CHARLES UNIVERSITY
FACULTY OF SOCIAL SCIENCES
Institute of Economic Studies

Can Machines Explain Stock Returns?

Master's thesis

Author: Bc. Karolína Chalupová
Study program: Economics and Finance
Supervisor: doc. PhDr. Jozef Baruník, Ph.D.
Consultant: Mgr. Martin Hronec
Year of defense: 2021

Declaration of Authorship

The author hereby declares that he or she compiled this thesis independently, using only the listed resources and literature, and the thesis has not been used to obtain any other academic title.

The author grants to Charles University permission to reproduce and to distribute copies of this thesis in whole or in part and agrees with the thesis being used for study and scientific purposes.

Prague, January 5, 2021

Karolina Chalupova

Abstract

Recent research shows that neural networks predict stock returns better than any other model. The networks' mathematically complicated nature is both their advantage, enabling to uncover complex patterns, and their curse, making them less readily interpretable, which obscures their strengths and weaknesses and complicates their usage. This thesis is one of the first attempts at overcoming this curse in the domain of stock returns prediction. Using some of the recently developed *machine learning interpretability* methods, it explains the networks' superior return forecasts. This gives new answers to the long-standing question of which variables explain differences in stock returns and clarifies the unparalleled ability of networks to identify future winners and losers among the stocks in the market. Building on 50 years of asset pricing research, this thesis is likely the first to uncover whether neural networks support the economic mechanisms proposed by the literature. To a finance practitioner, the thesis offers the transparency of decomposing any prediction into its drivers, while maintaining a state-of-the-art profitability in terms of Sharpe ratio. Additionally, a novel metric is proposed that is particularly suited to interpret return-predicting networks in financial practice. This thesis offers a usable and economically explainable account of how machines make stock return predictions.

JEL Classification	C45, G12
Keywords	(interpretable) machine learning, neural networks, stock returns

Title	Can Machines Explain Stock Returns?
Author's e-mail	`chalupova.karolina@gmail.com`
Supervisor's e-mail	`barunik@fsv.cuni.cz`
Consultant's e-mail	`martin.hronec@fsv.cuni.cz`

Abstrakt

Nedávný výzkum ukazuje, že neuronové sítě dokážou předpovídat akciové výnosy lépe, než kterýkoli jiný model. Metematicky komplikovaná povaha sítí je zároveň jejich výhodou, umožňující odhalovat komplexní vzorce, a jejich prokletím, znesnadňujícím jejich interpretaci, což zamlžuje výhody a nevýhody sítí a komplikuje jejich užití. Tato práce je jedním z prvních pokusů toto prokletí

překonat. Za použití nově vyvinutých metod *interpretovatelného strojového učení* objasňuje, jak sítě vytvářejí své vynikající předpovědi výnosů. Poskytuje tak nové odpovědi na starou otázku, které proměnné určují rozdíly v akciových výnosech, a vysvětluje, co stojí za bezkonkurenční schopností neuronových sítí identifikovat mezi akciemi na trhu budoucí vítěze a poražené. Tato práce je pravděpodobně první, která zjišťuje, zda neuronové sítě podporují ekonomické mechanismy, které během posledních 50 let přinesl výzkum v oblasti oceňování aktiv. Z hlediska aplikace pro finanční praxi práce nabízí transparentnost, kterou přináší dekompozice každé předpovědi na vlivy jednotlivých vstupních proměnných; zároveň si práce zachovává Sharpe ratio na úrovni současné vědy. Navíc je představena nová metrika, která je zvláště vhodná pro interpretaci neuronových sítí ve finanční praxi. Tato práce nabízí aplikovatelný a ekonomicky vysvětlitelný popis toho, jak stroje předpovídají výnosy akcií.

Klasifikace JEL	C45, G12
Klíčová slova	(interpretovatelné) strojové učení, neuronové sítě, akciové výnosy
Název práce	Mohou stroje vysvětlit akciové výnosy?
E-mail autora	`chalupova.karolina@gmail.com`
E-mail vedoucího práce	`barunik@fsv.cuni.cz`
E-mail konzultanta práce	`martin.hronec@fsv.cuni.cz`

Acknowledgments

The author is especially grateful to doc. PhDr. Jozef Baruník, Ph.D., as well as Mgr. Martin Hronec for guiding, inspiring and offering valuable critics to her work both in the big picture and the technical details, always pointing to the right direction. The author is once again thankful to Mgr. Martin Hronec and his colleague MPhil. Ondřej Tobek, Ph.D, for sharing their meticulously cleaned dataset. The author would also like to thank Bc. Tomáš Turlík for drawing her attention to an important detail in data cleaning. A further thank you extends to RNDr. Milan Straka, Ph.D., for his outstanding courses in machine learning and advising this thesis on hyperparameter tuning in Python. This thesis is part of a project that has received funding from the European Unions Horizon 2020 research and innovation programme under the Marie Skodowska-Curie grant agreement No. 681228. Any mistakes are, of course, the author's own.

Typeset in LaTeXusing the IES Thesis Template.

Bibliographic Record

Chalupova, Karolina: *Can Machines Explain Stock Returns?*. Master's thesis. Charles University, Faculty of Social Sciences, Institute of Economic Studies, Prague. 2021, pages 101. Advisor: doc. PhDr. Jozef Baruník, Ph.D.

Contents

List of Tables

List of Figures

Acronyms

ML Machine Learning

Notation

The terms **characteristics**, **predictors** and **features** are used interchangeably and refer to the firm-specific variables based on which the returns are predicted (inputs to the neural network). The term *characteristics* is used in asset-pricing literature, the term *predictors* in econometrics, and *features* in machine learning.

The following mathematical symbols are used throughout the thesis.

Scalars, vectors, and matrices are denoted using italics, for example:

$$a, \boldsymbol{a}, \boldsymbol{A}$$

for scalar, vector and matrix respectively. An element of a vector $\boldsymbol{a}$ is denoted as

$$a_i$$

and an element of a matrix $\boldsymbol{A}$ is denoted as

$$a_{i,j}.$$

Random variables are denoted using uppercase italics, for example:

$$A$$

and they always represent a scalar random variable.

Master's Thesis Proposal

Author	Bc. Karolína Chalupová
Supervisor	doc. PhDr. Jozef Baruník, Ph.D.
Proposed topic	Can Machines Explain Stock Returns?

Motivation Explaining why different assets earn different average returns is one of the most studied problems in finance. While the academia is typically interested in identifying common risk factors in returns, the industry strives to predict the returns to identify which assets to buy and which to sell. The task is similar: build a model that correctly explains (predicts) asset returns.

A first such model is the 1970s Capital Asset Pricing Model (CAPM), which attributes an assets expected return to its exposure to the market risk factor. Fama and French (1993) find two additional sources of risk, resulting in a three-factor model, later augmented to five factors (Fama and French 2015). Since Fama and Frenchs publications, hundreds of papers have been reporting discoveries of yet new factors: Harvey, Liu, and Zhu (2016) count 316 different published factors, considering top journals only, and Cochrane (2011, 1047) describes the state of the research as a zoo of factors.

One of the issues with the zoo of factors is that the predictors were typically found controlling only for the usual three or five factors and not the rest of the zoo, which is one of the leading reasons why most claimed research findings in financial economics are likely false (Harvey, Liu, and Zhu 2016, 1). Search for a good model is further complicated by possible nonlinearities and factor interactions (Gu, Kelly, and Xiu 2018). Cochrane (2011, 1060) calls for solving this multidimensional challenge: First, which characteristics really provide independent information about average returns? Which are subsumed by others? Second, does each new anomaly variable also correspond to a new factor formed on those same anomalies? (...) Third, how many of these new factors are really important?

Meanwhile, practitioners in finance industry are celebrating unparalleled results when they predict stock returns with machine learning (ML) models. Gu, Kelly, and Xiu (2018, 1) find that ML offers an improved description of expected return

behavior compared to the standard linear regression and an unprecedented R2 in the United States. Tobek and Hronec (2018) confirm this finding internationally, with their ML models 4 times more profitable than linear regression and offering 2 times the Sharpe ratio.

As argued by Gu, Kelly, and Xiu (2018), ML is particularly suited for addressing the multidimensional challenge. First, the sheer amount of 316 candidate explanatory variables (even without considering unknown number and form of their interactions), traditional estimation techniques loose many degrees of freedom. Second, high correlation of characteristics induces multicollinearity, increasing the variance of the estimates. Third, allowing for nonlinarities seems crucial for predicting returns (Gu, Kelly, and Xiu, 2018). ML methods are often build for dimension reduction and nonlinearity.

For the academia to answer the multidimensional challenge and for finance ML practitioners to understand their models better, it is necessary to interpret ML models, i.e., be able to predict how they link inputs to returns predictions. ML models are scarcely interpretable per se, but additional methods can be used to extract interpretable information from them.

Hypotheses

1. Which characteristics provide information about average equity returns?
2. Which interactions of characteristics are important?
3. What is the effect of changing a characteristics value on average returns?

Methodology First, I am going to obtain data from CRSP, Compustat and Datastream on publically traded equity. Second, I am going to calculate the characteristics proposed by prior research as return predictors. Third, I am going to train several ML models with these characteristics as features. In doing so, I am going to pay particular attention to high correlation of characteristics to answer the research question meaningfully. This includes considerations of stability of the feature selection. Finally, I am going to examine the models to answer the research questions.

