values, according to panel B findings.

We now assess the statistical significance of LassoNet's forecasts against all the benchmark models via the Diebold and Mariano (1995) (DM) test for each OOS subperiod. Table 1.C.2. reports the relevant results. The overall picture shows that LassoNet outperforms the benchmark models for most subperiods and across all industries. The forecasting superiority is achieved at a 1% or 5% significance level. Only the XGBoost model demonstrates an almost equivalent performance in some subperiods and industries (e.g., 2015 – 2019), but this is not statistically significant against LassoNet. However, even in these subperiods, LassoNet significantly outperforms XGBoost in three out of ten industry portfolios forecasted. In addition, LassoNet outperforms all benchmarks during the financial crisis period (i.e., 2007 – 2009) so it represents a robust forecasting specification that is fully interpretable at the same time.

Table 1.C.1. OOS statistical performance for the LassoNet and the employed benchmark models across subperiods.

The table reports the OOS statistical performance across the four subperiods. For each industry, we compare the performance of the LassoNet model against the employed benchmarks (i.e., OLS-Regression, Group-Lasso, Elastic-Net, MLP-NN, Lasso-ANN-MLP, and XGBoost). We report the root mean squared error (RMSE) and the mean absolute Error (MAE) as error metrics. In Panel B, we also present the range for the RMSE and MAE metrics, which are calculated by the difference between each error metric's highest and lowest values. The lowest values are reported in bold.

Panel *A*: *Loss Criteria*	**LASSONET**	**OLS**	**GROUP LASSO**	**ELASTIC NET**	**MLP-NN**	**LASSO & MLP-NN**	**XGBOOST**		**LASSONET**	**OLS**	**GROUP LASSO**	**ELASTIC NET**	**MLP-NN**	**LASSO & MLP-NN**	**XGBOOST**
	DURBL								**ENRGY**						
RMSE	**0.0785**	0.2017	0.1002	0.1092	0.183	0.2226	0.0874		**0.0655**	0.4851	0.1229	0.1123	0.1433	0.1401	0.0696
MAE	**0.0566**	0.1522	0.0751	0.082	0.1423	0.178	0.0669		**0.053**	0.4152	0.1016	0.0903	0.1146	0.113	0.0550
	HLTH								**NODUR**						
RMSE	**0.0426**	6.2205	0.0651	0.0746	0.0872	0.0915	0.0549		**0.0395**	0.1714	0.0728	0.0745	0.1556	0.1528	0.0524
MAE	**0.0338**	2.4352	0.053	0.0601	0.0687	0.0718	0.0440		**0.0315**	0.1398	0.0606	0.0616	0.1214	0.1154	0.0417
	TELCM								**UTILS**						
RMSE	**0.0495**	0.2437	0.0802	0.1007	0.1143	0.1303	0.0667		**0.0402**	0.3541	0.212	0.252	0.1234	0.1813	0.0481
MAE	**0.0385**	0.1759	0.0601	0.0749	0.0899	0.1068	0.0510		**0.0316**	0.2787	0.1729	0.2017	0.0957	0.147	0.0367
	MANUF								**HITEC**						
RMSE	**0.0525**	0.1574	0.1068	0.1361	0.1283	0.1615	0.0595		**0.0733**	0.2884	0.1585	0.1494	0.161	0.1562	0.0952
MAE	**0.0401**	0.1385	0.0953	0.1228	0.1056	0.1197	0.0435		**0.0565**	0.2511	0.1284	0.118	0.1361	0.126	0.0733
	SHOPS								**OTHER**						
RMSE	**0.0467**	3.6683	0.1109	0.1292	0.1347	0.1055	0.0589		**0.052**	0.3097	0.132	0.1482	0.1486	0.1421	0.0629
MAE	**0.0377**	2.3331	0.09	0.1101	0.0982	0.0848	0.0473		**0.0389**	0.22	0.1039	0.116	0.1289	0.1132	0.0499

Panel B: *Error Metric Range*	**LASSONET**	**OLS**	**GROUP LASSO**	**ELASTIC NET**	**MLP-NN**	**LASSO & MLP-NN**	**XGBOOST**
RMSE Range	**0.0390**	6.0631	0.1469	0.1775	0.0958	0.1311	0.0471
MAE Range	**0.0251**	2.2967	0.1199	0.1416	0.0736	0.1062	0.0365

Table 1.C.2. Diebold Mariano test results for the LassoNet against benchmark models across subperiods.

The table displays the *t*-statistics and *p*-values of the DM (1995) test for the LassoNet against each benchmark pairwise for the four OOS subperiods separately. The null hypothesis tests that LassoNet and the benchmark forecast have equal predictive ability. Bold *p*-values and *t*-statistics indicate that we reject the null hypothesis of the two forecasts' equivalence and show the superiority of LassoNet against the benchmark.

	2000-2006											**2007-2009**									
	DURBL	**ENRGY**	**HITEC**	**HLTH**	**MANUF**	**NODUR**	**OTHER**	**SHOPS**	**TELCM**	**UTILS**		**DURBL**	**ENRGY**	**HITEC**	**HLTH**	**MANUF**	**NODUR**	**OTHER**	**SHOPS**	**TELCM**	**UTILS**
OLS	**-5.69 (0.00)**	**-9.27 (0.00)**	**-9.87 (0.00)**	**-9.34 (0.00)**	**-11.20 (0.00)**	**-4.42 (0.00)**	**-6.61 (0.00)**	**-13.14 (0.00)**	**-6.30 (0.00)**	**-11.22 (0.00)**		**-4.03 (0.00)**	**-5.39 (0.00)**	**-6.07 (0.00)**	**-5.91 (0.00)**	**-6.27 (0.00)**	**-6.09 (0.00)**	**-3.05 (0.00)**	**-3.95 (0.00)**	**-4.64 (0.00)**	**-5.87 (0.00)**
GROUP LASSO	**-4.72 (0.00)**	**-6.74 (0.00)**	**-3.88 (0.00)**	**-8.00 (0.00)**	**-5.89 (0.00)**	**-4.46 (0.00)**	**-6.75 (0.00)**	**-11.70 (0.00)**	**-5.05 (0.00)**	**-11.98 (0.00)**		**-2.40 (0.02)**	**-3.17 (0.00)**	**-3.13 (0.00)**	**-3.35 (0.00)**	**-6.19 (0.00)**	**-4.90 (0.00)**	**-2.42 (0.02)**	**-3.56 (0.00)**	**-2.63 (0.01)**	**-6.07 (0.00)**
ELASTIC NET	**-4.79 (0.00)**	**-5.34 (0.00)**	**-4.10 (0.00)**	**-8.46 (0.00)**	**-8.56 (0.00)**	**-3.91 (0.00)**	**-6.84 (0.00)**	**-12.96 (0.00)**	**-5.20 (0.00)**	**-11.65 (0.00)**		**-2.86 (0.00)**	**-3.70 (0.00)**	**-3.09 (0.00)**	**-3.52 (0.00)**	**-6.59 (0.00)**	**-4.55 (0.00)**	**-3.20 (0.00)**	**-4.04 (0.00)**	**-2.95 (0.00)**	**-6.10 (0.00)**
MLP NN	**-5.40 (0.00)**	**-5.13 (0.00)**	**-3.88 (0.00)**	**-5.03 (0.00)**	**-5.66 (0.00)**	**-7.32 (0.00)**	**-8.34 (0.00)**	**-4.61 (0.00)**	**-4.01 (0.00)**	**-5.85 (0.00)**		**-2.23 (0.03)**	**-3.71 (0.00)**	**-4.31 (0.00)**	**-2.11 (0.03)**	**-5.58 (0.00)**	**-4.36 (0.00)**	**-6.96 (0.00)**	**-2.71 (0.01)**	**-4.40 (0.00)**	**-3.72 (0.00)**
LASSO & MLP-NN	**-8.01 (0.00)**	**-5.49 (0.00)**	**-3.25 (0.00)**	**-5.59 (0.00)**	**-7.13 (0.00)**	**-4.56 (0.00)**	**-3.63 (0.00)**	**-4.81 (0.00)**	**-6.61 (0.00)**	**-6.15 (0.00)**		**-3.26 (0.00)**	**-3.16 (0.00)**	**-4.44 (0.00)**	**-3.06 (0.00)**	**-3.76 (0.00)**	**-3.62 (0.00)**	**-4.36 (0.00)**	**-2.75 (0.01)**	**-3.71 (0.00)**	**-4.69 (0.00)**
XGBOOST	**-2.46 (0.01)**	-0.96 (0.33)	**-3.87 (0.00)**	**-6.89 (0.00)**	**-1.85 (0.06)**	**-5.25 (0.00)**	**-4.97 (0.00)**	-0.16 (0.87)	**-4.75 (0.00)**	-0.33 (0.73)		-0.64 (0.51)	**-2.66 (0.01)**	**-2.59 (0.01)**	0.56 (0.56)	-1.53 (0.12)	**-1.67 (0.09)**	**-1.93 (0.05)**	**-1.70 (0.08)**	-1.59 (0.10)	**-2.30 (0.02)**
	2010-2014											**2015-2019**									
	DURBL	**ENRGY**	**HITEC**	**HLTH**	**MANUF**	**NODUR**	**OTHER**	**SHOPS**	**TELCM**	**UTILS**		**DURBL**	**ENRGY**	**HITEC**	**HLTH**	**MANUF**	**NODUR**	**OTHER**	**SHOPS**	**TELCM**	**UTILS**
OLS	**-5.22 (0.00)**	**-15.21 (0.00)**	**-8.75 (0.00)**	**-6.55 (0.00)**	**-6.85 (0.00)**	**-7.20 (0.00)**	**-5.43 (0.00)**	**-4.86 (0.00)**	**-4.53 (0.00)**	**-7.46 (0.00)**		**-3.49 (0.00)**	**-6.60 (0.00)**	**-5.83 (0.00)**	**-2.61 (0.01)**	**-5.32 (0.00)**	**-6.43 (0.00)**	**-3.92 (0.00)**	**-3.87 (0.00)**	**-4.23 (0.00)**	**-4.03 (0.00)**
GROUP LASSO	1.71 (0.09)	**-3.80 (0.00)**	**-5.30 (0.00)**	**-2.38 (0.02)**	**-9.19 (0.00)**	**-4.03 (0.00)**	**-4.07 (0.00)**	**-6.03 (0.00)**	**-2.12 (0.03)**	**-6.08 (0.00)**		**-2.71 (0.00)**	**-6.54 (0.00)**	**-5.14 (0.00)**	-1.20 (0.23)	**-2.09 (0.04)**	**-5.48 (0.00)**	**-6.10 (0.00)**	**-2.65 (0.01)**	**-2.13 (0.03)**	**-3.82 (0.00)**
ELASTIC NET	1.16 (0.24)	**-2.00 (0.04)**	**-4.78 (0.00)**	**-2.74 (0.00)**	**-10.50 (0.00)**	**-2.27 (0.02)**	**-4.29 (0.00)**	**-7.50 (0.00)**	-1.05 (0.29)	**-6.05 (0.00)**		**-3.04 (0.00)**	**-5.13 (0.00)**	**-4.72 (0.00)**	-1.27 (0.20)	**-3.59 (0.00)**	**-7.54 (0.00)**	**-4.80 (0.00)**	**-2.91 (0.00)**	**-4.00 (0.00)**	**-3.88 (0.00)**
MLP NN	**-3.60 (0.00)**	**-4.69 (0.00)**	**-3.07 (0.00)**	**-5.71 (0.00)**	**-2.49 (0.01)**	**-2.87 (0.00)**	**-5.20 (0.00)**	**-6.15 (0.00)**	**-6.89 (0.00)**	**-5.09 (0.00)**		**-5.06 (0.00)**	**-4.58 (0.00)**	**-13.21 (0.00)**	**-4.00 (0.00)**	**-5.20 (0.00)**	**-4.26 (0.00)**	**-7.46 (0.00)**	**-4.04 (0.00)**	**-4.21 (0.00)**	**-3.71 (0.00)**
LASSO & MLP-NN	**-5.03 (0.00)**	**-4.72 (0.00)**	**-2.92 (0.00)**	**-4.92 (0.00)**	**-4.63 (0.00)**	**-3.47 (0.00)**	**-4.74 (0.00)**	**-5.92 (0.00)**	**-6.32 (0.00)**	**-4.68 (0.00)**		**-4.10 (0.00)**	**-5.86 (0.00)**	**-7.66 (0.00)**	**-4.74 (0.00)**	**-2.58 (0.01)**	**-6.90 (0.00)**	**-5.41 (0.00)**	**-5.57 (0.00)**	**-5.31 (0.00)**	**-6.98 (0.00)**
XGBOOST	**-2.27 (0.02)**	2.70 (0.00)	2.28 (0.02)	**-1.84 (0.06)**	0.33 (0.73)	**-3.17 (0.00)**	-1.20 (0.22)	**-5.24 (0.00)**	**-2.35 (0.02)**	**-2.38 (0.02)**		0.25 (0.85)	-0.60 (0.54)	-0.97 (0.32)	**-3.31 (0.00)**	-1.30 (0.19)	2.44 (0.01)	0.15 (0.88)	**-3.01 (0.00)**	**-2.93 (0.00)**	**-3.46 (0.00)**

We also evaluate the statistical significance of LassoNet's forecasts against all the benchmark models via the *Superior Predictive Ability* (SPA) test of Hansen (2005) and the *Model Confidence Set* (MCS) test of Hansen et al. (2011) for each of the four OOS subperiods Tables 1.C.3 and 1.C.4 report the findings of the SPA and MCS tests, respectively. Table 1.C.3 shows that LassoNet is not inferior to most benchmarks and for the subperiods examined. Only for the ENRGY and NODUR industries and for the 2010 – 2014 and 2015 – 2019 industries, respectively, is LassoNet inferior to the benchmark models. Table 1.C.4 demonstrates that the LassoNet almost always belongs to the *model confidence set* (i.e., *p*-value greater than nominal statistical levels) and generates the highest *p*-value most of the time. There are only a few exemptions (i.e., ENRGY, HITEC, and NODUR), which does not hold for the 2010 – 2014 and 2015 – 2019 periods. The second-best model is the XGBoost since it is frequently included in the confidence set, while Group Lasso is the third-best, especially for the most recent subperiods. Finally, we do not perform the Giacomini and Rossi (2010) (GR) fluctuation test for each subperiod as it assesses the relative forecasting performance of the proposed model with each of the benchmark models over time based on a rolling window of observations (i.e., 60 months). Hence, the GR test will not provide meaningful results, especially for our subperiods, which are less or equal to five years.

Table 1.C.3. SPA test results for the LassoNet and the employed benchmark models.

The table displays the *p*-values for the SPA test of Hansen (2005) for each OOS subperiod. The test provides insights regarding the relative performance of LassoNet against the employed benchmarks (OLS-Regression, Group Lasso, Elastic-Net, MLP-NN, Lasso-MLP-NN, and XGBoost). Bold *p*-values indicate a failure to reject the null hypothesis that LassoNet is not inferior to the benchmarks.

		DURBL	ENRGY	HITEC	HLTH	MANUF	NODUR	OTHER	SHOPS	TELCM	UTILS
P-VALUE		2000-2006									
	RMSE	**0.515**	**0.928**	**0.717**	**0.509**	**0.525**	**0.542**	**0.733**	**0.542**	**0.501**	**0.602**
	MAE	**0.512**	**0.957**	**0.828**	**0.519**	**0.838**	**0.554**	**0.607**	**0.350**	**0.54**	**0.394**
		2007-2009									
	RMSE	**0.992**	**0.605**	**0.937**	**0.17**	**0.868**	**0.516**	**0.947**	**0.942**	**0.964**	**0.503**
	MAE	**1.000**	**0.699**	**0.82**	**0.142**	**0.901**	**0.896**	**0.985**	**0.922**	**0.907**	**0.513**
		2010-2014									
	RMSE	**0.165**	0.009	**0.408**	**0.585**	**0.407**	**0.520**	**0.541**	**0.608**	**0.874**	**0.537**
	MAE	**0.361**	0.000	**0.338**	**0.938**	**0.302**	**0.529**	**0.791**	**0.601**	**0.882**	**0.514**
		2015-2019									
	RMSE	**0.383**	**0.714**	**0.899**	**0.739**	**0.991**	0.015	**0.451**	**0.691**	**0.879**	**0.806**
	MAE	**0.384**	**0.725**	**0.918**	**0.964**	**0.945**	0.029	**0.695**	**0.776**	**0.886**	**0.504**

Table 1.C.4. MCS test results for the LassoNet and the employed benchmark models.

The table reports *p*-values for Hansen et al.'s (2011) MCS procedure over each OOS subperiod and each industry at a 10% confidence level. A sequence of significance tests is performed to find model forecasts that are not inferior to others. *P*-values that exceed the nominal significance levels (i.e., 1%, 5%, and 10%) show that the model belongs to the MCS. The models belonging to the confidence set at the 10% significance level are reported in bold.

	2000-2006											2007-2009									
	DURBL	ENRGY	HITEC	HLTH	MANUF	NODUR	OTHER	SHOPS	TELCM	UTILS		DURBL	ENRGY	HITEC	HLTH	MANUF	NODUR	OTHER	SHOPS	TELCM	UTILS
OLS	0.010	0.002	0.006	0.000	0.000	0.001	0.034	0.000	0.034	0.000		0.001	0.000	0.000	0.010	0.000	0.001	**0.176**	0.009	0.003	0.002
GROUP LASSO	0.059	0.000	0.006	0.000	0.013	0.046	0.034	0.000	0.070	0.000		**0.229**	0.064	**0.111**	**0.121**	0.000	0.011	**0.176**	0.057	**0.103**	0.002
ELASTIC NET	0.059	0.002	0.015	0.000	0.000	0.065	0.034	0.000	0.067	0.000		**0.120**	0.013	**0.113**	**0.121**	0.000	0.012	0.066	0.009	0.016	0.001
MLP NN	0.000	0.000	0.015	0.000	0.000	0.001	0.003	0.087	0.070	0.002		0.023	0.000	0.000	**0.121**	0.000	0.000	0.000	0.057	0.000	0.000
LASSO & MLP-NN	0.000	0.000	0.000	0.000	0.000	0.012	0.013	0.000	0.000	0.006		0.000	0.013	0.000	**0.116**	0.000	0.000	0.000	0.01	0.000	0.002
XGBOOST	0.001	**0.160**	0.006	0.000	0.053	0.001	0.034	0.906	0.031	**0.805**		**0.496**	0.013	**0.118**	**1.000**	**0.253**	0.082	**0.176**	**0.137**	**0.197**	0.002
LASSONET	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**		**1.000**	**1.000**	**1.000**	**0.355**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	2010-2014											2015-2019									
	DURBL	ENRGY	HITEC	HLTH	MANUF	NODUR	OTHER	SHOPS	TELCM	UTILS		DURBL	ENRGY	HITEC	HLTH	MANUF	NODUR	OTHER	SHOPS	TELCM	UTILS
OLS	0.065	0.000	0.000	0.010	0.000	0.002	0.092	0.015	0.026	0.001		0.064	0.000	0.000	**0.193**	0.022	0.011	**0.106**	0.082	0.097	0.060
GROUP LASSO	**1.000**	0.003	0.005	0.093	0.000	0.000	**0.113**	0.002	0.026	0.002		0.064	0.010	0.017	**0.193**	**0.392**	0.000	0.007	**0.146**	**0.222**	0.062
ELASTIC NET	**0.870**	0.003	0.005	0.093	0.000	0.002	**0.113**	0.002	**0.222**	0.002		0.001	0.044	0.018	**0.193**	0.029	0.002	0.044	**0.146**	0.097	0.060
MLP NN	0.001	0.003	0.036	0.005	0.000	0.005	0.000	0.000	0.000	0.000		0.013	0.000	0.000	0.000	0.001	0.011	0.000	0.025	0.057	0.035
LASSO & MLP-NN	0.001	0.003	0.000	0.003	0.000	0.002	0.000	0.000	0.000	0.000		0.010	0.000	0.000	0.000	0.050	0.000	0.000	0.005	0.000	0.000
XGBOOST	0.001	**1.000**	**1.000**	0.093	**1.000**	0.000	0.113	0.000	0.026	0.001		**1.000**	**0.588**	**0.191**	0.000	**0.392**	**1.000**	**0.912**	0.082	0.043	0.003
LASSONET	**0.312**	0.001	0.015	**1.000**	**0.758**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**		**0.777**	**1.000**	**1.000**	**1.000**	**1.000**	0.012	**1.000**	**1.000**	**1.000**	**1.000**

1.C.2. Covariates importance

1.C.2.1 SAGE value estimation

Positive SAGE values reflect a covariate's contribution to lowering the model's prediction error. Figure 1.C.2 presents the covariates with the three highest SAGE values for every industry and the OOS subperiods examined (i.e., 2000-2006, 2007-2009, 2010-2014, 2015-2019). By observing Figure 1.C.2, we arrive at four significant conclusions. First, the valuation ratios are the covariate category most frequently achieves the highest SAGE values. This finding indicates that valuation ratios are instrumental for predictive tasks involving industry returns and should be regarded as an integral part of the input variable set, especially when forecasting periods characterized by financial distress. In terms of importance, valuation ratios are followed by industry and *other-industry* lagged returns categories. Lagged returns, and therefore past performance, contain valuable information when predicting future returns. Additionally, we prove that solid links and interdependencies exist among the industries reported in previous studies (see Rapach et al., 2015, 2019).

Like the main paper, we calculate the fraction of covariates belonging to the group with the three highest SAGE values across the four OOS subperiods and the ten industries in total. We repeat that computation for every category. Figure 1.C.3 presents the estimated selection rates for the different covariates' categories across the four OOS periods and all industries.

From Figure 1.C.3, it is evident that the valuation ratios have again the highest aggregate presence, as indicated by a selection rate close to 50%. In the second and third place, the industry's lagged returns and *other-industry* lagged returns categories possess similar selection rates, 18.33% and 17.5%, respectively. This time, the macroeconomic variables and financial soundness categories follow with a selection rate of 5.83% and 4.16%, while the remaining categories achieve even lower selection rates.

Figure 1.C.2. SAGE values bar plots.

The figure displays the three covariates with the highest SAGE values for every industry across the four OOS subperiods. We restrict our results to the three covariates with the highest SAGE values to investigate the most significant variables (and the categories they belong to). The bar graphs also include 95% confidence intervals for the SAGE value of each covariate.

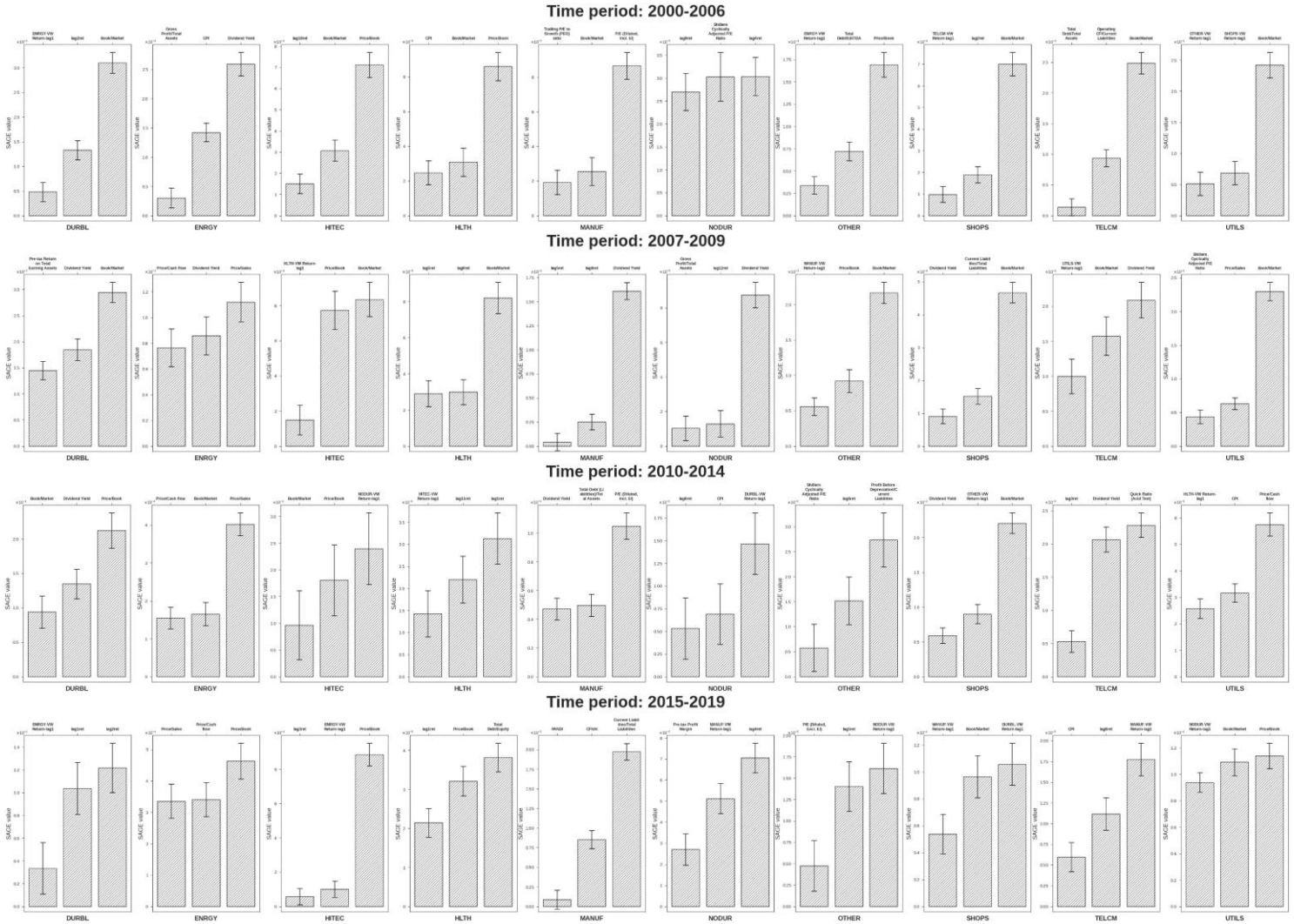
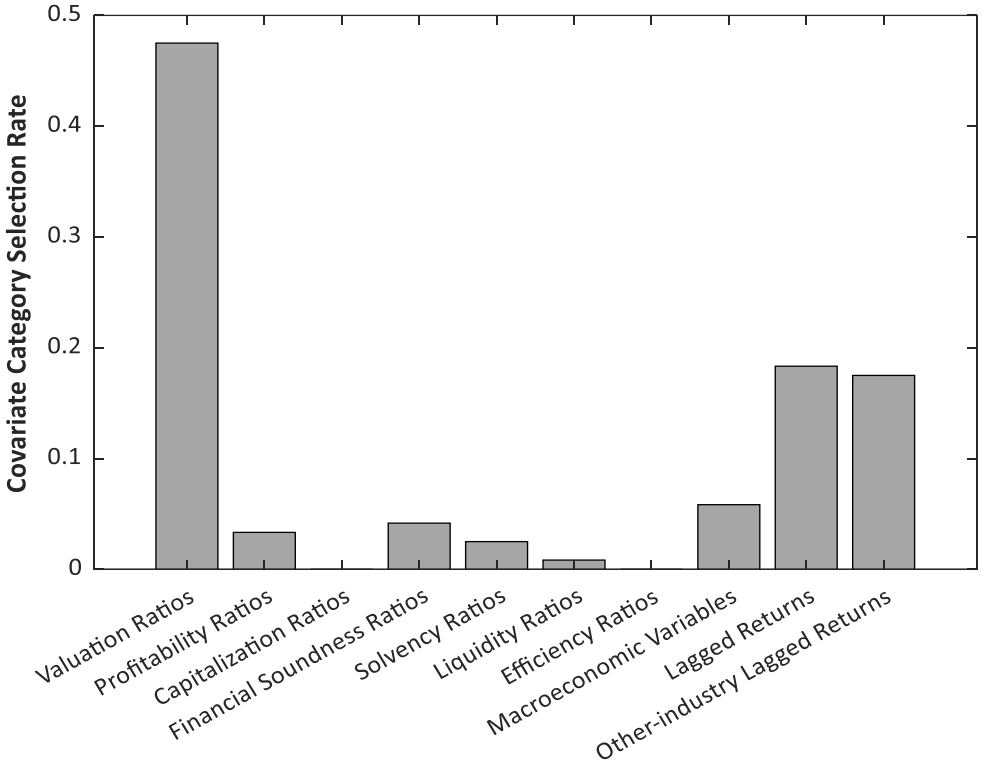


Figure 1.C.3. Selection rates for the covariates' categories.

The figure displays the selection rate for each category's covariates within the three highest positions regarding their corresponding SAGE values across the four OOS subperiods and ten industries.



To quantitatively examine for fluctuations in the positive contribution of each covariate category across time, we conduct cross-industry summations of the SAGE values for each OOS subperiod. The results are displayed in Table 1.C.5 and validate that valuation ratios, industry, and *other*-industry lagged returns categories become increasingly crucial for the model's predictive accuracy during the financial crisis years. In terms of the 2007-2009 period, the positive contribution of the three most crucial categories, we find that the valuation ratios achieve a higher positive contribution to the model when compared to both the industry and *other*-industry lagged returns. This result is directly derived by examining the SAGE value sums' relative magnitude for the covariate categories.

1.C.2.2 Statistical significance of SAGE values across subperiods

Another question is how a category's positive contribution varies over time. To provide a quantitative answer, we perform hypothesis testing. Specifically, we create a set of ten aggregate SAGE values corresponding to each covariate category across all industries for each OOS subperiod. For instance, we sum the SAGE values of all covariates belonging to the same category across all industries.

Table 1.C.5. Aggregate SAGE values across the covariates' categories.

The table reports the cross-industry summation of SAGE values for each of the four OOS subperiods. We obtain an aggregate measure of a category's overall positive contribution to the LassoNet's performance by summing all covariates belonging to a specific category for every OOS subperiod.

		Valuation Ratios	Profitability Ratios	Capitalization Ratios	Financial Soundness Ratios	Solvency Ratios	Liquidity Ratios	Efficiency Ratios	Macroecon. Variables	Lagged Returns	Other-industry Lagged Returns
2000-2006		$3.29\text{x}10^{-3}$	$8.10\text{x}10^{-5}$	-	$2.24\text{x}10^{-4}$	$1.36\text{x}10^{-5}$	-	-	$3.31\text{x}10^{-4}$	$7.33\text{x}10^{-4}$	$3.00\text{x}10^{-4}$
2007-2009		$1.58\text{x}10^{-2}$	$1.54\text{x}10^{-3}$	$5.15\text{x}10^{-6}$	$8.83\text{x}10^{-4}$	$3.23\text{x}10^{-4}$	$2.93\text{x}10^{-4}$	$3.85x10^{-4}$	$1.10\text{x}10^{-3}$	$4.20\text{x}10^{-3}$	$2.18\text{x}10^{-3}$
2010-2014		$3.37\text{x}10^{-3}$	$1.48\text{x}10^{-6}$	$1.44\text{x}10^{-4}$	$5.35\text{x}10^{-6}$	$5.04\text{x}10^{-4}$	$2.28\text{x}10^{-5}$	$5.20\ x10^{-6}$	$1.03\text{x}10^{-4}$	$5.07\text{x}10^{-4}$	$7.55\text{x}10^{-4}$
2015-2019		$2.50\text{x}10^{-3}$	$4.60\text{x}10^{-5}$	-	$2.30\text{x}10^{-4}$	$5.34\text{x}10^{-4}$	$4.02\text{x}10^{-6}$	$3.15\ x10^{-6}$	$2.24\text{x}10^{-5}$	$6.59\text{x}10^{-4}$	$6.47\text{x}10^{-4}$

Then, we create a set of 10 category aggregate SAGE values for each OOS subperiod, and for consecutive periods, we conduct two-tailed pairwise *t*-tests between the aggregate SAGE value sets. Effectively, we conduct hypothesis tests for the following period pairs: (2000-2006) – (2007-2009), (2007-2009) – (2010-2014), (2010-2014) – (2015-2019). This task aims to investigate variations in a category's importance across periods. Table 1.C.6 demonstrates the *t*-statistics of the hypothesis tests for the ten covariate categories and the four OOS periods.

Table 1.C.6. SAGE values hypothesis tests between periods.

The table reports the two-tailed *t*-statistic between category aggregate SAGE values of consecutive OOS subperiods. We sum all industries' SAGE values of all covariates in the same category. We create a set of 10 aggregate SAGE values for all covariate categories and each OOS independently. Given that our first hypothesis test compares 2007-2009 and 2000-2006, the first row of the table is empty. Empty table cells also denote that a hypothesis could not be performed if the LassoNet did not choose any positively contributing covariates for a particular category and any of the periods being compared in the corresponding *t*-test. *, **, *** denote significance at the 10%, 5% and 1% level, respectively.

OOS periods	Val. Ratios	Prof. Ratios	Capt. Ratios	Fin. Sound. Ratios	Solvency Ratios	Liquidity Ratios	Efficiency Ratios	Macro. Variables	Lagged Rets	*OI* Lagged Rets
2000-2006	-	-	-	-	-	-	-	-	-	-
2007-2009	2.00*	0.98	-	1.20	1.62	-	-	0.88	1.88*	2.02*
2010-2014	-1.95*	-1.04	0.96	-1.63	0.34	-1.08	-1.43	-1.16	-2.01*	-1.50
2015-2019	-0.43	0.97	-	1.91*	-0.91	-0.81	-0.34	-1.17	0.46	-0.36

Consistent with previously reported findings, Table C.6 exhibits that the valuation ratios positive contribution increased during the financial crisis (i.e., 2007-2009). Specifically, we observe a positive and statistically significant test statistic of 2.00 at the 5% level for the hypothesis test comparing 2007-2009 with 2000-2006 (i.e., H_0: $\mu_{2007-2009} - \mu_{2000-2006} = 0$) and a negative test statistic of -1.95 for the test comparing 2007-2009 and 2010-2014 periods (i.e., H_0: $\mu_{2010-2014} - \mu_{2007-2009} = 0$). The hypothesis test confirms the significance of the valuation ratios category for any forecasting exercise, especially when analyzing periods characterized by increased financial uncertainty and distress. Our findings are consistent with Bianchi and McAlinn (2021), who report increased explanatory power for financial ratios during recessions when those are used to forecast industry returns. We can also observe increasing importance for the *other-industry* lagged returns category for the financial crisis years. This finding directly results from a positive test statistic of 2.02 for the hypothesis test between the 2007-2009 and 2000-2006 periods (i.e., H_0: $\mu_{2007-2009} - \mu_{2000-2006} = 0$). The economic interpretation of this finding is the existence of stronger cross-industry links and interdependencies during periods of economic crisis, which has crucial implications for any investment and portfolio construction attempt. This finding aligns with the evidence provided in previous literature. Specifically, Rapach et al. (2015, 2019) show that an investment strategy based on model predictions using as inputs individual and *other*-industry lagged returns achieves substantial gains during business cycle

recessions. Additionally, the positive contribution of the lagged returns category also increases during the same period (*t*-statistic = 1.88), while for the financial soundness category, we witness increased importance for the years 2015-2019 compared to 2010-2014.

In the second stage, we employ pairwise hypothesis tests between the covariates' categories to statistically evaluate differences in their positive contribution to predictability. After determining the positive contribution of a specific category, as previously described in this subsection, we repeat this task for the four OOS periods. We effectively create four sets of 10 aggregate SAGE values. Then, we concatenate the four discrete sets into one set for each covariate category.[18] by the time period.

Each set, corresponding to a specific category, has 40 aggregate SAGE values across all OOS subperiods. Finally, we employ pairwise two-tailed *t*-tests between the SAGE values of the covariates categories.

Table 1.C.7 reports the *t*-statistics and the corresponding *p*-values of the cross-category hypothesis tests. Specifically, the hypothesis tests comparing the mean of the valuation ratios against the mean of every other covariate category have a positive and statistically significant *t*-statistic at the 1% level in most cases. This result is not unexpected since valuation ratios have the highest selection rates in the three positions with the highest SAGE values across the industries and the OOS periods. In addition, the *t*-tests also reveal a favourable and statistically significant outcome for the mean positive contribution of the lagged returns category compared to other covariate categories. In this case, the statistical significance is established on average at the 5% level. The only exceptions are the valuation ratios and the *other-industry* lagged returns category. When comparing the mean contribution of lagged returns against *other-industry* lagged returns, the *t*-statistic is positive (i.e., 0.96) but insignificant. The results for the *other-industry* lagged returns are similar to the lagged returns category, except for the *t*-test with the profitability ratios. In this case, the outcome still favours the *other-industry* lagged returns category but is not statistically significant.