For interpretable models, the research questions can be answered directly. These models include regularized linear regression, principal components regression or partial least squares. A disadvantage of linear models is their significantly lower predictive power compared to complex models (Gu, Kelly, and Xiu 2018).

For complex models, the following methods can be used to make them interpretable and tackle the respective research questions:

1. Feature Importance. Feature importance measures describe how much a model relies on a feature to produce a prediction. Model reliance (Fisher, Rudin, and

Dominici 2018) measures feature importance by calculating error in prediction resulting from permuting a feature. Intuitively, if permuting a feature (randomly shuffling its values across examples), does not have an impact on prediction error, the feature is not important for the prediction.

2. Feature Interaction. Friedman and Propescu (2008) propose H-statistic to measure a) to what extent two features interact with each other, b) to what extent a feature interacts with all other features. The statistic ranges between 0 and 1 and measures the variance of model output explained by the interaction. An advantage of H-statistic is that it captures all functional forms of interactions.

3. Accumulated Local Effects (ALE). Apley (2016) develops ALE to measure how a feature on average influences the prediction. For a given feature, ALE first defines a grid of values. Second, for each grid segment, ALE looks at examples such that their feature values fall in the given grid segment. Third, it calculates average prediction pretending the feature values are slightly higher (lower) than in the grid segment. Fourth, it subtracts these two averages. Therefore, one can interpret ALE as increasing value of this feature by this unit increases output by x units, holding other features fixed at the average of the group, where the group consists of examples with similar values of analyzed feature.

4. Finally, a nonlinear yet interpretable model can be used, as in the new paper by Bryzgalova et al. (2019), which introduces a variant of random trees, so called asset-pricing trees, to tackle the question of which variables and interactions are important for predicting stock returns. Through a new method of pruning random trees, the authors are able to identify common risk factors in stock returns.

Expected Contribution My thesis would contribute to a deeper understanding of equity returns and of the ML models trying to predict them. All my research questions are designed to address the first part of Cochranes (2011, 1060) multidimensional challenge: First, which characteristics really provide independent information about average returns? Which are subsumed by others?. Answering this question is essential for understanding of equity returns behavior and building any model, explanatory or predictive. My thesis fits into this strand of literature in the following ways.

Several studies have use ML to study my first research question. Gu, Kelly, and Xiu (2018) calculate feature importance of their 94 characteristics in estimating equity returns. Their different ML models agree on selection of the most important characteristics (price trends, return reversal, momentum, stock liquidity, stock

volatility, and valuation ratios). Kelly, Pruit, and Su (2017) use instrumental principal component analysis to instrument risk factors from 36 characteristics, with result similar to Gu, Kelly, and Xiu (2018) that only a small fraction of characteristics is important, again including return reversal and valuation ratios. I expect to use these studies as a benchmark.

The academic contribution is twofold. While Gu, Kelly, and Xiu (2018) find that feature interactions seems crucial for returns prediction, I am not aware of a paper trying to identify, measure, and interpret the interactions. By studying my second research question, I am going to shed light on which interactions are important. In my third research question, I am going to shift from the task of identifying important features to measuring their effect on predictions, which is a key step in interpreting return-predicting ML models.

In addition to the academic contribution, my thesis can have a practical impact. ML models are becoming the norm in the finance industry. To debug, improve, and sell an ML model, interpretability of the model is highly important. From a practitioners perspective, my thesis can serve as a case study of some of the methods, issues, and results that occur when interpreting return-predicting ML models.

Outline

1. Motivation: ML is suitable for answering Cochranes (2011) multidimensional challenge and solving the current issues of asset pricing research.
2. Literature review: State of current asset pricing research, existing answers to Cochranes (2011) multidimensional challenge.
3. Data: Calculation and inspection of features.
4. Methodology: ML models used, interpretation techniques.
5. Results: Answers to my three research questions.
6. Conclusion: Implications of my results, areas for further research.

Core Bibliography Apley, D. W. (2016). Visualizing the effects of predictor variables in black box supervised learning models. arXiv preprint arXiv:1612.08468.

Bryzgalova, S., Pelger, M., & Zhu, J. (2019). Forest through the trees: Building cross-sections of stock returns. Available at SSRN 3493458.

Cochrane, J. H. (2011). Presidential address: Discount rates. The Journal of finance, 66(4), 1047-1108. 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.

Fisher, A., Rudin, C., & Dominici, F. (2018). All Models are Wrong but many are Useful: Variable Importance for Black-Box, Proprietary, or Misspecified Prediction Models, using Model Class Reliance. arXiv preprint arXiv:1801.01489.

Friedman, J. H., & Popescu, B. E. (2008). Predictive learning via rule ensembles. The Annals of Applied Statistics, 2(3), 916-954.

Gu, S., Kelly, B., & Xiu, D. (2018). Empirical asset pricing via machine learning (No. w25398). National Bureau of Economic Research.

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

Tobek, O., & Hronec, M. (2018). Does it Pay to Follow Anomalies Research? International Evidence.

Author

Supervisor

Chapter 1

Introduction

The question of why different assets earn different returns has been occupying finance for over half a century. Still, instead of providing a clear answer, literature proposes more and more variables that purportedly explain asset returns, accumulating *hundreds* of diverse candidate answers, form exposure to underlying risks of the investment to behavioral biases of the investors themselves or imperfections in the practical functioning of the stock market. At the same time, a promising strand of literature emerges that brings machine learning (ML) methods to this financial field – and to a great avail: machines are able to predict asset returns unprecedentedly well. However, machines are mathematically complicated and the knowledge they extract from the data thus remains hidden in their complex structure, leaving their predictions unexplained and the question of what drives asset returns still unanswered. In other words, the onset of machine learning in finance brings improvements in forecasting, but does not satisfactorily account for a desire of an economist to identify and interpret the effects driving those improved results. This thesis is one of the first attempts to fill this gap. With the use of so-called machine learning interpretability methods, it uncovers which variables the machines focus on when forecasting stock returns.

Machine learning has seen a surge in many fields, both academic and applied. Machines, neural networks in particular, recently achieved state-of-the-art results in image and speech recognition, object detection, game playing or language translation, often surpassing human abilities. Machines are also spreading to more ethically sensitive areas, such as car driving, credit rating, or medical diagnostics. In finance, they are applied to a wide range of tasks, including bankruptcy and bank failure prediction, bond rating, inflation fore-

casting and, yes, stock returns prediction. In virtually all these applications, they out-perform the more traditional approaches, such as linear regression or discriminant analysis (Fadlalla & Lin 2001). This also attracts financial investors: in stock returns prediction, neural networks offer 2 to 4 times the profitability of the older approaches (Gu *et al.* 2020; Tobek & Hronec 2020).

With this advance of artificial intelligence, a need arises to understand the machines on a deeper level, to explain their decisions – in technical terms, to *interpret* the machines. This is important for ethical, scientific and practical reasons. First, from the perspective of ethics, it is crucial to understand how a model works before using it to make decisions that affect human lives, particularly in as sensitive areas as medical diagnostics or loan eligibility. Second, from the perspective of scientific curiosity, studying how exactly artificial intelligence works can bring inspiring results. For example, interpreting neural networks used in image recognition revealed surprising similarities between the hidden layers of the networks and human retinal cortex (Olah *et al.* 2017). Finally, it is necessary to interpret the machines for practical purposes: to develop, debug and improve them. For instance, by uncovering which input variables are important, a machine can be further improved by fine-tuning the vital inputs (De Prado 2018). To meet these needs, interpretable ML is now a quickly developing field (Molnar 2020).

In finance, interpretable ML is still in its infancy. Bryzgalova *et al.* (2019) arrive at a data-driven discount factor for stock prices by creatively using random forests, an interpretable ML model. Li *et al.* (2020) decompose exchange rate predictions into linear and non-linear effects and show how the variables inside the networks interact to produce exchange rate prediction. Gu *et al.* (2020); Tobek & Hronec (2020) use neural networks to predict stock returns, but they focus on performance and profitability rather than interpretation of the machines. They show that neural networks predict stock returns better than any other model, but the investigation into the economic forces behind these results is very limited. This work aims to fill this void.

This thesis is likely a first deeper dive into ML interpretability in the domain of stock returns prediction. First, this offers a data-driven perspective on the long-standing economic question of what accounts for the differences in returns between stocks. It builds on the 50 years of research in finance, by using the top variables the field has accumulated as the most robust stock returns predictors – such as momentum, reversal, liquidity, value effect or accruals – and uses them as inputs to neural network, which forecasts the stocks' returns in the next

month. The thesis then uses ML interpretability methods to extract insights about which of these variables influence the predicted return the most, and, importantly, in which direction. This direction is important to add economic meaning to the network: all input variables have theoretical underpinnings from prior literature, which suggests which sign (direction) each particular variable should have in order to comply with economic theory. This work is likely the first to study if machines agree or disagree with the economic mechanisms the field has proposed over the last century. Apart from this economic perspective, this thesis also offers a more applied interpretation of the networks: in particular, it uncovers which variables are important for the profitability of networks in forming long-short portfolios, that is, it shows which firm characteristics the networks rely on to identify the future winners and losers among the stocks in the market. Both these contributions are to the best of my knowledge unseen in prior literature.