[18] Since we are interested exclusively in the cross-category comparison for this task, we do not separate the aggregated SAGE values for a specific category

Table 1.C.7 SAGE values pair-wise hypothesis tests between the covariates' categories.

We sum the SAGE values of all covariates belonging to the same category for each of the ten industries over the four OOS subperiods. For all covariate categories, we create a set of 10 aggregate SAGE values. We then employ pairwise *t*-tests between the covariates' categories to statistically evaluate differences in the positive contribution. We present the *t-statistics* and the corresponding *p*-values for the hypothesis tests in parenthesis. We additionally report the corresponding *p*-values under the Hommel (1988) criterion for *p*-value correction in brackets. The *t*-statistics and *p*-values of the column category with a statistically significant higher positive contribution than row one are presented in bold. *, **, *** denote significance at the 10%, 5% and 1% level, respectively.

	Valuation Ratios	Lagged Returns	Other-industry Lagged Returns	Macroeconomic Variables	Financial Soundness Ratios	Solvency Ratios	Profitability Ratios	Capitalization Ratios	Liquidity Ratios	Efficiency Ratios
Valuation Ratios	-	-	-	-	-	-	-	-	-	-
Lagged Returns	**2.503 (0.016**) [0.016**]**	-	-	-	-	-	-	-	-	-
Other-industry Lagged Returns	**2.878 (0.006***) [0.013**]**	0.962 (0.34) [0.34]	-	-	-	-	-	-	-	-
Macroeconomic Variables	**3.204 (0.003***) [0.008***]**	**2.027 (0.048**) [0.128]**	**1.689 (0.095*) [0.191]**	-	-	-	-	-	-	-
Financial Soundness Ratios	**3.248 (0.002***) [0.007***]**	**2.229 (0.031**) [0.095*]**	**2.125 (0.038**) [0.113]**	0.204 (0.839) [0.949]	-	-	-	-	-	-
Solvency Rations	**3.311 (0.002***) [0.006***]**	**2.456 (0.018**) [0.077*]**	**2.553 (0.013**) [0.053*]**	0.643 (0.523) [0.949]	0.575 (0.567) [0.838]	-	-	-	-	-
Profitability Ratios	**3.147 (0.003***) [0.009***]**	**1.745 (0.085*) [0.171]**	1.214 (0.229) [0.229]	-0.064 (0.949) [0.949]	-0.205 (0.838) [0.838]	-0.492 (0.625) [0.625]	-	-	-	-
Capitalization Ratios	**3.422 (0.001***) [0.006***]**	**2.891 (0.006***) [0.037**]**	**3.539 (0.001***) [0.006***]**	1.56 (0.127) [0.546]	**2.003 (0.051*) [0.24]**	1.362 (0.18) [0.54]	1.017 (0.315) [0.404]	-	-	-
Liquidity Ratios	**3.397 (0.002***) [0.006***]**	**2.794 (0.008***) [0.048**]**	**3.313 (0.002***) [0.009***]**	1.336 (0.188) [0.753]	1.621 (0.111) [0.333]	0.981 (0.331) [0.625]	0.894 (0.376) [0.404]	-0.586 (0.56) [0.56]	-	-
Efficiency Ratios	**3.387 (0.002***) [0.006***]**	**2.754 (0.009***) [0.053*]**	**3.227 (0.002***) [0.012**]**	1.248 (0.218) [0.873]	1.482 (0.144) [0.432]	0.84 (0.404) [0.625]	0.843 (0.404) [0.404]	-0.781 (0.438) [0.56]	-0.195 (0.846) [0.846]	-

1.D. Forecasting accuracy in changing conditions

We compare the relative forecasting performance of the LassoNet with each benchmark model over time and assess the stability over time given changing conditions with the Giacomini and Rossi (2010) (GR) fluctuation test. The GR test defines the local relative loss for the two models as the sequence of rolling OOS loss differences. We consider rolling windows of 90 observations for the overall OOS period (i.e., 2010 – 2019). The null hypothesis assesses the equality of forecasting performance versus the one-sided alternative that the benchmark model forecasts are worse than those of LassoNet at least one point in time. Table 1.D.1 reports the relevant results along with the critical values generated by the GR test at 5% and 10% statistical significance levels. In particular, we present the maximum test statistic as Giacomini and Rossi (2010) suggested for statistical inference under unstable conditions along with the average proportion of rejections of the null hypothesis across industries. Our findings show that the LassoNet is the best model for most industries and all benchmarks, except for a few cases where the null is not rejected (DURABL, NODUR, TELCM, and UTILS).

Table 1.D.1. Giacomini and Rossi (2010) test statistics for the LassoNet and benchmark models for 2010 – 2019.

The table reports the Giacomini and Rossi (2010) fluctuation test statistics for the LassoNet against each benchmark and across all industries. We performed a one-sided test, H0: E[Delta L(t,h)] = 0 for all months t, vs. [Delta L(t,h)] > 0, where DeltaL(t,h) is the difference between the squared forecast errors of the two competing models, L1 being the benchmark model and L2 being the LassoNet. Giacomini and Rossi (2010) report that the t-statistic is calculated as the largest t-statistic over the sequence of the local relative forecast error losses over the 2010 – 2019 OOS period. We also report the proportion of rejections over the OOS. Bold test statistics indicate rejection of the null hypothesis at 5% and 10% significance level (i.e., critical values of 2.08 and 1.74).

Industry	OLS	Group Lasso	Elastic Net	MLP	Lasso-MLP	XGBoost
DURBL	2.45137**	1.58897	2.11564**	4.48722**	1.64475	2.36937**
ENERG	4.42121**	2.69759**	2.51923**	2.50882**	4.07156**	2.96719**
HITECH	3.20637**	2.91216**	3.00463**	5.12442**	5.12289**	4.55903**
HLTH	3.16127**	3.01982**	3.74091**	3.61146**	3.71238**	2.70526**
MANUF	4.73910**	2.65211**	3.42005**	4.56712**	4.88276**	2.10526**
NODUR	2.68160**	2.09735**	1.59148	8.15421**	5.57494**	1.92356*
OTHER	2.21763**	2.24457**	2.51885**	2.87921**	3.38487**	2.27009**
SHOPS	4.17061**	2.14354**	2.22786**	2.73764**	1.85687*	3.10657**
TELCM	4.37874**	1.55920	2.27454**	6.40765**	3.77355**	3.27609**
UTILS	4.57595**	1.97674*	2.62336**	4.17115**	7.37913**	1.21159
Proportion of rejections	0.69677	0.22903	0.39677	0.88710	0.82903	0.48710

1.E. Trading application – robustness checks

In the section, we utilize the out-of-sample (OOS) industry returns predictions based on rolling-window forecasting to create portfolios with their corresponding ETFs by considering more conservative expense ratios and portfolios of the industry portfolio themselves.

1.E.1. ETFs trading application

First, we build monthly long-short portfolios for industry ETFs by selecting the best and worst-performing industries based on the forecasts generated in the preceding sections, , but this time we consider a more conservative expense ratio of 0.50% every time we execute trades on the ETFs. Similarly to the main paper, we report the performance of *Max1-Min1*, *Max2-Min2*, and the *Max3-Min3* spread portfolios. Table 1.E.1 presents the performance of ETF's long-short portfolios constructed based on LassoNet, Group Lasso, Elastic net and XGBoost forecasts as well as numerous benchmarks (i.e., value-weighted CRSP and S&P500 composite indices, equally weighted portfolios of the industry portfolios and selected industry ETFs, one-month ahead spread portfolio of industry ETFs) as described in the main text. Once again, ETF portfolios constructed based on LassoNet's forecasts beat all benchmarks, validating the results of the main paper. The most profitable portfolio is the *Max3-Min3* generating an annualized Sharpe ratio of 2.20 and an annualized five-factor alpha of 18.77%, statistically significant at a 1% significance level.

1.E.2. Industry returns trading application

Second, we use the OOS industry returns forecasts to form directly spread portfolios of different industries. To do so, we sort the industry portfolios each month based on the corresponding return predictions of LassoNet and the rest of the benchmark models (i.e., Group Lasso, Elastic Net and XGBoost). We create long-short portfolios as in the ETF's exercise (i.e., *Max1-Min1*, *Max2-Min2*, and *Max3-Min3)*. We consider ten basis points transaction costs for our strategy. Following Balvers and Wu (2006), we divide the total transaction cost among the industry portfolios, forming the strategy in each portfolio. For instance, the *Max2-Min2* strategy is formed monthly using four industry portfolios, with two of which we construct our long position, while the remaining two form our short position. For the *Max1–Min1, Max2-Min2, and Max3–Min3* portfolios, we count each switch of one of the two, three, or six industry portfolios as 1⁄2, 1⁄4, and 1⁄6 of the total transaction costs. Suppose for a specific month, we switch only the long position in the *Max1-Min1* portfolio to a different industry portfolio. This action will account for 1/2 of the total transaction costs since the short position will remain unaltered. We keep almost the same benchmarks as in the ETF's exercise, apart from the equally-weighted portfolio of ETFs, while this time, we create a one-month ahead portfolio, which short the

industries' returns in the current month to trade their spread on the following month. Table 1.E.2 reports the relevant findings for the investment strategies' performance.

Notably, all our deep learning investment portfolios, except the XGBoost, generate higher returns and positive and statistically significant alphas than the benchmarks. LassoNet is once again the superior model in terms of economic significance. At the same time, the overall findings demonstrate that it can effectively generate abnormal returns not captured by seminal factor models while minimizing the downside risk. Hence, it authenticates LassoNet's ability to provide economically meaningful forecasts, which fund managers and trading desks can utilize.

Table 1.E.1 Performance of industry ETF portfolios based on OOS forecasts.

The table demonstrates performance metrics for trading strategies of industry ETFs based on LassoNet's, Group Lasso, Elastic Net and XGBoost forecasts and benchmark strategies over the 2010 – 2019 OOS period. The *Max1-Min1*, *Max2-Min2*, and *Max3-Min3* industry ETFs spread portfolios are constructed based on the highest and lowest-performing industries according to their corresponding forecasts while considering a 0.50% expense ratio. We report the annualized mean return and Sharpe ratio, maximum drawdown and annualized alphas. The reported annualized alphas are obtained using the 4-factor (Carhart, 1997) and 5-factor (Fama & French, 2015) models. *, **, *** denote significance at the 10%, 5% and 1% level, respectively.

	Portfolios				**Benchmark Strategies**				
***Panel A:* Max1-Min1**	**LASSONET**	**GROUP LASSO**	**ELASTIC NET**	**XGBOOST**	**VW-CRSP**	**SP500**	**EW-Ind. Port**	**EW-Ind. ETFs**	**1 MAH**
Ann. Mean Return (%)	19.89	18.13	19.00	4.46	12.32	11.46	6.24	7.28	-9.10
Volatility (%)	14.74	15.07	14.65	12.49	12.72	12.46	12.15	12.46	16.53
Ann. Sharpe ratio	1.30	1.16	1.26	0.31	0.93	0.88	0.47	0.54	-0.58
Max Drawdown (%)	20.80	19.80	16.41	33.89	18.52	17.04	17.69	17.86	100.09
Ann. 4-factor alpha (%)	18.85***	17.40***	17.71***	3.56	-	-	5.23**	6.22**	-11.57**
Ann. 5-factor alpha (%)	17.92***	16.50***	16.95***	2.12	-	-	5.39**	6.18**	-12.94***
Panel B: Max2-Min2	**LASSONET**	**GROUP LASSO**	**ELASTIC NET**	**XGBOOST**	**VW-CRSP**	**SP500**	**EW-Ind. Port**	**EW-Ind. ETFs**	**1 MAH**
Ann. Mean Return (%)	20.07	15.45	15.02	1.29	12.32	11.46	6.24	7.28	-4.52
Volatility (%)	10.16	10.39	10.26	9.86	12.72	12.46	12.15	12.46	12.80
Ann. Sharpe ratio	1.92	1.44	1.41	0.08	0.93	0.88	0.47	0.54	-0.39
Max Drawdown (%)	6.93	8.18	9.09	18.25	18.52	17.04	17.69	17.86	54.44
Ann. 4-factor alpha (%)	19.62***	15.00**	14.61**	0.87	-	-	5.23**	6.22**	-6.94*
Ann. 5-factor alpha (%)	19.13***	14.34**	14.14**	0.26	-	-	5.39**	6.18**	-7.02*
Panel C: Max3-Min3	**LASSONET**	**GROUP LASSO**	**ELASTIC NET**	**XGBOOST**	**VW-CRSP**	**SP500**	**EW-Ind. Port**	**EW-Ind. ETFs**	**1 MAH**
Ann. Mean Return (%)	18.61	11.57	10.59	0.45	12.32	11.46	6.24	7.28	-7.33
Volatility (%)	8.21	8.35	8.05	7.34	12.72	12.46	12.15	12.46	10.37
Ann. Sharpe ratio	2.20	1.32	1.25	-0.09	0.93	0.88	0.47	0.54	-0.72
Max Drawdown (%)	4.70	6.64	10.29	18.18	18.52	17.04	17.69	17.86	6.65
Ann. 4-factor alpha (%)	18.28***	9.94**	8.02**	0.08	-	-	5.23**	6.22**	-8.75**
Ann. 5-factor alpha (%)	17.76***	9.00**	7.22**	-0.77	-	-	5.39**	6.18**	-9.05**

Table 1.E.2 Performance of industry portfolios based on OOS forecasts.

The table demonstrates performance metrics for trading strategies of industry portfolios based on LassoNet's, Group Lasso, Elastic Net and XGBoost forecasts and benchmark strategies over the 2010 – 2019 OOS period. The *Max1-Min1*, *Max2-Min2*, and *Max3-Min3* industry spread portfolios are constructed based on the highest and lowest-performing industries according to their corresponding forecasts and considering one-way transaction costs of 10 basis points. We report the annualized mean return and Sharpe ratio, maximum drawdown and annualized alphas. The reported annualized alphas are obtained using the 4-factor (Carhart, 1997) and 5-factor (Fama & French, 2015) models. *, **, *** denote significance at the 10%, 5% and 1% level, respectively.

	Portfolios				**Benchmark Strategies**			
***Panel A :* Max1-Min1**	**LASSONET**	**GROUP LASSO**	**ELASTIC NET**	**XGBOOST**	**VW-CRSP**	**SP500**	**EW-Ind. Port**	**1 MAH**
Ann. Mean Return (%)	21.33	20.13	20.64	5.55	12.32	11.46	12.68	-2.58
Volatility (%))	14.46	15.96	15.11	13.31	12.72	12.46	12.46	16.37
Ann. Sharpe ratio	1.42	1.22	1.33	0.38	0.93	0.88	0.97	-0.18
Max Drawdown (%)	-21.04	-23.00	-18.75	-34.65	18.52	17.04	17.86	52.22
Ann. 4-factor alpha (%)	19.46***	18.53***	18.61***	4.23	-	-	11.72***	-3.87
Ann. 5-factor alpha (%)	18.74***	17.71***	18.05***	3.25	-	-	10.33***	-4.28*
Panel B: Max2-Min2	**LASSONET**	**GROUP LASSO**	**ELASTIC NET**	**XGBOOST**	**VW-CRSP**	**SP500**	**EW-Ind. Port**	**1 MAH**
Ann. Mean Return (%)	22.75	17.97	17.38	5.20	12.32	11.46	12.68	-1.34
Volatility (%))	10.12	11.03	11.11	9.98	12.72	12.46	12.46	12.62
Ann. Sharpe ratio	2.19	1.58	1.51	0.47	0.93	0.88	0.97	0.07
Max Drawdown (%)	-6.95	-9.78	-9.91	-11.99	18.52	17.04	17.86	27.53
Ann. 4-factor alpha (%)	19.98***	16.94***	16.54***	4.07	-	-	11.72***	-1.32
Ann. 5-factor alpha (%)	19.09***	16.03***	16.00***	2.79	-	-	10.33***	-2.51
Panel C: Max3-Min3	**LASSONET**	**GROUP LASSO**	**ELASTIC NET**	**XGBOOST**	**VW-CRSP**	**SP500**	**EW-Ind. Port**	**1 MAH**
Ann. Mean Return (%)	14.52	10.34	9.56	2.48	12.32	11.46	12.68	-0.52
Volatility (%))	6.17	6.92	6.91	6.32	12.72	12.46	12.46	10.47
Ann. Sharpe ratio	2.26	1.42	1.31	0.32	0.93	0.88	0.97	-0.09
Max Drawdown (%)	-3.60	-8.19	-7.99	18.86	18.52	17.04	17.86	33.35
Ann. 4-factor alpha (%)	13.55***	10.14***	9.00***	1.65	-	-	11.72***	-2.18
Ann. 5-factor alpha (%)	12.27***	9.22***	8.65***	0.86	-	-	10.33***	-3.13

1.F. Robustness check: 49 industry portfolios

1.F.1. Forecasting accuracy

Finally, we run the same forecasting experiment using a more granular U.S. industry portfolio returns. In particular, we use 49 industry portfolios from Kenneth French's website.[19] Again, we employ the same rolling-window forecasting exercise for 1985-2019, using January 2010 to December 2019 as the OOS, in which the forecasting horizon is one month ahead. We also use the same covariates (i.e., 88 covariates). However, this time, we use a panel data format for our prediction task. This is for practical reasons in presenting the forecasting performance of LassoNet and the benchmark models for 49 industries. Table 1.F.1 reports the average OOS RMSE and MAE criteria across the industries examined. The overall picture validates the relevant findings of the main paper. LassoNet is the superior model in forecasting industry portfolios, with XGBoost, Elastic Net and Group Lasso being the second, third and fourth-best models. Interestingly, the simple linear regression performs better than the remaining deep learning models (i.e., MLP and Lasso-MLP).

Table 1.F.1. OOS statistical performance for the LassoNet and the employed benchmark models.

The table reports the OOS statistical performance over the 2010 – 2019 period across 49 industries. We compare the performance of the LassoNet model against the employed benchmarks (i.e., OLS-Regression, Group-Lasso, Elastic-Net, MLP-NN, Lasso-ANN-MLP, and XGBOOST). We report the root mean squared error (RMSE) and the mean absolute Error (MAE) as error metrics. The lowest values are reported in bold.

Panel B: *Error Metric*	**LASSONET**	**OLS**	**GROUP LASSO**	**ELASTIC NET**	**MLP**	**LASSO - MLP**	**XGBOOST**
RMSE	**0.0542**	0.0677	0.0662	0.0656	0.0812	0.0749	0.0563
MAE	**0.0420**	0.0511	0.0503	0.0501	0.0637	0.0595	0.0437

To assess the statistical significance of LassoNet's forecast against the benchmark models, we conduct the Diebold and Mariano (1995) (DM) test for forecasting accuracy while using the squared error as a loss function. Once again, a negative and significant t-statistic rejects the null hypothesis that two forecasts have equal predictive ability, and it shows the superiority of LassoNet against the benchmark (i.e., lower loss). Table 1.F.2., presents the DM t-statistics and their corresponding *p-values* (in parenthesis). All *p-values* are below the significance thresholds, and their corresponding *t*-statistics are negative enough to prove that LassoNet has superior predictive ability against the benchmarks for all the 49 industries.

[19] The industry definitions and their corresponding returns can be found on Kenneth French's website: https://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html.

Table 1.F.2. Diebold Mariano test results for the LassoNet against benchmark models.

The table displays the t-statistics and p-values of the D.M. (1995) test for LassoNet against each benchmark pairwise across 49 industries for the 2010-2019 OOS period. The null hypothesis is defined as the LassoNet's and benchmark forecasts having equal predictive ability. Bold *p*-values and *t*-statistics indicate that we reject the null hypothesis of the two forecasts' equivalence and show the superiority of LassoNet against the benchmark.

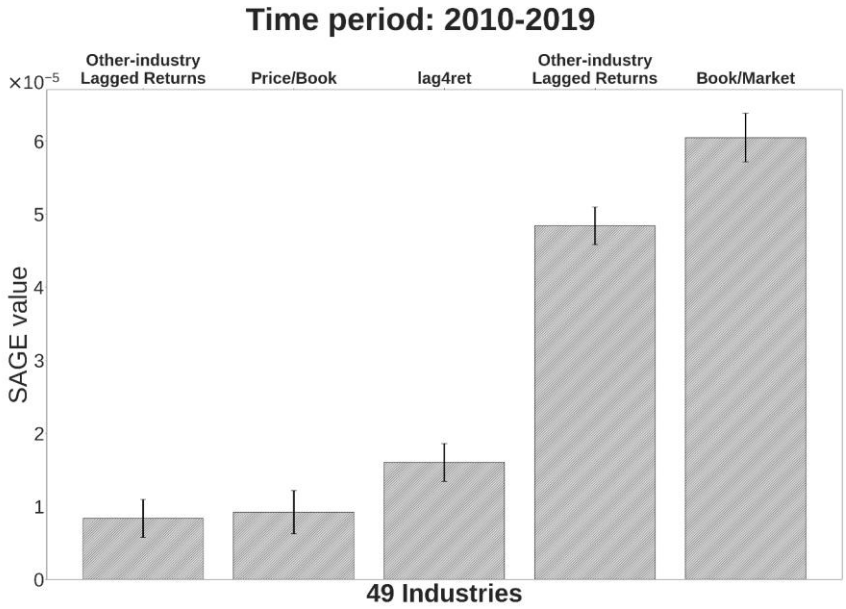
DM test: *t-statistic* (*p*-value)					
LASSONET					
OLS	GROUP LASSO	ELASTIC_NET	MLP	LASSO_MLP	XGBOOST
-2.85 (0.002)	**-2.85 (0.002)**	**-2.91 (0.001)**	**-7.08 (0.000)**	**-6.62 (0.000)**	**-1.84 (0.073)**

1.F.2. Covariates importance

In this section, we examine the covariates driving the LassoNet forecasts across the 49 industries by computing the aggregate SAGE values of the forecasts. Figure 1.F.1 presents the five covariates with the highest SAGE values OOS across all industries, along with 95% confidence intervals of the mean SAGE value of each covariate.[20]

Figure 1.F.1. SAGE values bar plots.

The figure displays the three covariates with the highest SAGE values for all 48 industries across the 2010 – 2019 OOS period. We restrict our results to the five covariates with the highest SAGE values to investigate the most significant variables (and the categories they belong to). The bar graphs also include 95% confidence intervals around the mean SAGE value of each covariate.



[20] We decided that presenting the five covariates with the highest SAGE values would be more informative since we forecast 49 industries in a panel data framework.

Consistent with our findings on forecasting ten industry portfolios, valuation ratios, specifically Book/Market and Price/Book, along with other-industry lagged returns and the lagged returns of the same industry, are the most important in contributing to LassoNet's predictability.

1.F.3. Trading application

We also evaluate the economic significance of the 49 industry portfolios' forecasts. We follow the same trading exercise as in the main manuscript (i.e., forming spread portfolios), but this time, we invest directly in the industry portfolios because of the lack of availability of ETFs for each industry. To do so, we form long-short industry portfolios each month based on the corresponding return predictions of LassoNet and the rest of the benchmark models (i.e., Group Lasso, Elastic Net and XGBoost). We consider top-bottom 5%, 10% and 15% portfolios of the highest and lowest-performing industries. Our target is to evaluate the performance of a more granular subset of industry spread portfolios since we forecast a large set of industries (i.e., 49). However, we know that a passive investor would choose to invest only in a few ETFs, contrary to an active investor who trades numerous equities. Again, we consider ten basis points transaction costs and divide the total transaction cost among the industry portfolios, forming our strategy while considering the same benchmarks. Table 1.F.3. reports the same performance metrics of each portfolio and benchmarks as the ones computed in the main manuscript and previous sections. All long-short portfolios constructed based on LassoNet industry forecasts outperform the portfolios constructed on the rest of deep learning benchmarks and benchmark strategies.

Table 1.F.3 Performance of 49 industry portfolios based on OOS forecasts.

The table demonstrates performance metrics for trading strategies of 49 industry portfolios based on LassoNet's, Group Lasso's, Elastic Net and XGBoost's forecasts and those of benchmark strategies over the 2010 – 2019 OOS period. The *top-bottom 5%, 10% and 15%* industry spread portfolios are constructed based on the highest and lowest-performing industries according to their corresponding forecasts and by considering one-way transaction costs of 10 basis points. We report the annualized mean return and Sharpe ratio, maximum drawdown and annualized alphas. The reported annualized alphas are obtained using the 4-factor (Carhart, 1997) and 5-factor (Fama & French, 2015) models. *, **, *** denote significance at the 10%, 5% and 1% level, respectively.

	Portfolios				**Benchmark Strategies**			
Panel A: Top-Bottom 5%	**LASSONET**	**GROUP LASSO**	**ELASTIC NET**	**XGBOOST**	**VW-CRSP**	**SP500**	**EW-Ind. Port**	**1 MAH**
Ann. Mean Return (%)	27.98	26.82	24.54	24.23	12.32	11.46	12.74	-4.67
Volatility (%))	17.52	17.36	17.25	15.68	12.72	12.46	14.20	26.94
Ann. Sharpe ratio	1.59	1.51	1.39	1.51	0.93	0.88	0.86	-0.19
Max Drawdown (%)	24.04	22.20	22.20	9.69	18.52	17.04	22.25	76.97
Ann. 4-factor alpha (%)	26.83***	26.68***	24.06***	22.41***	-	-	10.76**	-8.06**
Ann. 5-factor alpha (%)	26.24***	26.00***	23.76***	22.16***	-	-	9.77**	-8.01**
Panel B: Top-Bottom 10%	**LASSONET**	**GROUP LASSO**	**ELASTIC NET**	**XGBOOST**	**VW-CRSP**	**SP500**	**EW-Ind. Port**	**1 MAH**
Ann. Mean Return (%)	24.81	24.15	24.08	18.85	12.32	11.46	12.74	-0.66
Volatility (%))	11.15	11.22	11.87	8.92	12.72	12.46	14.20	16.41
Ann. Sharpe ratio	2.17	2.10	2.02	2.05	0.93	0.88	0.86	-0.07
Max Drawdown (%)	6.60	6.86	12.27	7.82	18.52	17.04	22.25	40.53
Ann. 4-factor alpha (%)	24.24***	23.54***	22.67***	18.47***	-	-	10.76**	-3.44
Ann. 5-factor alpha (%)	22.80***	22.00***	20.10***	18.18***	-	-	9.77**	-3.61
Panel C: Top-Bottom 15%	**LASSONET**	**GROUP LASSO**	**ELASTIC NET**	**XGBOOST**	**VW-CRSP**	**SP500**	**EW-Ind. Port**	**1 MAH**
Ann. Mean Return (%)	21.33	21.30	20.78	16.63	12.32	11.46	12.74	0.26
Volatility (%)	7.90	7.74	8.97	6.98	12.72	12.46	14.20	13.30
Ann. Sharpe ratio	2.68	2.63	2.31	2.30	0.93	0.88	0.86	-0.02
Max Drawdown (%)	4.63	4.02	8.14	3.65	18.52	17.04	22.25	25.23
Ann. 4-factor alpha (%)	20.56***	20.37***	20.09***	16.02***	-	-	10.76**	-2.58
Ann. 5-factor alpha (%)	19.89***	19.76***	19.11***	15.38***	-	-	9.77**	-2.43

CHAPTER 2

Imputing Hedge Fund Datasets via Bi-directional Deep Learning

2.1. Introduction

A common problem in financial datasets—especially those for hedge funds—is the presence of missing values. Missing values present challenges for many topics in asset pricing, including model estimation and out-of-sample return forecasting. Thus, the effective imputation of missing data can substantially improve empirical asset pricing, including out-of-sample asset return prediction. Despite its importance, the issue of missing values in financial datasets has only recently started to receive significant attention in the finance literature (e.g., Giglio et al., 2021; Freyberger et al., 2021; Bryzgalova et al., 2024; Beckmeyer and Wiedemann, 2023; Chen and McCoy, 2024). This paper examines the imputation of missing values in hedge fund datasets. Missing values are a particularly glaring issue for hedge fund datasets, so imputing missing data has important implications for empirical research on hedge funds. We consider imputing missing values for both hedge fund returns and predictors, analyzing the accuracy of the imputations and how they affect out-of-sample hedge fund return prediction.

To deal with missing data in hedge fund datasets, we employ a deep learning model that considers both time-series and cross-sectional properties to impute missing values for hedge fund returns and a large set of fund characteristics. Specifically, we use the bidirectional recurrent imputation network for time series (BRITS, Cao et al., 2018) method that applies to general settings for missing data. Several BRITS properties make it attractive to fill in missing values for hedge fund datasets. First, BRITS network uses information from a variable's past and future values to impute missing values. This bidirectional flow provides a richer information set that is especially useful in sparse datasets with relatively large numbers of missing values and helps the model achieve the same quality of imputation at the beginning and the end of a variable's time series. Second, the BRITS network uses information from the entire cross-section of available observations for a variable, which aligns with the dependency of financial variables on common factors such as business-cycle fluctuations. Third, BRITS attaches less weight to observations in the distant past when imputing missing values. This seems reasonable, as observations from the recent past are likely to have more of an effect on a variable's current value than those from the more distant past. Fourth, BRITS is a recurrent neural network that can effectively model data characterized by nonlinear dynamics. Overall, the properties of BRITS make it well suited for imputing missing values for financial data, including hedge fund data.

We consider data for 3,800 hedge funds from January 1994 to November 2021. Our dataset consists of a time series for hedge fund returns, along with 23 hedge fund predictors that are potentially relevant for

forecasting future hedge fund returns. Among the predictors we consider are past returns, return autocorrelations, higher return moments, and measures of fund manager skill. After making standard adjustments to the hedge fund return data and accounting for the availability of hedge fund predictors' data, the in-sample period is January 1998 to December 2012, while January 2013 to November 2021 serves as the out-of-sample period for evaluating hedge fund return forecasts based on the fund predictors and their interactions with a set of economic variables.

We first experiment over the in-sample period to investigate the accuracy of BRITS in imputing missing values for hedge fund returns and predictors relative to a set of benchmark methods. For benchmarks, we consider the cross-sectional mean (e.g., Haugen and Baker, 1996; Green et al., 2017; Light et al., 2017), the time-series mean, and the singular value thresholding algorithm for matrix completion (Cai et al., 2010). In the experiment, we randomly drop a portion of the observed values for a hedge fund variable and fill in the actual and artificial missing values using an imputation method. For the artificial missing values, we use the filled-in and actual values to compute the root mean squared error (RMSE), thereby providing a measure of the accuracy of the imputation method. The BRITS method produces a lower RMSE than the benchmarks for the artificial missing values, so BRITS imputes missing values for hedge fund variables with greater fidelity.

Next, we assess the performance of BRITS in an out-of-sample (OOS) forecasting and trading context. First, we forecast hedge fund returns using a large set of variables comprised of the 23 fund predictors and their interactions with four economic variables (for a total of 115 predictors). We generate a sequence of monthly hedge fund return forecasts for all available funds in a given month using a rolling window of 15 years (180 months). Specifically, to generate return forecasts for the month $t + 1$, we first consider available data for hedge fund returns and predictors for the month $t - 179$ to month t. We apply the BRITS method to fill in missing values for the return and predictors' data. We then use the complete set of available and filled-in observations for the returns and predictors (as well as the economic variables, which do not have missing values) for the month $t - 179$ to month t to train a prediction model that generates a set of return forecasts for the month $t + 1$. We do this sequentially to generate hedge fund return forecasts for January 2013 to November 2021. Second, we use the OOS forecasts to form equally-weighted decile portfolios. Specifically, we sort the hedge funds in the cross-section into deciles based on the OOS forecasts and then focus on the top decile of the highest-performing hedge funds to create equally-weighted portfolios of a long-only strategy of hedge funds. We exclusively form long positions on the top fund decile since hedge funds are not tradable assets that investors can short. Instead, investors can only decide which funds to allocate their capital to and from which funds to withdraw their capital. Therefore, by the end of each month, we invest evenly all capital available (i.e., form long positions) in the funds belonging to the top decile portfolio based on the forecasts.

For the prediction models, we employ machine learning methods from three classes: gradient-boosted trees, neural networks, and linear penalized regression. For gradient-boosted trees, we use the XGBoost (Chen and Guestrin, 2016), LightGBM (Ke et al., 2017), and CatBoost (Prokhorenkova et al., 2018) algorithms. We consider three neural network architectures, including deep neural networks. For linear penalized regression, we use the seminal least absolute shrinkage and selection operator (LASSO, Tibshirani, 1996) as well as two of its variants: adaptive LASSO (Zou, 2006), and sparse group LASSO (Simon et al., 2013). For the various machine learning methods, out-of-sample hedge fund return forecasts based on filling in missing values (for both hedge fund returns and predictors) with BRITS are substantially more accurate in terms of RMSE and MAE than return forecasts that rely on filling in missing values via the cross-sectional mean. Moreover, the trading application results outline the economic significance of BRITS imputations. The models' forecasts that were trained on BRITS imputed returns and predictors data lead to more profitable trading decisions as measured by higher portfolio annualized returns, Sharpe ratios, and alphas. On the contrary, the equally-weighted decile portfolios constructed based on the forecasts of the models trained on cross-sectional mean imputed data attain inferior portfolio performance metrics. In the context of forecasting hedge fund returns with a large pool of predictors and creating fund portfolios, BRITS provides an effective strategy for dealing with the plethora of missing values in hedge fund datasets.

In sum, we make three primary contributions to the literature. First, we show that BRITS is an efficacious deep-learning tool for dealing with missing values in hedge fund datasets. Because missing values are a pervasive problem in hedge fund datasets, this makes BRITS a valuable resource for empirical research on hedge funds. The effectiveness of BRITS in filling in missing values lies in its ability to glean information along both the time and cross-sectional dimensions and to accommodate nonlinear dynamics. BRITS also does not require strong data assumptions, providing a flexible approach for filling in missing values in financial datasets. Second, we provide the most comprehensive analysis on out-of-sample hedge fund return predictability. Our analysis incorporates many hedge fund return predictors, including numerous hedge fund characteristics and their interactions with a set of economic variables, as well as a broad array of machine learning models.[21] When we impute missing values for hedge fund datasets via BRITS, we find that combining many predictors and machine learning methods generates significant improvements in out-of-sample hedge fund return predictability. Third, the information advantage associated with BRITS imputed datasets enables the predictive models to generate more profitable forecasts, as highlighted in our trading application's results. The forecasts of machine learning methods, which were trained on fully recovered datasets by BRITS, can effectively identify which hedge funds will be the most profitable next month. Such crucial piece of information can be leveraged

[21] Existing studies that investigate out-of-sample hedge fund return predictability include Avramov et al. (2011, 2013) and Wu et al. (2021)

by institutional investors when constructing their portfolios and deciding on which hedge funds to allocate their capital.

The rest of the chapter is organized as follows. Section 2.2 describes the data. Section 2.3 outlines the BRITS network for imputing missing values. Section 2.4 reports the empirical results. Section 2.5 concludes.

2.2. Literature Review

Financial literature spreads on two broad categories of studies that refer to the issue of missing values. Those that employ conventional techniques to handle missing values, such as the cross-sectional mean (see, Haugen and Baker, 1996; Kelly and Jiang, 2014; Green et al., 2017; Light et al., 2017; Kozak et al., 2020; Gu et al., 2020) and those that develop new methods for imputing missing values to improve the performance of existing ones (see, Freyberger et al, 2021; Giglio et al., 2021; Beckmeyer and Wiedemann, 2022; Bryzgalova et al., 2022).

When it comes to the first class of studies, Haugen and Baker (1996) employ factor models to analyze the cross-section of stock returns using data from all stocks represented in the Russell 3000 stock index for the period from 1979 to 1993. The authors adopt the cross-sectional mean method to impute missing values, since it creates less bias than removing a stock entirely from the sample. Light et al. (2017) also handle missing observations by using the cross-sectional mean approach to fill in missing observations of U.S. monthly stock returns and firm characteristics for the period January 1980 to December 2014. Recently Kozak et al. (2020) treat missing U.S. stock market data by focusing on the cross-sectional mean imputation to form 50 portfolios of anomalies characteristics for the sample period of November 1973 to December 2017. Similarly, Gu et al. (2020) examine all listed firms in NYSE, AMEX, and NASDAQ and use the cross-sectional median to handle the missing entries for a comprehensive sample that spans from March 1957 until December 2016. On the other hand, Kelly and Jiang (2014) create a subset of the original dataset based on the condition that the included stocks will have at least 36 months observed out of the 120 months examined.