The dataset is a liquid universe of globally traded stocks, from 1990 to 2018, and offers 30 firm-level variables on 8,350 companies. The ability of the networks used in this thesis to predict the stock returns in the following month is at par with the state-of-the-art models, which further enhances its relevance for financial practice. The thesis studies 4 feed-forward neural networks, consisting of 1 to 4 hidden layers. This architecture is ubiquitous in many financial applications other than stock return prediction (Fadlalla & Lin 2001), which makes the work quite relevant to other areas as well.

The thesis is structured as follows: Chaper 2 offers a review of existing literature, focusing first on the economic motivation of stock return predictors, then on the current state of interpretable ML in stock returns, and finally on the selection of a suitable feature importance measure. Chapter 3 first describes the dataset and then explains the methodology behind training, evaluating and interpreting the networks used. Chapter 4 gives the results, first demonstrating the predictive ability and profitability of the networks, and then diving into their interpretation. Chapter 5 concludes and offers areas for further research.

Chapter 2

Literature Review

The literature review proceeds as follows. First, general approaches to modelling stock returns are reviewed (the factor and characteristics models). Second, the characteristics models are discussed in more detail and show the current literature's answer to which variables predict stock returns. Then, the exposition motivates the use of neural networks in stock returns prediction and reviews the advances of interpretable ML in this domain. Finally, ML interpretability literature is examined with the aim of choosing a suitable feature importance measure.

2.1 Approaches to Modeling Stock Returns

There are two major approaches in literature to modeling stock returns: factor models and characteristics models. Factor models explain the returns as

$$E_t(R^i_{t+1}) = R^f_{t+1} + \beta_{1,i,t}\lambda_{1,t} + \beta_{2,i,t}\lambda_{2,t} + \ldots + \beta_{K,i,t}\lambda_{K,t}. \tag{2.1}$$

In words, the expected return of stock i at time t is a function of the risk-free return at time t and priced exposures to K *risk factors*. $\beta_{1,i,t}$ to $\beta_{K,i,t}$, called *factor loadings*, are exposures of stock i to the corresponding risk factor and represent the amount of risk inherent in asset i due to its correlation to the risk factor. $\lambda_{1,t}$ to $\lambda_{K,t}$ are the prices of the K risk factors at time t, common to all stocks (Kelly *et al.* 2019). A well-known example of factor model is Capital Asset Pricing Model, where there is a single risk factor – the return on market portfolio (Cochrane 2009). In the view of the Capital Asset Pricing Model, an investor is rewarded for holding risk inherent to stock i that she cannot diversify away: the market risk. The assets that have negative

correlation with the market are precious, because they provide wealth when the economy (the market, or other sources of wealth in general) performs poorly. In case of more factors (classical examples are Fama & French (1996; 2015)), the investor is similarly rewarded for holding stocks that covary with other sources of undiversifiable risk.

The main issue with factor models is that the risk factors are unobserved and so are the risk prices, so we have no directly observed variables at the right hand side of the estimated equation. Typically, the factors must be constructed by the researchers themselves, before the risk exposures and finally risk prices can be estimated. Examples of this pre-specification include the market portfolio in the case of Capital Asset Pricing Model or portfolios organized based on a firm-specific variables, such as firm size in the Small-Minus-Big factor in Fama & French (1993). Recently, Bryzgalova *et al.* (2019) provide an interesting alternative to constructing portfolios using ML, particularly, random forests. In their study, the factors are not pre-specified by the researcher, but constructed using a data-driven decision tree. This has the advantage of not having to specify the factors *ex ante*, but letting the data speak instead.

The other approach to modeling stock returns is using firm-specific variables, called *characteristics*, such as firm size or past return, as explanatory variables:

$$E_t(r^i_{t+1}) = f(x_{1,i,t}, x_{2,i,t}, \ldots, x_{K,i,t}). \tag{2.2}$$

In words, the expected return of stock i in the next period $(t+1)$ is a function (f) of the K characteristics of stock i at time t $(\boldsymbol{x}_{i,t})$. Empirical examples include e.g., Gu *et al.* (2020); Tobek & Hronec (2020); Bryzgalova *et al.* (2019). This is also the approach taken by this thesis. The key advantage over factor models is that all explanatory variables are directly observed.

Characteristics models and factor models are related: the motivation of using characteristics is that they proxy for the firm's exposure to sources of risk (Bryzgalova *et al.* 2019; Kelly *et al.* 2019). This is instructively seen in Kelly *et al.* (2019), who use characteristics as proxies for risk exposure quite explicitly:

$$
\begin{aligned}
E_t(R^i_{t+1}) &= R^f_{t+1} + \boldsymbol{\beta}'_{i,t}\boldsymbol{\lambda}_t \\
R^i_{t+1} &= \alpha_{i,t} + \boldsymbol{\beta}'_{i,t}\boldsymbol{f}_{t+1} + \epsilon_{i,t+1} \\
\boldsymbol{\beta}'_{i,t} &= g(x_{1,i,t}, x_{2,i,t}, \ldots, x_{K,i,t}),
\end{aligned}
$$

where the first two rows are the typical factor model and the third row shows how the exposures $\boldsymbol{\beta}_{i,t}$ are instrumented using K characteristics $\boldsymbol{x}_{i,t}$. Under this approach, the risk exposures are first instrumented using characteristics and then the factor model is estimated. However, in case that uncovering the underlying factor structure is not the main focus, the simpler approach of using characteristics directly as per equation 2.2 is common (Gu *et al.* 2020; Tobek & Hronec 2020; Bryzgalova *et al.* 2019). This approach is now discussed in detail.

2.2 Modeling Stock Returns With Characteristics

> We have a lot of questions to answer: First, which characteristics really provide independent information about average returns?
>
> Cochrane (2011)

As just described, a common approach to model stock returns is to use firm-specific variables, called characteristics, as predictors of the return, as per the equation 2.2. The equation also shows that the task of the researcher is to determine which characteristics ($\boldsymbol{x}$) to use, and the general functional form f in which they should interact to produce the prediction. The parameters which pin down the particular manifestation of the functional form are then estimated empirically using data, which completes the model. The choice of the characteristics and of the functional form is now discussed in turn using existing literature.

2.2.1 Choosing the Characteristics

Literature has accumulated hundreds of variables that purportedly predict stock returns. Harvey *et al.* (2016) count 313 variables, considering the top journals only and calling the count "surely too low". Indeed, Cochrane (2011) refers to the state of asset pricing as a "zoo". Actually, many of these published predictors are spurious. As (Harvey *et al.* 2016, p. 5) put it rather

famously: "most claimed research findings in financial economics are likely false". Main reasons for this include publication bias and data-snooping bias (multiple-hypothesis testing bias, in-sample overfitting) and lack of replication studies in finance (Harvey *et al.* 2016; McLean & Pontiff 2016). 12% of variables cannot be replicated even in-sample, the rest is biased by about 10% (McLean & Pontiff 2016). Harvey *et al.* (2016) show that out of 296 published significant factors 158, 142, 132 and 80 are false discoveries, the precise number depending on statistical framework used.

Cochrane formulates the problem of separating the wheat from the chaff in stock returns prediction in his "multidimensional challenge" (Cochrane 2011, p. 1060): "We have a lot of questions to answer: First, which characteristics really provide independent information about average returns?" In response to this challenge, studies emerged that consider all the published characteristics in a single model, which allows to their effects to crowd each other out in a 'survival of the fittest', such as Gu *et al.* (2020) and Tobek & Hronec (2020).

Gu *et al.* (2020) and Tobek & Hronec (2020) study 94 (153) characteristics proposed by prior literature and use them in various return-prediction models. Their results suggest that some categories of characteristics are consistently more important than others across time and model specifications. This finding presents a crucial stepping stone for this thesis, allowing it to select 30 most important variables from Tobek & Hronec (2020) as stock return predictors, which can be considered a distillation of the current literature's answer to the Cochrane's multidimensional challenge.[1]

2.2.2 Economic Motivation of the Characteristics Used in This Thesis

All characteristics have economic motivation that suggests a particular mechanism of why they should predict stock returns. When we inspect this economic rationale, several broader categories emerge consistently. This section reviews these economic mechanisms and at the same time presents the 30 variables used in this thesis as members of these broader categories. This is summarized

[1]There are three advantages of Tobek & Hronec (2020) over Gu *et al.* (2020) for this purpose: First, like this thesis, Tobek & Hronec (2020) use a liquid universe of stocks, which means that their results are less likely to be driven by transactions costs. Second, their data are global, rather than US only, which allows to uncover more generally applicable findings. Finally, they use larger set of characteristics (153 rather than 93), which makes it less likely that important information is omitted.

in Table 2.1, which shows all characteristics used in this thesis (column *Features*), together with their authors and publishing journal, categorized by their economic motivation (column *Category*). Each category is discussed in detail its own paragraph below.

A first group of characteristics relates to how investors approach volatility. Traditional asset pricing theory assumes investor's approach to risk is symmetrical in gains and losses (Cochrane 2009). However, psychology suggests that people are at the same time risk-averse (case of insurance) and risk-loving (case of lottery) and have asymmetrical utility in gains and in losses (Kahneman & Tversky 2013). (This is in contrast to the standard utility maximization assumption of the traditional asset-pricing models, such as Consumption-Based model and Capital Asset Pricing Model.) This has direct implications for stock returns: it seems that lottery-like stocks, stocks with right skew and stocks with low correlation with market volatility can afford to pay lower average returns in exchange for their appealing risk profile (predictors *Maximum Return* (Bali *et al.* 2011), *Coskewness* (Harvey & Siddique 2000) and *Idiosyncratic Risk* (Ang *et al.* 2006), respectively).