Regarding new and more efficient approaches to imputing missing entries, Freyberger et al. (2021), develop a linear method that uses a GMM estimator to impute missing values. The authors use the common stock universe of firms trading in the NYSE, AMEX, and NASDAQ exchanges as their data application setting and state that their imputing procedure provides significant benefits in terms of OOS predictability. Furthermore, their framework operates under the limitation that 18 stock characteristics are observable so that they can drive the imputation process for the rest of the characteristics. The GMM procedure operates based on the setup that the imputed predictors should explain the stock return movements. Therefore, the imputation is oriented towards filling values that best explain stock returns and not uncovering the actual missing values of the predictors. In their research, Giglio et al. (2021) leverage a matrix completion method to fill hedge funds'

returns for the sample period 1994 – 2018. The authors use a nuclear norm penalized regression approach to recover missing hedge fund returns entries, which relies on the assumption that the returns matrix can be rewritten as a noisy low-rank matrix. While the low-rank assumption may be justified in the context of the study and the hedge fund returns case, it is generally restricting and potentially not optimal to handle other types of datasets, such as predictors' data. Bryzgalova et al. (2022), employ principal component analysis (PCA) and linear regression to handle missing values. Based on the latent factors of observed stock characteristics, the model can impute missing values for a sample that covers thousands of individual stocks and spans from January 1967 until December 2020. Their modeling framework can be adjusted to include cross-sectional and time series information, but it lacks capturing non-linear relationships. Finally, Beckmeyer and Wiedemann (2022) propose an advanced imputation method that does not make any assumptions on the data. Their sample contains 153 stock characteristics for all firms trading in NYSE, AMEX, and NASDAQ exchanges, and it covers the period from July 1962 to December 2020. Despite the advantage of the proposed framework in dealing with non-linear relations, the adopted deep learning setup only predicts in which percentile the missing stock characteristics belong and not the actual value of the characteristic. While the recovery of the missing observations' percentiles can be appropriate for specific financial applications, this method is not flexible enough to accommodate the direct filling of missing values.

Our imputing framework is not affected by any of the limitations described above. BRITS does not make any assumptions about the structure of the data and can effectively use both time series, and cross-sectional information to effectively impute missing observations. Finally, several innovative properties differentiate our modeling framework from other conventional and unconventional methods and can lead to an enhanced imputation fidelity that other models cannot achieve.

2.3. Data

We use hedge fund data from the Lipper Trading Advisor Selection System (TASS) database for January 1994 to November 2021. Following standard procedure in the hedge fund literature, we apply a set of filters to the data before using it for our analysis (e.g., Fung and Hsieh, 2000; Aragon, 2007; Bali et al., 2012, 2021; Chen et al., 2021; Wu et al., 2021). We follow Chen et al. (2021) and consider only US-oriented hedge funds (to avoid fund duplicates in different currencies) as well as funds that report net-of-fees returns. In terms of the filters, we preclude survivorship bias by including both live and defunct funds. To account for backfill bias, we delete each fund's first twelve months of returns. In addition, to remove multi-period sampling bias, we require each fund to have at least 30 return observations[22]. Finally, we discard funds with assets under management below $5 million, so we focus on large funds that are more relevant to investors and that are

[22] For robustness purposes, we follow Giglio et al. (2021) and require each fund to have at least 60 return observations across the full sample period for the imputation experiment in Section 2.4.2.

less likely to manipulate reporting to TASS. After applying the filters, we have 3,800 hedge funds for our analysis.

With respect to hedge fund return predictors, we consider a collection of 23 characteristics that are plausible fund return predictors and/or have been found to predict fund returns (e.g., Titman and Tiu, 2011; Bali et al., 2012, 2019; Heuson et al., 2020; Wu et al., 2021). We group the hedge fund predictors into four categories, as indicated in Panels A through D of Table 2.1. The first category consists of lagged cumulative returns ranging from one to 36 months, which are designed to capture short-, medium-, and long-term momentum effects. The second category is comprised of the first- through third-order autocorrelations in hedge fund returns over the last twelve months. These provide additional measures for capturing persistence in hedge fund returns. The next category is comprised of various return moments, including measures of volatility, coskewness, skewness, and kurtosis. Predictors that provide proxies for managerial skill make up the fourth category. This category includes alphas based on different multifactor models for hedge fund returns, the R^2 statistic in the context of the well-known Fung and Hsieh (2004) seven-factor model, assets under management, and the maximum return of the last twelve months. In addition, we consider four economic variables (see Panel E of Table 2.1, which we interact with the fund characteristics. This allows for the predictive relations between the predictors and future hedge fund returns to vary with economic conditions. The four economic variables are the equity market uncertainty index (Economic Policy Uncertainty website), the economic policy uncertainty index (Baker et al., 2016), the TED spread, and the CBOE volatility index (VIX). After allowing for the hedge fund predictors to interact with the economic variables, we have a total of 23 + 4 × 23 = 115 predictors.

Table 2.1. The employed hedge fund predictor set.

This table presents the predictor set used for the modeling purposes of our study along with the corresponding symbol of each predictor we use in the following section.

Predictor variable	Abbreviation
Panel A: Previous returns	
One month return	**PastR**
Three-month cumulative return	**CR3**
Six-month cumulative return	**CR6**
Nine-month cumulative return	**CR9**
12-month cumulative return	**CR12**
36-month cumulative return	**CR36**
Panel B: Previous return autocorrelations	
Lag 1 autocorrelation of the past 12 months' returns	**ALg1**

Lag 2 autocorrelation of the past 12 months' returns	**ALg2**
Lag 3 autocorrelation of the past 12 months' returns	**ALg3**
Panel C: Return moments	
Total volatility over the past 36 months	**T.Vol**
Unsystematic risk over the past 36 months	**UnVol**
Systematic risk over the past 36 months	**SysVol**
Coskewness over the past 36 months	**CoSkw**
Idiosyncratic skewness over the past 36 months	**Id.Skw**
Skewness over the past 36months	**T.Skw**
Kurtosis over the past 36months	**T.Kurt**
Panel D: Managerial skill	
7-Factor alpha over the past 12 months based on Fung and Hsieh (2004)	**FHA12**
7-Factor alpha over the past 36 months based on Fung and Hsieh (2004)	**FHA36**
9-Factor alpha over the past 24 months based on Bali et al. (2021)	**B.9FA**
11-Factor alpha over the past 36 months based on Chen et al (2021)	**C.11FA**
R-Squared	$\boldsymbol{R^2}$
Assets Under Management	**AUM**
Max return over the past 12 months (Upside potential of Bali et al. (2019)	**MaxR**
Panel E: Economic variables	
Equity Market Economic Uncertainty Index	**EMU**
Economic Policy Uncertainty Index	**EPU**
TED Spread	**TED**
Implied Volatility Index	**VIX**

2.4. Methodology

Our work builds on the research of Cao et al. (2018) and employs a specialized bidirectional long-short term memory (LSTM) architecture, so-called BRITS, that is explicitly modified to conduct multivariate time series imputation for the hedge fund datasets.[23] The most characteristic property of BRITS is the use of the two base layers that process time series data in a forward (i.e., positive time) and a backward (i.e., negative time)

[23] The Long-Short Term Memory (LSTM) network is a variant of a recurrent neural network (RNN) (see Hochreiter & Schmidhuber, 1997).

direction for our imputation exercise. The learning mechanism is achieved by jointly training the two layers specialized in handling sequential data. The recurrent neural network layers process the time series from the past towards the future, and vice versa, while the overall joint architecture extracts patterns from the data and optimizes the objective function. In the simplest scenario, the two layers use only time-series information to impute missing values and therefore assume no correlation between the hedge funds cross-section. The implementation of the imputing layers requires a recurrent or temporal component and a regression part. Hedge fund datasets, such as the Lipper Tass, possess an ideal setup for financial data imputation, given that missing values are conditioned on information that flows both from the past and the future. As a result, the bidirectional processing of information can assist in extracting valuable data patterns that can help the overall modeling framework fill in missing values with high fidelity. Such a property differentiates our proposed method from the other imputation techniques, which are restricted by relying exclusively on past data, or by assuming only linear data dynamics. The following subsections start with a detailed description of the data setup and they describe the operation of forward and backward layers in both time series and cross-sectional structure. Then, the hyperparameters optimization is presented, followed by a simulation study, a forecasting experiment, and a trading application, designed to evaluate the proposed method's imputation ability as well as the significance of fully recovering missing values via BRITS.

2.4.1. Initial setup

We denote a dataset of multivariate time series of hedge fund returns and predictors as $\{x_{t_1}, x_{t_2}, \dots, x_{t_N}\} \equiv \mathbb{X}$. These time series are observed at regularly-spaced intervals $\{t_1, t_2, \dots, t_N\}$, and for each observation holds that $x_t \in \mathbb{R}^D$ (i.e., the t-th observation x_t consists of D elements $\{x_t^1, x_t^2, \dots, x_t^D\}$). In our hedge fund dataset, it is usually the case that there exist some x_t's for which some or all D elements are missing. Figure 2.1 provides a representation of our data setting, illustrating the data's irregular missing value patterns.

Figure 2.1. Presence of missing values following irregular patterns.

This figure shows an arbitrary form of missingness that is present in both our hedge fund returns and predictors datasets. For each month t, we denote the observed values and the missing values for the multivariate time series matrix $\mathbb{X}$. Missing entries can appear randomly.

$\mathbb{X} =$

Date	Hedge Fund$_1$	Hedge Fund$_2$	Hedge Fund$_3$	Hedge Fund$_4$	Hedge Fund$_5$
Month 1	observed value	observed value	observed value	observed value	observed value
Month 2	Missing value X?	Missing value X?	Missing value X?	observed value	observed value
Month 3	Missing value X?	Missing value X?	Missing value X?	Missing value X?	observed value
Month 4	Missing value X?	observed value	observed value	Missing value X?	observed value
Month 5	Missing value X?	observed value	observed value	Missing value X?	observed value
Month 6	observed value	Missing value X?	observed value	Missing value X?	Missing value X?
Month 7	Missing value X?	Missing value X?	observed value	Missing value X?	Missing value X?

The multivariate time series $\mathbb{X}$ is incomplete, and a masking matrix m_t (M) is used to represent the missing components. In some cases, consecutive x_t's ($\in \mathbb{R}^D$) can be partially or fully missing for consecutive timesteps. Given that past time-series' observations have a decaying influence on future observations, we aim to capture this feature by creating a particular matrix that tracks for how long a value has been missing. Specifically, a tracking matrix, $\delta_t(\Delta)$ is used to capture the gap from the last non-missing observation to the current timestep t. Equations (2.1) and (2.2) below provide the calculation of the masking and tracking matrices for every timestep t.

$$\mathbf{m_t^d} = \begin{cases} 0, & if\ x_t^d\ is\ missing \\ 1, & if\ x_t^d\ is\ not\ missing \end{cases} \tag{2.1}$$

$$\delta_t = \begin{cases} 1 + \delta_{t-1}, & if\ t > 1,\ \mathbf{m_{t-1}^d} = 0 \\ 1, & if\ t > 1,\ \mathbf{m_{t-1}^d} = 1 \\ 0, & if\ t = 1 \end{cases} \tag{2.2}$$

Figure 2.2 shows a practical application given the matrix $\mathbb{X}$ of Figure 2.1. In the first column of the masking matrix in Figure 2.2, we observe that only the first and the sixth months have a value of 1, while the rest of the months have a value of zero. The sequence of ones and zeros follows the presence of missing values in the matrix of Figure 2.1, and, evidently, in the first column, we only have observed values for Month 1 and Month 6. The frequent presence of missing values for the first hedge fund (i.e., the first column) of matrix $\mathbb{X}$ is also depicted in the tracking matrix of Figure 2.2. For instance, by inspecting the first column and Month 6 of the tracking matrix we see the value of five. Given the current timestep t (i.e., Month 6), the last observation took place in Month 1, and therefore δ_t is calculated as: $\delta_6 = 6 - 1 = 5$.

Figure 2.2. Example estimation of the masking and tracking matrices.

This figure illustrates example instances of the masking and tracking matrices, which are named M and Δ, respectively. The construction of both matrices is based on matrix $\mathbb{X}$ of Figure 2.1 and the formulas of equations (2.1) and (2.2).

$\mathrm{M} =$

Date	$Hedge\ Fund_1$	$Hedge\ Fund_2$	$Hedge\ Fund_3$	$Hedge\ Fund_4$	$Hedge\ Fund_5$
Month 1	1	1	1	1	1
Month 2	0	0	0	1	1
Month 3	0	0	0	0	1
Month 4	0	1	1	0	1
Month 5	0	1	1	0	1
Month 6	1	0	1	0	0
Month 7	0	0	1	0	0

$$\Delta = \left[\begin{array}{|c|c|c|c|c|c|}\hline \text{Date} & Hedge\ Fund_1 & Hedge\ Fund_2 & Hedge\ Fund_3 & Hedge\ Fund_4 & Hedge\ Fund_5 \\ \hline \text{Month 1} & 0 & 0 & 0 & 0 & 0 \\ \hline \text{Month 2} & 1 & 1 & 1 & 1 & 1 \\ \hline \text{Month 3} & 2 & 2 & 2 & 1 & 1 \\ \hline \text{Month 4} & 3 & 3 & 3 & 2 & 1 \\ \hline \text{Month 5} & 4 & 1 & 1 & 3 & 1 \\ \hline \text{Month 6} & 5 & 1 & 1 & 4 & 1 \\ \hline \text{Month 7} & 1 & 2 & 1 & 5 & 2 \\ \hline \end{array}\right]$$

We impute the hedge fund returns and each of the predictor matrices separately by forming 24 cross-sectional matrices, upon which we apply the BRITS methodology. Also, the masking and tracking matrices are calculated separately for the returns and each predictor. The aforementioned setup is essential for training our deep learning architecture, which we describe in the following section.

2.4.2 Forward and backward layers

At each timestep t the forward imputing layer receives an observation (i.e., $x_{t_1}, x_{t_2}, \dots, x_{t_N}$) and constructs the so-called "memory cell", which consists of an ensemble of functions known as "gates". In each memory cell, several mathematical operations occur, such as the non-linear transformations on the data, extracting meaningful data patterns, and filtering out information not instrumental for the objective function. To model a complete sequence of time series observations, we need an equivalent in-length series of memory cells. Furthermore, the sequential nature of our data requires the use of recursive functions to construct the memory cell at each timestep t. The mathematical representation of the memory cell is as follows:

$$\begin{pmatrix} i_t \\ f_t \\ o_t \\ g_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ tanh \end{pmatrix} \mathbf{W} \begin{pmatrix} h_{t-1} \\ x_t \\ \mathbf{1} \end{pmatrix} \tag{2.3}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \tag{2.4}$$

$$h_t = o_t \odot \tanh(c_t) \tag{2.5}$$

where, i_t, f_t, and o_t, and g_t are functions known in the machine learning literature as *input gate, forget gate, output gate*, and *candidate cell state*, respectively; $\mathbf{W}$ represents the weight matrix receiving gradient updates for each gate function; σ and $tanh$ represent the sigmoid and hyperbolic tangent functions, respectively. $\odot$

denotes the element-wise multiplication; c_t and h_t are known as the cell state and the hidden state, respectively.[24]

Fan et al. (2019) note that c_t carries information of the modeled time series, while f_t selectively filters and removes information from the previous cell state, c_{t-1}. Krauss and Fisher (2018) note that o_t determines which information from the cell state is used as the output of the memory cell, which we refer to as the hidden state. Lastly, i_t decides how much information is added to the current cell state as a function of the input at time t and the memory cell's output at time $t-1$. According to Fan et al. (2019) all the gate functions would ideally obtain nearly binary values, which numerically would represent the filtering and adjustment of information stored in the cell state. The described mechanism exhibits how the LSTM can effectively learn temporal dependencies between the observations and extract data patterns essential.

In their study, Cao et al. (2018) state that the presence of missing values requires a modified version of the recurrent neural network learning algorithm. If there are missing values in the input x_t, then x_t cannot be used directly in equation (2.3). The authors note that a "complemented" version, x_t^c, needs to be constructed since there are missing values in x_t. For the generic case of a recurrent neural network, x_t^c is updated according to the following algorithm:

$$\widehat{x_t}^{forw} = \mathrm{W}_x h_{t-1} + b_x \tag{2.6}$$

$$x_t^c = m_t \odot x_t + (1 - m_t) \odot \widehat{x_t}^{forw} \tag{2.7}$$

$$\gamma_t = e^{\{-\max(0,\ W_\gamma \delta_t + b_\gamma)\}} \tag{2.8}$$

$$h_t = \sigma(\mathrm{W_h}[h_{t-1} \odot \gamma_t] + U_h[x_t^c \circ m_t] + b_h) \tag{2.9}$$

$$\ell_t = \langle m_t, \mathcal{L}_{MAE}(x_t, \widehat{x_t}^{forw}) \rangle \tag{2.10}$$

where, W_x, W_γ and U_h are weight matrices receiving gradient updates and ∘ denotes concatenation.

In equation (2.6) we provide the mathematical representation of the regression component of the forward imputing layer. Essentially, we calculate an optimal approximation of x_t by the previous timestep's hidden state, h_{t-1}. Across time, we get a series of approximations, $\{\widehat{x_{t_1}}^{forw}, \widehat{x_{t_2}}^{forw}, \dots, \widehat{x_{t_N}}^{forw}\}$. However, if some elements of x_t are present, it is reasonable to directly use these values instead of their approximation. Equation (2.7) shows the construction of the complement vector x_t^c, which is populated by the observed elements of x_t, when those exist, and their approximation $\widehat{x_t}$, when those are missing. This calculation is

[24] All the weights that compose the BRITS architecture are optimized via the gradient descent iterative algorithm. The gradient descent algorithm uses the gradient of the model's loss function at each iteration of the optimization and updates the weights in the opposite direction of the gradient.

feasible given that the masking vector m_t tracks the positions of missing values. Equation (2.8) describes the temporal decay factor (Cao et. al, 2018), one of the algorithm's most innovative parts that distinguishes the model from any other imputation method. For large values of δ_t, which is equivalent to a value being missing for many timesteps, we obtain a small value of γ_t, which in turn causes a significant decay to the hidden state carrying past information via the "$h_{t-1} \odot \gamma_t$" operation. γ_t captures the specific missing value patterns in the hedge fund returns and predictors time series and is responsible for the most effective imputation of their missing values. Importantly, γ_t is constructed with learnable parameters which obtain their optimal values with updates received from the neural network's iterative learning algorithm (i.e., gradient descent). Equation (2.9) shows that the calculation of the hidden state at time t relies on the hidden state of $t-1$ adjusted by the decay factor, and the complement vector x_t^c. Finally, equation (2.10) refers to the forward reconstruction loss function via which the error is calculated so that the neural network can receive the gradient updates for its weight matrices. The "masked" version of mean absolute error (MMAE) is used to proxy the reconstruction loss.[25] Specifically, the mean absolute error is obtained exclusively for the elements of x_t where we have estimated an approximation $\widehat{x_t}^{forw.}$, and at the same time, an observed value (i.e., the *ground truth*) is available. This series of calculations is possible given the masking vector m_t and at the end, we get a series of forward reconstruction errors $\{\ell_{t_1}^{forw}, \ell_{t_2}^{forw}, \dots, \ell_{t_N}^{forw}\}$. We practically update the recurrent component (i.e., hidden state) of the LSTM architecture via a modified version of equations (2.3) to (2.5), which can now be computed since we replace x_t, which contains missing values, with x_t^c that holds the observed values and model generated approximations ($\widehat{x_t}$) for the missing value cases. A pioneering aspect of the model is that the approximated $\widehat{x_t}$ returns or predictor values at each timestep are treated as a learnable parameter, not a constant number. Consequently, these values are optimized in a twofold manner. First, via the optimization of the weight matrices involved in the estimation of $\widehat{x_t}$. Second, via direct updates to the $\widehat{x_t}$ values themselves. This novel adaptation of the neural network's learning algorithm promotes higher convergence speed and performance for the whole model.

In cases where a return or predictor value has been missing for a long time, past information may be less influential than future information. For instance, suppose a predictor value has been missing from March until September for a specific year. Naturally, observed predictor values for October and November are expected to be more closely related to September's missing value compared to January and February's observed values. However, the forward imputing layer imputes missing values only by considering the relation with past information, which can limit the effectiveness of our method. To overcome this limitation, the backward imputing layer is implemented. The mathematical specification of the backward imputing layer is identical to the one of the forward imputing layer—the significant difference lies in how the time series is processed. The

[25] Where $\mathcal{L}_{MAE}(x_t, \widehat{x_t}) = \frac{1}{d}\sum_{i=1}^{d}|x_{ti} - \widehat{x_{ti}}|$.

backward imputing layer receives and processes the time series observations in the reverse order (i.e., the last observation becomes the first and so on). At each timestep, it outputs an approximation of x_t, the $\widehat{x_t}^{backw.}$, and across time, a series of approximated x_t values, $\{\widehat{x_{t_1}}^{backw}, \widehat{x_{t_2}}^{backw}, \dots, \widehat{x_{t_N}}^{backw}\}$. As in the forward imputing layer case, for each approximated x_t , we get an error ℓ_t, and as a result, a sequence of backward reconstruction errors $\{\ell_{t_1}^{backw}, \ell_{t_2}^{backw}, \dots, \ell_{t_N}^{backw}\}$. Cao et al. (2018) suggest employing the so-called *consistency loss* to ensure that each element of $\{\widehat{x_{t_1}}^{backw}, \widehat{x_{t_2}}^{backw}, \dots, \widehat{x_{t_N}}^{backw}\}$ is as close as possible to the corresponding element of $\{\widehat{x_{t_1}}^{forw}, \widehat{x_{t_2}}^{forw}, \dots, \widehat{x_{t_N}}^{forw}\}$. In other words, the $\widehat{x_t}$ Imputations obtained in the forward and backward direction are required to be consistent and with the least discrepancy possible. To measure the discrepancy, the absolute error is used (e.g., $\ell_{t_1}^{cons} = \left|\widehat{x_{t_1}}^{forw} - \widehat{x_{t_1}}^{backw}\right|$) and a series of consistency errors is obtained, $\{\ell_{t_1}^{cons}, \ell_{t_2}^{cons}, \dots, \ell_{t_N}^{cons}\}$. The total model loss via which the weight matrices and the imputed values are optimized is the summation of the forward reconstruction loss, the backward reconstruction loss, and the consistency loss. The final imputed values at each t are obtained with the following formula:

$$\widehat{x_t}^{final} = \frac{\left(\widehat{x_t}^{forw} + \widehat{x_t}^{backw}\right)}{2} \tag{2.11}$$

2.4.3. Cross-sectionally enhanced imputation

So far both the forward and the backward imputing layers are constructed based on sequential time series data. However, hedge fund returns, and financial assets' returns in general, are also characterized by their cross-sectional dependencies. It follows that using past, future, and cross-sectional information is expected to provide the best imputation fidelity, which is also the reason of the efficacy of the cross-sectional mean imputation method in financial literature (see, Haugen and Baker, 1996; Light et al., 2017; Kozak et al., 2020). To facilitate the inclusion of capturing cross-sectional information at time t, first, we obtain the complement vector x_t^c by equations (2.6) and (2.7). The cross-sectional approximation of $\widehat{x_t}$ is given by the following formula:

$$\widehat{x_t}^{cross} = \mathrm{W}_{cross} x_t^c + b_{cross} \tag{2.12}$$

where, W_{cross} and b_{cross} are weights receiving gradient updates. W_{cross} is restricted to having zero values in the diagonal. Given that $\widehat{x_t}^{cross}$ is based on x_t^c, actual cross-sectional values are used when they do exist, while their approximation is used when those are unavailable. The d-th element in each $\widehat{x_t}^{cross}$ approximates the d-th element in x_t using the cross-sectional values (i.e., exclusively cross-sectional information). For the forward (backward) layer, the past (future) time series information is combined with the cross-sectional information via the following formulas:

$$\beta_t = \sigma\big(W_\beta[\gamma_t \circ m_t] + b_\beta\big) \tag{2.13}$$

$$\widehat{c_t} = \beta_t \odot \widehat{x_t}^{cross} + (1 - \beta_t) \odot \widehat{x_t} \tag{2.14}$$

where, β_t, W_β, and b_β are weights receiving gradient updates. β_t is bounded on $[0,1]^D$ and determines the optimal weighting between the cross-sectional based estimation $\widehat{x_t}^{cross}$ and the $\widehat{x_t}$ estimation, which depends on past (future) time series observations for the case of the forward (backward) layer. The temporal decay factor γ_t is also involved in the estimation of β_t. As a result, the optimal weighting β_t can be adjusted when a value has been missing for several timesteps and, therefore, its influence diminishes. For instance, suppose a funds return has been missing for several months, and we want to impute its value at time t. The cross-sectional values are available at time t. Then, at this timestep, the learning algorithm should ideally adjust β_t to assign a larger weight toward the cross-sectional approximation of x_t (i.e., $\widehat{x_t}^{cross}$) and a smaller weight to the past information-based approximation (i.e., $\widehat{x_t}$) given that the predictor value has been missing for a long time. We obtain $\widehat{c_t}$ via equation (2.14) which is the final approximation of x_t. The inclusion of cross-sectional information for the imputation task and the newly introduced equations requires the adjustment of equations (2.7) and (2.9). The new complement vector c_t^c, the hidden state h_t, and the final loss are calculated as follows:

$$c_t^c = m_t \odot x_t + (1 - m_t) \odot \widehat{c_t} \tag{2.15}$$

$$h_t = \sigma(\mathrm{W_h}[h_{t-1} \odot \gamma_t] + U_h[c_t^c \circ m_t] + b_h) \tag{2.16}$$

$$\ell_t^{final} = \langle m_t, \mathcal{L}_{MAE}(x_t, \widehat{x_t}\,)\rangle + \langle m_t, \mathcal{L}_{MAE}(x_t, \widehat{x_t}^{cross}\,)\rangle + \langle m_t, \mathcal{L}_{MAE}(x_t, \widehat{c_t}\,)\rangle \tag{2.17}$$

Expectedly, the $\widehat{x_t}$ is replaced by its enhanced counterpart (i.e., $\widehat{c_t}$) that also leverages cross-sectional information for the approximation of x_t. The complement vector c_t^c dictates that we keep the truly observed values where those exist and use the approximations only for the cases of missing values. The hidden state is estimated in equation (2.16). The hidden states is involved in the calculation of other formulas and by recursively depending at time t on h_{t-1}, it encodes and "memorizes" historical information[26] about the time series. The final model loss uses the masking vector m_t to locate the truly observed values in x_t and compares them with the approximation that leverages historical information, the cross-sectional approximation, and the approximation which uses both historical and cross-sectional information. Essentially, the ℓ_t^{final} accumulates the model's error from three different sources so that the weight matrices can be updated effectively and reach their optimal values. The ℓ_t^{final} as well as equations (2.12) – (2.17) are estimated independently for the forward and backward imputing layers. For the forward and backward layers, we get two $\widehat{c_t}$, the $\widehat{c_t}^{forw}$ and

[26] Historical information is a general reference to the previous timesteps. In the forward and backward layers the meaning of the word "historical" is adapted accordingly based on which direction the time series is processed in each case.

$\widehat{c_t}^{backw}$, respectively. The BRITS generates the final imputed version of our returns and predictors' dataset by the following calculation:

$$x_t^{imputed} = m_t \odot x_t + (1 - m_t) \odot \left[\frac{\left(\widehat{c_t}^{forw} + \widehat{c_t}^{backw} \right)}{2} \right] \tag{2.18}$$

2.4.4. Optimization and hyperparameters

Similar to most neural network architectures, the weights of BRITS are optimized by the gradient descent iterative algorithm. In equation (2.17) we have described the model's loss function which is based on the MAE criterion and is minimized by the optimization process. At each step of the optimization, we evaluate the gradients and then update the BRITS weights in the opposite direction of the gradient (see also Gu et al., 2020; Beckmeyer & Wiedemann, 2022). The practical implementation of BRITS requires an additional optimization step known as hyperparameter optimization in the machine learning literature. This step involves deciding the highest-performing neural network specification out of all candidate architectures. In our study, we examine 16 candidate architectures exploring different values for the number of hidden neurons, epochs, early stopping patience, and learning rates. We display the hyperparameter search space in Table 2.3.

Table 2.3. The hyperparameter search space.

This table presents the different hyperparameter values for the BRITS specifications we explore. The optimized BRITS is the combination that achieves the lowest imputation error, which is calculated by the masked root-mean-square error (equation 2.20), for the hedge fund returns and each predictor dataset.

Hidden neurons	[64, 128]
Epochs	[50, 200]
Early stopping patience	[5, 25]
Learning rate	$[10^{-6}, 10^{-3}]$
Batch size	12
Optimizer	Adam

For the number hidden neurons, we explore two values in the power of two, as is common in the relevant literature. Cao et al. (2018) use 64 neurons, and we also try 128 neurons. For the number of epochs, we experimented with a smaller number (i.e., 50 epochs) and a larger number (i.e., 200 epochs). In terms of early stopping, we follow Gu et al. (2020) and we apply five epochs patience, as well as an even larger value of 25. We follow Cao et al. (2018) and Gu et al. (2020) and we apply learning rates of 10^{-6} and 10^{-3}. The batch size is set equal to 12 months (i.e., one year) and we use as our optimizer the Adam optimizer following again Cao et al. (2018) and Gu et al. (2020). The superior architecture, out of the candidates, achieves the lowest imputation error and it is the one we use for our the remaining of our study.

2.5. Imputation fidelity

We examine the performance of our proposed method by adopting a holistic approach that assesses the imputations obtained from BRITS methodology. Specifically, we estimate the imputation error via a simulation study in which we randomly drop a percentage of observed values and impute them along with the original missing entries. We track the location of the artificially dropped values via the indicative masking given by equation (2.19). This is important since those observed values act as the target for measuring the imputation performance of the BRITS model. For our in-sample dataset (i.e., January 1998 to December 2012), we develop two different simulation experiments. In the first one we randomly drop 10% of the observed values and we fill their missing entries along with those that originally missing. In the second experiment, we repeat the same process, but this time we randomly remove 20% of the original entries. We follow the recent literature on financial data imputation (see Bryzgalova et al., 2022), and adapt the root-mean-square error formula to calculate the imputation error for both simulation studies. A lower error reveals higher imputation performance, and, therefore, higher imputation fidelity. Specifically, in equation (2.20) we display the calculation via the masked root-mean-square (MRMSE) error formula. The MRMSE criterion serves as a proxy of the achieved imputation error.

$$\mathrm{I}_{\mathrm{t}}^{d} = \begin{cases} 1\,, & if\ x_t^d\ is\ artificially\ dropped \\ 0\,, & if\ x_t^d\ was\ originally\ missing \end{cases} \tag{2.19}$$

$$\ell_{MRMSE}\left(\text{imputation}, \text{target}, \mathrm{I}_{\mathrm{t}}^{d}\right) = \sqrt{\frac{\sum_{d=1}^{D}\sum_{t=1}^{T}(imputation - target)^2 \odot \mathrm{I}_{\mathrm{t}}^{d}}{\sum_{d=1}^{D}\sum_{t=1}^{T}\mathrm{I}_{\mathrm{t}}^{d}}} \tag{2.20}$$

We also compare the performance of BRITS in imputing hedge fund returns and predictors' values against popular counterparts used by financial studies. We apply three other imputation techniques for that purpose, the time-series mean, the cross-sectional mean (see, Haugen and Baker, 1996; Green et al., 2013; Light et al., 2017; Beckmeyer and Wiedemann, 2022; Bryzgalova et al., 2022), and the singular value thresholding for nuclear norm minimization (SVT.NNM) (see, of Giglio et al., 2021).[27] SVTNNM is our most powerful benchmark method, since it belongs to a class of algorithms known as the matrix completion methods, which are particularly used to fill in missing values in the data (Cai et al., 2010). Given a partially observed matrix A, the objective is to retrieve a low-rank matrix X. In our study, matrix A represents the stacked hedge fund returns or predictors' series in the cross-section of hedge funds, which contain missing values, while $\widehat{\mathrm{X}}$ represents the retrieved matrix with no missing values. To obtain $\widehat{\mathrm{X}}$, the following optimization problem is solved:

[27] Giglio's et al. (2021) differs from ours as they conduct matrix completion on the residual matrix of hedge returns obtain by factor models, for identifying funds with significant alphas via a false discovery rate control.

$$\widehat{\mathrm{X}} = argmin_{\mathrm{X}} \left\{ \frac{1}{2} \|\mathrm{P}_{\Omega}(\mathrm{X}) - \mathrm{P}_{\Omega}(A)\|_F^2 + \lambda \|\mathrm{X}\|_* \right\} \quad (2.21)$$

where, $\mathrm{P}_{\Omega}(\mathrm{X}) = \mathrm{X}_{ij}$ if the return or predictor value is observed and 0 otherwise, $\| \|_F$ is the Frobenius norm, λ is a tuning parameter, and $\| \|_*$ denotes the nuclear norm that is equal to the sum of singular values of X. In our implementation, we set $\lambda = 1$, the maximum number of iterations at 50, and tolerance level at 0.001.

Several Python libraries are used to analyze the data and implement the imputation methods. Numpy (Harris et al., 2020) and pandas (McKinney, 2010) are employed to pre-process the datasets. The BRITS model is implemented by utilizing the code associated with the original paper (Cao et al., 2018)[28] and the pypots (Du, 2023) library. The filling (Fuchs et al., 2021) R package is applied to estimate the matrix completion benchmark. To ensure reproducibility, where applicable, we fix the model's random seed to control for random components, such as the weight initialization in the BRITS architecture, and serialize the trained models using the pickle library. Serialization guarantees that the exact trained model, including its parameters and architecture, can be loaded and reused, which is crucial for consistency.

Tables 2.4 and 2.5 present the 10% and 20% simulation study results, respectively, for the hedge fund returns and the universe of our predictors. Both tables mainly report the mean MRMSE value. For the predictor's case, we provide aggregate metrics, across all predictors in the form of mean imputation error and the standard deviation of the error metric. A lower standard deviation of the imputation error indicates consistent imputation performance across all predictors. Additionally, Tables 2.6 and 2.7, provide a more in-depth picture of the imputation methods' performance across each predictor, by displaying their MRMSE separately. Table 2.4 clearly shows that BRITS achieves superior imputation performance for both the hedge fund returns and predictor variables. Concerning the returns imputation, BRITS yields an MRMSE of 3.4477, while the second performing model (i.e., cross-sectional mean) generates an error of 4.0381. More importantly, our proposed method outperforms both the more standard methods of imputing returns as well as the more powerful SVT.NNM matrix completion method. SVT.NNM produces a much higher MRMSE of 4.0656 compared to BRITS. To shed more light, the absolute difference, in terms of masked MRMSE criterion, between the BRITS and the SVT.NNM methods is 0.6179, while the relevant absolute difference between the SVT.NNM and the worst performing method (i.e., time series mean) is only 0.4647. Evidently, the imputation quality achieved by BRITS is substantially higher than any other approach. The SVT.NNM is also outperformed by the cross-sectional mean imputation method. With respect to the predictors, the results are in line with those of hedge fund returns, highlighting that BRITS yields the lowest mean MRMSE value, at the 4.1144 level. Furthermore, BRITS obtains the lowest imputation error standard deviation (i.e., 5.3038). The remaining methodologies generate similar mean imputation errors and corresponding standard deviations, which are by

[28] The code is available at the following GitHub repository: https://github.com/caow13/BRITS

far higher than those of the BRITS model. The matrix completion method, SVT.NNM, shows the second-best performance with a mean imputation error of 6.5140, while the time series mean achieves the second-lowest standard deviation of the imputation error at the 8.9549 level. The above findings regarding BRITS superiority are also confirmed by the 20% simulation study results presented in Table 2.6. Concerning hedge fund return imputation, BRITS generates again the lowest mean imputation error at 3.5863, while the cross-sectional mean follows with an imputation error of 4.1207. For the predictors case, BRITS yields also the lowest mean imputation error (i.e., 6.8818) and standard deviation of that error across predictors (i.e., 17.6176). The second-best method in terms of mean imputation error is that of SVT.NNM (i.e., 9.2462). Both benchmark methods also have similar performance for the standard deviation of imputation error, with the cross-sectional mean achieving a slightly lower value in comparison (i.e., 19.0318).