A second group of characteristics also relates to the behavioral science. It seems that investors are slow to revise their existing beliefs upon arrival of new information, which shows in stock returns as momentum (predictors *52-Week High* (George & Hwang 2004) and *Lagged Momentum* (Novy-Marx 2012)), and when they *do* revise their beliefs, they over-react, which shows as reversal – periods of high returns followed by periods of low returns (predictors *Short-Term Reversal* (Jegadeesh 1990) and *Momentum-Reversal* (Jegadeesh & Titman 1993)).

A third and fourth group of characteristics relate to liquidity. It appears that illiquid stocks (typically small stocks with low trading volume and large bid-ask spread) must offer a higher return to compensate for their higher trading costs (predictors *Amihud's Measure* (Amihud 2002), *Liquidity Shocks* (Bali *et al.* 2013) and *Volume over Market Value of Equity* (Haugen & Baker 1996)). It seems that investors avoid illiquidity to the degree a that mere higher chance of a stock becoming illiquid commands a higher return (predictors *Coefficient of Variation of Share Turnover* (Chordia *et al.* 2001) and *Liquidity Beta 3* and *Liquidity Beta 5* (Acharya & Pedersen 2005)).

A fifth group of characteristics focuses is due to limited attention of investors, which can result in mis-pricing. First, it seems that investors tend to overlook artificially bloated (manipulated or "dressed") balance sheets, and

Category	Feature	Author	Journal	Sign
Attitude To Risk	Coskewness	Harvey & Siddique (2000)	JF	-1
	Idiosyncratic Risk	Ang *et al.* (2006)	JF	-1
	Maximum Return	Bali *et al.* (2011)	JF	-1
Behavioral	52-Week High	George & Hwang (2004)	JF	1
	Lagged Momentum	Novy-Marx (2012)	JFE	1
	Momentum-Reversal	Jegadeesh & Titman (1993)	JF	-1
	Short-Term Reversal	Jegadeesh (1990)	JF	-1
Financial Constraints	Whited-Wu Index	Whited & Wu (2006)	RFS	1
Illiquidity	Amihud's Measure (Illiquidity)	Amihud (2002)	JFM	1
	Liquidity Shocks	Bali *et al.* (2013)	RFS	-1
	Volume / Market Value of Equity	Haugen & Baker (1996)	JFE	-1
Illiquidity Risk	Coefficient of Variation of Share Turnover	Chordia *et al.* (2001)	JFE	1
	Liquidity Beta 3	Acharya & Pedersen (2005)	JFE	-1
	Liquidity Beta 5	Acharya & Pedersen (2005)	JFE	1
Limited Attention	Accruals	Sloan (1996)	AR	-1
	Change in Common Equity	Richardson *et al.* (2006)	AR	-1
	Earnings Forecast-to-Price	Elgers *et al.* (2001)	AR	1
	Earnings Predictability	Francis *et al.* (2004)	AR	-1
	Net Operating Assets	Hirshleifer *et al.* (2004)	JAE	-1
	Operating Profits to Assets	Ball *et al.* (2016)	JFE	1
	Profit Margin	Soliman (2008)	AR	1
	RD / Market Equity	Chan *et al.* (2001)	JF	1
Seasonality	Seasonality	Heston & Sadka (2008)	JFE	1
	Seasonality 11-15 N	Heston & Sadka (2008)	JFE	-1
	Seasonality 2-5 A	Heston & Sadka (2008)	JFE	1
	Seasonality 2-5 N	Heston & Sadka (2008)	JFE	-1
	Seasonality 6-10 A	Heston & Sadka (2008)	JFE	1
	Seasonality 6-10 N	Heston & Sadka (2008)	JFE	-1
Value Effect	Duration of Equity	Dechow *et al.* (2004)	RAS	-1
	Leverage Component of Book/Price	Penman *et al.* (2007)	JAR	1

Table 2.1: Economic Motivation of Predictors Used in This Thesis

The table shows all the characteristics (features) used in this thesis grouped by their underlying economic motivation as predictors of stock returns. Column *Category* gives the main economic motivation of the variable and is discussed in detail in the text of this section. Columns *Author* and *Journal* specify the first academic publication of each characteristic. Column *Sign* shows the direction of the relationship between the characteristic and returns, as found by the original paper. The *Journal* shortcuts stand for, in alphabetic order: The Accounting Review (AR), Journal of Accounting and Economics (JAE), Journal of Accounting Research (JAR), Journal of Finance (JF), Journal of Financial Economics (JFE), Journal of Financial Markets (JFM), Review of Accounting Studies (RAS), and Review of Financial Studies (RFS).

tend to over-price firms with high accruals (predictors *Accruals* (Sloan 1996), *Net Operating Assets* (Hirshleifer *et al.* 2004), *Operating Profits to Assets* (Ball *et al.* 2016), and *Change in Common Equity* (Richardson *et al.* 2006)). Similarly, they tend to under-value firms with high research and development costs, as these are typically not present on the balance sheets (predictor *RD Over Market Equity* (Chan *et al.* 2001)). Finally, limited attention of investors manifests in overlooking important decompositions of profit margin (predictor *Profit Margin* (Soliman 2008)).

A sixth group of characteristics reflects the empirically observed seasonal patterns in stock returns. While the theoretical underpinning of these characteristics is still a matter of academic debate, the phenomenon is very strong and robust empirically (*Seasonality* predictors from Heston & Sadka (2008)).

A seventh group of characteristics are proxies for the well-known value effect Fama & French (1993), which shows that empirically, firms with high Book to Market ratio tend to have higher returns. The predictors used here are *Duration of Equity* (Dechow *et al.* 2004)) and *Leverage Component of Book to Price* (Penman *et al.* 2007), and their original papers discuss also the theoretical underpinnings of the phenomenon.

Finally, financially constrained firms tend to co-move, and since financial constraints are typically market-wide during economic crises, this risk is not diversified, and therefore priced. The predictor *Whited-Wu Index* (Whited & Wu 2006) measures the extent to which a stock is impacted by financial constraints.

2.2.3 Choosing the Functional Form: Why Neural Networks?

A crucial modeling decision to be made is choosing the general functional form f in which the characteristics should interact in equation 2.2. While classical asset-pricing models are linear (e.g. Fama & French 1993; 1996), the function f can be any model: Bryzgalova *et al.* (2019) use random forests and Gu *et al.* (2020); Tobek & Hronec (2020) employ the whole spectrum of methods, both linear (linear regression with and without regularization, principal components regression, elastic net) and nonlinear (generalized linear models, random forest, gradient-boosted regression trees and neural networks). There are two arguments why use neural networks for the stock return prediction: their superior performance and their ability to find non-linear relationships from data.

The first argument to using networks to predict stock returns is their su-

perior performance. Prior research indicates that neural networks can predict stock returns better than any other model. Gu *et al.* (2020) make a horse race of several ML and traditional methods. They show that neural networks out-perform all other models, both in explained variance and in profitability of their returns' forecasts. They are far better than linear models (linear regression with and without regularization, principal components regression, elastic net), and also beat other nonlinear models (generalized linear models, random forest, and gradient-boosted regression trees). Tobek & Hronec (2020) confirm these findings on international (as opposed to U.S.) data and liquid universe of stocks. Gu *et al.* (2020) statistically reject linear models in favor of the non-linear ones as those with better performance.

The second argument for using neural networks is their ability to find important non-linear relationships in the data. Recent ML literature suggests that non-linearity is vital for explaining stock returns (Gu *et al.* 2020; Bryzgalova *et al.* 2019) and that the linear models miss important information, particularly non-linearities and variable interactions. If a variable has a non-linear relationship with returns, the linear models can conclude there is no association, while in reality there is an important non-linear relationship. For example, in Gu *et al.* (2020), linear models deem size and volatility unimportant predictors, while non-linear models find the contrary. A similar problem may arise if predictors have zero association with return themselves, but not in interaction with other predictors. The nonlinearity seems to be a curcial aspect of stock return prediction: Bryzgalova *et al.* (2019) and Gu *et al.* (2020) agree that the ability to uncover non-linearities and interactions is the crucial driving force of the superior ML performance. In fact, neural networks are theoretically motivated as Universal Approximators: no matter the functional form f, a neural network already with one hidden layer can approximate any "well-behaved" function arbitrarily well. As there are so many possible forms in which the predictors can interact, it is unfeasible to search for the correct specification manually by pre-specifying the functional form. Thanks to their flexibility, neural networks thus allow for fitting the correct functional form in a data-driven manner.

A final advantage is not specific to neural networks, but applies to ML literature in general. Unlike in the traditional asset-pricing literature, which focuses on in-sample fit, model selection in ML is based on validation data, as opposed to in-sample (training) data. Similarly, model evaluation is done solely on held-out (testing) data. In the traditional approach to stock returns, model

is tuned by the researcher based on in-sample fit, which results in unreplicable spurious findings (McLean & Pontiff 2016). On the other hand, ML puts an emphasis on out-of-sample performance, which allows to select a robust model that generalizes well to previously unseen data.

2.3 Interpretable ML and Stock Returns: What Has Been Done?

To explain how the methods employed in this thesis fit into the broader context of ML interpretability literature, it is useful to first categorize them following Molnar (2020). A first distinction can be made between intrinsically interpretable models, which are explainable *per se* (linear and logistic regression, random trees), and post-hoc methods, which can be applied to extract interpretable knowledge from more complex models, such as neural networks, which is the case of this thesis. A further distinction can be made as to the applicability of the interpretability methods: some methods are *model-specific*, that is, can only be applied to some ML models, while others are *model-agnostic*, applicable to all models from linear regressions to random trees and neural networks. For purposes of generality, the methods selected in this thesis are model-agnostic — while they are well-suited to neural networks, they can also be used on any other model, such as linear regression. Yet other distinction can be made between *global* and *local* ML interpretability methods: while the former study how a model decides overall (across all observations), the latter can be used to interpret the drivers behind a single observation. Both approaches are taken in this thesis, as both global and local perspectives are important for model interpretability.

Further, there are several aspects of interpreting a neural network in particular. First, it is possible to study which input variables are important for the prediction (known as feature importance). Second, it is possible to see how the prediction changes in response to the input variables (marginal relationships). Finally, interaction effects and non-linearities can be uncovered from the hidden layers. This thesis takes the first two approaches, as it considers them natural starting points, and leaves the third for follow-up research. The three are now discussed in turn, in reverse order.

First, hidden layers can be analyzed to to uncover the feature interactions and patterns learned by the network. Hidden layer's values is a linear com-

bination of the previous layer, with then applied non-linear function. From this perspective, the hidden neurons can be considered as features learned, or engineered, by the network. The hidden layers can be inspected ex post to answer the question of which *interactions* between input features are important. For example, in the case of image recognition, it is possible to see that the network learns to recognize simple shapes (such as straight lines and circles) in the first hidden layers, and then proceeds to learn more intricate patterns (such as flowers and shapes of animals) in the deeper layers (Olah *et al.* 2017). In the returns prediction task, it could be analogously studied which feature combinations are important. However, there is no such paper to the best of my knowledge. The closest to it is likely the interesting study of interaction terms in Bryzgalova *et al.* (2019), but it is a different model (random forest rather than neural network, i.e., no hidden layers) and a different prediction task (construction of basis assets rather than returns prediction). Gu *et al.* (2020) *do* show interaction effects in a neural network predicting stock returns, but it is only an interaction of a single variable with 4 other variables (that is, 4 interactions out of 8649, and that is just counting the first-order), so these results are merely an illustration of an interesting area for future research.

Interestingly, Recurrent Neural Networks (RNNs), can be analyzed in similar manner to uncover the abstract patterns learned by the network. RNNs are used for modeling data with important time dimension, such as voice, text and financial time-series. Di Persio & Honchar (2016) employ RNNs to predict stock price, but without focusing on interpretability. RNN interpretability is demonstrated in Karpathy *et al.* (2015) on a language model: the authors are able to identify hidden cells that have interpretable meaning, such as neurons that "specialize" in recognizing passages in quotes or brackets. In the financial domain, Giles *et al.* (1997) find that RNNs learn well-known patterns in exchange rate movements, such as momentum and reversal. While this thesis views stock returns prediction from the cross-sectional perspective, so these time-series methods are not applicable here, this literature offers promise.

Second, it can be studied how the prediction changes in response to the input variables. To the best of my knowledge, there is a single attempt at this in the financial domain: Gu *et al.* (2020) plot the predicted return for different values of 4 selected predictors, each time holding the remaining 92 predictors at 0. In addition to the limited scope (only 4 variables out of 93 are studied), this has the obvious limitation of unrealistic datapoints: in reality, there are no observations with values of 0 for all but one feature; the correlation of data

is also ignored. This thesis offers the marginal relationships of all features, calculated with theoretically well-grounded methodology. This allows to see the typical sign and magnitude of all the predictors in the network.

Third, it can be studied which input variables are more important than others (global feature importance). Two papers are the closest to this thesis in this respect: Gu *et al.* (2020) and Tobek & Hronec (2020). Both study the very same prediction task as this thesis: to predict the return of a stock in the next month, given the characteristics of the stock as of the current month. Also like this thesis, both use neural networks to model the relationship between characteristics and stock returns. And, like this thesis, both calculate the global importance. However, both focus on performance of the networks and admit that their "goal in interpreting machine learning models is modest" (Gu *et al.* 2020, p. 2247). Specifically, the interpretation in both papers is reduced to a single global feature importance figure. This thesis builds on Gu *et al.* (2020) and Tobek & Hronec (2020) by diving fully into the global feature importance, while meeting their high performance. (The thesis uses the very same architecture of the networks as in Gu *et al.* (2020), and dataset from Tobek & Hronec (2020), so the high performance is to be credited wholly to their work.) The contributions in analysis of global feature importance over and above these two papers are several. First, this thesis chooses a feature importance metric that is well-grounded in theory and prior literature (2.4). Second, it suggests and uses another, novel metric which is particularly suited for uncovering why neural networks are good at predicting winners and losers in the stock market. Third, it always compares the interpretation to linear regression, which points to the most important non-linearities. Fourth, it compares the interpretation across networks with different depth, which allows to see of the added complexity changes the model interpretation. Fourth, it interprets not only the ensembles, but their constituent models as well, which further unveils why the ensembles perform so well.

2.4 Choosing Feature Importance Measure

A crucial aspect of interpreting a model is answering the question of why the model gives a particular prediction for a particular observation (this is called *local feature importance*). For example, why does the model predict Apple's return in the following month to be 15%? How do the individual features contribute to this prediction? In other words, how can the prediction be decom-

posed into (*attributed to*) individual features? For this reason, ML literature also calls this task *feature attribution*.

There are several properties that are naturally desirable for any attribution measure to have. First, we would like the measure to give a complete account of the prediction: if we sum the individual feature attributions, we would like to obtain the final prediction (called Completeness or Efficiency axiom in Sundararajan *et al.* (2017) and Molnar (2020) respectively). Second, the measure should give zero attribution score to any feature that is mathematically independent of the predicted variable (called Sensitivity(b) or Dummy axiom in Sundararajan *et al.* (2017) and Molnar (2020) respectively). A related desirable property is that if two observations differ in one feature and have different prediction, the attribution to the differing feature should be non-zero (called Sensitivity(a) axiom in Sundararajan *et al.* (2017)). Third, the measure should be additive: if a model is a linear combination of other models (such as an ensemble model), the attributions of the model should be the same linear combination of the attributions of the individual models (called Linearity or Additivity axiom in Sundararajan *et al.* (2017) and Molnar (2020) respectively). Fourth, if two models with different architectures produce the same predictions for the same set of inputs, the attributions of the two models should be identical (called Implementation Invariance axiom in Sundararajan *et al.* (2017)). As pointed out by Sundararajan *et al.* (2017), it is particularly important in the case of neural networks that Implementation Invariance be upheld: a single functional form of the model can be attained using more than one set of weights due to many degrees of freedom. The training of the network can converge to any of these sets, depending, for instance, on random initialization of weights. However, if the function arrived at is effectively the same, it is desirable that the feature attribution be the same as well. Finally, symmetric features (i.e., such that $f(x, y) = f(y, x) \forall x, y$) should receive the same attributions if $x = y$ for a particular observation (called Symmetry axiom in Shrikumar *et al.* (2017) and Molnar (2020).

Several local feature importance measures are put forward in the literature. **Simple Gradient Method** (Baehrens *et al.* 2010) applies a first order linear approximation of the model to find the sensitivity of the prediction to small changes in values of a given feature. It is a natural starting point for computing feature attribution, as it is very intuitive: if a small change in predictor's value has large effect on prediction, the gradient is large. Other approaches involve back-propagating the final prediction through all the layers down to the input

layer. These include **Guided Back-Propagation** (Springenberg *et al.* 2014), **Layer-Wise Relevance Propagation** (Binder *et al.* 2016) and **DeepLift** (Shrikumar *et al.* 2017). However, all these methods violate the most fundamental axioms, rendering the usage of the methods very problematic. Simple Gradient Method and Guided back-propagation violates Sensitivity(a) and Completeness, while DeepLift and Layer-Wise Relevance Propagation violate Implementation Invariance (Shrikumar *et al.* 2017; Sundararajan *et al.* 2017). The only measures used so far in the domain of stock-returns prediction (in Gu *et al.* (2020), Tobek & Hronec (2020)) are variants of Simple Gradient Method, and thus suffer the same issues.

There are only two methods that satisfy the above axioms: Integrated Gradients (Sundararajan *et al.* 2017) and Shapley Values (Shapley & Shubik 1971).

Integrated Gradients (Sundararajan *et al.* 2017) is a method similar to Simple Gradient Method in that it considers the gradient of the prediction with respect to the feature values. The gradient captures how the prediction changes for small changes in given feature. (Using the gradient is advantageous as it must be already calculated (with respect to weights) during network training.) The measure alters the Simple Gradient Method by considering the gradient not only exactly at the observation, but also at all points on the path from a baseline (usually 0) to the observation at hand. This adjustment is already enough to achieve Completeness and Sensitivity(a) axioms that Simple Gradient Method breaks.

Shapley Values (Shapley & Shubik 1971) originate in game theory and their use in ML is intuitively explained in Molnar (2020): for a single observation, imagine that the feature values enter a room in random order. At each point, all feature values present in the room make a prediction. Shapley value for feature j is the average change in prediction that happens when feature j enters the room. The average here can be taken either across all possible random orders as in Shapley & Shubik (1971) (there are $K!$ of them for K features) or approximated using several samples as in Štrumbelj & Kononenko (2014).