Tables 2.6 and 2.7 validate the above results, revealing that our proposed model achieves the lowest imputation error for most predictors, specifically for 17 out of the 23 predictors for 10% simulation study and for 19 out of 23 predictors for the 20% simulation study. SVT.NNM is the top imputation performer, while BRITS holds the second-best position for most of the remaining predictors. Additional significant insights are derived from the above tables by comparing the imputation error's minimum and maximum values across all predictors. For example, the corresponding imputation error range is [0.0105 – 18.9780] for BRITS, while this range is at much higher levels, [0.0152 – 29.8859] for SVT.NNM. For the cross-sectional mean imputing method, the imputation error ranges [0.0415 – 29.9949], while for the time series mean method [0.0374 – 32.1998]. These findings suggest that BRITS has the most consistent imputing ability with smaller performance deviations than all benchmarks.

Table 2.4. 10% Simulation study results.

This table presents the 10% simulation study's results for the hedge fund returns and predictors datasets. For the 23 predictors, we display a mean imputation error. For the case of funds returns, the imputation error is calculated by the masked RMSE for the artificially dropped observed values.

Imputation method	Hedge Fund Returns		PREDICTORS	
	Mean Imputation Error		Mean Imputation Error	Std. Deviation of Imputation Error
Cross-sectional Mean (X-Mean)	4.0381		6.8857	8.9549
Time-Series Mean (TS-Mean)	4.5028		6.8945	9.3650
SVT.NNM	4.0656		6.5140	9.1547
Optimized BRITS	3.4477		4.1144	5.3083

Table 2.5. 20% Simulation study results.

This table presents the 20% simulation study's results for the hedge fund returns and predictors datasets. For the 23 predictors, we display a mean imputation error. For the case of funds returns, the imputation error is calculated by the masked RMSE for the artificially dropped observed values.

Imputation method	Hedge Fund Returns		PREDICTORS	
	Mean Imputation Error		Mean Imputation Error	Std. Deviation of Imputation Error
Cross-sectional Mean (X-Mean)	4.1207		9.4925	19.0318
Time-Series Mean (TS-Mean)	4.5598		9.4216	19.1810
SVT.NNM	4.1901		9.2462	19.2209
Optimized BRITS	3.5863		6.8818	17.6176

Table 2.6. 10% Simulation study results for the predictors.

This table displays the 10% simulation study's results for the 23 predictors examined in our research. For all predictors, we present the imputation error, which is calculated by the masked RMSE for the artificially dropped observed values.

Imp. Method	PastR	CR3	CR6	CR9	CR12	CR36	ALg1	ALg2	ALg3	T.Vol	UnVol	SysVol	CoSkw	Id.Skw	T.Skw	T.Kurt	FHA12	FHA36	B.9FA	C.11FA	R^2	AUM	MaxR
X-Mean	7.910	10.125	11.755	16.094	20.714	28.375	0.312	0.316	0.337	4.843	4.594	0.866	0.041	0.839	0.930	2.942	3.969	1.352	1.257	29.995	0.196	1.615	8.991
TS-Mean	8.145	10.764	13.391	17.386	21.729	27.074	0.311	0.322	0.349	3.596	3.508	0.496	0.037	0.666	0.771	2.405	3.970	1.106	1.181	32.200	0.127	0.976	8.063
SVT.NNM	7.958	10.090	11.933	16.284	20.988	28.255	0.175	0.175	0.185	4.001	3.972	0.065	0.015	0.213	0.224	1.512	3.736	1.030	0.677	29.886	0.032	0.346	8.072
BRITS	7.700	6.936	5.898	8.806	9.861	15.797	0.174	0.173	0.183	1.587	1.535	0.152	0.011	0.223	0.252	1.711	3.311	0.263	0.262	18.978	0.110	7.102	3.607

Table 2.7. 20% Simulation study results for the predictors.

This table displays the 20% simulation study's results for the 23 predictors examined in our research. For all predictors, we present the imputation error, which is calculated by the masked RMSE for the artificially dropped observed values.

Imp. Method	PastR	CR3	CR6	CR9	CR12	CR36	ALg1	ALg2	ALg3	T.Vol	UnVol	SysVol	CoSkw	Id.Skw	T.Skw	T.Kurt	FHA12	FHA36	B.9FA	C.11FA	R^2	AUM	MaxR
X-Mean	6.270	8.934	13.056	20.107	19.020	32.009	0.313	0.316	0.337	4.134	3.845	0.873	0.043	0.830	0.929	2.937	3.870	1.157	1.387	88.538	0.197	1.619	7.605
TS-Mean	6.567	9.760	14.315	20.835	20.404	30.008	0.312	0.323	0.349	3.072	2.974	0.500	0.038	0.664	0.771	2.372	3.880	0.972	1.319	89.153	0.127	0.986	6.996
SVT.NNM	6.349	9.181	13.610	20.703	19.857	32.414	0.181	0.184	0.194	3.289	3.191	0.074	0.016	0.237	0.253	1.788	3.663	0.804	0.924	88.457	0.033	0.448	6.810
BRITS	6.027	5.964	7.042	8.887	9.186	15.465	0.181	0.183	0.194	1.582	1.443	0.158	0.011	0.236	0.276	1.687	3.343	0.279	0.325	85.529	0.110	7.032	3.142

2.6. Assessing imputation predictability on realized returns

After establishing that BRITS achieves superior imputation performance, we investigate whether that imputation significantly improves the forecasting accuracy of realized hedge fund returns. The link between imputation fidelity and higher forecasting accuracy lies in the informational advantage gained when a forecasting model is estimated on a "complete" version of the dataset including both realized and retrieved entries. In addition, more accurate predictions can lead to more profitable investment decisions and therefore, such an informational advantage of data imputation can hold economic value as well. In this section, we employ a battery of powerful machine learning models forecasting financial assets and we focus on evaluating the predictive models' accuracy fully OOS. Each machine learning specification is trained independently using the "complete" version of predictors' dataset as this is recovered by BRITS and has a forecast target realized hedge fund returns only (i.e., original data including missing entries). The OOS period ranges from January 2013 to November 2021 and we utilize the RMSE and MAE formulas for evaluation. The MRMSE and MMAE formulas, presented earlier on, guarantee that our forecasting application is realistic, and so we do not calculate the prediction error for months that we do not observe any realized funds returns. We also compare the forecasting performance of the machine learning models using the BRITS method for the imputation of our forecasting inputs, with the equivalent performance using the cross-sectional mean imputation method. We mainly select the cross-sectional mean approach as our main benchmark due to its popularity in imputing financial data in financial literature (see, Haugen and Baker, 1996; Green et al., 2017; Light et al., 2017; Kozak et al., 2020; Gu et al., 2020). Also, the cross-sectional mean achieves the second-best performance in imputing hedge fund returns as shown in our simulation experiment in the previous section.

To realistically impute the missing values in-sample dataset we apply a rolling window forward format for all our imputation methods. We use the optimal BRITS architecture tuned for each of the hedge fund returns and predictor matrices during the simulation study in which we randomly drop 10% of the observed values of the data. We use that architecture to impute the first 15 years (i.e., 180 months) of data for both the hedge fund returns (i.e., $r_{i,t+1} - r_{i,t+180}$) and predictors' sets (i.e., $\mathbb{X}_{i,t} - \mathbb{X}_{i,t+179}$) and we then shift forward the data by one month and re-apply BRITS on the next 15-year window for the returns (i.e., $r_{i,t+2} - r_{i,t+181}$) and predictors (i.e., $\mathbb{X}_{i,t} - \mathbb{X}_{i,t+179}$). We repeat this process sequentially until the end of our sample. For our forecasting application, we adopt the following general panel regression framework, which describes the expected excess return $r_{i,t+1}$ of hedge fund i at time $t+1$ as a function of predictors $\mathbb{X}_{i,t}$ via the following additive prediction error model (see, Gu et al., 2020):

$$r_{i,t+1} = \mathbb{E}_t\left(r_{i,t+1}\right) + e_{i,t+1} = g\left(\mathrm{X}_{i,t}\right) + e_{i,t+1} \quad (2.22)$$

where, hedge funds are indexed by i, and months by $t = 1, \ldots, T$. The conditional expected excess return g (■) term represents a flexible function that a machine learning model parameterizes, $\mathbb{X}_{i,t}$ is a D-dimensional vector of predictors and $e_{i,t+1}$ is the error term. In our case, we use a balanced panel dataset $\left\{\left(\mathbb{X}_{i,t}, r_{i,t+1}\right)\right\}_{1\le i\le n}$ spanning across the set of predictors and 3800 hedge funds. To estimate the machine learning models and generate the OOS predictions, we also use a rolling-window method and conduct annual model re-estimations. Specifically, we use 15-years as in-sample observations to estimate our models and produce forecasts for the following one year (i.e., 12 months) following Gu et al. (2020). We then roll forward our in-sample dataset by one year and re-estimate our predictive models before generating again OOS forecasts for the next 12 months. This process is repeated until the end of our dataset.

We apply a comprehensive set of 10 machine learning models belonging to the generic families of gradient-boosted trees, deep neural networks, and penalized regressions (see Gu et al., 2020; Filippou et al., 2022; Gunnarsson et al., 2021; DeMiguel et al., 2023). Specifically, we use three gradient-boosted tree models (i.e., XGBoost, LightGBM and Catboost), three deep neural network architectures (i.e., NNs with different number of hidden layers and neurons), and three penalized regressions (i.e., Lasso, Sparse Group Lasso, and Adaptive Lasso). We present the exact methodologies of those approaches in Appendix 2.A. Since all forecasting models require setting up specific parameterizations, we construct a grid of candidate hyperparameters closely following the relevant literature (see Kingma & Ba, 2017; Gu et al., 2020; Mendez-Civieta et al. 2020; Gunnarsson et al. 2021; Filippou et al., 2022). The full universe of those hyperparameters tested for each model are displayed in Table 2.8. For the gradient-boosted tree models, we explore different values for the number of trees, fraction of inputs used to construct each tree, maximum tree depth, and learning rate.

For the deep NN architectures, we set the batch size to 10000, the number of epochs to 100, the optimizer to Adam, the learning rate to 0.001, the l_1- weight regularization to 0.001, and the early stopping patience at five epochs. The early stopping mechanism prevents model-overfitting, which is equivalent to a high in-sample and poor OOS performance. In practice, to conduct early stopping, we use a validation dataset. At each iteration of the training algorithm, the NN will generate predictions for the validation sample, and the training algorithm will be stopped prematurely if the validation sample error rises for five consecutive training iterations (i.e., epochs). Regarding the number of hidden layers of NNs, we consider up to five hidden layers. For the number of neurons of the first hidden layer we compromise between two popular rules of thumb, namely, half or the square root of

the number of predictors (Filippou et al., 2022). As for the number of neurons for the consecutive layers we follow the geometric pyramid rule (see Gu et al., 2020).

Table 2.8. The forecasting models.

This table displays the different predictive models employed for the forecasting task. We employ a battery of commonly used machine learning models to form a representative and complete set of predictive techniques.

Model type	Model Specification
Gradient-Boosted Trees	
XGBoost	Number of regression trees: {50, 100, 150] Fraction of inputs used: [0.6, 0.8] Maximum tree depth: [1, 2, 3] Learning rate: [0.3, 0.4]
LightGBM	
Catboost	
Neural Network Models	
Deep Neural Network 3 Hidden Layers neurons [32, 16, 8]	Batch size: 10000 Epochs: 100 Optimizer: Adam Early stopping: 5 epochs Learning rate: 0.001 Weight regularization: 0.001
Deep Neural Network 4 Hidden Layers neurons: [32, 16, 8, 4]	
Deep Neural Network 5 Hidden Layers neurons: [32, 16, 8, 4, 2]	
Penalized Regressions	
Lasso	$\lambda \epsilon \{10^{-4}, 10^{-1}\}$
Sparse Group Lasso	$\lambda \epsilon \{10^{-4}, 10^{-1}\}, \alpha \in [0,0.25,0.5,0.5,1]$
Adaptive Lasso	$\lambda \epsilon \{10^{-4}, 10^{-1}\}, \gamma \in [0,8,1]$

Finally, for the regression models, we explore different values for the respective penalty parameters as presented in Table 2.8. Finally, we perform grid-search, and to choose the optimal combination of hyperparameter values which generate the lowest mean squared error in the validation dataset for each model. We achieve that by setting the validation dataset equal to 30% of our in-sample dataset for the tuning of our forecasting models. To respect the time series structure of our data, we do not use cross-validation. Instead, we construct the validation sample by subtracting the last 30% of

observations from our in-sample dataset. The validation dataset is not available to the model during the training phase and therefore, a low mean-squared error indicates potentially superior OOS performance.

With respect to the implementation of the forecasting models we use different libraries offered within the Python programming language. Specifically, lightgbm (Ke et al., 2017), xgboost (Chen & Guestrin, 2016), and catboost (Prokhorenkova et al., 2018) libraries are utilized for the estimation of the gradient-boosted trees. The asgl (Mendez-Civieta et al., 2021) library is used for the penalized regression benchmarks, while MLP neural network benchmarks are implemented with tensorFlow/keras (Abadi et al., 2016) library. To ensure reproducibility, we also apply the random seed technique and serialize the trained models using the pickle library.

Since our OOS dataset contains missing values, we take the necessary steps to calculate the OOS prediction error only for the cases where a realized return exists. As a first step, we apply equation (2.1) and calculate the masking matrix for our OOS returns dataset. The generated masking matrix provides information on which months we have realized returns for each hedge fund. As a second step, we use modified formulas for the root-mean-squared-error and mean-absolute-error to estimate the OOS prediction error for our models. The modified formulas use only information from months we observe realized returns OOS. The masked root-mean-squared error and masked mean-absolute-error formulas are as follows:

$$\aleph = \sum_{d=1}^{D} \sum_{t=1}^{T} OOSm_t^d \tag{2.23}$$

$$L_{RMSE}\left(\hat{r}_{t+1}, r_{t+1}, OOSm_t^d\right) = \sqrt{\frac{\sum_{d=1}^{D} \sum_{t=1}^{T} \left(\left(\hat{r}_{t+1} - r_{t+1} \right) \odot OOSm_t^d \right)^2}{\aleph}} \tag{2.24}$$

$$L_{MMAE}\left(\hat{r}_{t+1}, r_{t+1}, OOSm_t^d\right) = \frac{\sum_{d=1}^{D} \sum_{t=1}^{T} |\hat{r}_{t+1} - r_{t+1}| \odot OOSm_t^d}{\aleph} \tag{2.25}$$

where, $\aleph$ counts the total number of realized hedge fund returns on our OOS dataset with the masking matrix term $OOSm_t^d$ taking a value of 1 when a return exists and 0 otherwise, we use the Hadamard product $\odot$ and the OOS masking matrix to calculate across time t and the different hedge funds d the squared error (i.e., $\left(\left(\hat{r}_{t+1} - r_{t+1} \right) \odot OOSm_t^d \right)^2$) and absolute error (i.e., $|\hat{r}_{t+1} - r_{t+1}| \odot OOSm_t^d$) between predicted returns and realized returns only when the latter exist on the OOS dataset

In Table 2.9, we compile the results for our forecasting experiment. The reported findings confirm the information advantage and improved predictive accuracy gained for all forecasting models estimated on the fully recovered datasets. Notably, these results are consistent and robust across all

model categories. When the forecasting models use as inputs the imputed datasets generated by the BRITS method, the MRMSE ranges from 4.431 to 4.748. On the other hand, when the forecasting models are estimated on a cross-sectional mean imputed dataset, the MRMSE ranges at higher levels in comparison and precisely 4.457 to 4.825. We arrive at a similar conclusion by investigating the reported MMAE results. The MMAE ranges at lower levels (i.e., 2.207 to 2.352) when the machine learning models are trained on BRITS imputed data compared to cross-sectional mean imputed data (i.e., 2.236 to 2.504).

Comparing the forecasting performance of the employed models, the penalized regressions and gradient-boosted trees outperform the NN architectures irrespective of the imputation method used to recover the missing entries. The lowest MRMSE is achieved by the Lasso penalized regression model (i.e., 4.431) followed by the CatBoost gradient-boosted tree model (i.e., 4.437). With respect to the lowest MMAE, CatBoost achieves superior performance (i.e., 2.207). Examining the models' performance when trained on BRITS imputed dataset, the penalized regressions on average attain the lowest MRMSE at the 4.463 level, while the mean MRMSE (i.e., 4.466) of gradient-boosted trees family is only slightly higher than that yielded by the family of penalized regressions, with the family of deep NNs coming last. Hence, deep NN architectures report the weakest performance compared to the other models with a mean MRMSE of 4.698, and a mean MMAE of 2.319. The relevant performance when using the cross-sectional mean method is similar based on the mean MRMSE criterion, however, when it comes to the lowest mean MMAE error, gradient-boosted trees are the top performers. The Lasso model attains again the superior performance (i.e., MRMSE: 4.457, MMAE: 2.236) out of all forecasting models. Thus, irrespective of the imputation method used, the deep NNs present the weakest predictive performance as confirmed by both the MRMSE and MMAE. The above findings highlight that complex and sophisticated specifications, such as deep NNs, are not always the most efficient for predicting hedge fund returns. In contrast, linear and gradient-boosted tree models can potentially provide superior results when forecasting fund returns.

Table 2.9. The forecasting models' OOS prediction error.

This table presents the OOS prediction error results for the predictive models when trained on BRITS imputed data and the cross-sectional mean imputed data. The OOS period ranges from January 2013 to November 2021, while the models are trained in a rolling window format with annual re-estimations. In our study, we employ machine learning models that belong to three distinct categories, specifically, gradient-boosted trees, deep neural networks, and penalized regressions. With no exceptions, the reported results prove that all the machine learning models achieved superior performance when trained on BRITS imputed data compared to cross-sectional mean imputed data.

	OOS: January 2013 – November 2021		**OOS: January 2013 – November 2021**	
	OOS MRMSE		**OOS MMAE**	
Predictive Models	**BRITS training data**	**Cross-sectional Mean training data**	**BRITS training data**	**Cross-sectional Mean training data**
Gradient-Boosted Trees (GBT)				
XGBoost	**4.523**	4.558	**2.211**	2.240
LightGBM	**4.438**	4.460	**2.210**	2.252
CatBoost	**4.437**	4.467	**2.207**	2.264
Mean GBA Error Metric	**4.466**	4.495	**2.209**	2.252
Multi-Layer Perceptron Neural Networks (NN)				
5-Hidden Layers NN	**4.745**	4.801	**2.314**	2.504
4-Hidden Layers NN	**4.748**	4.825	**2.352**	2.503
3-Hidden Layers NN	**4.603**	4.754	**2.292**	2.497
Mean MLP-NN Error Metric	**4.698**	4.793	**2.319**	2.501
Penalized Regressions				
Lasso	**4.431**	4.457	**2.211**	2.236
Sparse Group Lasso	**4.513**	4.529	**2.330**	2.358
Adaptive Lasso	**4.444**	4.470	**2.219**	2.258
Mean Penalized Regr. Error Metric	**4.463**	4.485	**2.253**	2.284

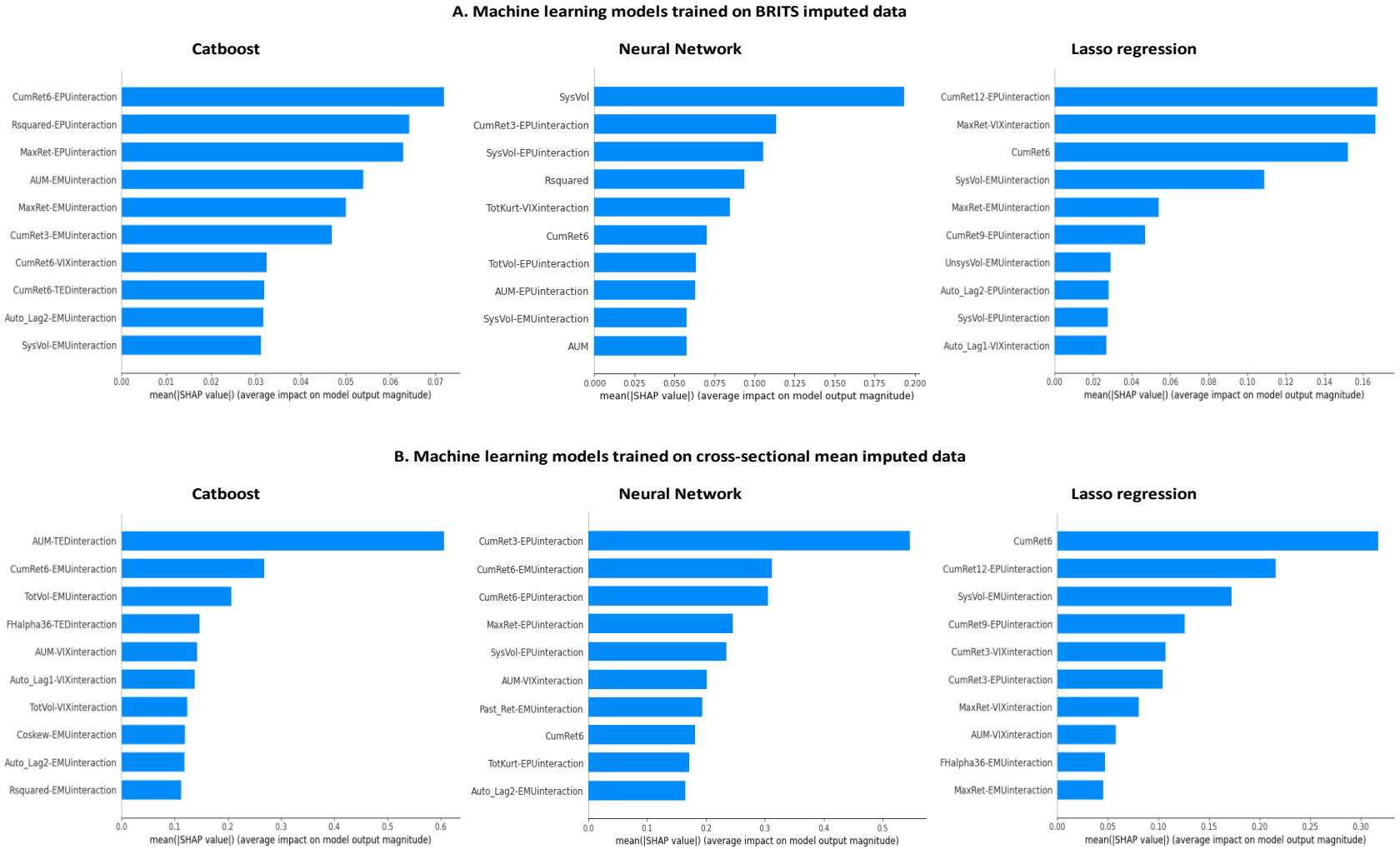
2.7. Measuring predictors' importance

In this section, we investigate the importance of the fund-specific predictors and their interactions with the top-performing forecasting models from each category of machine learning specifications. We use the Shapley Additive exPlanations (SHAP) methodology to calculate the SHAP values of our predictors. SHAP is based on a cooperative game theory framework, which aggregates the Shapley values across all predictors to measure the contribution of each predictor to the forecasting exercise. The method considers the fluctuations in the forecasting model output (i.e., the prediction) with and without a specific predictor, while including the rest. The predictions from the two models are compared and the difference in predictions is calculated. Then, the SHAP values are estimated as the weighted average of all possible differences for each predictor, respectively (Lundberg and Lee, 2017). For the estimation of Shapley values, we use the shap (Lundberg & Lee, 2017) Python library. Figure 2.3 provides the 10 most important predictors according to their SHAP values for CatBoost, NN with three layers and Lasso models. We focus on these models, since, per model category, these 3 techniques achieved the lowest OOS forecasting error. We calculate the SHAP values as the mean of the absolute SHAP value across all observations for each predictor as well as their interactions with the macroeconomic factors, as those have been imputed by BRITS. To gain further insights and conduct a comparative analysis, we also estimate the SHAP values when the same machine learning models are trained on cross-sectional mean imputed data. Upon estimating the SHAP values, we evaluate the predictor's importance, on the last OOS window covering the period January 2021 to November 2021.

In Figure 2.3.A., we observe that the interactions of fund-specific predictors with macroeconomic interaction variables dominate the standalone fund-specific predictors since they are among the top 10 most important overall predictors most of the time and for all models. This is a sound finding since powerful machine learning methodologies cannot only capture nonlinearities in the data but also take advantage of their interactions to provide better predictions, which is also explained by the fact that CatBoost is a top-performing model. In terms of importance, the interaction of cumulative returns appears more frequently in the top places, while interactions of predictors presenting managerial skill, such as the maximum return over the past year as well as the assets under management of funds and R^2 come second. We also observe, interactions and standalone predictors of systematic volatility to be important for predictions mainly for the case of NNs. In addition, the EMU and EPU indices, are the macroeconomic factors that appear more frequently in the most important interactions with fund-specific predictors. We also estimate the corresponding SHAP values of the predictors imputed by the cross-sectional mean method in Figure 2.3.B. Similarly, to Figure 2.3.A., the macroeconomic interaction variables are the most significant across all models compared to individual predictors.

Figure 2.3. SHAP values graphs.

The figure displays the OOS SHAP values for the top-performing forecasting models per model category. In Figure 2.3.A., we present the SHAP values for the CatBoost, NN with three hidden layers, and Lasso regression models, which were trained on hedge fund returns and predictors imputed by BRITS. In Figure 2.3.B., we present the SHAP values when the aforementioned models are trained on imputed data generated by the cross-sectional mean method. Irrespective of the imputation method used, the interactions between the predictors and the economic variables achieve higher significance for the model's predictions compared to the individual predictors.



The interactions with cumulative returns and variables that belong to the manager skill category are the most instrumental for the model's predictions. Therefore, we can conclude that irrespective of the imputation method used, the macroeconomic interaction variables dominate individual predictors in terms of importance, while the previous returns and manager skill variable categories create the most informative interactions for the models' forecasts.

2.8. Trading application

As a final experiment, we construct an investment application. The trading application will provide us with insights regarding the economic significance of the forecasts generated by the machine learning models which were trained on BRITS imputed returns and predictors datasets. For each month, we sort the hedge funds in the cross-section into deciles based on each machine learning model's forecast. We form equally-weighted decile portfolios, but we mainly focus on the top decile portfolio, which includes the top-performing hedge funds. Focusing on the top-performing hedge funds (i.e., top decile) is a natural decision since investors can not short a hedge fund as is the case with other assets (e.g., stocks), but instead, they can only choose in which fund to invest the available capital. Hence, we invest evenly all capital available by the end of each month in the top decile portfolio, which is equivalent to holding long positions on these specific funds. At the end of each month, we compute the monthly return of the top-decile portfolio and present its annualized mean return, the annualized Sharpe ratio, and the Fung and Hsieh (2004) seven-factor model annualized alpha. We compute the Newey-West t-statistic with three lags for the annualized return and alpha. We also report the R^2 statistic[29], which measures how well the machine learning model predicts the realized top decile portfolio return. If the realized abnormal return factor is predicted more accurately, then an investor knows better how much funds in the top decile will outperform in the next period (see Kaniel et al. 2023).

Table 2.10 presents the performance of top decile fund portfolios for each of the top machine learning forecasters based on the MRMSE criterion, along with the performance of an ensemble of the mean of forecasts for the top-performing models. Based on the forecasting application results (see Table 2.9) and for the case of BRITS imputed data, the lowest forecasting error is achieved by CatBoost, NN with three layers, and Lasso regression. For the case of cross-sectional mean imputed data, the top forecasters are LGBM, NN with three layers, and Lasso regression, and we use those forecasts to construct our benchmark portfolios. Table 2.11 reports the performance of top decile fund portfolios

[29] We calculate the R^2 statistic following Kaniel et al. (2023). We denote F_t the realized returns and $\widehat{F_t}$ the predicted return of the long portfolio based on the prediction-sorted deciles. The predictive R^2 statistic is calculated as: $R^2 = 1 - \frac{\sum_1^T (\widehat{F_t} - F_t)^2}{\sum_1^T (\widehat{F_t})^2}$

constructed based on the ensembles of mean forecasts of each category of machine learning models employed (i.e., gradient-boosted trees, neural networks, penalized regressions). We also report the performance on the ensembles of mean forecasts generated by all machine learning models for each imputation method in the bottom row of each panel.

Table 2.10 highlights that the BRITS-generated imputations of the hedge fund returns and predictors datasets provide an information advantage that enables the machine learning models to produce more profitable forecasts. The decile portfolios which were formed based on the forecasts of the models that were trained on BRITS imputed datasets attain higher annualized returns, Sharpe ratios, and alpha values compared to the benchmark portfolios. The Lasso regression's forecasts, which rely on BRITS' effective imputation of the data, display the highest probability since the constructed portfolio has an annualized return of 15.11%, a Sharpe ratio of 1.22, a statistically significant and positive alpha of 4.19%, and a predictive R^2 of 21.51. The second-highest annualized return, alpha, and predictive R^2 is achieved by the CatBoost-BRITS portfolio with values of 12.91%, 2.96%, and 18.31%, respectively. The NN-BRITS portfolio has a lower performance in comparison, but, still outperforms the NN-cross-sectional mean benchmark portfolio in terms of annualized return, Sharpe ratio, and alpha values. The ensemble portfolio that relies on the mean forecast of the top machine learning models has a superior performance when the models are trained on BRITS data. The aforementioned portfolio has a higher annualized return (i.e., 15.11%), Sharpe ratio (i.e., 1.22), and alpha (i.e., 3.44%) compared to the cross-sectional mean benchmark portfolio.

The results presented in Table 2.11 further validate the economic significance of BRITS imputations. For the three model categories, the decile portfolios attain higher annualized returns, alpha values, and predictive R^2 compared to the cross-sectional mean ensemble portfolios. The penalized regression category achieves the highest profitability, while the decile portfolio achieves an annualized return of 14.41%, a Sharpe ratio of 1.41, a positive and statistically significant Sharpe ratio of 4.68%, and a predictive R^2 of 21.11%. Considering the ensemble portfolio of the three model categories, we observe superior performance for the BRITS case. In more detail, the BRITS ensemble portfolio has a higher annualized return (i.e., 15.24%), Sharpe ratio (i.e., 1.26), and alpha value (i.e., 3.24%). The cross-sectional mean ensemble portfolio of the three model categories has an inferior performance across all metrics except for predictive R^2. Finally, in both BRITS and cross-sectional mean cases, the decile portfolios constructed based on the ensemble of the penalized regression category outperform the ensemble portfolios related to the other model categories (i.e., gradient-boosted trees and neural networks).

Table 2.10. Decile portfolio for top-performing machine learning models.

The table presents the performance metrics for the equally-weighted decile portfolios which were formed based on the top-performing (per model category) machine learning models' forecasts. The machine learning models are trained on BRITS imputed datasets (Panel A) and cross-sectional mean imputed datasets (Panel B), in order to perform a comparative analysis considering the portfolios' performance. We focus on the top decile portfolio, which includes the top-performing hedge funds. At the end of each month, we compute the monthly return of the top-decile portfolio and present its annualized mean return, the annualized Sharpe ratio, and the Fung and Hsieh (2004) seven-factor model annualized alpha. We compute the Newey-West *t*-statistic with three lags for the annualized return and alpha. We also report the predictive R^2 statistic following Kaniel et al. (2023).

Top models (RMSE)	**Ann. Mean Return (%)**	**Return *t*-stat**	**Ann. Sharpe ratio**	**FH (2004) alpha (%)**	**Alpha t-stat**	**Pred. R^2(%)**
Panel A: BRITS						
CatBoost	**12.91*** **	2.76	1.11	**2.96**	1.29	**18.31**
NN-3layers	**12.47*** **	2.74	**1.02**	**1.14**	0.54	-0.55
Lasso	**15.14*** **	3.49	**1.31**	**4.19** **	1.99	**21.51**
Ensemble-Top (Mean)	**15.11*** **	3.22	**1.22**	**3.44**	1.61	17.56
Panel B: Cross-sectional mean						
LGBM	11.16***	3.01	**1.13**	2.32	1.18	12.47
NN-3layers	10.42**	2.43	0.91	-0.07	-0.04	-0.39
Lasso	12.97***	3.21	1.18	2.44	1.27	14.94
Ensemble-Top (Mean)	12.11***	2.93	1.08	1.61	0.78	**18.11**

Table 2.11. Decile portfolio for the three model categories.

The table presents the performance metrics for the equally-weighted decile portfolios which were formed based on the mean ensemble forecast of each of the three machine learning model categories we examine. The machine learning models are trained on BRITS imputed datasets (Panel A) and cross-sectional mean imputed datasets (Panel B), in order to perform a comparative analysis considering the portfolios' performance. We focus on the top decile portfolio, which includes the top-performing hedge funds. At the end of each month, we compute the monthly return of the top-decile portfolio and present its annualized mean return, the annualized Sharpe ratio, and the Fung and Hsieh (2004) seven-factor model annualized alpha. We compute the Newey-West *t*-statistic with three lags for the annualized return and alpha. We also report the predictive R^2 statistic following Kaniel et al. (2023).

Top models (RMSE)	**Ann. Mean Return (%)**	**Return *t*-stat**	**Ann. Sharpe ratio**	**FH (2004) alpha (%)**	**Alpha t-stat**	**Pred. R^2(%)**
Panel A: BRITS						
Gradient-Boosted Trees	**11.94****	2.52	1.01	**2.04**	0.76	**15.2**
Neural Networks	**13.17*****	3.06	**1.08**	**1.08**	0.62	**-0.65**
Penalized regressions	**14.41*****	3.8	**1.41**	**4.68****	2.41	**21.11**
Ensemble- All	**15.24*****	3.51	**1.26**	**3.24***	1.71	15.9
Panel B: Cross-sectional mean						
Gradient-Boosted Trees	10.89***	2.88	**1.07**	**2.04**	1.09	11.6
Neural Networks	11.27**	2.48	0.96	0.72	0.31	-29.5
Penalized regressions	13.17***	3.63	**1.41**	4.56***	2.53	18.6
Ensemble-All	12.80***	3.25	1.23	2.64	1.4	**20.31**

2.9. Conclusion

The study proposes a bidirectional recurrent imputation neural network for imputing hedge fund returns and their corresponding predictors. The model handles missing values by capturing time series and cross-sectional information without making any assumptions about the hedge fund data's structure and distribution. Hence, we can effectively recover funds' missing entries. To establish the imputation fidelity of BRITS, we construct two simulation studies in which we artificially drop a random 10% and 20% of the observed values and a battery of benchmarks used in the financial literature for data imputation. Additionally, we examine the importance of BRITS generating full informative predictors datasets for forecasting hedge fund returns. We train well-established machine learning methodologies on the imputed set of predictors for our forecasting experiment.

Our findings indicate that BRITS outperforms all benchmarks in terms of imputation error and provides a recovered set of predictors, which can generate accurate forecasts when fed to gradient-boosted trees, NNs, and penalized regressions. The simulation study validates that our framework achieves superior results in imputing hedge fund returns and predictors' values. This result is robust in the 10% and 20% simulation studies. The forecasting task's results confirm that machine learning models estimated on the BRITS imputed datasets show higher forecasting accuracy than those imputed using the cross-sectional mean method, the standard method in the financial literature. Hence, BRITS can provide an informational advantage, boosting the OOS performance of forecasting models. Regarding predictor importance, we provide evidence that interactions of fund-specific predictors with macroeconomic variables dominate the standalone fund-specific predictors. As shown by the SHAP interpretability method, macroeconomic interactions with fund-specific characteristics are among the top 10 most important predictors, most of the time considering all models. This outcome favoring the significance of macroeconomic interactions holds irrespective of the imputation method we use to fill the missing values. Finally, we prove via our trading application that adopting an effective imputation method such as BRITS can assist the forecasting models in generating more profitable and economically meaningful forecasts. When trained on BRITS imputed data, machine learning models can more accurately identify the hedge funds that achieve the highest returns in the next month. Such an outcome has important implications for institutional investors, who can benefit from the enhanced accuracy of their predictive models and make optimal decisions concerning which hedge funds to include in their portfolio and from which to withdraw their capital. Out of the three considered model categories, penalized regressions display the highest profitability. The superior economic performance of penalized regressions is consistent irrespective of the imputation method we use to fill in the missing values in the hedge fund datasets.