Actually, it can be shown (Sundararajan *et al.* 2017) that the axioms of Completeness, Implementation Invariance, Sensitivity and Linearity are satisfied *only* by a particular group of methods, called *path methods*, which are all very much like Integrated Gradients, but may take other than straight-line path between the baseline and the observation at hand. Within path methods, Integrated Gradients can be shown (Sundararajan *et al.* 2017) to be the only

one that is symmetry preserving. Shapley Values are related to the path methods in that they average over *multiple* paths, instead of just considering a single path. This is unfortunately the reason behind their computational intensity, as discussed below.

Integrated Gradients have one crucial practical advantage over Shapley Values. Even the approximate version of Shapley Value (Štrumbelj & Kononenko 2014) is very computationally intensive: sampling a single random order of features then takes K calls to the network — a prediction is made *each time* a feature "enters the room". So to calculate even the approximate Shapley Value for a single observation, the network must be called $K \cdot M$ times, where $M < K!$ is the number of steps in the approximation. On the other hand, the Integrated Gradients only take M calls to the network. In case of many features and (or) many observations for which we wish to calculate local feature importance, and (or) long prediction times (note that all three are often the case in ML), the difference in computation times is crucial. Even though the number of features is only 30 in this thesis, the difference in computation times is already substantial. Moreover, much more features are to be expected in practice, so using a scalable measure seems vital. Therefore, this thesis uses Integrated Gradients as the feature importance measure. The technical aspects of calculating Integrated Gradients are described in the Data and Methodology.

The methods discussed so far are based on local interpretation, i.e., they allow to decompose the prediction into individual features. In addition to these local measures, there are *global* feature importance measures, which do not allow for such decomposition, but can nonetheless offer insight into the overall workings of the model. In the domain of stock returns prediction, Gu *et al.* (2020) use **Decrease of R Square** (R^2, variance in the target variable explained by the model) that results from setting a particular feature's values to 0. This measure is problematic from two reasons. First, R^2 is notoriously unstable in returns prediction task: the metric fluctuates greatly due to the high ratio of noise to signal, which makes it even less informative than usual. Second, setting feature values to 0 completely denounces the marginal distribution of the feature and makes it flat, which is less realistic than the available alternatives (Fisher *et al.* 2019). **Model Reliance** proposed by (Fisher *et al.* 2019) solves both these disadvantages. It measures how much of model's loss (here, it is the Mean Squared Error) is lost when a feature is permuted. This has the advantage of preserving the marginal distribution of the feature, while making it independent of the prediction. If the particular feature is important, permuting

it will increase the model's loss. Model Reliance of the networks used in this thesis is given in the Appendix. Unfortunately, the Mean Squared Error of networks in the domain of stock return prediction is quite large and noisy, again due to the high ratio of noise to signal, which makes Model Reliance less useful than in other applications (see Fisher *et al.* (2019) and Molnar (2020)). This thesis uses instead a variant of Model Reliance: it replaces the Mean Squared Error by the decrease in return of the implied long-short portfolios (see 3). This has the advantage of using a measure that is stable in the domain of predicting stock returns and of focusing on what the stock-predicting returns do best: predict winners and losers in the stock market. Using a corresponding metric allows to identify the sources of this out-performance. The metric, along with Model Reliance, is further described in Data and Methodology.

Chapter 3

Data and Methodology

3.1 Data

3.1.1 Universe of Stocks

The dataset comprises of liquid publicly traded shares from global developed countries. These include Europe, Japan and Asia-Pacific (Australia, New Zealand, Hong Kong, and Singapore).[1] The data is monthly and spans from 1990 to 2018. It comprises 8,350 companies, totaling 657,141 observations. An average firms is present for 6.5 years (79 monthly observations). The data are rather evenly distributed in time, with 1,888 observations (firms) in an average month.

It is advantageous that the universe of these stocks is highly liquid, because it means that the firms are easily tradable with high volume and low transaction costs. This makes the results of this thesis more realistic (in terms of profitability of the trading strategy) and relevant (in terms of model interpretability), as the findings are not likely to be driven by illiquidity issues and microstructure noise (Asparouhova *et al.* 2010). The liquidity filter is obtained from Tobek & Hronec (2020).

[1]Note that the United States are not included. Research shows there is regional specificity in the data, for example due to different accounting standards in the U.S. (Tobek & Hronec 2020). It is therefore prudent to perform estimations on U.S. and international data separately. Thus, to bring insights about U.S. stock market, the analysis in this thesis could be extended by replication on U.S. data. For further discussion on the value of international vs. U.S. evidence, see Tobek & Hronec (2020).

3.1.2 Predictors

A single observation (in ML terminology, an example), represents a single firm in a given month. An observation consists of 30 predictors (characteristics of that firm, as of the given month, calculated from underlying past data), and the predicted variable, the stock's return, in the next month. All characteristics are calculated by Tobek & Hronec (2020).

Table 3.1 shows all 30 predictors studied in this thesis. According to measures employed in Tobek & Hronec (2020), these variables are the most important predictors of equity returns out of 153 candidate variables from leading financial and accounting journals.[2] For each variable, Table 3.1 gives the bibliographic reference to the original paper that proposed it as equity return predictor, the underlying economic motivation of the variable (discussed in Section 2.2.2). The table also shows that most predictors are calculated using market data (19 of them), rather than the firms' accounting numbers (10 predictors); one predictor is calculated using analysts' expectations from Institutional Brokers Estimate System. Typically, the market variables are updated monthly, and the accounting predictors change values with yearly frequency.

3.1.3 Data Cleaning

Before feeding the data to the neural network, it is crucial to suitably clean them so that they are as informative as possible. As neural networks work best when the values of all features are on the same scale, the data are centered (by subtracting the mean) and then fitted into the range between the values -1 and 1 using their original minima and maxima to define the scale, as in Gu *et al.* (2020). However, this operation would be highly problematic in the presence of outliers: if a feature has observations several times as high (or as small) as the rest of the data, scaling between -1 and 1 would set the outliers' values close to 1 (or -1), while setting the remaining majority of the values close to 0, thus rendering the data completely uninformative. To avoid this, the outliers are treated before scaling.

Outliers are treated using winsorization, which preserves the observation being treated, but clips it to the value of the corresponding percentile (1%

[2]See Figure 8 in Tobek & Hronec (2020). Global liquid universe of stocks is considered. Tobek & Hronec (2020) serve as a pre-selection of important variables for this thesis, which leaves the remaining 123 variables outside the scope of this work. The results in Tobek & Hronec (2020) suggest that these excluded variables are considerably less important, which allows to omit them in this thesis without loosing too much information.

Feature	Data Source	Category	Author	Journal	Frequency
52-Week High	Market	Behavioral	George & Hwang (2004)	JF	M
Short-Term Reversal	Market	Behavioral	Jegadeesh (1990)	JF	M
Idiosyncratic Risk	Market	Attitude to risk	Ang *et al.* (2006)	JF	M
Volume / Market Value of Equity	Market	Illiquidity	Haugen & Baker (1996)	JFE	M
Coefficient of Variation of Share Turnover	Market	Illiquidity Risk	Chordia *et al.* (2001)	JFE	M
Maximum Return	Market	Attitude to risk	Bali *et al.* (2011)	JF	M
Whited-Wu Index	Accounting	Financial Constraints	Whited & Wu (2006)	RFS	M
Coskewness	Market	Attitude to risk	Harvey & Siddique (2000)	JF	M
Operating Profits to Assets	Accounting	Limited attention	Ball *et al.* (2016)	JFE	Y
Lagged Momentum	Market	Behavioral	Novy-Marx (2012)	JFE	M
Liquidity Beta 5	Market	Illiquidity Risk	Acharya & Pedersen (2005)	JFE	M
RD / Market Equity	Accounting	Limited attention	Chan *et al.* (2001)	JF	Y
Seasonality 6-10 A	Market	Seasonality	Heston & Sadka (2008)	JFE	M
Seasonality 11-15 N	Market	Seasonality	Heston & Sadka (2008)	JFE	M
Seasonality 2-5 N	Market	Seasonality	Heston & Sadka (2008)	JFE	M
Momentum-Reversal	Market	Behavioral	Jegadeesh & Titman (1993)	JF	M
Amihud's Measure (Illiquidity)	Market	Illiquidity	Amihud (2002)	JFM	M
Net Operating Assets	Accounting	Limited attention	Hirshleifer *et al.* (2004)	JAE	M
Seasonality 6-10 N	Market	Seasonality	Heston & Sadka (2008)	JFE	M
Seasonality	Market	Seasonality	Heston & Sadka (2008)	JFE	M
Seasonality 2-5 A	Market	Seasonality	Heston & Sadka (2008)	JFE	M
Accruals	Accounting	Limited attention	Sloan (1996)	AR	Y
Duration of Equity	Accounting	Value effect	Dechow *et al.* (2004)	RAS	Y
Change in Common Equity	Accounting	Limited attention	Richardson *et al.* (2006)	AR	Y
Profit Margin	Accounting	Limited attention	Soliman (2008)	AR	Y
Liquidity Beta 3	Market	Illiquidity Risk	Acharya & Pedersen (2005)	JFE	M
Liquidity Shocks	Market	Illiquidity	Bali *et al.* (2013)	RFS	M
Leverage Component of Book/Price	Accounting	Value effect	Penman *et al.* (2007)	JAR	Y
Earnings Predictability	Accounting	Attitude to risk	Francis *et al.* (2004)	AR	Y
Earnings Forecast-to-Price	Analyst Forecasts	Limited attention	Elgers *et al.* (2001)	AR	M

Table 3.1: Predictors

The table is a list of all predictors used in this thesis. Column *Feature* gives the names of the variables, (the naming comes from Tobek & Hronec (2020), which is the source of the dataset used in this thesis). Column *Data Source* informs whether the variable is calculated using the firm's accounting data, market data, or data on forecasts of analysts. Column *Category* presents the categorization of the features by the underlying economic motivation of the variable (discussed in Section 2.2.2). Column *Frequency* shows how often the feature changes values (is updated as new underlying data becomes available): yearly (Y) or monthly (M). All the variables are calculated by Tobek & Hronec (2020), which offer more information on their construction. The *Journal* shortcuts stand for, in alphabetic order: The Accounting Review (AR), Journal of Accounting and Economics (JAE), Journal of Accounting Research (JAR), Journal of Finance (JF), Journal of Financial Economics (JFE), Journal of Financial Markets (JFM), Review of Accounting Studies (RAS), and Review of Financial Studies (RFS).

and 99%). This approach to treating outliers was selected by inspecting the resulting histograms of all variables.