Appendix 2

2.A. Forecasting models

For the forecasting task, we employ a battery of commonly-used machine learning models described in this section of the Appendix.

2.A.1. Gradient-boosted tree algorithms

2.A.1.1 EXtreme Gradient Boosting (XGBoost)

The XGBoost model was introduced in the work of Chen and Guestrin (2016) and extends the boosting algorithm developed by Friedman (2001). Jabeur et al. (2021) state that XGBoost is an ensemble model based on decision trees that utilizes an optimization process leading to superior performance compared to individual techniques. Nobre and Neves (2019) note that the output of the XGBoost model can be calculated with the formula:

$$\hat{r} = \sum_{k=1}^{K} f_k(\mathrm{X}), \;\; f_k \in \mathcal{F}$$

where, f is a function in the functional space $\mathcal{F}$, $\mathcal{F} = \{f(\mathrm{X}) = w_{q(\mathrm{X})}\}$ is the space of the regression trees, q is the structure of each regression tree, w is the leaf weight, f_k is the k-th regression tree, K is the number of regression trees.

The loss function that is optimized to train the model effectively is the following:

$$L = \sum_{t} l(\hat{r}, r) + \sum_{k} \Omega(f_k)$$

where, l is the squared error loss function measuring the difference between the predicted return $\hat{r}$ and the realized return r, and Ω is a regularization term. Ω is specified by the following formula:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \| w\|^2$$

where, γ is a regularization hyperparameter, T is the number of leaves in each regression tree, and λ is a regularization hyperparameter.

In our XGBoost implementation we use the xgboost Python library (Chen & Guestrin, 2016) associated with the original paper, and utilize the library's default parameters. Specifically, the

number of trees is set to 100, the maximum depth of a tree to 3, the learning rate to 0.1, γ to 0, and λ to 1.

2.A.1.2 Light gradient boosting machine (LightGBM)

The LightGBM model was introduced by Ke et al. (2017). To effectively deal with a large number of data instances and explanatory variables, the model's training algorithm consists of two novel techniques: gradient-based one-side sampling and exclusive feature bundling. The objective function of the model integrates a number of regression trees to approximate the final model (Sun et al., 2020) as follows:

$$f_T(X) = \sum_{t=1}^{T} f_t(X)$$

where, $f_t(X)$ denotes a regression tree.

The objective function is estimated by Newton's method (Sun et al., 2020). According to Ken et al. (2017), LightGBM can outperform the XGBoost model in terms of computational speed and memory consumption.

2.A.1.3 Categorical Boosting (CatBoost)

The CatBoost model was proposed by Prokhorenkova et al. (2018) and belongs to the class of gradient-boosting algorithms. The authors state that CatBoost outperforms other gradient-boosted models due to its properties. The model's training algorithm consists of two innovative techniques, specifically ordered boosting, which is a permutation-driven alternative to the classic algorithm, and a novel algorithm for processing categorical features. Instead of the conventional regression trees, CatBoost is constructed based on oblivious decision trees (Gulin et al., 2011; Ferov& Modrý, 2016), which are more balanced and less prone to overfitting. As noted in Jabeur et al. (2021), the function of each decision tree h^t in the t-th step of the sequence of approximations is the following:

$$h^t = \arg\ min \frac{1}{N} \sum \left(-f^t(\mathrm{X}, \mathrm{r}) - h(\mathrm{X})\right)^2$$

where, f^t is a least squares approximation by the Newton method.

2.A.2 Deep Neural Network

Fan et al. (2021) indicate that artificial neural networks use a composition of a series of non-linear functions to model non-linearity. In mathematical notation, they take the following form:

$$H^{(h)} = f^{(h)} \circ f^{(h-1)} \circ f^{(h-2)} \circ \ldots \circ f^{(1)}(x)$$

where, ○ illustrates the composition of two functions, h is the number of hidden layers.

By letting $H^0 \triangleq x$, we can define recursively $H^{(l)} = f^l(H^{(l-1)})$ for all $l = 1, \ldots, L$ (Fan et al., 2021). To retrieve the so-called MLP architecture, we define the output of its hidden layer with ReLU activation functions as follows:

$$H^{(l)} = f^{(l)}\left(H^{(l-1)}\right) \triangleq \mathrm{ReLU}\left(W^{(l)}H^{(l-1)} + b^{(l)}\right)$$

where, $W^{(l)}$ is the weight matrix fully connecting the previous layer with every neuron in the l-th layer, $b^{(l)}$ is the bias (intercept) term. Finally, the output of the final hidden layer, $H^{(L)}$, and the corresponding observed value from the training dataset are used to estimate the loss function we minimize. We do not apply an activation function for the neural network's final output to avoid binding it to a specific value range.

2.A.3.1 Lasso regression

The Lasso regression (Tibshirani, 1996) introduces sparsity in the predictors' set by shrinking some coefficients and setting others to zero. The Lasso estimator is given by:

$$\vartheta^* = \underset{\vartheta}{argmin} \left\| r - \mathrm{X}^{\mathrm{T}}\vartheta \right\|_2^2 + \lambda \sum_{d=1}^{D} |\vartheta_d|_1$$

where, λ is a tuning penalty parameter controlling the level of sparsity.

We use a validation dataset to decide on the optimal value of λ and following (Gu et al., 2020) our search space is $\{10^{-4}, 10^{-1}\}$.

A.3.2 Group Lasso

Yuan and Lin (2006) suggest the following Group Lasso estimator:

$$\vartheta^* = \underset{\vartheta_0, \vartheta}{argmin} \left\| r - \mathrm{X}^{\mathrm{T}}\vartheta \right\|_2^2 + \lambda \sum_{\xi=1}^{K} \sqrt{d_\xi} \left\| \vartheta^\xi \right\|_2$$

where, K is the number of categories the predictors are divided into, the term $\sqrt{d_\xi}$ weights each category according to its size and d_ξ is the size of the ξ category, ϑ^ξ is a sub-vector of coefficients from ϑ with components that correspond to the covariates in ξ category. We use a validation dataset to decide on the optimal value of λ and following (Gu et al., 2020) our search space is $\{10^{-4}, 10^{-1}\}$. In Section 3 of our main paper, we mention that our predictors can be divided into 3 major categories (i.e., past returns and autocorrelation, second and higher moments and skill of hedge fund managers).

We also introduce a fourth category by including macroeconomic factors and their interactions with the predictors (i.e., $K = 4$). We use a validation dataset to decide on the optimal value of λ and following (Gu et al., 2020) our search space is $\{10^{-4}, 10^{-1}\}$.

2.A.3.3 Sparse Group Lasso

The Sparse Group Lasso model was introduced in Friedman et al. (2010) and combines the Lasso and Group Lasso penalization under the following mathematical formulation:

$$\vartheta^* = \underset{\vartheta}{argmin} \ \left\| r - \mathrm{X}^{\mathrm{T}}\vartheta \right\|_2^2 + \ a\lambda \sum_{d=1}^{D} |\vartheta|_1 + (1-a)\lambda \sum_{\xi=1}^{K} \sqrt{d_\xi} \left\| \vartheta^\xi \right\|_2$$

where, a is bounded in $[0, 1]$ and controls the penalization between Lasso and Group Lasso.

We use a validation dataset to decide on the optimal value of λ, and following (Gu et al., 2020) our search space is $\{10^{-4}, 10^{-1}\}$. Regarding the tunable parameter a, we explore the following values: $\alpha \in [0, 0.25, 0.5, 0.75, 1]$

2.A.3.4 Adaptive Lasso

The Adaptive Lasso was introduced in Zhou (2006). The authors suggest that the model satisfies the so-called oracle properties, and it performs as well as if the true model was provided in advance. The Adaptive Lasso estimator is the following:

$$\vartheta^* = \underset{\vartheta}{argmin} \ \left\| r - \mathrm{X}^{\mathrm{T}}\vartheta \right\|_2^2 + \ \lambda \sum_{d=1}^{D} \widetilde{w_d} |\vartheta_d|_1$$

$$\widetilde{w_d} \ = \frac{1}{|\vartheta_d^\gamma|}$$

where, $\widetilde{w_d}$ is a weight corresponding to ϑ coefficient of d predictor.

To determine $\widetilde{w_d}$ we leverage a recent technique proposed by Mendez-Civieta et al. (2021) that uses partial least squares components. Under the condition that collinearity is not an issue, Zhou (2006) proposes first to run an OLS regression and then feed these weights (i.e., ϑ_d) to $\widetilde{w_d}$. However, a high number of variables in our pool of predictors are correlated. Alternatively, when collinearity is present in the dataset, Mendez-Civieta et al. (2021) propose to run a partial least squares (PLS) regression instead of an ordinary OLS regression and then feed the obtained weights in $\widetilde{w_d}$. To determine the number of utilized PLS components, the researcher needs to specify the desired amount of variability in X (the matrix of predictors) explained. For this setting, we selected a number of PLS components that can explain 90% of the variability in our predictors' matrix. For the λ and γ

tunable parameters, we used a validation dataset to decide on the optimal combination given the defined search spaces: $\lambda \epsilon \{10^{-4}, 10^{-1}\}, \gamma \in [0{,}8, 1]$.

CHAPTER 3

Directional Predictability of Industry Returns via White-Box Deep Learning

3.1. Introduction

Effectively predicting the directional movements of the industries' portfolios' excess returns is significant to portfolio managers and financial institutions. Predicting the directional movement of an asset's return (i.e., directional forecasts) refers to predicting the sign of the next period's return using past information. This task holds important implications concerning market timing and shifting wealth between financial assets and risk-free investments. Earlier research has suggested that directional accuracy is the only conventional measure of forecast quality related to profits (see Leitch & Tanner, 1991), which is also supported by the finding of Leung et al. (2000). More recent studies (see Nyberg, 2011; Nyberg & Pönkä, 2016; Pönkä, 2017; Becker & Leschinski, 2018; Iworiso & Vrontos, 2019) reiterate this point and highlight that successful trading strategies and investment decisions rely on the accuracy of the directional forecasts. Despite its economic importance, there is scarce literature exploring the directional predictability of returns, especially in the returns of industries' portfolios. Most financial literature has focused on modelling the conditional mean of stock returns (see, among others, Gu et al., 2020; Chen et al., 2021; Gu et al., 2021) rather than considering directional forecasts and industry-aggregated datasets. The current state of the literature and the ever-growing adoption of machine learning models for financial applications motivate our research objective. We aim to explore the ability of these models to generate accurate directional forecasts in the context of industry return predictability. However, to avoid using a machine learning model as a "black box", it is essential also to identify the determinants of predictability (Becker & Leschinski, 2018) and, therefore, identifying the most informative covariates when predicting directional movements of industry returns.

In this study, we adopt a state-of-the-art and interpretable neural network model to predict the directional movements of industries' excess returns effectively. Essentially, predicting directional movements of returns is a classification task related to sign prediction (i.e., positive or negative). Our modelling framework adopts the TabNet neural network architecture, which was introduced in the study of Arik and Pfister (2021). Three fundamental properties distinguish TabNet from other neural network architectures and the standard model (i.e., the logistic regression) employed in classification-oriented financial tasks. First, contrary to standard neural networks, TabNet is an interpretable architecture with the ability to conduct sparse covariate selection and quantify the importance of each covariate. TabNet derives its predictions by utilising a sequential process with multiple decision steps.

Each decision step relies on three computational blocks, which perform mathematical operations and are responsible for non-linear data transformations and extracting patterns from the data. The computational blocks also select the most informative covariates driving the model's prediction and provide insights regarding the level of importance for each covariate.

Second, TabNet is a parameter-efficient neural network architecture. Typically, the number of weights in neural network architectures increases as the number of covariates increases. On the other hand, TabNet conducts covariate selection and exclusively reserves the learning capacity (i.e., the estimated model weights) to the most informative covariates without wasting "resources" on irrelevant covariates. Furthermore, at each decision step of the model's architecture, TabNet possesses two sets of weights. The first set of weights is unique to each decision step, while the second set is shared across decision steps. This component of the model and the act of sharing weights at different parts of the model's architecture ensure parameter efficiency and distinguish TabNet from other neural networks.

Third, TabNet's architecture performs non-linear data transformations and extracts meaningful data patterns via the feature transformer computational blocks. Feature transformers utilize modern neural network techniques such as gated linear unit non-linearity and batch normalization, which benefit the model's training process and, eventually, the model's performance (see among others, Dauphin et al. 2016; Gu et al., 2020). Additionally, TabNet does not impose any assumption on the distribution and structure of the data. The properties mentioned above provide TabNet a predictive advantage over linear models, such as the logistic regression, which is the standard benchmark for financial classification applications (see Fischer & Krauss, 2018; Gunnarsson et al., 2021) and imposes several assumptions on the data (e.g., the linearity assumption between the log-odds of the target variable and the covariates, absence of multicollinearity).

These key components are novel to TabNet architecture since no other neural network technique can facilitate covariate selection and the quantification of covariate importance under the same architecture, while being parameter efficient. Recently introduced methods in the machine learning literature can help explain individual model predictions and provide approximations regarding covariate importance. For instance, the Local Interpretable Model-agnostic Explanations (LIME) method utilises a local surrogate and more interpretable model that explains a single output from a "black-box" model (see Ribeiro et al., 2016; Molnar, 2022). Lundberg and Lee (2017) use Shapley values to analyse individual model predictions and then assign covariate importance scores using game-theoretic concepts. However, these methods, and all similar techniques, are applied to the model post-estimation and can only provide approximations to covariate importance without being

directly integrated into the model's architecture. On the contrary, TabNet has a built-in mechanism that determines covariate importance. Notably, the model learns to select and base its predictions on the most informative covariates. We can derive the most informative set of covariates and their relative importance post-estimation. Therefore, TabNet is a "white-box" architecture that effectively solves the "black-box" criticism that neural networks receive.

We contribute to the literature in multiple ways. First, we contribute to the statistical and forecasting literature by adopting a state-of-the-art, parameter-efficient, and interpretable neural network technique trained on a financial panel dataset that can effectively provide accurate predictions. To our knowledge, this is the first paper that uses TabNet to predict the directional movements of 49 industries' excess returns. Importantly, TabNet has an integrated mechanism as part of its architecture that can shed light on the most informative covariates driving the model's predictions without needing an "external" algorithm that can provide an approximation of covariate importance. Since financial applications often require model interpretability and transparency, the proposed "white-box" neural network can be readily implemented by portfolio managers and financial practitioners as part of their predictive toolbox. Second, we extend the scarce financial literature that employs a machine learning technique for a classification task. Only limited studies explore the performance of neural networks for predicting the directional movements of financial returns. With our study, we aim to fill this gap in the literature and provide evidence that sophisticated architectures can provide accurate and economically meaningful forecasts. Third, we use extensive datasets of both industry portfolios and covariates as predictors, contrary to previous studies (see Rapach et al., 2015; 2019; Bianchi and McAlinn, 2021). For instance, our forecasting experiment uses a rich set of 49 industries as given by the Fama-French (1997) industry classification system and 127 covariates. On the other hand, Rapach et al. (2015; 2019) and Bianchi and McAlinn (2021) take a less granular approach by focusing on 30 and 10 industries, respectively. Moreover, the studies of Rapach et al. (2015; 2019) and Bianchi and McAlinn (2021) use a limited set of input variables compared to our covariate set. Rapach et al. (2015; 2019) utilize exclusively cross-industry lagged returns. Bianchi and McAlinn (2021) do not include lagged returns and cross-industry lagged returns as covariates and limit their covariate set to 70 financial ratios and five macroeconomic variables. Our covariate set consists of financial ratios, lagged returns, cross-industry lagged returns and macroeconomic variables. Fourth, TabNet's covariate selection mechanism also captures seasonality effects and cross-industry interdependencies, which holds crucial implications for portfolio managers when deciding asset allocation and the portfolio's exposure to industry sectors.

To conduct our analysis, we examine 49 industries' portfolios from January 1976 to December 2022. The out-of-sample (OOS) period ranges from January 2013 to December 2022, approximately

20% of the total sample. Our research focuses on predicting the directional movements of industries' excess returns. We follow a panel data format structure for our in-sample dataset. The constructed panel dataset is used to train the proposed model and all benchmarks, generating out-of-sample predictions for 2013-2022. We evaluate the model's predictions with multiple performance metrics (i.e., accuracy, balanced accuracy, logistic loss, brier score loss, and the Area-Under-the-Curve statistic) and statistical tests, such as the McNemar's pairwise test (McNemar, 1947; Edwards, 1948), Cochran's Q-test (Cochran, 1950), F-test (Snedecor & Cochran, 1989), Pesaran-Timmerman directional accuracy test (Pesaran & Timmermann, 1992) and Anatolyev-Gerko excess profitability test (Anatolyev & Gerko, 2005). We also construct a trading application to evaluate the economic significance of TabNet's predictions. Based on TabNet's predictions, we form the trading positions and shift wealth between the industries' portfolios and a risk-free investment (i.e., the one-month Treasury Bill). As benchmarks, we include three buy-and-hold strategies of three market indices (i.e., the CRSP value and equally-weighted indices and the Standard and Poor's 500 index), and a trading strategy based on the predictions of the best-performing technique from the benchmark model set.

Our empirical results indicate that TabNet achieves the highest OOS predictive accuracy across all performance metrics compared to other linear (i.e., logistic regression) and non-linear machine learning models (i.e., variations of extreme gradient boosting, explainable boosting machine, and random forest). The employed statistical tests also validate these results. The attained accuracy for TabNet is 64.30%, whereas for the second best-performing model, the logistic regression, the accuracy is 61.11%. Furthermore, TabNet has the lowest logistic loss and brier score loss, 0.6458 and 0.2266, respectively. The logistic regression has a logistic loss at the 0.8293 level and a brier score loss at 0.2578. Finally, we establish the economic significance of TabNet's predictions via the trading application results. The strategy formed based on TabNet's predictions achieves the highest annualized return (i.e., 18.04%), Sharpe ratio (i.e., 1.8807), and positive and statistically significant alpha values against the four and five-factor models. Based on the logistic regression predictions, the trading strategy attains an annualized return of 14.79% and Sharpe ratio at the 1.0947 level. TabNet outperforms all other benchmarks even after accounting for five and ten basis points monthly transaction costs.

Our research is related to different strands of literature. In the first strand, we categorize financial papers that employ machine learning techniques to predict stock market and industry portfolio returns. Notable examples include the work of Krauss et al. (2017), Rapach et al. (2019), Gu et al. (2020), and Bianchi and McAlinn (2021). In their work, Krauss et al. (2017) examine the performance of machine learning models using daily total returns of all stocks belonging to the S&P 500 index and conclude that their findings pose a severe challenge to the semi-strong form of market efficiency.

Rapach et al. (2019) use a machine learning framework to examine industry return predictability and underline the economic significance of the model's predictions. The authors also uncover significant industry interdependencies and economic links via their modeling framework. In a different study, Gu et al. (2020) use US stock market data and indicate that machine learning algorithms models offer an improved description of the expected return behaviour relative to traditional forecast methods. Bianchi and McAlinn (2021) analyze industry and aggregate stock market excess returns for 1970-2020 using an ensemble of linear models. They conclude that financial ratios provide valuable information for forecasting stock returns at the industry and aggregate market levels.

In the second strand of literature, we categorize the financial research papers that utilize a machine learning model for a classification task, in which discrete class labels are predicted. The research of Leung et al. (2000) and Iworiso and Vrontos (2019) belong to the second cluster of papers. Leung et al. (2000) analyze a dataset containing three globally traded stock market indices from January 1967 to December 1995. The authors first use a group of classification models to predict the direction (i.e., sign) of the stock index excess return movement, and then a second group of models to conduct a level estimation of the stock index excess return (i.e., the regression models). The empirical results prove that classification models outperform regression models regarding sign prediction and profitability. Iworiso and Vrontos (2019) examine the US stock market from January 1960 to December 2016. The empirical results validate that the machine learning models outperform the benchmark econometric techniques accuracy and the forecasts' profitability. Gunnarsson et al. (2021) conduct a comparative analysis of several machine learning models for the credit scoring classification task and conclude that the XGBoost model outperforms other machine learning techniques, including neural networks and the logistic regression, which is the industry standard for credit scoring tasks. Our proposed model differentiates from machine learning models used in other studies by being interpretable and parameter-efficient. For the case of TabNet, we do not need to use additional algorithms to determine covariate importance since the model itself can provide insights into the most significant covariates. Additionally, TabNet leverages under the same architecture advanced techniques, such as gated linear unit non-linearity, batch normalization, and transformer computational blocks, which enhance its training process and OOS performance. The combination of these techniques is not present in the standard neural network models.

The remainder of this paper is structured as follows. Section 3.2 discusses the related literature. Section 3.3 describes the employed dataset and the covariate set. Section 3.4 presents our methodology. Section 3.5 covers the empirical results concerning the proposed model's predictive and trading performance. Finally, section 3.6 concludes.

3.2. Literature review

Our work is linked to different strands of financial literature. In the first cluster, we group the studies that employ a machine learning technique to predict financial market returns, specifically stock market and industry portfolio returns (see among others, Beller et al., 1998; Rapach et al., 2015; Krauss et al., 2017; Rasekhschaffe & Jones, 2019; Rapach et al., 2019; Huck, 2019; Gu et al., 2020; Gu et al., 2021; Chen et al., 2021; Bianchi & McAlinn, 2021, Avramov et al., 2022). In the second cluster, we group the financial research papers that employ a machine learning model for a classification task, in which discrete class labels are predicted instead of a continuous quantity of a target variable (i.e., asset returns). The concept of classification takes the form of predicting the directional movements (i.e., positive or negative) of stock or indices' returns, whether a stock return will over-(under-) perform the cross-sectional median stock return, and credit scoring tasks (see among others, Leung et al., 2000; Nyberg, 2011; Fischer & Krauss, 2018; Iworiso & Vrontos, 2019; Karhunen, 2019; Dumitrescu et al., 2022; Gunnarsson et al., 2021, McDonnell et al., 2023).

3.2.1. Machine Learning for stock market predictability

This subsection presents the studies that employ a machine learning model for stock return prediction. To analyze the U.S. stock market, Krauss et al. (2017) investigate the effectiveness of ML techniques in statistical arbitrage. The dataset consists of daily total returns of all stocks belonging to the S&P 500 index from January 1990 until October 2015. The authors state that their findings severely challenge the semi-strong form of market efficiency. Rasekhschaffe and Jones (2019) analyze 22 developed markets for 1994-2016, using a covariate set of 194 factors. Their findings suggest that ML techniques can provide a better alternative to linear models when forecasting returns, and their performance is consistent across both the US as well as other developed markers. Huck (2019) examines the performance of machine learning models in a "big data" setting. The author constructs a covariate set with more than 600 predictors and analyses a sample from 1990 to 2015 for the US stock market. The predictor set includes lagged returns, firm-specific information, time information, indices, risk factors, and commodities. The results show that positive excess returns are negated when accounting for transaction costs and that adding covariates does not guarantee a boost in the profitability of the machine learning models' forecasts. Gu et al. (2020) perform a comparative analysis of ML algorithms in the setting of cross-section and time-series stock return prediction. The authors use a dataset from March 1957 to December 2016 that includes all NYSE-, AMEX-, and NASDAQ-listed firms. The study concludes that these models offer an improved description of the expected return behavior relative to traditional forecast methods. Moreover, the authors note that neural networks - and, to a lesser extent, regression trees- are the top-performing models. In a similar data setting, Chen

et al. (2021) use monthly equity return data for all securities on the CRSP universe and a sample period from January 1967 to December 2016. The researchers employ a deep neural network architecture for their asset pricing model, which outperforms all benchmark approaches with respect to the Sharpe ratio value, explained variation, and pricing errors. Gu et al. (2021) propose a new latent factor conditional asset pricing model based on an autoencoder neural network architecture. The dataset consists of monthly individual stock returns from CRSP for all firms listed in the NYSE, AMEX, and NASDAQ indices. The autoencoder allows for factor exposures as a non-linear function of predictors and performs dimensionality reduction. The findings also support the superiority of the autoencoder neural network, given that the out-of-sample pricing errors are far more minor compared to those of other leading factor models (i.e., Fama-French factor models). In their work, Avramov et al. (2022) investigate whether machine learning techniques can generate profitable signals for investment decisions. The authors use a large sample of US stocks for the period spanning from 1987 to 2017. Their findings indicate that machine learning methods can identify mispriced stocks, while the generated signals can be profitable in long positions and display low downside risk. However, transaction costs can deteriorate the performance of machine learning-based strategies.

3.2.2. Industry return predictability

This subsection discusses the studies that investigate industry return predictability. Rapach et al. (2015) construct a predictive regression framework that examines industry interdependencies and cross-industry return predictability while allowing for direct and indirect sectoral links. For this research objective, an adaptive Lasso model and network analysis are applied to a dataset consisting of monthly data from 1960 until 2014 for 30 industry portfolios' returns from Kenneth French's Data Library. The results provide strong evidence that lagged returns of interdependent industries are significant predictors of individual industry returns. This finding is also reinforced by a profitable long-short trading strategy that uses the adaptive Lasso's predictions to construct portfolios. Another work contributing to the scarce literature on industry return predictability is the study of Rapach et al. (2019), which leverages an ML framework to model 30 industry portfolios' returns spanning 1960 until 2016. The employed model generates economically meaningful predictions when used as part of a trading strategy and for constructing long-short investment portfolios. Industry returns are modeled using a two-step process, using the Lasso model as the first step to conduct covariate selection and then an OLS regression on the selected covariates. An additional study that analyzes industry and aggregate stock market excess returns for the period 1970-2020 is the work of Bianchi and McAlinn (2021). This research uses an ensemble of linear models for its forecasting objective and a rich set of predictors, which consist of financial ratios and macroeconomic variables. The authors conclude that financial ratios provide valuable information for forecasting stock returns at the industry and

aggregate market level. Moreover, they show that their ensemble method can achieve significant OOS economic gains while outperforming other linear and non-linear predictive models. An earlier study that analyzes industry returns under a Bayesian framework is Beller et al. (1998). Beller et al. (1998) use a Bayesian multivariate regression model to analyze a dataset with quarterly equal- and capitalization-weighted returns for 55 U.S. industries from 1973 until 1995. Their OOS findings indicate that industry returns are predictable, and forecasting models, when combined with mean-variance optimization criteria, can provide economically valuable guidance during portfolio selection.

3.2.3. Classification models and financial applications

This subsection discusses the scarce literature that employs a classification predictive framework for financial applications. Leung et al. (2000) use a dataset containing three globally traded stock market indices (i.e., SP500 for the US, FTSE 100 for the UK, and Nikkei 225 for Japan) from January 1967 to December 1995. The authors first use a group of classification models to predict the direction (i.e., sign) of the stock index excess return movement, and then a group of models to conduct level estimation of the stock index excess return. The first group of classification models includes linear discriminant analysis, logit, probit, and probabilistic neural network techniques. In the second group of regression models that predict excess returns as a continuous target variable, the authors include exponential smoothing, multivariate transfer function, vector autoregression with Kalman filter, and multilayered feedforward neural network techniques. The empirical results prove that classification models outperform regression models in terms of predicting the direction of the stock index movement and achieving higher returns in the scope of a trading strategy. Nyberg (2011) constructs a predictive framework to forecast the direction of US stock excess return movements. The author utilizes a dataset from January 1968 to December 2006 and a model set consisting of linear classification models, ARMAX models, and predictive models based on volatility forecasts. The empirical results show that the direction of the SP500 stock index excess return is predictable in-sample; however, the predictability becomes weaker OOS.

In their study, Fischer and Krauss (2018) analyze a dataset consisting of daily total returns of all stocks belonging to the S&P 500 index from January 1990 until October 2015. They utilize a battery of predictive models to predict whether a stock return will be above or below the cross-sectional median stock return. The Long Short-Term Memory (neural) network outperforms the logistic regression, standard deep neural networks, and random forest models when considering the forecasts' predictive accuracy and profitability. Iworiso and Vrontos (2019) examine the US stock market for the period spanning from January 1960 to December 2016. Their analysis includes multiple forecasting techniques such as the binary probit models, classification and regression trees, and penalized binary

probit models. The predictive binary classification task is to forecast whether US stock excess return will be positive or non-positive (i.e., zero or negative). The reported results support that sophisticated machine learning models outperform the benchmark binary probit models with respect to their statistical accuracy and the economic significance of their forecasts. Karhunen (2019) employs several statistical and machine learning algorithms to predict the direction of excess returns for the stock market indices of 11 developed countries. The analysis focuses on the period from March 1980 to February 2010. The empirical results indicate mild predictability in the stock market when considering the statistical significance of the forecasts; however, in some cases there is evidence of highly significant results in the economic sense.

In a different application and data setting, Dumitrescu et al. (2022) examine the performance of machine learning models for the credit scoring classification task. Their objective is to predict a loan default, while their model set consists of logistic regression, random forest, support vector machine, neural network, and the proposed model, which combines decision trees and logistic regression under a joint framework. The authors note that the logistic regression remains the benchmark scoring model in the credit industry and show that their proposed technique outperforms the standard logistic regression and compares competitively to the random forest. In a different study, Gunnarsson et al. (2021) examine the performance of the XGBoost model against the logistic regression, decision tree, random forest, and several neural network models for the classification task of credit scoring. All models are evaluated on ten retail credit scoring data sets. The results indicate that the XGBoost technique outperforms all other models, including the industry standard (i.e., the logistic regression) and the neural network architectures. Finally, McDonnell et al. (2023) examine machine learning classification models for insurance risk classification via claims prediction. They use a limited model set and compare TabNet against logistic regression and XGBoost. Their data application involves a synthetic dataset with 100000 observations, modeled based on a real dataset provided by a Canadian insurer. The authors prove that TabNet has higher accuracy compared to the logistic regression and XGBoost models, and therefore, its adoption is encouraged for predictive tasks related to insurance risk pricing.

3.3. Data and covariate set

We forecast the monthly directional movements of 49 U.S. industry portfolio returns using the in-sample period from January 1976 to December 2012, and the OOS period from January 2013 to December 2022. The in-sample dataset is constructed by stacking for all 49 industries the corresponding in-sample datasets in a panel format data structure (see Gu et al., 2020; Filippou et al., 2022). We use the panel above data structure of the in-sample dataset to train all predictive models.

Adopting a panel data structure can significantly increase the size of our in-sample dataset and solve the issue of limited observations arising from our data's monthly frequency. The classification of 49 industries employed follows the Fama-French paradigm and the descriptions on Kenneth French's website. The 49 industry sectors we examine in this study are presented in Appendix 3.A. Accordingly, the industry returns correspond to the value-weighted average of their constituent stocks.

For our prediction task, we construct a set consisting of 127 covariates. To create our dataset, we use the Compustat database from the Wharton Research Data Services (WRDS) platform and precisely 63 industry financial ratios, which belong to valuation, profitability, capitalisation, financial soundness, solvency, liquidity, and efficiency categories, respectively. Capitalisation ratios measure the debt component of a firm's total capital structure; efficiency ratios capture the effectiveness of the firm's usage of assets and liability; financial soundness and solvency ratios capture the firm's ability to meet long-term obligations; liquidity ratios measure a firm's ability to meet its short-term obligations; profitability ratios measure the ability of a firm to generate profit; valuation ratios estimate the attractiveness of a firm's stock. The median value from the group of companies belonging to the specific industry is taken to arrive at the industry-level aggregation for each financial ratio.[30] Based on previous research on industry return predictability (see Rapach et al., 2015), we also decided to include other industries' lagged returns to extend our dataset of covariates further. For each U.S. industry portfolio, we also include up to 12-month excess lagged returns and the 1-month value-weighted excess lagged returns of all other 48 industries as additional covariates.

Motivated by other financial studies that outline the importance of macroeconomic information when predicting financial returns (see, Dangl & Halling, 2012; Bianchi & McAlinn, 2021), our dataset is extended and also includes macroeconomic variables. We use four macroeconomic variables downloaded from the FRED database, namely the Chicago Fed National Financial Conditions Index (NFCI), Chicago Fed National Activity Index (CFNAI), Chicago Fed National Activity Index: Production and Income (PANDI), and the Consumer Price Index (CPI). The NFCI index captures U.S. financial conditions in money, debt and equity markets as well as the traditional and "shadow" banking systems. The PANDI index provides information regarding the national economy's expansion with respect to its historical trend rate of growth. The CFNAI index captures overall economic activity and the related inflationary pressure, while the CPI is used as an inflation index.

[30] The aggregation of the data to the industry-level and the respective pre-processing steps are carried out by the WRDS research team. For more information we direct the reader to the WRDS Industry Financial Ratio manual (2016) published by the WRDS Research Team.

3.4. Methodology

This section discusses the proposed model's methodology for effectively predicting the directional movement of industries' excess returns. In the first subsection, we start with some preliminaries and the mathematical formulation of the predictive task. The following subsections discuss TabNet's building blocks for forming innovative neural network architecture. To compare TabNet's predictive ability, we use an extensive benchmark model set consisting of several machine learning techniques, including state-of-the-art models and statistical tests. Specifically, for our benchmark model set, we consider the logistic regression model, the logistic regression model with l_2 penalty (Fischer & Krauss, 2018), three variations of the EXtreme Gradient Boosting (XGBoost) model (Chen & Guestrin, 2016), the Explainable Boosting Machine (EBM) model (Norti et al., 2019), and the Random Forest model (Breiman, 2001). We present the benchmark models in detail in Appendix 3.B.

3.4.1. Initial setup

In this study, our objective is to predict the directional movements of monthly industry excess returns. We denote the monthly industry excess return over the one-month risk-free rate as r_t. We use an indicator variable to convert the continuous variable r_t to the binary space [0, 1] based on the sign of the industry excess return. For this task, the following rule is applied:

$$\begin{cases} \boldsymbol{y_t = 1}, & \boldsymbol{if\ r_t > 0} \\ \boldsymbol{y_t = 0}, & \boldsymbol{if\ r_t \leq 0} \end{cases} \tag{3.1}$$

The binary variable y_t captures the directional movements (i.e., positive or negative) of industry excess returns. Therefore, we distinguish the positive class, which is related to a positive excess return, and the negative class, which corresponds to the negative excess return case[31]. Our objective is to create accurate forecasts of y_t conditioned on available information up until $t-1$. To generate $\widehat{y_t}$ forecasts, we leverage a non-linear and interpretable (i.e., "white-box") neural network specification to construct a probabilistic framework that provides estimates in the form of:

$$\begin{cases} \Pr(r_t > 0 | \, \mathrm{X_{t-1}}) = \ \Pr(y_t = 1 | \, \mathrm{X_{t-1}}) \\ \Pr(r_t \leq 0 | \, \mathrm{X_{t-1}}) = \ \Pr(y_t = 0 | \, \mathrm{X_{t-1}}) = 1 - \ \Pr(y_t = 1 | \, \mathrm{X_{t-1}}) \end{cases} \tag{3.2}$$

where, $\Pr(r_t > 0 | \, \mathrm{X_{t-1}})$ is a probabilistic estimate of a greater than zero industry excess return for next month (i.e., positive class), $\Pr(r_t > 0 | \, \mathrm{X_{t-1}})$ is equivalent to the probabilistic estimate of y_t being equal to one, $\Pr(r_t \leq 0 | \, \mathrm{X_{t-1}})$ is a probabilistic estimate of a less than or equal to zero industry excess return for next month (i.e., negative class); following the rules of probability, it holds that

[31] In practice, the negative class includes every non-positive industry excess return.

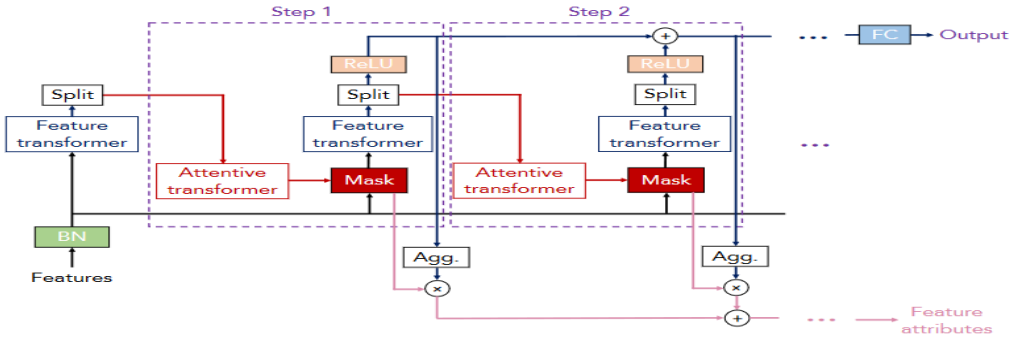
$\Pr(y_t = 0|\, \mathrm{X}_{t-1}) = 1 - \Pr(y_t = 1|\, \mathrm{X}_{t-1})$, all probabilistic predictions for next month's directional movement of industry excess returns are conditioned on past information set compiled by the chosen covariates X_{t-1}.

3.4.2 TabNet's neural network architecture

The probabilistic framework is constructed based on the research of Arik and Pfister (2021), which introduced the novel TabNet deep learning model[32]. TabNet is an interpretable neural network since we directly derive the most informative covariates from the model. The model's architecture employs a sequential process with multiple decision steps to generate its predictions. In practice, each decision step i receives as an input the processed information from the $[i-1]$ step and performs covariate selection as well as the non-linear transformation of the covariate vector. The transformed covariate vectors from each step are aggregated and then used by the model to generate predictions. Each decision step of the sequential process consists of three computational blocks, the feature transformer, the attentive transformer, and the Mask. These computational blocks carry out the necessary mathematical operations responsible for the non-linear data transformations and the effective covariate selection. Figure 3.1 provides an overview of the model's neural network architecture.

Figure 3.1. Overview of TabNet's neural network architecture.

The figure displays an overview of TabNet's neural network architecture consisting of the feature transformer, the attentive transformer, and covariate masking computational blocks that carry out the mathematical operations required for the model's final prediction. The feature transformer block converts the input to a more informative non-linear representation, while the attentive transformer block selects which covariates are passed to the following processing steps. The Mask block conducts mathematical operations and contains information revealing the most informative and, therefore, significant covariates.



Note: From "Tabnet: Attentive interpretable tabular learning", by S. Ö Arik,., and T. Pfister, 2021, *Proceedings of the AAAI Conference on Artificial Intelligence*, *35*(8), 6679-6687. Copyright 2021 by Association for the Advancement of Artificial Intelligence.

[32] TabNet is available in Python programming language via the PyTorch library implementation (Dreamquark, 2023) at: https://github.com/dreamquark-ai/tabnet.

3.4.3 The attentive transformer

The mechanism responsible for covariate selection is the attentive transformer which constructs a learnable mask $M[i] \in R^{BxD}$ at each i decision step. The masking is multiplicative, and to effectively enforce covariate sparsity, the mask $M[i]$ is multiplied with the covariates vector $f \in R^{BxD}$ [33] received as input at each decision step (i.e., $M[i] \cdot f$). Covariate selection is a fundamental property of the architecture since the model's learning capacity is devoted exclusively to the most informative covariates while ignoring the irrelevant ones (Arik and Pfister, 2021). As a result, TabNet becomes a parameter-efficient architecture since no trainable weights are assigned to non-informative covariates that were not selected by the masking process. Several mathematical operations occur inside the attentive transformer block to construct the learnable mask. The attentive transformer possesses a separate neural network architecture consisting of fully connected and batch normalization layers as well as the sparsemax normalization operation (Martins & Astudillo, 2016). The sparsemax normalization enforces sparsity on the covariate vector and then projects these covariates onto a probability map in Euclidean space. Each covariate gets assigned an associated probability reflecting how much a specific covariate will influence the model's output at each decision step. In practice, at each decision step i, the attentive transformer receives a processed covariate vector from the previous time step $[i-1]$ and constructs the mask under the following mathematical formulation:

$$M[i] = sparsemax\big(P[i-1] \cdot h_i(a[i-1])\big) \tag{3.3}$$

$$P[i] = \prod_{j=1}^{i} (\gamma - M[j]) \tag{3.4}$$

$$\sum_{j=1}^{D} M[i]_{b,j} = 1 \tag{3.5}$$

where, h_i is a trainable function parameterized by fully connected and batch normalization neural network layers, as shown in Figure 3.2 (b), $a[i-1]$ is the output of a previous decisions step's feature transformer (i.e., a processed covariate vector from the previous decision step), $P[i-1]$ is the so-called "prior scale term" described by the relaxation hyperparameter γ; when $\gamma = 1$, a covariate is enforced to be used exclusively at a single decision step, whereas as γ increases, more flexibility is provided in terms of using a covariate at multiple decision steps, $P[0]$ is initialized as all ones (i.e.,

[33] Where B is the batch size.

1^{BxD}) and no prior information is enforced on the selected covariates, for a single data sample the sum of the elements of $M[i]$ equals one[34].

The sparsity level can be adjusted on the overall TabNet architecture by introducing a regularization term to the model's cross-entropy objective function (see also Krauss et al., 2017)[35]. Specifically, the added term takes the following form:

$$\sum_{i=1}^{N_{steps}} \sum_{b=1}^{B} \sum_{j=1}^{D} \frac{-M_{b,j}[i] \log\left(M_{b,j}[i] + \varepsilon\right)}{N_{steps} \cdot B} \tag{3.6}$$

where, ε is a small number added for numerical stability, the regularization term can be added to TabNet's objective (i.e., loss) function via a coefficient λ_{sparse} that is treated as a hyperparameter. According to Arik and Pfister (2021), adding the regularization term in TabNet's cross entropy objective function can provide benefits in the presence of datasets where a large number of features is redundant.

3.4.4 The feature transformer

Each n-th decision step receives D-dimensional covariates f and outputs to a feature transformer a sparse covariate vector (McDonnell et al., 2023). The feature transformer block has a separate neural network architecture, transforming the input into a more informative and high-dimensional representation. This representation of the covariate vector is then split into the decision step output (i.e., $d[i] \in R^{BxN_d}$) and the information that is passed to a subsequent decision step of the architecture (i.e., $a[i] \in R^{BxN_a}$). The architecture of the feature transformer consists of multiple neural network layers that are either unique (i.e., step-dependent) to each decision step or shared across decision steps. Constructing feature transformers with step-dependent as well as shared neural network layers promotes parameter efficiency and robust learning (Arik & Pfister, 2021). Specifically, the feature transformer consists of fully connected neural network layers (FC), a batch normalization layer (BN) (see Gu et al., 2020), and Gated Linear Unit (GLU) non-linear activation functions (Dauphin et al., 2016). Additionally, the GLU connects to a residual connection (He et al., 2016) with normalization. The normalization with $\sqrt{0.5}$ enhances the model's optimization (i.e., training) process and stabilizes the variance throughout TabNet's architecture (Gehring et al., 2017; Arik & Pfister,

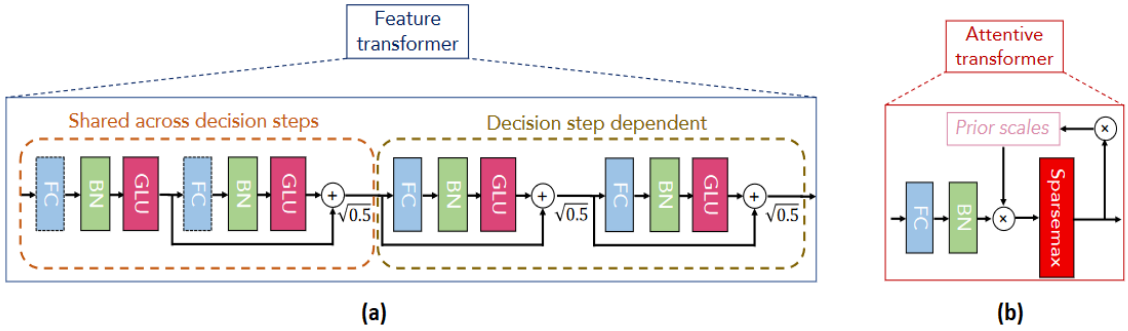
[34] This is an expected mathematical outcome, since $M[i]$ is derived by a sparsemax operation and covariates are projected on a probability map.

[35] For a training sample b, the loss is calculated as follows: $L_{binary\ cross\ entropy} = -(ylog(p) + (1-y)\log(1-p))$.
where, y is the true class label taking a value 1 or 0, and p is the predicted softmax probability for class 1, while the predicted probability of sample b belonging to class 0 is given by $(1-p)$.

2021). Finally, we apply a ReLU[36] activation function on the feature transformer-generated representations of the covariate vectors at each step (i.e., the decision step output, $d[i]$), and aggregate the result across all decision steps (i.e., $d_{out} = \sum_{i=1}^{N_{steps}} \mathrm{ReLU}(d[i])$). TabNet's final prediction is computed after applying a linear mapping to d_{out} which is performed by a set of weights W_{final}. Figure 3.2 (a) provides an illustrative overview of the feature transformer and its computational components responsible for converting the input vector to a more informative high-dimensional representation.

Figure 3.2. The components of the feature transformer and the attentive transformer blocks.

The figure displays the components of the feature transformer and attentive transformer computational blocks of TabNet's neural network architecture. A) The feature transformer block transforms the received input into a more informative, high-dimensional representation. The high-dimensional transformation is achieved by passing the input through a series of neural network layers (i.e., fully connected neural network layers (FN), batch normalization layers (BN), and Gated Linear Unit (GLU) non-linear activation functions) and a residual connection (He et al., 2016) with normalization. B) The attentive transformer possesses a separate neural network architecture consisting of FN and BN layers as well as the sparsemax normalization operation (Martins & Astudillo, 2016), which enforces sparsity on the covariate vector. The attentive transformer assigns each covariate an associated probability reflecting how much a specific covariate will influence the model's output.



Note: From "Tabnet: Attentive interpretable tabular learning", by S. Ö Arik,., and T. Pfister, 2021, *Proceedings of the AAAI Conference on Artificial Intelligence*, *35*(8), 6679-6687. Copyright 2021 by Association for the Advancement of Artificial Intelligence.

3.4.5 TabNet's global interpretability and the mask

TabNet is an interpretable architecture since the covariate selection masks at each decision step reveal the selected features. For instance, if $M_{b,j}[2] = 0$, then the j^{th} covariate of the b^{th} sample does not have any contribution to the 2^{nd} steps output $d[2]$. Arik and Pfister (2021) underline the significance of calculating aggregate covariate importance by combining the masks across the decision steps. However, each step can contribute at a different magnitude to the overall model's prediction;

[36] ReLU activation function: $f(x) = \max(0, x)$.

for this reason, a weighting mechanism should also be applied to determine each step's relative importance. The authors propose using the following coefficient:

$$\eta_b[i] = \sum_{c=1}^{N_d} \text{ReLU}\left(d_{b,c}[i]\right) \tag{3.7}$$

It follows that $\eta_b[i]$ denotes the contribution of the decision step i to the overall model's prediction for the b^{th} sample. In an extreme case, if $d_{b,c}[i] < 0$, then all covariates at the i^{th} decision step should have zero contribution to the overall model prediction for the b^{th} sample. As the value of $\eta_b[i]$ increases, then the contribution of the i^{th} decision step to the overall model output also increases. For each b sample, the $\eta_b[i]$ coefficient scales the mask $M[i]$ at each decision step, while the formula for the aggregate covariate importance mask is presented in equation (3.8). To arrive at the global covariate importance across the OOS dataset, we average $M_{agg-b,j}$ for each covariate j across all b OOS data points as shown in equation (3.9).

$$M_{aggregate-b,j} = \frac{\sum_{i=1}^{N_{steps}} \eta_b[i] M_{b,j}[i]}{\sum_{j=1}^{D}\sum_{i=1}^{N_{steps}} \eta_b[i]\, M_{b,j}[i]} \tag{3.8}$$

$$M_{aggregate-j} = \frac{M_{agg-b,j}}{b}, \qquad \forall j \tag{3.9}$$

3.4.6 TabNet hyperparameters

TabNet's neural network architecture contains several hyperparameters, the value of which is decided based on the model's performance on a validation dataset. Deciding on the optimal hyperparameter values based on a validation dataset is the standard approach in the financial machine learning literature (see Fischer and Krauss, 2018; Gu et al., 2020). For each industry, we retain the last 30% of in-sample observations (i.e., the training dataset) as the validation observations, corresponding to approximately 20% of the total sample (see also Granger, 1993; Gu et al., 2020). We effectively create 49 validation data matrices. The aggregate validation dataset is constructed by stacking these validation data matrices on a panel data format structure. Regarding the hyperparameter, we explore every possible combination and retain the optimal one as the optimal TabNet architecture used to generate the OOS predictions. Naturally, the optimal architecture is the one that achieves the highest accuracy metric on the validation dataset. We explore different values for the batch size and epochs (i.e., number of training iterations) as well as the number of steps, N_{steps}, and λ_{sparse} coefficient we described in sections 3.4.2-3.4.4 of our methodology. The financial literature has no consensus regarding the optimal batch size range and number of epochs. We try two

different values for each hyperparameter and heuristically arrive at the optimal one. Regarding the N_{steps} hyperparameter, we follow Arik and Pfister (2021) who propose using values between three and ten. The authors note that higher values for N_{steps} can potentially lead to model over-fit. In our study, we explore two values for N_{steps}; specifically, the most conservative possible out of those recommended (i.e., $N_{steps} = 3$) as well as a higher candidate value (i.e., $N_{steps} = 6$). Finally, for the λ_{sparse} coefficient, we consider values of 0.001, 0.1. $\lambda_{sparse} = 0.001$ is the Python library's default parameter value, and we also include a higher λ_{sparse} value (i.e., $\lambda_{sparse} = 0.1$). Table 3.1 compiles the different hyperparameters we examined to optimize TabNet's architecture.

Table 3.1. The hyperparameters' search space for TabNet's architecture.

The table displays the different hyperparameters of TabNet's architecture we optimized for using a validation dataset. The validation dataset holds a panel data structure and was constructed using the last 30% of time series observations from each industry's in-sample dataset.

TabNet Hyperparameters	
Batch size	200, 300
Epochs	25, 50
N_{steps}	3, 6
λ_{sparse}	0.001, 0.1
Total number of candidate architectures	**8**

3.5. Empirical results

3.5.1 Forecasting accuracy

We compare TabNet's OOS performance with a battery of machine learning models, including state-of-the-art techniques (Appendix 3.B.1-3.B.4). All models are trained on the in-sample dataset ranging from January 1976 to December 2012, which has a panel format data structure as described in our Data section. The trained models are then used to generate predictions for the OOS dataset, which spans from January 2013 to December 2022. All models are evaluated on the OOS dataset, equivalent to approximately 20% of the total sample.

We use several Python libraries to conduct our analysis. Numpy (Harris et al., 2020) and pandas (McKinney, 2010) are applied for data preprocessing tasks. The TabNet model is developed using PyTorch (Paszke et al., 2019) and pytorch_tabnet (DreamQuark-ai, 2023) libraries. Logistic regression

and random forest (RF) benchmark models are implemented using scikit-learn (Pedregosa et al., 2011), while xgboost (Chen & Guestrin, 2016) library is used to estimate the XGBoost (XGB) model. Moreover, the XGBoost-Generalized Additive Model (XGB-GAM) is implemented using the nodegam (Changet al., 2021) library, while the explainable boosting machine (EBM) model is implemented with the interpret (Nori et al., 2019) library. To ensure reproducibility, we fix the random and serialize the trained models using the pickle library, enabling consistent reuse and further analysis.

As suggested in other studies investigating the performance of classification models for financial applications (see Lessmann et al., 2015; Gunnarsson et al., 2021) we include multiple metrics to effectively evaluate the model's classification accuracy and do not rely on a single measure. We use five different classification performance metrics, and precisely, accuracy, balanced accuracy, logistic loss, brier score loss, and the area under the receiver operating characteristic (ROC) curve (AUC statistic). In Appendix 3.C., we provide descriptions of the accuracy metrics employed. A higher value indicates a more accurate classification model for the accuracy metric, balanced accuracy metric, and AUC statistic. In contrast, a lower value denotes a relatively higher predictive accuracy for the logistic and the brier score loss. Additionally, for all models, we display the ROC curves from which the AUC statistic can be derived, as explained in Appendix 3.C.3. The AUC statistic equals the probability that a randomly chosen positive case will be ranked higher (i.e., receive a higher probability score) than a randomly chosen negative case; therefore, higher values are equivalent to higher predictive accuracy.

Table 3.2 validates that TabNet achieves the highest classification accuracy when predicting the directional movements of industry excess returns. This result is supported unanimously by all the performance metrics. In detail, TabNet achieves the highest accuracy (i.e., 0.6430), balanced accuracy (i.e., 0.6066), and AUC statistic (i.e., 0.6486), and the lowest logistic loss (i.e., 0.6458) and brier score loss (i.e., 0.2266). The second best-performing model is the logistic regression model, as dictated by 3 out of 5 performance metrics (accuracy, balanced accuracy, and the AUC statistic). Expectedly, the logistic regression with L_2 penalty achieves a very similar performance to the standard logistic regression model. The logistic regression with L_2 penalty outperforms the logistic regression in 2 out of 5 performance metrics (i.e., the logistic and brier score loss) and, for this reason, is ranked as the third best-performing model. In terms of accuracy, the XGB-GAM achieves the worst performance, while the RF model has a balanced accuracy of slightly over 50%. Since balanced accuracy considers potential class imbalances when calculating the accuracy score for a model, a large discrepancy between the accuracy and balanced accuracy metrics can potentially signal a model being biased towards the more dominant class in the dataset. The RF model has the largest discrepancy between its accuracy and balanced accuracy metrics (i.e., 0.0784), while TabNets achieves one of the smallest such discrepancies (i.e., 0.0364) among all models. The difference between the attained accuracy and

balance accuracy for the logistic regression (i.e., the second best-performing model) is 0.0601, which is approximately double the equivalent value for the case of TabNet. With respect to the logistic loss and the AUC statistic, TabNet achieves the best values by a large margin compared to the benchmark models. The logistic models have similar AUC values, at 0.5834 and 0.5833, respectively, while the XGB default model follows with an AUC value of 0.5494. Again, the RF model attains the lowest position in terms of the AUC statistic.

Finally, Figure 3.3 displays for all models the ROC curves, which are constructed based on the predictions for the OOS period. The position of the ROC curve reflects the accuracy of a classification model. The ROC of random classifier is placed on the diagonal line, while a greater area under the curve denotes better prediction power. Figure 3.3 establishes that the ROC curve associated with TabNet covers greater area when compared to all other benchmark models. Therefore, the graphs authenticate that TabNet generates the most accurate predictions.

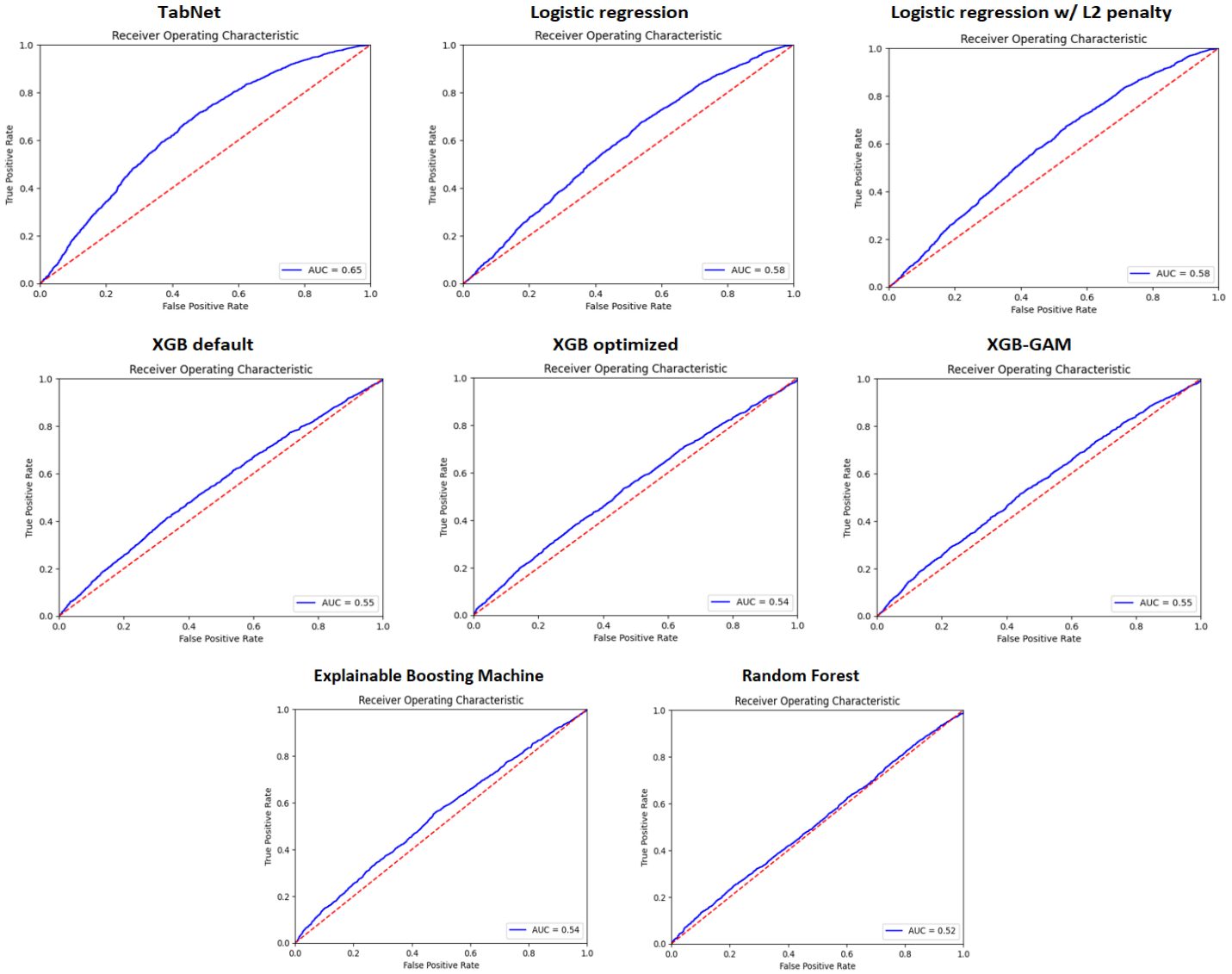
Table 3.2. The OOS classification accuracy metrics for the predictive models.

The table reports the classification accuracy metrics which evaluate the models' predictions. All models are evaluated on the OOS period spanning from January 2013 to December 2022. In our study, we utilize 5 different performance metrics. For the accuracy and balanced accuracy metrics, as well as the AUC statistic, a higher value indicates a relative higher predictive accuracy. On the other hand, for the logistic and brier score loss, a lower value denotes relatively higher predictive accuracy.

Directional Accuracy Metric		**TabNet**	**Logistic Regression**	**Logistic Regression (l_2 penalty)**	**XGB default**	**XGB optimized**	**XGB-GAM**	**Explainable Boosting Machine**	**Random Forest**
Accuracy (Acc)		0.6430	0.6111	0.6099	0.5738	0.5719	0.5565	0.5650	0.5827
Balanced Accuracy (BA)		0.6066	0.5510	0.5494	0.5286	0.5178	0.5298	0.5222	0.5043
$\|Acc- - BA\|$		0.0364	0.0601	0.0605	0.0452	0.0541	0.0267	0.0428	0.0784
Logistic Loss		0.6458	0.8293	0.8241	0.8118	0.7502	0.7640	0.7416	0.6826
Brier Score loss		0.2266	0.2578	0.2573	0.2803	0.2662	0.2705	0.2650	0.2447
AUC statistic		0.6486	0.5834	0.5833	0.5494	0.5428	0.5456	0.5441	0.5171

Figure 3.3. The ROC curves for TabNet and the benchmark models.

The figure displays the ROC for all the employed models in our study. Based on the ROC curves, we also derive the AUC statistic which is reported on the bottom-right position in each of the ROC graphs. The ROC curves are constructed using the model's predictions for the OOS period spanning from January 2013 to December 2022.



3.5.1.2 Statistical tests

This subsection presents the statistical tests we employ to evaluate the classification models. To compare TabNet against all benchmark models, we use both pairwise (i.e., McNemar's test) and multiple comparison statistical tests such as (Cochran's Q-test and F-test[37]). For the pairwise test, we employ the necessary p-value correction. Moreover, following relevant literature (see, Fischer and Krauss, 2018), we use the Pesaran-Timmermann test to assess the predictive accuracy of the models. In Appendix 3.D., we provide extensive descriptions for all statistical tests.

The McNemar's test (McNemar, 1947; Edwards, 1948) conducts pairwise comparisons between two machine learning classification models. Under the null hypothesis of the test, the first and the second model have the same error rate. Therefore, rejecting the null hypothesis denotes that there is a significant difference in the classification accuracy of the two models. In our study, we utilize the correction proposed by Edwards (1948) and apply McNemar's test to compare TabNet against all benchmark models in a pairwise manner. Table 3.3 Panel A presents the test results. Based on these findings, we can safely conclude that we reject the null hypothesis in all cases. Consequently, there is a statistically significant difference between the error rates of TabNet and all benchmark models. Therefore, the test highlights significant differences in the predictive performance of TabNet when compared with all other models. Since the test is performed pairwise, we also include two p-value correction procedures, specifically the Bonferroni (1936) and the Hommel (1998) p-value correction methods. The p-value correction process further validates the initial results and does not change the initial picture. The χ^2-statistics range from 23.9186 (*p*-value: 0.0000) for the logistic regression model to 129.2258 (*p*-value: 0.0000) for the XGB-GAM model.

Cochran's test (Cochran, 1950) can be used to evaluate multiple classifiers and be regarded as a generalization of McNemar's test. For a set $\{ML_1, ML_2, \dots, ML_L\}$ of classification models, under the null hypothesis, there is no difference between the classification accuracies p_{acc} of the machine learning models (i.e., $H_0: p_1 = p_2 = \cdots = p_L$). Similar to Cochran's test, the test developed by Snedecor and Cochran (1989) can assist in evaluating multiple machine learning classification models. The F-test assesses the null hypothesis that there is no difference in the classification accuracies of a set of L machine learning models (i.e., $H_0: p_1 = p_2 = \cdots = p_L$). For a set of classifiers $\{ML_1, ML_2, \dots, ML_L\}$, if the models do not perform differently on the OOS dataset, then the test statistic (i.e., F-statistic) follows the F distribution with $(L-1)$ and $(L-1)\,\mathrm{x}\,T$ degrees of freedom.

[37] The McNemar test, The Cochran's Q test, and the F-test, are implemented using the mlxtend Python library (Raschka, 2018).

Table 3.3. The McNemar's pairwise test, Cochran's Q test, and F-test results for the predictive models.

The table displays the results for the McNemar's test, Cochran's Q test, and F-test for all predictive models explored in this study. The tests are applied to the models' OOS predictions for the period January 2013 to December 2022. For the case of the McNemar's test (Panel A), we also provide a Bonferroni (1936) and Hommel (1998) *p*-value correction, since the test is applied in a pairwise manner. Finally, in Panel B, we present the results for Cochran's Q test and the F-test. These tests do not require a *p*-value correction since they do not operate based on pairwise comparisons but consider the entire model set. *, **, *** denote significance at the 10%, 5% and 1% level, respectively.

Panel A	**McNemar's pairwise test**				
	Benchmark classification models	**χ^2-statistic**	***p*-value**	**Bonferroni (1936) p-value correction**	**Hommel (1998) p-value correction**
TabNet	**Logistic Regression**	23.9186	0.0000***	0.0000***	0.0000***
	Logistic Regression (w/ l_2 penalty)	25.8312	0.0000***	0.0000***	0.0000***
	XGB default	84.1429	0.0000***	0.0000***	0.0000***
	XGB optimized	100.8637	0.0000***	0.0000***	0.0000***
	XGB-GAM	129.2258	0.0000***	0.0000***	0.0000***
	Explainable Boosting Machine	106.7501	0.0000***	0.0000***	0.0000***
	Random Forest	75.6282	0.0000***	0.0000***	0.0000***

Panel B	**t-statistic**	**p-value**
Cochran's Q test	129.3037	0.0000***
F-test	21.6262	0.0000***

Rejecting the null hypothesis of Cochran's test and the F-test highlights significant differences in the performance of the classification models. We apply the two tests on the OOS predictions generated by TabNet and all benchmarks.

The Cochran's Q and F-test results in Panel B of Table 3.3 further confirm the outcome of McNemar's test. Both tests assess the null hypothesis that there is no difference between the classification accuracies of the examined classification models. Given that for Cochran's Q test, the t-statistic is 129.3037, and for the F-test, the t-statistic is 21.6262, we can reject the null hypothesis of both tests. The *p*-values for the same tests are as expected at the 0.0000 level. The above findings authenticate a statistically significant difference between the predictive accuracy of TabNet and the benchmark models. This observation should be considered in conjunction with the insights from Table 3.2, compiling the performance metrics. We can statistically significantly argue that TabNet has the best performance for predicting the directional movements of industries' excess returns.

Finally, we leverage the Pesaran and Timmermann (1992) test that evaluates the OOS classification accuracy of machine learning models. The null hypothesis is that a model's predictions and the target variable y_t are independently distributed (see also Fischer & Krauss, 2018). Therefore, failing to reject the null hypothesis signifies an inferior forecasting model. We implement the test for TabNet and each of the benchmarks using their OOS predictions and report the results in Table 3.4. Considering the results for TabNet, we can reject the null hypothesis with high confidence (*p*-value: 0.0000). Notably, TaNet achieves the highest test statistic (i.e., 17.8477) across all models. These results confirm that TabNet's predictions are not independently distributed from the target variable, which in our case captures the directional movements of industries' excess returns. Except for the RF model, we reject the null hypothesis for the rest of the benchmark models. These results indicate that all other models exhibit statistically significant forecasting accuracy except for the RF model.

Based on the above observations and considering both the performance metrics of Table 3.2 as well as the statistical test results, we can derive several insights. First, a state-of-the-art neural network architecture can outperform linear (i.e., logistic regressions) and tree-based models when predicting the directional movements of industry excess returns. Neural networks can effectively capture non-linearities and complex data patterns, leading to higher forecasting performance. While a relatively vast literature (see among others, Gu et al., 2020; Filippou et al., 2022) has explored the predictive ability of neural networks in the context of predicting financial returns as a continuous target variable (i.e., a regression task), there is scarce literature exploring the predictive power of neural networks for classification objectives. Our results establish that neural networks can be equally effective for classification tasks such as the one we explore in this study.

Table 3.4. The results for the Pesaran-Timmerman directional accuracy test.

The table demonstrates the test results for the Anatolyev-Gerko excess profitability test and Pesaran-Timmerman directional accuracy test across the model set. The tests are performed based on the models' predictions for the OOS period spanning from January 2013 to December 2022. For each test, we report the *t*-statistic, as well as the *p*-value. *, **, *** denote significance at the 10%, 5% and 1% level, respectively.

	Pesaran-Timmerman Directional Accuracy test	
	***t*-statistic**	***p*-value**
TabNet	17.8477	0.0000***
Logistic Regression	10.3761	0.0000***
Logistic Regression (w/ l_2 penalty)	10.0994	0.0000***
XGB default	4.98757	0.0000***
XGB optimized	3.34482	0.0000***
XGB-GAM	4.71625	0.0000***
Explainable Boosting Machine	3.79651	0.0000***
Random Forest	1.2376	0.1079

Second, contrary to the conclusions drawn by Gunnarsson et al. (2021) regarding the relative ineffectiveness of neural networks for credit scoring classification tasks, we show that TabNet can outperform all benchmarks, including the state-of-the-art XGB model and its variants. These results are aligned with the evidence in Fischer and Krauss (2018), which supports the utility of a neural network architecture for the classification task of a stock return over-(under-) performing the cross-sectional median return. Third, our analysis shows that a linear model can outperform tree algorithms, even the more advanced versions (i.e., the XGB variants). This outcome supports that adopting a sophisticated forecasting model should not be a panacea for financial tasks. On the other hand, non-linear neural network models, such as TabNet, can provide statistically significantly more accurate predictions than all other models. We can conclude that not all non-linear models outperform their linear counterparts by default; however, those that do can potentially provide significant predictive insights and benefits. For this reason, portfolio managers and investors should experiment beyond the realm of conventional linear models and are

encouraged to include TabNet in their modeling toolbox, given its exceptional performance and interpretability.

3.5.2 Covariate importance

In Figure 3.4, we display the covariate importance for the OOS period and rank the covariates from higher importance to relatively lower. We calculate importance as discussed in Section 4.5 of our methodology and present the results for the 15 most influential covariates (i.e., approximately 10% of the total covariate pool) driving the model's OOS predictions. By examining the figure, we conclude that the most crucial covariate categories by frequency of presence in these 15 positions are valuation ratios, lagged returns, other-industry lagged returns and financial soundness. By focusing on the five most important covariate categories, we can observe that the three top positions are dominated by valuations ratios, followed by an other-industry lagged return and a 12-month lagged return covariates. A large body of literature supports the significance of valuation ratios for predicting financial returns (Keim & Stambaugh, 1986; Fama & French, 1988; Campbell & Shiller, 1988; Campbell & Thompson, 2008). Our results confirm that valuation ratios can be equally important in predicting the directional movements of financial returns, specifically industries' excess returns. Additionally, we underline that our modelling setup differs from the typical setup in which a predictive model forecasts a continuous variable, usually excess asset returns. However, we prove that TabNet's neural network architecture can still extract the most informative data patterns present in the valuation ratios even for a vastly different predictive objective, that is, a classification of directional movements.

We derive additional insights by observing that three out of the ten most significant covariates belong to the lagged excess returns covariate category, and 3 out of the 15 most significant covariates belong to the other-industry lagged excess return category. The results establish that information held in lagged excess returns can assist TabNet in creating accurate predictions. Moreover, as shown in Figure 3.4, TabNet has chosen the 12-month, 10-month, and 6-month excess return lags as the most informative. The 12-month and the 6-month lag can hold information and data patterns relative to serial correlation and seasonality effects as reported in the financial literature (Jegadeesh, 1990; see also Heston & Sadka, 2008). These data patterns were captured and leveraged by TabNet when its classification decisions were made. Moreover, the fact that 20% of the 15 most important covariates belong to the other-industry return category is aligned with the literature exploring the predictability of industry returns and the diffusion of information across the industries. Rapach et al. (2015; 2019) have provided extensive evidence that other industry-lagged excess returns can assist in predicting the individual industry's excess returns. In line with the studies mentioned above, we prove the existence of solid economic links between the industries. In practice, these economic links can take

the form of "buyer-seller" relationships, supply chain interactions, common reactions towards macroeconomic conditions and policy decisions, and technology spillovers. Finally, the financial soundness covariate category also has a strong presence with 5 out of the 15 covariates belonging to this category (i.e., Short-Term Debt/Total Debt, Receivables/Current Assets, Current Liabilities/Total Liabilities, Interest/Average Total Debt, Long-Term Debt/Book Equity), and the profitability category has a minor presence with only one covariate (i.e., Return on Assets).