Since all data fed to the neural network must be real numbers, missing observations must be imputed. Here, the missing data are replaced by 0, which is suitable, since the scaling between -1 and 1 allows to interpret the value of 0 as "no signal". The predictors vary considerably as to the percentage of missing data, as documented in Figure 3.1. This is important, as the predictors with many missing values are likely to be less informative, as they offer less space for learning. The variables calculated directly from past returns, such as *52-Week High*, *Short-Term Reversal*, *Idiosyncratic Risk* and *Maximum*, have almost no missing values. On the other hand, the ratio of research and development expenses to market value (*RD / Market Equity*) has more than 60% of values missing. Other variables with many missing values are *Earnings Predictability* and *Earnings Forecast-to-Price* (around 50% and 40%). The rest of variables have typically between 5% and 20% of missing values.

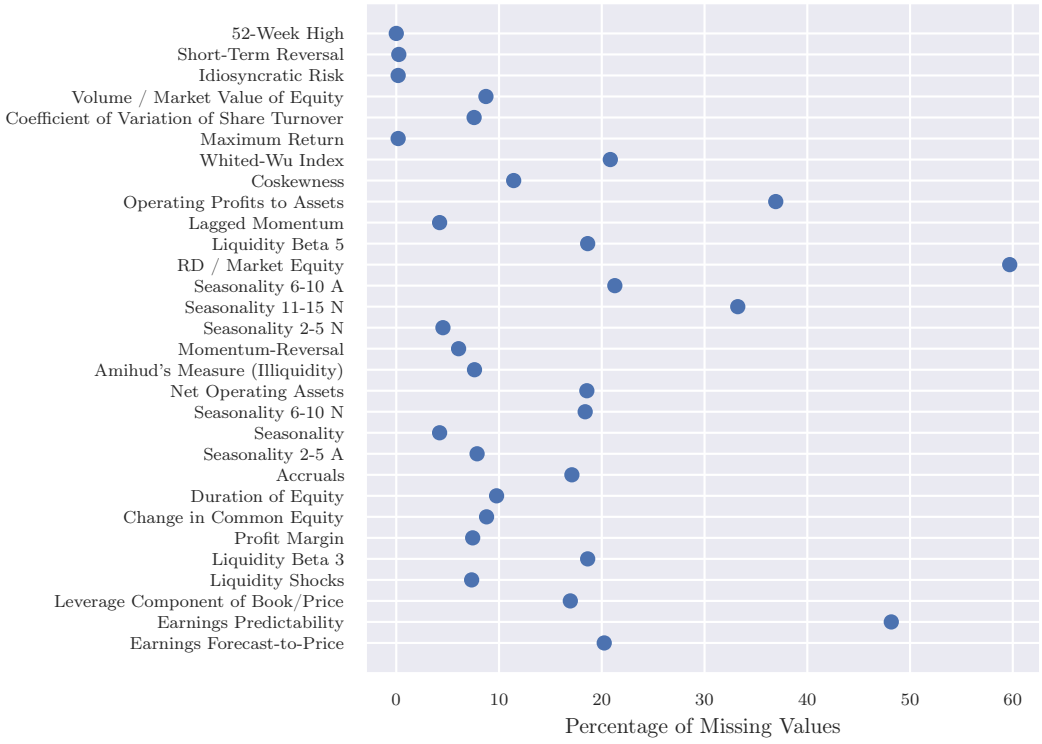


Figure 3.1: Amount of Missing Values in Individual Features

These data-cleaning operations must be employed in a suitable order, which is: imputing missing values by 0, winsorizing, centering and scaling into the

−1 and 1 interval. Of course, all the transformations are performed feature-wise, that is, always considering a single feature at a time. Less obviously, the transformations are performed on data grouped by years, that is, first splitting the data into groups by year, then applying the transformations and finally combining back into single dataset. The main motivation for this is to prevent information from spilling between training, validation and test sets, which are also organized by years.

3.1.4 Predictor Descriptives

This section describes the basic statistical properties of individual predictors after cleaning, i.e., the properties of data entering the neural network. The terms *predictors* and *features* are used interchangeably.

It is particularly important to focus on the standard deviation of the features, as it is crucial for them to be informative of future returns. To appreciate this, consider a degenerate example of a feature with all values equal to zero: since the feature values are the same irrespective of the return, the data offer no room for the model to learn. Thus, features with very low standard deviation may offer too little space for learning and be unimportant as consequence. Figure 3.2 shows the standard deviations of all features. It typically ranges between 0.2 and 0.35, which makes that the within-feature variability large enough for learning. *Whited-Wu Index*, *Liquidity Beta 5*, *Coskewness* and *52-Week High* are the variables with highest dispersion, while *Earnings Predictability*, *Amihud's Measure* and *Liquidity Shocks* are the most narrowly distributed around mean. However, the predictors have rather similar standard deviations overall.

Figure 3.3 shows the correlations of all features. For the purposes of interpretation, correlation is an important aspect of the predictors' distribution. If a pair of predictors is highly correlated, they might substitute each other in the model. As a result, one predictor can appear important and the other unimportant depending on chance. A similar issue is well-known in linear regression as multicollinearity: even though both correlated predictors are important, their statistical significance may be low. Therefore, in order to ascertain importance of the predictors, it is more desirable that they be little correlated. The correlations are shown in Figure 3.3. The diagonal of the matrix is the correlation

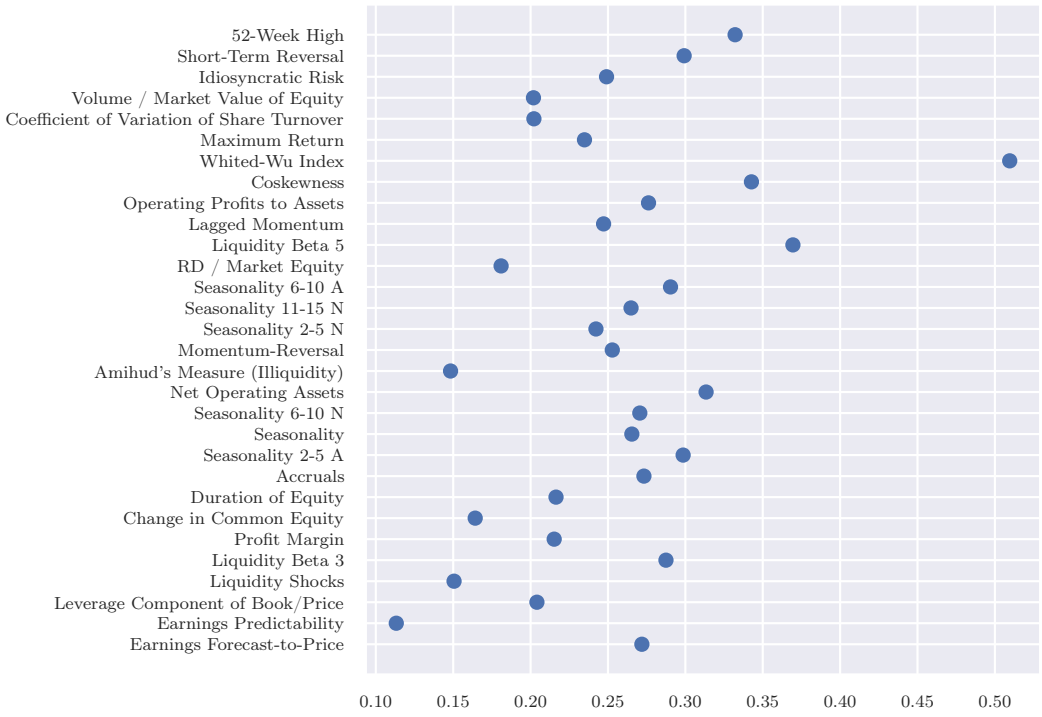


Figure 3.2: Standard Deviation of the Features

of each feature with itself, which is 1 by definition. The rest of the matrix visualizes the pairwise correlation coefficients of all features, where negative (positive) correlations are represented with red (blue) squares. The strength of the correlation is represented both by the hue and size of the squares: the stronger the relationship between two predictors, the darker and larger the square. The numerical values are given in the Appendix A. Overall, the figure shows that the correlation in the data is small: vast majority of correlations is around 0, which is advantageous.