3.5.3 Trading application

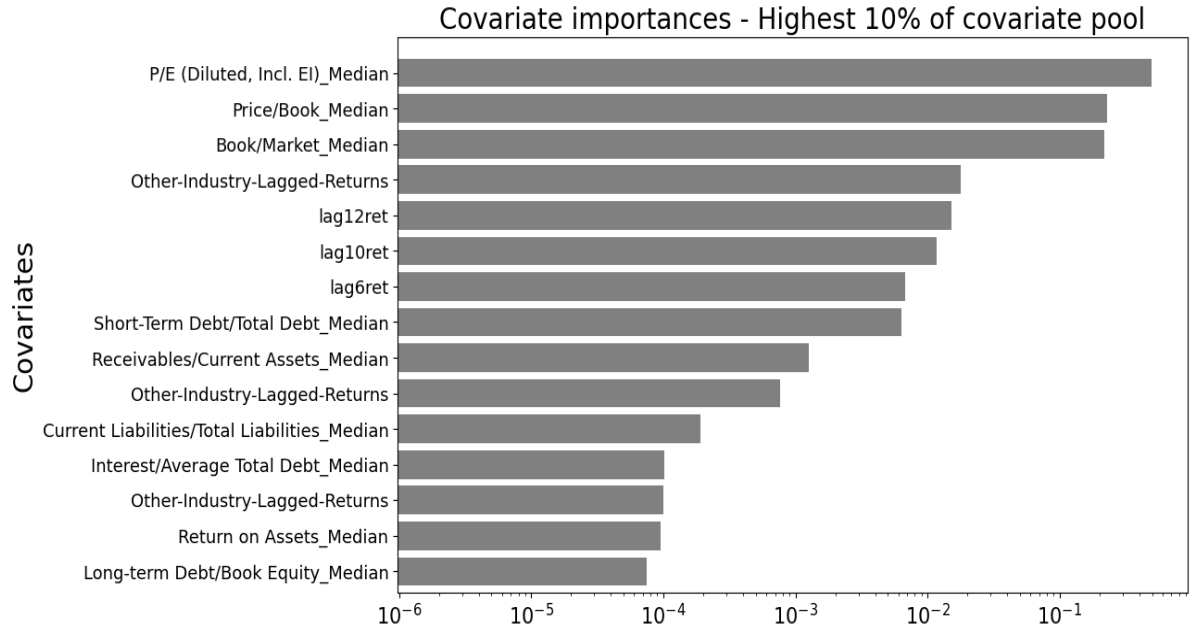
We construct a trading application to assess the economic significance of TabNet's predictions in the OOS period. To construct the trading application, we follow relevant literature (Leung et al., 2000; Karhunen, 2019; Iworiso & Vrontos, 2019) and assume that each month, an investor shifts their wealth between an industry portfolio and the one-month Treasury Bill (T-Bill). To make an informative decision, the investor uses a predictive model, such as TabNet. Based on the model's predicted probability of a positive excess industry return for next month, the following decision rule is employed to construct the trading strategy:

$$\begin{cases} \boldsymbol{If}\ \Pr(r_t > 0 | \mathrm{X}_{t-1}) = \Pr(y_t = 1 | \mathrm{X}_{t-1}) > 0.5 \ \rightarrow\ purchase\ the\ industry\ portfolio \\ \boldsymbol{If}\ \Pr(r_t > 0 | \mathrm{X}_{t-1}) = \Pr(y_t = 1 | \mathrm{X}_{t-1}) \leq 0.5 \ \rightarrow\ purchase\ the\ TBill \end{cases} \tag{3.10}$$

In our study, we examine 49 industry portfolios. It follows that each month, the investor will have to decide for which industries to go long and for which cases it would be preferable to purchase the TBill, as dictated by the model's predictions. After determining for each industry which position to hold, the investor equal-weights all 49 trading positions and forms the final investing portfolio for each month t. We then calculate the trading strategy's performance based on these weights and the OOS realized returns. The trading strategy formed based on TabNet's forecasts is evaluated in two ways. First, we repeat the same technique to construct an investing portfolio, but instead of using TabNet's predictions, we use the forecasts of a different predictive model. Naturally, we use the forecasts of the most accurate model out of the benchmark model set. The most accurate benchmark model will be decided by inspecting the OOS accuracy metrics described in section 3.5.1. Second, we compare our trading strategy against a buy-and-hold strategy on three market indices, specifically the CRSP Value-Weighted index (CRSP-VW), the CRSP Equal-Weighted index (CRSP-EW), and the Standard and Poor's 500 (SP500) index. Finally, for all trading strategies, we calculate the annualized return, annualized Sharpe ratio, and the alpha values against the four-factor (Fama & French, 1993; Carhart, 1997) and five-factor (Fama & French, 2015) models.

Figure 3.4. Covariate importance derived from the TabNet model.

The figure displays the 15 most informative covariates according to the TabNet model. These 15 covariates, representing approximately 10% of the total covariate pool, have on average (i.e., across the OOS period; January 2013 – December 2022) the highest contribution to the model's predictions as described in our methodology section. The covariates are ranked in order of importance, from higher to relatively lower.



3.5.3.1 Anatolyev-Gerko excess profitability test

The Anatolyev and Gerko (2005) test assesses a trading strategy that issues a buy signal if the prediction for the next period's return (here, excess return) is positive and a short signal otherwise. The average return of the strategy above is compared with a benchmark strategy that forms buy/sell signals randomly to test for the significance of return predictability and excess profit (Liu et al., 2019). Under the null hypothesis, the average return of the trading strategy formed based on the sign of the return predictions should statistically be equal to a benchmark strategy that issues buy/sell signals at random with probabilities corresponding to the proportion of "buys" and "sells" implied ex-post by the trading strategy (Anatolyev & Gerko, 2005). Failing to reject the null hypothesis demonstrates that a classification model does not perform significantly better than a classifier that would randomly produce a positive or negative sign prediction for the next period's excess return. Appendix 3.D.5. provides a detailed description of the test.

3.5.3.2 Trading application results

In this subsection, we present the trading application results for the OOS period spanning from January 2013 to December 2012. The positions are constructed using TabNet's predictions. In case of a positive class prediction (i.e., $y_t = 1$) for next month, we purchase the industry portfolio, whereas

in case of a negative class prediction, we purchase the TBill. This decision rule is applied across all 49 industries, and then all positions are equal-weighted. In Table 3.5 we compile the trading application results, which prove that TabNet can generate economically meaningful forecasts. Specifically, the trading strategy that is constructed using TabNet's predictions generates the highest annualized return and Sharpe ratio compared to both the trading strategy that uses the logistic regression's predictions and the buy-and-hold strategies on the three market indices.

TabNet's equal-weighted (EW) trading strategy has an annualized return of 18.04% and a Sharpe ratio at the 1.8807 level. In contrast, for the case of the logistic regression, the strategy attains an annualized return of 14.79% and a Sharpe ratio at 1.1914. Out of the three market indices, the buy-and-hold strategy on the VW-CRSP index achieves the highest annualized return (i.e., 11.43%) and Sharpe ratio (i.e., 0.7659). The profitability of TabNet's EW trading strategy persists after we apply monthly transaction costs at five and ten basis points. For the ten basis points monthly transaction cost, we observe in Table 3.5 that TabNet has an annualized return of 16.84% and Sharpe ratio at the 1.7556 level. These performance metrics are superior to the corresponding metrics of the trading strategy based on the logistic regression's predictions and the VW-CRSP index buy-and-hold strategy. The alpha values obtained by the four-factor and five-factor model regressions provide similar insights favouring TabNet. In more detail, the alpha values for trading strategy based on TabNet's predictions are 11.69% and 11.03% for the four-factor and five-factor model, respectively. Both these alpha values are significant at 1% level and remain statistically significant at 1% even after applying the more conservative (i.e., ten basis points) monthly transaction cost. For the case of the logistic regression EW trading strategy, the alpha values are at a lower level in comparison, precisely at 6.61% and 4.39% for the four-factor and five-factor models, respectively. For the case of the logistic regression trading strategy, the five-factor alpha value does not remain significant at the 1% level when we apply five and ten basis points monthly transaction costs. Finally, the portfolio constructed based on the proposed model's directional forecasts has the lowest volatility and maximum drawdown values. The above results further outline TabNet's superior predictive accuracy and ability to generate profitable predictions.

Finally, in Table 3.6, we display the results for the Anatolyev-Gerko excess profitability test, which reveals the same outcome as the Pesaran-Timmerman test. Considering the test results for TabNet, we can reject the null hypothesis with high confidence (p-value: 0.0000). These results confirm that TabNet's forecasts can achieve excess profitability in the context of the Anatolyev-Gerko test. Except for the RF model, we can also reject the null hypothesis for the rest of the benchmark models. The highest test statistic for the Anatolyev-Gerko (i.e., 20.4898) test is achieved by TabNet.

Table 3.5. Trading application results for the OOS period.

The table reports the trading application results for the OOS period from January 2013 to December 2022. We use TabNet's monthly predictions to form an equally-weighted strategy across the 49 industries, in which we purchase the industry portfolio in case of a positive prediction and the TBill otherwise. We compare the economic significance of TabNet's predictions against the logistic regression predictions (i.e., the second-best performing model) and a "buy-and-hold" strategy on three market indices (i.e., VW-CRSP, EW-CRSP, SP500). For the trading strategies based on model predictions, we also include monthly transaction costs at the five and ten basis points level. We report the annualized return and Sharpe ratio performance metrics and the alpha values derived by the four-factor and five-factor Fama-French models. *, **, *** denote significance at the 10%, 5% and 1% level, respectively.

	TabNet			**Model Benchmark Logistic Regression[38]**			**Indices Benchmarks**		
	EW-trading strategy			**EW-trading strategy**			**Buy-and-Hold**		
	No TC[39]	**5bp TC**	**10bp TC**	**No TC**	**5bp TC**	**10bp TC**	**VW-CRSP**	**EW-CRSP**	**SP500**
Ann. Mean Return (%)	18.04	17.44	16.84	14.79	14.19	13.59	11.43	8.06	11.03
Volatility (%)	9.59	9.59	9.59	12.41	12.41	12.41	14.93	16.71	14.76
Annualized Sharpe Ratio	1.8807	1.8182	1.7556	1.1914	1.1430	1.0947	0.7659	0.4826	0.7475
Max Drawdown (%)	5.24	5.67	6.19	13.03	13.21	13.39	24.68	30.60	24.77
Ann. 4-factor alpha (%)	11.69***	11.09***	10.49***	6.61***	6.01***	5.41***	-	-	-
Ann. 5-factor alpha (%)	11.03***	10.43***	9.83***	4.39***	3.79**	3.19**	-	-	-

[38] Logistic regression is the second-best performing model after TabNet in terms of predictive accuracy metrics (and the best model from the benchmark model set).

[39] TC = Transaction (T) Costs (C).

Table 3.6. Results for the Anatolyev-Gerko excess profitability test.

The table demonstrates the test results for the Anatolyev-Gerko excess profitability test and Pesaran-Timmerman directional accuracy test across the model set. The tests are performed based on the models' predictions for the OOS period spanning from January 2013 to December 2022. For each test, we report the *t*-statistic, as well as the *p*-value. *, **, *** denote significance at the 10%, 5% and 1% level, respectively.

	Anatolyev-Gerko Excess Profitability test	
	***t*-statistic**	***p*-value**
TabNet	20.4898	0.0000***
Logistic Regression	13.3723	0.0000***
Logistic Regression (w/ l_2 penalty)	13.1944	0.0000***
XGB default	3.8578	0.0001***
XGB optimized	1.7456	0.0404**
XGB-GAM	5.1952	0.0000***
Explainable Boosting Machine	5.6578	0.0000***
Random Forest	-0.2467	0.5974

3.6. Conclusion

This paper uses a state-of-the-art and "white-box" deep learning method to predict the directional movements of industries' excess returns. We employ the TabNet model, a neural network architecture that uses a multi-step sequential processing mechanism to extract meaningful data patterns, derive covariate importance, and generate accurate predictions. To evaluate the OOS predictive ability of TabNet, we employ a benchmark model set consisting of linear and machine learning models, including state-of-the-art techniques, and use multiple performance metrics and statistical tests. To assess the economic significance of TabNet's predictions, we construct an EW trading strategy and compare it to a benchmark trading strategy and buy-and-hold strategies on three market indices.

Our findings show that TabNet achieves superior predictive accuracy and generates profitable predictions in the scope of a trading strategy. The forecasting application's results validate that TabNet has the highest performance considering all performance metrics (i.e., accuracy, balanced accuracy, logistic loss, brier score loss, and the AUC statistic) and statistical tests (i.e., Cochrans Q-test, the F-

test, the Anatolyev-Gerko and Pesaran-Timmerman tests). The second best-performing model is the logistic regression model, and the worst is the random forest. We derive covariate importance directly from TabNet's architecture and conclude that the most significant categories are the valuation ratios, lagged returns, other-industry lagged returns, and financial soundness. The three top positions with the most significant covariates belong exclusively to the valuation ratios category, while the fourth and fifth most informative covariates belong to the other-industry lagged return and lagged return categories. The reported significance of the other-industry lagged return category confirms the presence of strong economic links and interdependencies among the industries and the diffusion of information across the economic sectors. The 12-month lagged return is the fifth most influential covariate, which indicates the presence of seasonality effects and that past excess returns can hold important information when predicting the directional movements of future excess returns. Regarding the trading application, the EW trading strategy constructed using TabNet's predictions is the most profitable, has the higher Sharpe ratio and attains positive and statistically significant alphas against the four-factor and five-factor models. The alpha values remain positive and statistically significant even after including transaction costs.

Machine learning models like TabNet provide significant advantages in empirical asset pricing by capturing complex, non-linear relationships that traditional models like Fama-French (2015) often overlook. TabNet's ability to handle large datasets and dynamic feature selection is highly valuable, but challenges like reduced connection to economic theory, data intensity, and computational complexity remain. Future research could explore hybrid models that combine the interpretability of traditional methods with TabNet's flexibility, leveraging its performance to discover new factors while grounding insights in well-established financial theories. The aforementioned research directions could further contribute to creating theoretically sound models that are better equipped to handle real-world complexities.

Overall, our results should persuade portfolio managers to incorporate state-of-the-art and interpretable neural networks, such as TabNet, as part of their predictive toolbox. TabNet exhibits superior predictive ability against linear and other machine learning models without sacrificing performance for interpretability, as with other neural network models. Finally, predicting the directional movements of industries' returns via an effective framework can be profitable and generate statistically significantly abnormal returns as the factor models indicate.

Appendix 3

3.A. 49 industry sectors

The table presents the 49 industry sectors we examine in our study. The 49 industries' classification follows the Fama-French paradigm and the descriptions provided on Kenneth French's website[40].

Industry sector	Symbol
Agriculture	AGRIC
Food Products	FOOD
Candy & Soda	SODA
Beer & Liquor	BEER
Tobacco Products	SMOKE
Recreation	TOYS
Entertainment	FUN
Printing and Publishing	BOOKS
Consumer Goods	HSHLD
Apparel	CLTHS
Healthcare	HLTH
Medical Equipment	MEDEQ
Pharmaceutical Products	DRUGS
Chemicals	CHEMS
Rubber and Plastic Products	RUBBR
Textiles	TXTLS
Construction Materials	BLDMT
Construction	CNSTR
Steel Works Etc	STEEL
Fabricated Products	FABPR
Machinery	MACH
Electrical Equipment	ELCEQ
Automobiles and Trucks	AUTOS
Aero	AIRCRAFT
Shipbuilding, Railroad Equipment	SHIPS
Defense	GUNS
Precious Metals	GOLD
Non-Metallic and Industrial Metal Mining	MINES
Coal	COAL
Petroleum and Natural Gas	OIL
Utilities	UTIL
Communication	TELCM
Personal Services	PERSV
Business Services	BUSSV
Computers	HARDW
Computer Software	SOFTW
Electronic Equipment	CHIPS
Measuring and Control Equipment	LABEQ
Business Supplies	PAPER

[40] Kenneth French website: http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/index.html

Shipping Containers	BOXES
Transportation	TRANS
Wholesale	WHLSL
Retail	RTAIL
Restaurants, Hotels, Motels	MEALS
Banking	BANKS
Insurance	INSUR
Real Estate	RLEST
Trading	FIN
Almost Nothing	OTHER

3.B. Benchmark models

To adequately assess TabNet's accuracy, we utilize an extensive set of benchmark predictive models.

3.B.1 Logistic regression

The logistic regression model has been widely used in financial applications involving classification tasks (see among others, Fischer & Krauss, 2018; Dumitrescu et al., 2022), and its advantage is its simple interpretation. The model searches for a single linear decision boundary in the covariates' space to generate the class predictions (Dumitrescu et al., 2022). The logistic regression admits the following mathematical formulation:

$$\Pr(y_t = 1 | \mathrm{X}_{\mathrm{t}-1}) = F\big(\xi(\mathrm{X}_{\mathrm{t}-1}; \beta)\big) = \frac{1}{1 + e^{(-\xi(\mathrm{X}_{\mathrm{t}-1}; \beta))}}$$

$$\xi(\mathrm{X}_{\mathrm{t}-1}; \beta) = \beta_0 + \sum_{j=1}^{D} \beta_j x_{t-1,j}$$

where, F is the logistic cumulative distribution function and β is a vector of parameters.

The parameters' vector β is obtained by maximizing the log-likelihood function (see Dumitrescu et al., 2022). In addition to the formulation above, we also include a logistic regression with L_2 penalty term in our benchmark model set. The L_2 The penalty is added to the logistic regression's loss function and acts as a regularization term. To optimize for the value of the L_2 we follow Fischer and Krauss (2018). We use 5-fold cross-validation and select the optimal choice out of 100 L_2 candidate values on a logarithmic scale between 0.0001 and 10000.

3.B.2 EXtreme Gradient Boosting and the Generalized Additive Model variation

The EXtreme Gradient Boosting (XGBoost) model was introduced in the work of Chen and Guestrin (2016) and extends the boosting algorithm developed by Friedman (2001). Gunnarsson et al. (2021) show that XGBoost can outperform state-of-the-art neural network models in their credit-scoring

financial application. Motivated by this research, we add XGBoost to the benchmark model set and compare the attained performance with the one achieved by the TabNet neural network architecture. Regarding the mathematical formulation of XGBoost, Nobre and Neves (2019) note that the model's output can be calculated with the formula:

$$\widehat{y_t} = \sum_{k=1}^{K} f_k(\mathrm{X}_{\mathrm{t}-1}), \;\; f_k \in \mathcal{F}$$

where, f is a function in the functional space $\mathcal{F}$, $\mathcal{F} = \{f(\mathrm{X}) = w_{q(\mathrm{X})}\}$ is the space of the classification trees[41], q is the structure of each classification tree, w is the leaf weight, f_k is the k-th classification tree, K is the number of classification trees.

The loss function that is optimized to train the model effectively is the following:

$$L = \sum_{t} l(\widehat{y_t}, y_t) + \sum_{k} \Omega(f_k)$$

where, l is the cross-entropy loss function measuring the difference between the predicted class $\hat{y}$ and the true class y, and Ω is a regularization term. Ω is specified by the following formula:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \| w\|^2$$

where, γ is a regularization hyperparameter, T is the number of leaves in each classification tree, and λ is a regularization hyperparameter.

In our XGBoost implementation, we use the xgboost Python library (Chen & Guestrin, 2016) associated with the original paper. In our benchmark model set, we include two versions of the XGBoost model. For the XGBoost version utilizing the Python library's default parameters (XGB default, hereafter), the number of trees is set to 100, the maximum depth of a tree to 3, the learning rate to 0.1, γ to 0, and λ to 1. The second XGBoost version is optimized using 5-fold cross-validation and the parameter search space of Gunnarsson et al. (2021). In more detail, we explore three different values for the number of trees (i.e., 50, 100, 150) and the maximum depth of a tree (i.e., 1, 2, 3), 2 different values for the fraction of inputs used to construct each classification tree (i.e., 0.6, 0.8) and the learning rate (i.e., 0.3, 0.4), and keep the rest of the parameters to the Python library's default settings.

We extend the benchmark model set by an additional XGBoost variation, specifically a Generalized Additive Model (GAM) version of XGBoost. Chang et al. (2021) suggest that converting an XGBoost

[41] Hereafter, we refer to the decision tree model applied for a classification task as "classification tree".

model to a GAM structure (XGB-GAM) requires fixing the maximum depth of the classification trees to 1 so that the trees do not learn feature interactions. In their study, the authors provide evidence that GAM models based on tree-methods, such as XGB-GAM, outperform other GAM variations in terms of accuracy and discovery of data patterns. For the implementation of this XGBoost variation, we use the Python library associated with the original paper[42]. The number of trees is set at 5000, all inputs are used to construct each classification tree (i.e., not a fraction of them), the learning rate is 0.1, γ is set to 0, and λ to 1.

3.B.3 Explainable Boosting Machine

Norti et al. (2019) introduced the Explainable Boosting Machine (EBM), which is a "white-box" machine learning model offering high interpretability. According to the authors, EBM also achieves accuracy equal to other state-of-the-art predictive techniques. On a general level, EBM is a GAM model with the following form:

$$g\big(E(y_t)\big) = \beta_0 + \sum_{j=1}^{D} f_j(x_{t-1,j})$$

where, g is the logistic link function that adapts EBM to the classification setting.

Norti et al. (2019) state that EBM has several advantages over traditional GAM models. EBM learns each f_j function using modern machine learning techniques, such as bagging (Breiman, 1996) and gradient boosting (Friedman, 2001). The model's boosting algorithm cycles through the features in an effective way that mitigates the effects of co-linearity and ensures that feature order does not matter. As a result, EBM can optimally learn the best covariate function f_j. Moreover, EBM's training algorithm can detect and include only those covariate interactions (i.e., $\sum f_{i,j}(x_{t-1,i}, x_{t-1,j})$ that can benefit the model's performance and avoid including non-informative covariate interactions. For additional details regarding the algorithmic procedure and the selection of the most informative pairwise covariate interactions, we direct the reader to Lu et al. (2012), Lu et al. (2013), and Caruana et al. (2015). For the EBM's implementation, we use the Python library associated with the original paper[43] and keep the default parameter settings for the model. For our predictive task, the f_j is replaced by classification trees optimized by EBM's training algorithm.

3.B.4 Random Forest

The Random Forest (RF) machine learning technique was introduced by Breiman (2001) and is considered a type of bootstrap aggregation based on a subset of the input covariates that was

[42] The Python library is available at the following web-link: https://github.com/zzzace2000/nodegam.
[43] The Python library is available at the following web-link: https://github.com/interpretml/interpret.

randomly drown (Iworiso & Vrontos, 2020). In more detail, Fischer and Krauss (2018) note that an RF model consists of multiple classification trees built on different bootstrapped training data samples. To build each classification tree, a different random sample of the covariates is drawn from the full covariate set. Moreover, the large number of classification trees forming the RF model construct a voting ensemble that generates the predictions. Specifically, Iworiso and Vrontos (2020) note that this voting ensemble assigns to the input D-dimensional vector of covariates the predicted class that attained the majority of votes:

$$\hat{\delta}_{RF}^{M} = Majority\ Vote\left\{\hat{\delta}_m(X_{t-1})\right\}_{m=1}^{M}$$

where, $\hat{\delta}_b(X_{t-1})$ is the class prediction for the m_{th} random forest classification tree.

In our implementation, we use the Gini index (Breiman et al., 1984) to determine the best split at each classification tree node. We follow Fischer and Krauss (2018) and set the number of trees to 1000, the maximum depth to 20, and the size of the randomly drawn covariate set to the square root of the number of covariates.

3.C. Predictive accuracy metrics and notation

In this section we present the accuracy metrics we use in our study to assess the models' classification accuracy. We present the required notation and terminology, and subsequently describe the formulas used to calculate the accuracy metrics.

3.C.1 Preliminary notation and terminology

Positive class (for positive excess returns): $y_t = 1$

Negative class (for negative excess returns): $y_t = 0$

n: Total number of observations (i.e., months) in the forecast series

P: The number of instances in the positive class

N: The number of instances in the positive class

TP: A true positive is an outcome where the model correctly predicts the positive class

TN: A true negative is an outcome where the model correctly predicts the negative class

FP: A false positive is an outcome where the model incorrectly predicts the positive class

FN: A false negative is an outcome where the model incorrectly predicts the negative class

C.2 Accuracy metrics

Accuracy[44]: $\boldsymbol{ACC} = \frac{TP+TN}{n}$

[44] The so-called directional accuracy in Finance literature, or binary accuracy in machine learning literature.

Balanced Accuracy (average per-class accuracy): $\boldsymbol{BA} = \frac{1}{2}\left(\frac{TP}{P} + \frac{TN}{N}\right)$

Logistic loss:

$$\boldsymbol{L_{log}}\left(y_t, \overset{\text{TabNet}}{\Pr}(y_t = 1 \,|\mathrm{X}_{t-1})\right) = -\boldsymbol{y}(\boldsymbol{log}\left(\overset{\text{TabNet}}{\Pr}(y_t = 1 \,|\mathrm{X}_{t-1})\right) + (1-y)\log\left(1 - \overset{\text{TabNet}}{\Pr}(y_t = 1 \,|\mathrm{X}_{t-1})\right))$$

Brier Score[45]: $\boldsymbol{BS} = \frac{\mathbf{1}}{\boldsymbol{n}}(\sum_{\boldsymbol{t=1}}^{\boldsymbol{n}} \overset{\text{TabNet}}{\Pr}(y_t = 1 \,|\mathrm{X}_{t-1}) - \boldsymbol{y_t})$

3.C.3 Area under the Receiver Operating Characteristic Curve (ROC AUC): Graph and AUC statistic

3.C.3.1 Preliminary terminology

$\boldsymbol{TPR}$**: True Positive Rate:** The true positive rate gives the proportion of correct predictions in predictions of positive class : $\boldsymbol{TPR} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FN}}$

$\boldsymbol{FPR}$**: False Positive Rate:** The false positive rate gives the proportion of incorrect predictions in predictions of positive class : $\boldsymbol{FPR} = \frac{\boldsymbol{FP}}{\boldsymbol{FP+TN}}$

$\boldsymbol{TNR}$:**True Negative Rate:** The true negative rate gives the proportion of correct predictions in predictions of negative class : $\boldsymbol{TNR} = \frac{\boldsymbol{TN}}{\boldsymbol{TN+FP}}$

$\boldsymbol{FNR}$**: False Negative Rate:** The false negative rate gives the proportion of incorrect predictions in predictions of negative class: $\boldsymbol{FNR} = \frac{\boldsymbol{FN}}{\boldsymbol{FN+TP}}$

Table 3.C.1. The elements of a confusion matrix.

The confusion matrix summarizes the classification accuracy of model on a dataset by displaying the number of accurate and inaccurate instances given the generated predictions.

		Actual values	
		Positive class (1)	Negative class (0)
Predicted values	Positive class (1)	**TP**	**FP**
	Negative class (0)	**FN**	**TN**

[45] Gunnarsson et al. (2021) note that the brier score assesses the directional accuracy by computing the mean-squared error between the predicted probability of the positive class and the binary response variable.

3.C.3.2 The Receiver Operating Characteristic (ROC) Curve

Japkowicz and Shah (2011) provide an extensive description of the ROC graphs, and how they are used for the comparative evaluation of classification predictive models. The authors describe the ROC curve as the plot in which the horizontal axis denotes the false-positive rate FPR and the vertical axis denotes the true-positive rate TPR of a classification model. Since it holds: $\mathbf{0} \leq \boldsymbol{TPR} \leq \mathbf{1}$ and $\mathbf{0} \leq \boldsymbol{FPR} \leq \mathbf{1}$, it follows that the defined ROC space of the graph is unit square. The point of origin (0,0) refers to a trivial classification model that predicts only for the negative class (here, would be $y_t = 0$, i.e., the class associated with negative industry excess return), and therefore TPR and FPR are equal to zero. Naturally, the point (1,1) corresponds to a trivial classification model that would constantly predict the positive class and therefore $TPR = 1$, and $FPR = 1$. The diagonal line connecting the aforementioned points (0,0) and (1,1) corresponds to a classification model which achieves $TPR = FPR$, and therefore has the property of a random classifier assigning binary predictions randomly. Classification models above the diagonal line are better than a random classifier, whereas models below the diagonal line perform worse than a random classifier. Finally, the other two extreme graph values are (1, 0) and (0, 1). The (1, 0) point corresponds to a classifier that would make all the predictions wrong. The (1, 0) point corresponds to a flawless classifier that would correctly predict all cases of the positive class and make no mistakes on the negative class.

For every classifier, a decision rule/decision threshold identifies the value region corresponding to each of the discrete classes the model predicts. Japkowicz and Shah (2011) indicate that a so-called "operating point" in the ROC space (i.e., a pair of coordinates in the ROC graph) corresponds to a particular decision threshold for a classifier. Each operating point reveals the achieved TPR and FPR pair by the classifier, and is associated with a confusion matrix. As a result, an ROC curve is a collection of various confusion matrices over different and varying decision thresholds for a classification model. Exploring different decision thresholds (i.e., any value in the $\in [0,1]$) can give rise to a series of observed TPR and FPR pairs, that are depicted by a given coordinate in the ROC graph. Connecting all these [TPR, FPR] pairs creates the so-called ROC curve. If the ROC curve of a classification model is oriented towards the (0,1) coordinate, then this classifier outperforms another classifier with an ROC curve that is "nested" inside.

Finally, the Area Under the Curve (AUC) is a frequently used statistic for classification problems that quantifies the ROC analysis via a single value estimated by calculating the area covered by a classification model's ROC curve. By construction, a higher AUC statistic is preferable and indicates a better classifier compared to a lower AUC statistic achieved by a different classifier. The AUC is

bounded on $[0,1]$. An AUC of 1 indicates the perfect classification model, whereas an AUC of 0.5 corresponds to a random classifier.

3.D. Statistical tests

In this section we provide detailed descriptions of the statistical tests we used to assess the models' classification accuracy and excess profitability.

3.D.1. McNemar's test

The McNemar's test (McNemar, 1947; Edwards, 1948) conducts pairwise comparison between two machine learning classification models. Given the OOS predictions of two machine learning models, we construct the so-called contingency table, as shown in Table 3.D.1. The contingency table holds important information that can shed light on the two models' classification performance and provide.

Table 3.D.1. McNemar's test contingency table.

A 2x2 table, the contingency table, is used to compare two different models under the McNemar's test. The table holds information regarding the two models' classification accuracy and can provide insights with respect to their performance. n_{00}: Number of cases the two models misclassified y_t; n_{01}: Number of cases the first model misclassified y_t, but not the second model; n_{10}: Number of cases the second model misclassified y_t, but not the first model; n_{11}: Number of cases y_t was misclassified by neither the first nor the second model.

n_{00}	n_{01}
n_{10}	n_{11}

Note: $n_{00} + n_{01} + n_{10} + n_{11}$ *= the size of the OOS dataset*

Under the null hypothesis of the test, the first and the second model have the same error rate. The McNemar's test is a $\chi^2 - test$ that conducts a comparison between the number of counts expected given the null hypothesis to the observed counts (Dietterich, 1998). Under the null hypothesis, the expected counts are displayed on Table 3.D.1. The test statistic used to perform the test follows the χ^2 distribution with 1 degree of freedom. In our study, we utilize the correction proposed by Edwards (1948), and the final form of the test statistic is displayed in Table 3.D.2.

Table 3.D.2. McNemar's test null hypothesis and the test statistic.

The table presents the expected number of counts under McNemar's test null hypothesis and the calculation of the test statistic. The test statistic test follows the χ^2 distribution with 1 degree of freedom.

The expected number of counts under H_0	
n_{00}	$(n_{01} + n_{10})/2$
$(n_{01} + n_{10})/2$	n_{11}
McNemar's test statistic	$\frac{(\lvert n_{01} - n_{10} \rvert - 1)^2}{n_{01} + n_{10}}$

3.D.2. Cochran's test

Cochran's test (Cochran, 1950) can be used to evaluate multiple classifiers and be regarded as a generalization of McNemar's test. For a set $\{ML_1, ML_2, \dots, ML_L\}$ of classification models with size L, Cochran's Q statistic follows the χ^2 distribution with $L-1$ degrees of freedom. Under the null hypothesis, there is no difference between the classification accuracies p_{acc} of the machine learning models (i.e., $H_0: p_1 = p_2 = \cdots = p_L$). Cochran's Q statistic is defined as follows:

$$Q_c = (L-1)\frac{L\sum_{h=1}^{L} G_h^2 - V^2}{LV - \sum_{j=1}^{T} L_j^2}$$

where, G_h is the number of cases out of the total T cases that are correctly classified by the $ML_l = 1, 2, \dots, L$, L_j is the number of classification models out of L that correctly classified the covariate vector $z_j \in Z_T$, where Z_T is the dataset of covariates for the OOS period, V is the number of correct predictions among the L classification models, it holds that $V = \sum_{h=1}^{L} G_h = \sum_{j=1}^{T} L_j$.

3.D.3. F-test

The test developed by Snedecor and Cochran (1989) can assist in evaluating multiple machine learning classification models. The F-test assesses the null hypothesis that there is no difference in the classification accuracies of a set of L machine learning models (i.e., $H_0: p_1 = p_2 = \cdots = p_L$). For a set of classifiers $\{ML_1, ML_2, \dots, ML_L\}$, if the models do not perform differently on the OOS dataset, then the test statistic (i.e., F-statistic) follows the F distribution with $(L-1)$ and $(L-1)\,\mathrm{x}\,T$ degrees of freedom. As noted by Looney (1988), the F statistic can be calculated by the following steps:

We start by defining $ACCavg$ as the average of the accuracies of the different machine learning classification models, and then calculate the sum of squares of the classifiers (SSA).

$$ACCavg = \frac{1}{L}\sum_{j=1}^{L} ACC_j$$

$$SSA = T\sum_{h=1}^{L} G_h^2 - T\ L\ ACCavg$$

where G_h is the proportion of the T instances classified correctly by classifier h.

The sum of squares of the classified covariate vectors is calculated as:

$$SSB = \frac{1}{L}\sum_{j=1}^{T} L_j^2 - \ T\ L\ ACCavg^2$$

where, L_j is the number of classification models that correctly classified the covariate vector $z_j \in Z_T$

We proceed with the estimation of the total sum of squares and the following statistics before arriving at the F-statistic formula:

$$SST = T\ L\ \ ACCavg(1 - \ ACCavg)$$

$$SSAB = \ SST - SSA - SSB$$

$$MeanSA = \frac{SSA}{T - 1}$$

$$MeanSAB = \frac{SSAB}{(M - 1)(n - 1)}$$

$$F - statistic = \frac{MeanSA}{MeanSAB}$$

3.D.4. Pesaran-Timmermann directional accuracy test

The Pesaran and Timmermann (1992) test that can be used to evaluate the OOS classification accuracy of machine learning models. The null hypothesis is that a model's predictions and the target variable y_t are independently distributed (see also Fischer & Krauss, 2018). We adopt the test description to the notation we introduced in the methodology section 4.1. The Pesaran and Timmermann test statistic (PT-statistic) can be calculated by first defining $\widetilde{A_T}$ and $\widetilde{B_T}$ as:

$$\widetilde{A_T} = \frac{1}{T}\sum_{t=1}^{T} sign(\hat{y}_t)sign(y_t)$$

where, $\hat{y}_t$ is the model prediction, T is the size of the OOS dataset, and sign(.) takes the value of +1 when the argument is positive and -1 otherwise.

$$\widetilde{B_T} = (\frac{1}{T}\sum_{t=1}^{T} sign(\hat{y}_t))\,(\frac{1}{T}\sum_{t=1}^{T} sign(y_t))$$

The PT-statistic is given by:

$$PT - stastistic\ \equiv \frac{\widetilde{A_T} - \widetilde{B_T}}{\sqrt{\widehat{V_{PT}}}} \xrightarrow{d} N(0,1)$$

where,

$$\hat{V}_{PT} = \frac{16(T - 1)}{T^2}\hat{p}_{\hat{y}}\left(1 - \hat{p}_{\hat{y}}\right)\hat{p}_y\left(1 - \hat{p}_y\right)$$

$$\hat{p}_y = \frac{1}{2}\left(1 + \frac{1}{T}\sum_{t=1}^{T} sign(y_t)\right)$$

3.D.5. Anatolyev-Gerko excess profitability test

The Anatolyev and Gerko (2005) test assesses a trading strategy that issues a buy signal if the prediction for the next period's return (here, excess return) is positive and a short signal otherwise. The average return of the aforementioned strategy is compared with a benchmark strategy that forms buy/sell signals randomly to test for the significance of return predictability and excess profit (Liu et al., 2019). We adopt the test description to the notation we introduced in the methodology section 4.1. The one-period return of the trading strategy is given by:

$$R_t = sign(\widehat{y_t})r_t$$

where, $\widehat{y_t}$ is the model prediction, and sign(.) takes the value of +1 when the argument is positive and -1 otherwise.

The idea behind the test statistic (AG-statistic) is the following:

$$E[\text{sign}(\hat{y}_t)]\mathbb{E}[r_t] = E[\text{sign}(\hat{y}_t)E[r_t]]$$

$$\overset{H_0}{=} E[\text{sign}(\hat{y}_t)E[r_t \mid X_{t-1}]]$$

$$= E\big[E[\text{sign}(\hat{y}_t)r_t \mid X_{t-1}]\big]$$

$$= E[\text{sign}\,(\hat{y}_t)r_t] = E[R_t]$$

Given the above equations, and under the null hypothesis, the average return of the trading strategy formed based on the sign of the return predictions should statistically be equal to a benchmark strategy that issues buy/sell signals at random with probabilities corresponding to the proportion of “buys” and “sells” implied ex post by the trading strategy (Anatolyev & Gerko, 2005). The test statistic and the corresponding estimators of the above expectations are calculated using the sample data. The estimator of $E[R_t]$ and the estimator of $E[\text{sign}(\hat{y}_t)]\mathbb{E}[r_t]$ is given by:

$$A_T = \frac{1}{T}\sum_{t=1}^{T} R_t$$

$$B_T = \left(\frac{1}{T}\sum_{t=1}^{T} \text{sign}(\hat{y}_t)\right)\left(\frac{1}{T}\sum_{t=1}^{T} r_t\right)$$

Under the null hypothesis, the variance of $A_T - B_T$ is given by the following equation:

$$\text{Var}[A_T - B_T] = \frac{4(T-1)}{T^2}p_{\hat{y}}\big(1 - p_{\hat{y}}\big)\,\text{Var}[r_t]$$

where, $p_{\hat{y}} = \Pr\,(\text{sign}(\hat{y}_t) = 1)$

The estimator for $\mathrm{Var}\,[A_T - B_T]$ and $p_{\hat{y}}$ are calculated by the sample data. Finally, we provide the estimation of Hausman-type AG-statistic and its distribution (i.e., asymptotically normal).

$$\hat{V}_{AG} = \frac{4}{T^2}\hat{p}_{\hat{y}}\left(1 - \hat{p}_{\hat{y}}\right)\sum_{t=1}^{T}(r_t - \bar{r})^2$$

$$\hat{p}_{\hat{y}} = \frac{1}{2}\left(1 + \frac{1}{T}\sum_{t=1}^{T}\mathrm{sign}(\hat{y}_t)\right)$$

$$AG - statistic \equiv \frac{A_T - B_T}{\sqrt{\hat{V}_{AG}}} \xrightarrow{d} N(0,1)$$

CONCLUSION

This thesis explores the utility of state-of-the-art machine learning techniques in financial applications. The research does not serve the purpose of estimating sophisticated models only for estimation purposes. Instead, it provides an unbiased and robust evaluation of interpretable machine learning in predicting financial returns and constructing investment portfolios. Interpretability is of paramount importance in the financial domain. Portfolio managers and investors can benefit from sophisticated and high-performing models under the condition that the model generating the predictions is transparent. Critical applications such as financial returns forecasting, portfolio management, and asset allocation require understanding which variables drive the model's predictions. For this reason, in this thesis, we refrain from using neural networks as "black-boxes" and adopt interpretable techniques that can shed light on the decision-making process of the models. Given the flexibility of machine learning methods, we also highlight how these methods can be useful not only in the financial forecasting setting but also in filling missing values in highly sparse datasets. Effectively imputing missing values is often treated as a minor task, receiving limited attention in the financial literature despite its importance. It is an undeniable fact that almost all datasets present some form of missingness. Our research provides evidence that machine learning can outperform other commonly used methods to fill in missing values and generate imputations that enhance the predictive accuracy of forecasting models.

In the first chapter, we apply an interpretable machine learning framework, the LassoNet, to forecast U.S. industry portfolio returns over the 2010–2019 period based on a data-rich environment of 88 predictors. We compare the performance of LassoNet with several linear and non-linear models. Specifically, we include linear regression, Group Lasso, Elastic-Net, XGBoost, and neural networks in our benchmark model set. To quantify covariate importance, we leverage SAGE, the global importance interpretability method, on the covariate set selected by LassoNet. We develop a trading application to evaluate the economic impact of LassoNet's forecasts. Our findings reveal that state-of-the-art interpretable deep learning specifications can capture non-linear patterns and interactions among our predictors, which leads to more accurate industry return forecasts than other linear and non-linear machine learning techniques. LassoNet achieves significantly smaller forecasting errors across most industries examined than other econometric and machine learning models. Several statistical tests authenticate these results. Therefore, we prove that integrating covariate selection into a deep learning model specification, as performed by LassoNet, is an effective combination for achieving superior forecasting accuracy. Second, applying the SAGE method shows that valuation ratios and

individual and cross-industry lagged industry returns generate the highest SAGE values. The aforementioned covariate categories are critical determinants when predicting industry portfolio returns.

Our empirical evidence complements the relevant findings of studies using linear asset pricing for return predictability, mainly revealing profitability and liquidity ratios as essential factors of stock returns. We prove the economic significance of LassoNet forecasts in constructing more profitable spread portfolios than buy-and-hold strategies on market indices. The constructed portfolios based on LassoNet's forecasts achieve the best performance metrics and positive and statistically significant multifactor alphas. Notably, the trading application's results are robust when considering transaction costs. Overall, we expand the previous studies reporting the ability of Lasso methods to accurately predict industry portfolios' returns by successfully applying a Lasso-based deep learning method for non-linear environments and constructing a large-scale set of predictors for the same task. For all the above reasons, our findings can greatly interest academics and practitioners in portfolio and asset management industries.

In the second chapter, we adopt a bidirectional recurrent imputation neural network for imputing hedge fund returns and their corresponding predictors. The model handles missing values by capturing time series and cross-sectional information without making assumptions about the hedge fund data's structure and distribution. Hence, we can effectively recover funds' missing entries. To establish the imputation fidelity of BRITS, we construct a simulation study in which we artificially drop a random 10% and 20% of the observed values and compare the proposed method with a battery of benchmarks used in the financial literature for data imputation. Additionally, we examine the importance of BRITS in generating fully recovered predictor datasets for forecasting hedge fund returns. We train well-established machine learning models on the imputed returns and predictors datasets for our forecasting experiment. The employed machine learning models belong to three classes: gradient-boosted trees, neural networks, and linear penalized regressions. We also construct a trading application using the forecasts to construct equally weighted decile portfolios and focus exclusively on the top decile to form the long positions.

Our findings indicate that BRITS outperforms all benchmarks regarding imputation error and provides a recovered set of predictors, which can generate accurate forecasts when fed to linear and machine learning techniques. The simulation study validates that our framework achieves superior results in imputing hedge fund returns and predictors' missing entries. The forecasting task's results confirm that machine learning models estimated on the BRITS imputed datasets show higher forecasting accuracy than those imputed using the cross-sectional mean method, the standard method

employed in the financial literature to fill missing values. Therefore, BRITS can provide an information advantage, boosting the OOS performance of forecasting models. We use the SHAP interpretability method to uncover predictor importance and provide evidence that interactions of fund-specific predictors with macroeconomic variables dominate the standalone fund-specific predictors. Macroeconomic interactions with fund-specific characteristics are among the ten most important predictors most of the time and across all forecasting models. The trading application's results show that the equally-weighted decile portfolios formed based on the forecasts of machine learning models trained on BRITS imputed data achieve higher annualized returns, Sharpe ratios, and alpha values. These findings outline that the information advantage provided by BRITS imputations leads to higher OOS forecasting accuracy and superior profitability.

In the third chapter, we use a state-of-the-art and "white-box" deep learning method to predict the directional movements of industries' excess returns. We employ the TabNet model, a deep learning architecture that can provide the benefits of neural network models in terms of higher predictive accuracy without sacrificing interpretability. A key component of TabNet is that it is interpretable by construction; therefore, no external algorithm is required to derive covariate importance. To evaluate the OOS predictive ability of TabNet, we employ a benchmark model set consisting of linear and machine learning models and use multiple performance metrics and statistical tests. To assess the economic significance of TabNet's predictions, we construct an equally weighted trading strategy and compare it to a benchmark trading strategy based on the logistic regression forecasts and buy-and-hold strategies on three market indices.

Our results show that TabNet achieves superior predictive accuracy and generates profitable predictions. The forecasting application's results validate that TabNet has the highest performance considering all employed performance metrics and statistical tests. The second best-performing model is the logistic regression model, and the worst-performing is the random forest. We derive covariate importance directly from TabNet's architecture and conclude that the most significant categories are the valuation ratios, lagged returns, other-industry lagged returns, and financial soundness. The three top positions with the most significant covariates belong exclusively to the valuation ratios category. In contrast, the fourth and fifth most informative covariates belong to the other-industry lagged return and lagged return categories. The reported significance of the other-industry lagged return category confirms the presence of strong economic links and interdependencies among the industries and the diffusion of information across the economic sectors. The significance of the 12-month lagged return indicates the seasonality effects. It validates that past excess returns can hold important information when predicting the directional movements of future excess returns. Regarding the trading application results, the equal-weighted trading strategy constructed using TabNet's predictions is the most

profitable. It attains positive and statistically significant alphas against the four-factor and five-factor models. TabNet's profitability persists even in the presence of transaction costs.

Our findings should persuade portfolio managers to incorporate state-of-the-art deep learning methods into their predictive toolbox. Interpretable neural network architectures such as those explored in this thesis exhibit superior predictive ability without sacrificing performance for interpretability and vice-versa. In addition, deep learning methods such as BRITS can effectively impute missing values in financial datasets and provide crucial information advantages to models trained on fully recovered datasets. As seminal factor models indicate, predicting the directional movements of industries' returns via a deep learning framework can provide statistically significantly abnormal returns. Directional forecasts can be an alternative to the standard objective in financial research, which is to predict the level of returns. Therefore, financial practitioners are advised also to include classification methods when conducting their quantitative analyses.

The adoption of neural network models is not a topic restricted to portfolio managers and financial practitioners, since central banks and policymakers can also benefit from such powerful predictive techniques. Machine learning techniques can assist central banks in improving their operations and decision-making processes in multiple ways. Some key applications include economic forecasting, portfolio management, fraud detection, and risk assessment. First, machine learning models can be used in the time-series forecasting domain to process large volumes of data and produce highly accurate forecasts of economic variables (e.g., GDP, inflation, employment figures) and financial market returns (e.g., stock market and exchange rates). Second, these models can help central banks better predict economic conditions and adjust their monetary policies accordingly. Central banks can also use machine learning models to manage their assets and reserves by leveraging these cutting-edge techniques in investment decisions and optimizing asset allocation strategies.

Another area where machine learning can make a difference is in monitoring financial institutions and payment systems. Fraud detection and anomaly detection are becoming increasingly critical for the financial sector. Machine learning models can process transaction data and identify potentially fraudulent or unusual activity, contributing to more effective oversight. Research in this area could explore which specific algorithms and models are most effective at anomaly detection, particularly in the context of central banks' regulatory responsibilities. Furthermore, machine learning models can also be applied to risk assessment, analyzing market data, balance sheets, and macroeconomic indicators to identify vulnerabilities and systemic risks.

Future research can address the limitations within the scope of the thesis. First, the focus of this study was primarily on monthly data from U.S. industry portfolios and hedge funds, which may limit

the generalizability of the findings to other markets or types of data. Expanding this research to daily or intra-day financial data could significantly extend the models' applications, especially in high-frequency trading environments. Second, despite the significant predictive power of LassoNet and the insights provided by the SAGE method, there are limitations inherent in using Shapley values (as applied via SAGE) to interpret variable importance. While Shapley values offer a flexible, data-driven measure of covariate importance, their lack of direct linkage to traditional financial theories can pose challenges. For instance, traditional asset pricing models, like the Fama-French factors, offer an intuitive, economically grounded explanation of returns based on theoretically established factors such as market risk, size, and value. Shapley values, on the other hand, operate purely from a data-driven perspective, which can limit their interpretability in financial contexts where clear economic rationale is often sought. Future research could explore bridging this gap by combining Shapley-based interpretability with more traditional, theory-driven approaches to asset pricing.

Another limitation of this thesis is the lack of integration of unstructured textual data into the models, such as financial news, analyst reports, and social media. Textual data has become a crucial source of insights into market sentiment and investor behavior, which can provide early signals of market movements that may not be reflected in traditional financial metrics. Future research could incorporate natural language processing (NLP) techniques, such as sentiment analysis and topic modeling, to analyze and quantify market sentiment from these sources. More accurate forecasting models could be developed by combining traditional financial data with real-time sentiment analysis from news and social media. This approach could improve models' ability to predict market movements and detect early signals of major financial events, offering a more comprehensive understanding of market dynamics. In conclusion, while this thesis provides meaningful contributions to the field of financial forecasting through the use of interpretable machine learning models, it also highlights the potential for future research to explore more complex models, additional datasets, and the integration of new data types. The above directions for future research could enhance the accuracy, applicability, and interpretability of machine learning in financial markets, further advancing both academic understanding and practical implementation in the field.

BIBLIOGRAPHY

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., ... & Zheng, X. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467.

Anatolyev, S., & Gerko, A. (2005). A trading approach to testing for predictability. *Journal of Business & Economic Statistics*, *23*(4), 455-461.

Aragon, G. O. (2007). Share restrictions and asset pricing: Evidence from the hedge fund industry. *Journal of financial economics*, *83*(1), 33-58.

Arik, S. Ö., & Pfister, T. (2021). TabNet: Attentive Interpretable Tabular Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, *35*(8), 6679-6687.

Asness, C. S., Moskowitz, T. J., & Pedersen, L. H. (2013). Value and momentum everywhere. *The journal of finance*, *68*(3), 929-985.

Avramov, D. (2004). Stock Return Predictability and Asset Pricing Models. *Review of Financial Studies,* 17 (3), 699–738.

Avramov, D., Cheng, S., & Metzker, L. (2023). Machine learning vs. economic restrictions: Evidence from stock return predictability. *Management Science*, *69*(5), 2587-2619.

Bali, T. G., Brown, S. J., & Caglayan, M. O. (2012). Systematic risk and the cross section of hedge fund returns. *Journal of Financial Economics*, *106*(1), 114-131.

Bali, T. G., Brown, S. J., & Caglayan, M. O. (2019). Upside potential of hedge funds as a predictor of future performance. *Journal of Banking & Finance*, *98*, 212-229.

Bali, T. G., Brown, S. J., Caglayan, M. O., & Celiker, U. (2021). Does Industry Timing Ability of Hedge Funds Predict Their Future Performance, Survival, and Fund Flows?. *Journal of Financial and Quantitative Analysis*, *56*(6), 2136-2169.

Ball, R., Gerakos, J., Linnainmaa, J. T., and Nikolaev, V. (2016). Accruals, cash flows, and operating profitability in the cross-section of stock returns. *Journal of Financial Economics*, 121(1), 28-45.

Balvers, R. J., & Wu, Y. (2006). Momentum and mean reversion across national equity markets. Journal of Empirical Finance, 13(1), 24-48.

Bank of England (2022, October 11). "*Machine learning in UK financial services*". Bank of England.

Bartram, S. M., Branke, J., & Motahari, M. (2020). Artificial intelligence in asset management. Financial Analysts Journal, 76(1), 5–30.

https://www.bankofengland.co.uk/report/2022/machine-learning-in-uk-financial-services.

Becker, J., & Leschinski, C. (2018). *Directional predictability of daily stock returns* (No. 624). Hannover Economic Papers (HEP).

Beckmeyer, H., & Wiedemann, T. (2022). Recovering Missing Firm Characteristics with Attention-Based Machine Learning. *Available at SSRN*.

Beller, K. R., Kling, J. L., & Levinson, M. J. (1998). Are industry stock returns predictable? *Financial Analysts Journal*, *54*(5), 42–57.

Bianchi, D., & McAlinn, K. (2021). Divide and Conquer: Financial Ratios and Industry Returns Predictability. *SSRN Electronic Journal*.

Brigo, D., Huang, X., Pallavicini, A., & De Oscariz-Borde, H.,S. (2021). Interpretability in deep learning for finance: a case study for the Heston model. Working paper, SSRN.

Bonferroni, C. (1936). Teoria statistica delle classi e calcolo delle probabilita. *Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commericiali di Firenze*, *8*, 3-62.

Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). Classification and Regression Trees. Boca Raton, FL: CRC Press.

Breiman, L. (1996). Bagging predictors. *Machine learning*, *24*, 123-140.

Breiman, L. (2001). Random forests. *Machine learning*, *45*, 5-32.

Bryzgalova, S., Lerner, S., Lettau, M., & Pelger, M. (2022). Missing Financial Data. *Available at SSRN*.

Bussmann, N., Giudici, P., Marinelli, D., & Papenbrock, J. (2021). Explainable machine learning in credit risk management. Computational Economics, 57, 203–216.

Cai, J. F., Candès, E. J., & Shen, Z. (2010). A singular value thresholding algorithm for matrix completion. *SIAM Journal on optimization*, *20*(4), 1956-1982.

Cao, W., Zhou, H., Wang, D., Li, Y., Li, J., & Li, L. (2018). BRITS: Bidirectional recurrent imputation for time series. *Advances in Neural Information Processing Systems*, *2018-Decem*(NeurIPS), 6775–6785.

Campbell, J. Y., & Shiller, R. J. (1988). Stock prices, earnings, and expected dividends. *The Journal of Finance*, *43*(3), 661-676.

Campbell, J. Y., & Yogo, M. (2006). Efficient tests of stock return predictability. *Journal of Financial Economics*, *81*(1), 27–60.

Campbell, J. Y., & Thompson, S. B. (2008). Predicting excess stock returns out of sample: Can anything beat the historical average? *The Review of Financial Studies*, *21*(4), 1509–1531.

Carhart, M. M. (1997). On Persistence in Mutual Fund Performance. *The Journal of Finance*, *52*(1), 57–82.

Caruana, R., Lou, Y., Gehrke, J., Koch, P., Sturm, M., & Elhadad, N. (2015, August). Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1721-1730).

Cochran, W. G. (1950). The comparison of percentages in matched samples. *Biometrika*, *37*(3/4), 256-266.

Chang, C. H., Tan, S., Lengerich, B., Goldenberg, A., & Caruana, R. (2021, August). How interpretable and trustworthy are GAMs?. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining* (pp. 95-105).

Che, Z., Purushotham, S., Cho, K., Sontag, D., & Liu, Y. (2018). Recurrent Neural Networks for Multivariate Time Series with Missing Values. *Scientific Reports*, *8*(1), 1–12. https://doi.org/10.1038/s41598-018-24271-9.

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree-boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*.

Chen, Y., Cliff, M., & Zhao, H. (2017). Hedge funds: The good, the bad, and the lucky. *Journal of Financial and Quantitative Analysis*, *52*(3), 1081-1109.

Chen, Y., Han, B., & Pan, J. (2021). Sentiment trading and hedge fund returns. *The Journal of Finance*, *76*(4), 2001-2033.

Chen, L., Pelger, M., & Zhu, J. (2021). Deep Learning in Asset Pricing. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.3350138

Chen, L., Pelger, M., Zhu, J., (2023). Deep Learning in Asset Pricing. *Management Science*, articles in advance.

Chollet, F. (2016). Keras. https://github.com/fchollet/keras.

Christoffersen, P. F., & Diebold, F. X. (2006). Financial asset returns, direction-of-change forecasting, and volatility dynamics. *Management Science*, *52*(8), 1273-1287.

Civieta, Á. M., Aguilera-Morillo, M. C., & Lillo, R. E. (2021). Asgl: A Python Package for Penalized Linear and Quantile Regression. arXiv preprint arXiv: http://arxiv.org/abs/2111.00472.

Cohen, L. & Frazzini, A. (2008). Economic links and predictable returns. *Journal of Finance,* 63, 1977–2011.

Covert, I., Lundberg, S. M., & Lee, S. I. (2020). Understanding global feature contributions with additive importance measures. *Advances in Neural Information Processing Systems*, *33*, 17212-17223.

Covert, I., & Lee, S. I. (2021, March). Improving kernelshap: Practical shapley value estimation using linear regression. In *International Conference on Artificial Intelligence and Statistics* (pp. 3457-3465). PMLR.

Cui, Z., Ke, R., Pu, Z., & Wang, Y. (2018). *Deep Bidirectional and Unidirectional LSTM Recurrent Neural Network for Network-wide Traffic Speed Prediction*. 1–11. http://arxiv.org/abs/1801.02143

Dangl, T., & Halling, M. (2012). Predictive regressions with time-varying coefficients. *Journal of Financial Economics*, *106*(1), 157-181.

Dauphin, Y. N., Fan, A., Auli, M., & Grangier, D. (2016). Language modeling with gated convolutional networks. arXiv. *Computation and Language*.

Delen, D. , Sharda, R. , & Kumar, P. (2007). Movie forecast guru: A web-based dss for hollywood managers. *Decision Support Systems*, 43 (4), 1151–1170.

DeMiguel, V., Gil-Bazo, J., J. Nogales, F., & Santos, A.A.P. (2023). Machine learning and fund characteristics help to select mutual funds with positive alpha. *Journal of Financial Economics* 150 (3).

Diebold, F. X., & Mariano, R. S. (1995). Comparing Predictive Accuracy. *Journal of Business & Economic Statistics, 13* (3), 253–263.

Dietterich, T. G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural computation*, *10*(7), 1895-1923.

Dimopoulos, Y., P. Bourret, & S. Lek. (1995). Use of some sensitivity criteria for choosing networks with good generalization ability. *Neural Processing Letters* (2), 1–4.

Dong, X., & Li, Y.,. Rapach, D., E., & Guofu Zhou (2022). "Anomalies and the Expected Market Return,"

Journal of Finance, 77 (1), 639-681.

DreamQuark-ai. (2023). TabNet: Attentive interpretable tabular learning [Computer software]. GitHub. Retrieved from https://github.com/dreamquark-ai/tabnet.

Du, W., Côté, D., & Liu, Y. (2022). SAITS: Self-Attention-based Imputation for Time Series. *arXiv preprint arXiv:2202.08516*.

Dumitrescu, E., Hué, S., Hurlin, C., & Tokpavi, S. (2022). Machine learning for credit scoring: Improving logistic regression with non-linear decision-tree effects. *European Journal of Operational Research*, *297*(3), 1178-1192.

Dunis, C. L., Laws, J., & Sermpinis, G. (2011). Higher order and recurrent neural architectures for trading the EUR/USD exchange rate. *Quantitative Finance*, *11*(4), 615–629.

Edwards, A. L. (1948). Note on the "correction for continuity" in testing the significance of the difference between correlated proportions. *Psychometrika*, *13*(3), 185-187.

Fama, E. F. (1970). Efficient capital markets. *Journal of finance*, *25*(2), 383-417.

Fama, E. F., & French, K. R. (1988). Dividend yields and expected stock returns. *Journal of Financial Economics*, *22*(1), 3–25.

Fama, E. F., & French, K. R. (1993). Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, *33*(1), 3–56.

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

Fan, J., Ma, C., & Zhong, Y. (2021). A Selective Overview of Deep Learning. *Statistical Science*, *36*(2), 264–290.

Feng, G., He, J., & Polson, N. G. (2018). Deep learning for predicting asset returns. arXiv preprint arXiv:1804.09314.

Ferov, M., & Modrý, M. (2016). Enhancing lambdamart using oblivious trees. *arXiv preprint arXiv:1609.05610*.

Ferson, W. E. & Harvey, C. R. (1991). The Variation of Economic Risk Premiums. *Journal of Political Economy,* 99 (2), 385–415.

Ferson, W. E. & Korajczyk, R. A. (1995). Do arbitrage pricing models explain the predictability of stock returns? *Journal of Business*, pages 309-349.

Ferson, W. E. & Harvey, C. R. (1999). Conditioning Variables and the Cross Section of Stock Returns. *Journal of Finance*, 54 (4), 1325–1360.

Filippou, I., Rapach, D., Taylor, M. P., & Zhou, G. (2022). Out-of-Sample Exchange Rate Prediction: A Machine Learning Perspective. *Available at SSRN*.

Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, *270*(2), 654–669.

Freyberger, J., Höppner, B., Neuhierl, A., & Weber, M. (2021). Missing Data in Asset Pricing Panels. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.3932438

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, 1189–1232.

Friedman, J., Hastie, T., & Tibshirani, R. (2010). A note on the group lasso and a sparse group lasso. *arXiv preprint arXiv:1001.0736*.

Fuchs, R., Burget, R., & Jurica, P. (2021). Matrix completion benchmark and filling package in R. Journal of Machine Learning Research.

Fung, W., & Hsieh, D. A. (2000). Performance characteristics of hedge funds and commodity funds: Natural vs. spurious biases. *Journal of Financial and Quantitative analysis*, *35*(3), 291-307.

Fung, W., & Hsieh, D. A. (2004). Hedge fund benchmarks: A risk-based approach. *Financial Analysts Journal*, *60*(5), 65-80.

Gehring, J., Auli, M., Grangier, D., Yarats, D., & Dauphin, Y. N. (2017, July). Convolutional sequence to sequence learning. In *International conference on machine learning* (pp. 1243-1252). PMLR.

Giacomini, R., & White, H. (2006). Tests of conditional predictive ability. *Econometrica, 74*(6)*,* 1545-1578

Giacomini, R., & Rossi B. (2010). Forecast comparisons in unstable environments. *Journal of Applied Econometrics* 25, 595-620.

Giglio, S., Liao, Y., & Xiu, D. (2021). Thousands of alpha tests. *The Review of Financial Studies*, *34*(7), 3456-3496.

Granger, C. W. (1993). Strategies for modelling nonlinear time-series relationships. *Economic Record*, *69*(3), 233–238.

Green, J., Hand, J. R., & Zhang, X. F. (2017). The characteristics that provide independent information

about average US monthly stock returns. *The Review of Financial Studies*, *30*(12), 4389-4436.

Grønborg, S.,N., Lunde, A., Timmermann, A., & Wermers, R. (2021). Picking funds with confidence. Journal of Financial Economics, 139 (1) 1-28.

Gu, S., Kelly, B., & Xiu, D. (2020). Empirical Asset Pricing via Machine Learning. *Review of Financial Studies*, *33*(5), 2223–2273.

Gu, S., Kelly, B., & Xiu, D. (2021). Autoencoder asset pricing models. *Journal of Econometrics*, *222*(1), 429–450.

Gulin, A., Kuralenok, I., & Pavlov, D. (2011). Winning the transfer learning track of yahoo!'s learning to rank challenge with yetirank. In *Proceedings of the Learning to Rank Challenge* (pp. 63-76). PMLR.

Gunnarsson, B. R., Vanden Broucke, S., Baesens, B., Óskarsdóttir, M., & Lemahieu, W. (2021). Deep learning for credit scoring: Do or don't?. *European Journal of Operational Research*, *295*(1), 292-305.

Hansen, P. R. (2005). A Test for Superior Predictive Ability. *Journal of Business & Economic Statistics*, *23*(4), 365–380.

Haugen, R. A., & Baker, N. L. (1996). Commonality in the determinants of expected stock returns. *Journal of financial economics*, *41*(3), 401-439.

Hansen, P. R., Lunde, A., & Nason, J. M. (2011). The model confidence set. *Econometrica*, *79*(2), 453-497.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., ... & Oliphant, T. E. (2020). Array programming with NumPy. Nature, 585(7825), 357-362.

Harvey, C., R. & Liu, Y. (2015). Backtesting. *SSRN Working Paper.*

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

Heston, S. L., & Sadka, R. (2008). Seasonality in the cross-section of stock returns. *Journal of Financial Economics*, *87*(2), 418-445.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735-1780.

Hommel, G. (1988). A stagewise rejective multiple test procedure based on a modified Bonferroni

test. Biometrika, 75(2), 383-386.

Huck, N. (2019). Large data sets and machine learning: Applications to statistical arbitrage. *European Journal of Operational Research*, *278*(1), 330-342.

Iworiso, J., & Vrontos, S. (2020). On the directional predictability of equity premium using machine learning techniques. *Journal of Forecasting*, *39*(3), 449-469.

Jabeur, S. B., Mefteh-Wali, S., & Viviani, J. L. (2021). Forecasting gold price with the XGBoost algorithm and SHAP interaction values. *Annals of Operations Research*, 1-21.

Jacobs, H. (2015). What explains the dynamics of 100 anomalies?. *Journal of Banking & Finance*, *57*, 65-85.

Japkowicz, N., & Shah, M. (2011). *Evaluating learning algorithms: a classification perspective*. Cambridge University Press.

Jegadeesh, N. (1990). Evidence of predictable behavior of security returns. *The Journal of finance*, *45*(3), 881-898.

Jensen, M. C. (1978). Some anomalous evidence regarding market efficiency. *Journal of financial economics*, *6*(2/3), 95-101.

Karhunen, M. (2019). Algorithmic sign prediction and covariate selection across eleven international stock markets. *Expert Systems with Applications*, *115*, 256-263.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., ... & Liu, T. Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, *30*.

Keim, D. B., & Stambaugh, R. F. (1986). Predicting returns in the stock and bond markets. *Journal of financial Economics*, *17*(2), 357-390.

Kelly, B., & Jiang, H. (2014). Tail risk and asset prices. *The Review of Financial Studies*, *27*(10), 2841-2871.

Kim, A., Yang, Y., Lessmann, S., Ma, T., Sung, M., C., & Johnson J., E., V. (2020). Can deep learning predict risky retail investors? A case study in financial risk behaviour forecasting. *European Journal of Operational Research, 283 (1), 217-234.*

Kim, B., Khanna, R., & Koyejo, O. O. (2016). Examples are not enough, learn to criticize! criticism for interpretability. *In Advances in neural information processing systems*, pages 2280–2288.

Kingma, D. P., & Ba, J. L. (2017). Adam: A Method for Stochastic Optimization. arXiv preprint arXiv:

https://arxiv.org/abs/1412.6980.

Kosowski, R., Naik, N. Y., & Teo, M. (2007). Do hedge funds deliver alpha? A Bayesian and bootstrap analysis. *Journal of Financial Economics*, *84*(1), 229-264.

Kozak, S., Nagel, S., & Santosh, S. (2020). Shrinking the cross-section. *Journal of Financial Economics*, *135*(2), 271-292.

Krauss, C., Do, X. A., & Huck, N. (2017). Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500. *European Journal of Operational Research*, *259*(2), 689–702.

Leitch, G., & Tanner, J. E. (1991). Economic forecast evaluation: profits versus the conventional error measures. *The American Economic Review*, 580-590.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436-444.

Lemhadri, I., Ruan, F., Abraham, L., & Tibshirani, R. (2021). Lassonet: A neural network with feature sparsity. *Journal of Machine Learning Research*, *22*, 1–29.

Lessmann, S., Baesens, B., Seow, H. V., & Thomas, L. C. (2015). Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. *European Journal of Operational Research*, *247*(1), 124-136.

Leung, M. T., Daouk, H., & Chen, A. S. (2000). Forecasting stock indices: a comparison of classification and level estimation models. *International Journal of forecasting*, *16*(2), 173-190.

Lewellen, J., Nagel, S., & Shanken, J. (2010). A skeptical appraisal of asset pricing tests. Journal of Financial Economics, *96* (2), 175-194.

Looney, S. W. (1988). A statistical technique for comparing the accuracies of several classifiers. *Pattern Recognition Letters*, *8*(1), 5-9.

Lou, Y., Caruana, R., & Gehrke, J. (2012, August). Intelligible models for classification and regression. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 150-158).

Lou, Y., Caruana, R., Gehrke, J., & Hooker, G. (2013, August). Accurate intelligible models with pairwise interactions. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 623-631).

Lundberg, S.M., Lee, S.-I., (2017). A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems*, *30*, 1–10.

Light, N., Maslov, D., & Rytchkov, O. (2017). Aggregation of information about the cross section of stock returns: A latent variable approach. *The Review of Financial Studies*, *30*(4), 1339-1381.

Liu, L., Bu, R., Pan, Z., & Xu, Y. (2019). Are financial returns really predictable out-of-sample?: Evidence from a new bootstrap test. *Economic modelling*, *81*, 124-135.

Martins, A., & Astudillo, R. (2016, June). From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International conference on machine learning* (pp. 1614-1623). PMLR.

McDonnell, K., Murphy, F., Sheehan, B., Masello, L., & Castignani, G. (2023). Deep learning in insurance: Accuracy and model interpretability using TabNet. *Expert Systems with Applications*, *217*, 119543.

McKinney, W. (2010). Data structures for statistical computing in Python. In Proceedings of the 9th Python in Science Conference.

McNemar, Q. (1947). Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, *12*(2), 153-157.

Mendez-Civieta, A., Aguilera-Morillo, M. C., & Lillo, R. E. (2021). Adaptive sparse group LASSO in quantile regression. *Advances in Data Analysis and Classification*, *15*(3), 547-573.

Menzly, L. & O. Ozbas, O. (2010). Market segmentation and the cross-predictability of returns. *Journal of Finance,* 65, 1555–1580.

Mhaskar, H., Liao, Q., & Poggio, T. (2016). Learning Functions: When Is Deep Better Than Shallow. *arXiv preprint arXiv:* http://arxiv.org/abs/1603.00988.

Molnar, C. (2022). Interpretable Machine Learning: A Guide for Making Black Box Models Explainable (2nd ed.). christophm.github.io/interpretable-ml-book/.

Moskowitz and Grinblatt (1999). "Do industries explain momentum?" Journal of Finance, 54(4) 1249-1290.

Nobre, J., & Neves, R. F. (2019). Combining principal component analysis, discrete wavelet transform and XGBoost to trade in the financial markets. *Expert Systems with Applications*, *125*, 181-194.

Nori, H., Jenkins, S., Koch, P., & Caruana, R. (2019). Interpretml: A unified framework for machine learning interpretability. *arXiv preprint arXiv:1909.09223*.

Nyberg, H. (2011). Forecasting the direction of the US stock market with dynamic binary probit models. *International Journal of Forecasting*, *27*(2), 561-578.

Nyberg, H., & Pönkä, H. (2016). International sign predictability of stock returns: The role of the United States. *Economic Modelling*, *58*, 323-338.

Pastor, L. & Stambaugh, R. F. (2003). Liquidity risk and expected stock returns. *Journal of Political Economy*, 111(3), 642–685.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... & Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. In Advances in Neural Information Processing Systems (NeurIPS).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825-2830.

Pesaran, M. H., & Timmermann, A. (1992). A simple nonparametric test of predictive performance. *Journal of Business & Economic Statistics*, *10*(4), 461-465.

Pönkä, H. (2017). Predicting the direction of US stock markets using industry returns. *Empirical Economics*, *52*, 1451-1480.

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2018). CatBoost: unbiased boosting with categorical features. *Advances in neural information processing systems*, *31*.

Psaradellis, I., & Sermpinis, G. (2016). Modelling and trading the U.S. implied volatility indices. Evidence from the VIX, VXN and VXD indices. *International Journal of Forecasting*, *32*(4), 1268-1283.

Psaradellis, I., Laws, J., Pantelous, A. A., & Sermpinis, G. (2018). Pairs Trading, Technical Analysis and Data Snooping: Mean Reversion vs Momentum. *SSRN Electronic Journal*.

Rapach, D. E., Strauss, J. K., Tu, J., & Zhou, G. (2015). Industry Interdependencies and Cross-Industry Return Predictability. *Research Collection Lee Kong Chian School Of Business*.

Rapach, D. E., Strauss, J. K., Tu, J., & Zhou, G. (2019). Industry Return Predictability: A Machine Learning Approach. *The Journal of Financial Data Science*, *1*(3), 9–28.

Raschka, S. (2018). MLxtend: Providing machine learning and data science utilities and extensions to Python's scientific computing stack. *Journal of open source software*, *3*(24), 638.

Rasekhschaffe, K. C., & Jones, R. C. (2019). Machine learning for stock selection. *Financial Analysts Journal*, *75*(3), 70-88.

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016, August). " Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1135-1144).

Shapley, L. S. (1953). 17. A Value for n-Person Games. In H. W. Kuhn & A. W. Tucker (Eds.), *Contributions to the Theory of Games (AM-28), Volume II* (pp. 307–318). Princeton University Press.

Snedecor, G., & Cochran, W. (1989). *Statistical Methods* (8th ed.). Ames, IA: Iowa State University Press.

Sun, X., Liu, M., & Sima, Z. (2020). A novel cryptocurrency price trend forecasting model based on LightGBM. *Finance Research Letters*, *32*, 101084.

Tibshirani, R. (1996). Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, *58*(1), 267–288.

Timmermann, A., & Granger, C. W. (2004). Efficient market hypothesis and forecasting. *International Journal of forecasting*, *20*(1), 15-27.

Titman, S., & Tiu, C. (2011). Do the best hedge funds hedge?. *The Review of Financial Studies*, *24*(1), 123-168.

Wu, W., Chen, J., Yang, Z., & Tindall, M. L. (2020). A cross-sectional machine learning approach for hedge fund return prediction and selection. *Management Science*, *67*(7), 4577-4601.

Yuan, M., & Lin, Y. (2006). Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *68*(1), 49–67.

Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *67*(2), 301–320.

Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American statistical association*, *101*(476), 1418-1429.