Figure 3.4 shows a smaller part of the correlation matrix, subset so that the 10 most correlated feature pairs are more clearly visible. Panel (a) visualizes the correlation coefficients, while Panel (b) gives their precise numerical values. Looking at the 10 most correlated pairs, it is quite expected that these variables have strong correlation, as they capture similar economic phenomena and are calculated using similar raw variables (see Table 3.1) For example, *Idiosyncratic Risk* and *Maximum Return*, which constitute the most correlated pair, both capture attitude of investors to risk and are calculated using market return. Other examples include *Short-Term Reversal* and *52-Week High* – both capture

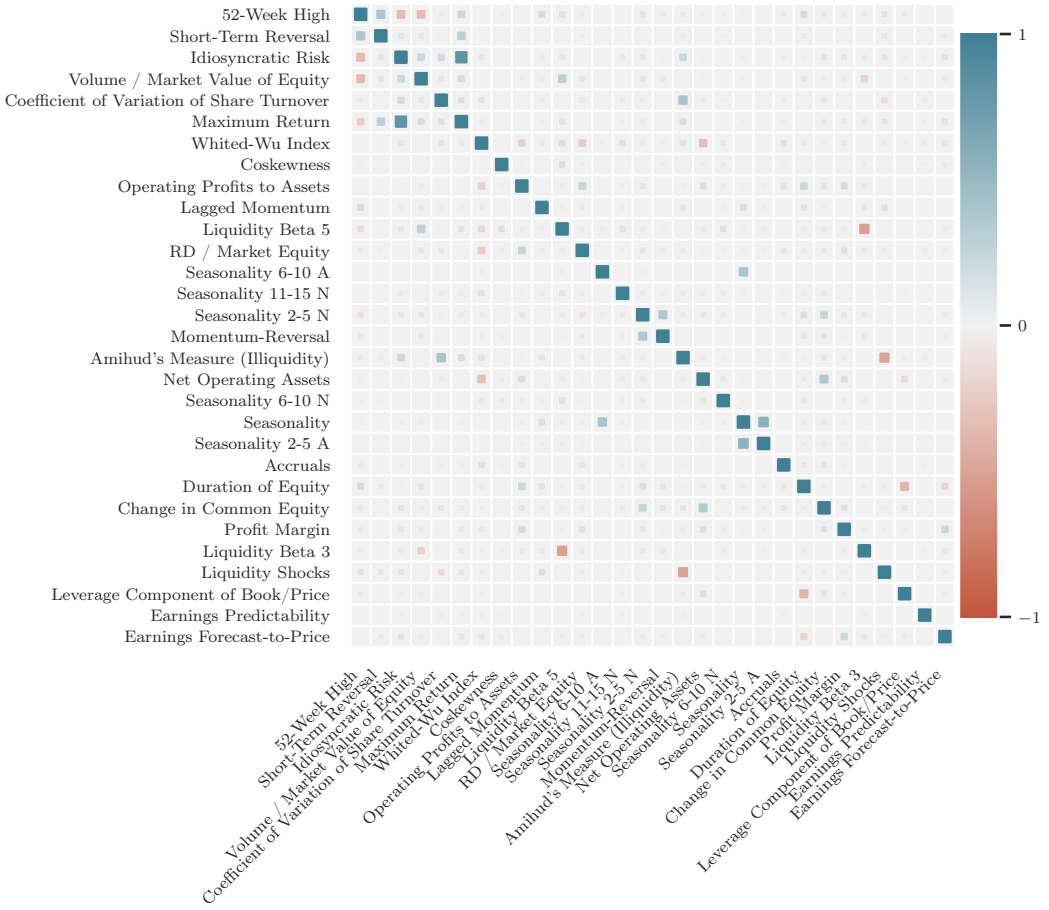
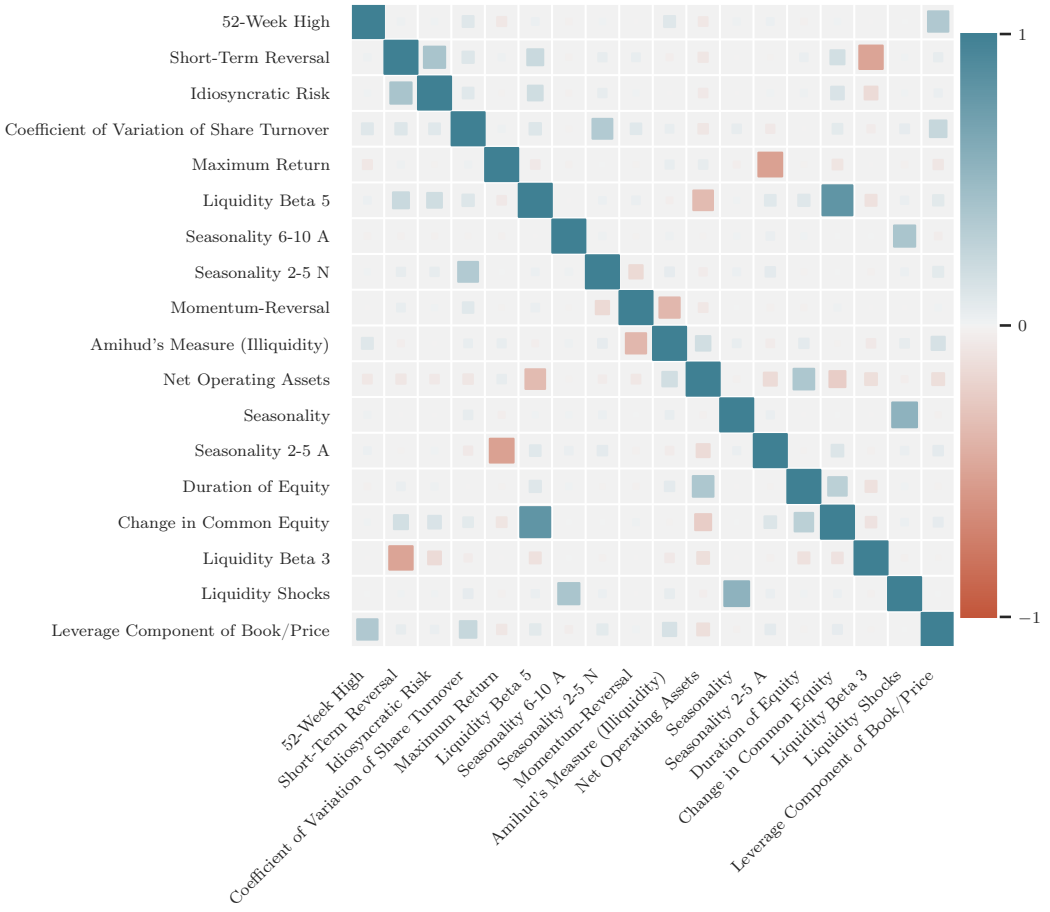


Figure 3.3: Features Correlation Matrix

The figure shows pairwise correlation coefficients of all features. Each square represents the value of the particular correlation coefficient. The value of the coefficient is represented both by size of the square (large absolute values are shown with large squares) and by the square's color (large absolute values have higher intensity, positive correlation is plotted with blue, negative with red).

behavioral biases of investors and are again calculated using past market returns – and *Duration of Equity* and *Leverage Component of Book to Price*, where both variables capture the value effect. To summarize, even though there are some rather correlated variables, their number is very small and their economic interpretation is close, which makes the data quite well-suited for assessing importance of individual economic effects.

More details about distributions of the features are provided in the Appendix A: Figure A.1 shows histograms of all features and Table A.1 lists the mean, standard deviation, minimum, maximum, and 25th, 50th and 75th percentile. As described in the previous section, all the features are scaled to have 0 mean and to fit into the interval -1 and 1. This is clearly visible in the



(a) Correlation Matrix

		Correlation Coefficient
Idiosyncratic Risk	Maximum Return	0.817
Seasonality	Seasonality 2-5 A	0.555
Liquidity Beta 5	Liquidity Beta 3	-0.512
Amihud's Measure (Illiquidity)	Liquidity Shocks	-0.494
Coefficient of Variation of Share Turnover	Amihud's Measure (Illiquidity)	0.405
Seasonality	Seasonality 6-10 A	0.403
Short-Term Reversal	52-Week High	0.382
Momentum-Reversal	Seasonality 2-5 N	0.370
Duration of Equity	Leverage Component of Book/Price	-0.367
Change in Common Equity	Net Operating Assets	0.353

(b) 10 Highest Correlation Coefficients

Figure 3.4: 10 Most Correlated Pairs of Features

Panel (a) is a detail of the correlation matrix (Figure 3.3), subset so as to show 10 most correlated features. Same as in Figure 3.3, each square represents the value of the particular correlation coefficient.The value of the coefficient is represented both by size of the square (large absolute values are shown with large squares) and by the square's color (large absolute values have higher intensity, positive correlation is plotted with blue, negative with red). Panel (b) shows the numerical values of the 10 highest correlation coefficients in the data.

descriptives given in the Appendix.

3.1.5 Descriptives of Predicted Variable

The predicted variable is the return of stock i in a given month. Figure 3.5 shows different aspects of the distribution of monthly returns. Panel (a) shows histogram, boxplot and percentiles, and Panels (b) and (c) summarize the same information numerically. The Figure shows that the monthly stock returns typically range between -6 and 6 percentage points (25th and 75th percentile), and that values outside the range of -25 and 25 percentage points are uncommon. The mean monthly return is small, but positive, around 0.4 percentage points.

3.2 Methodology

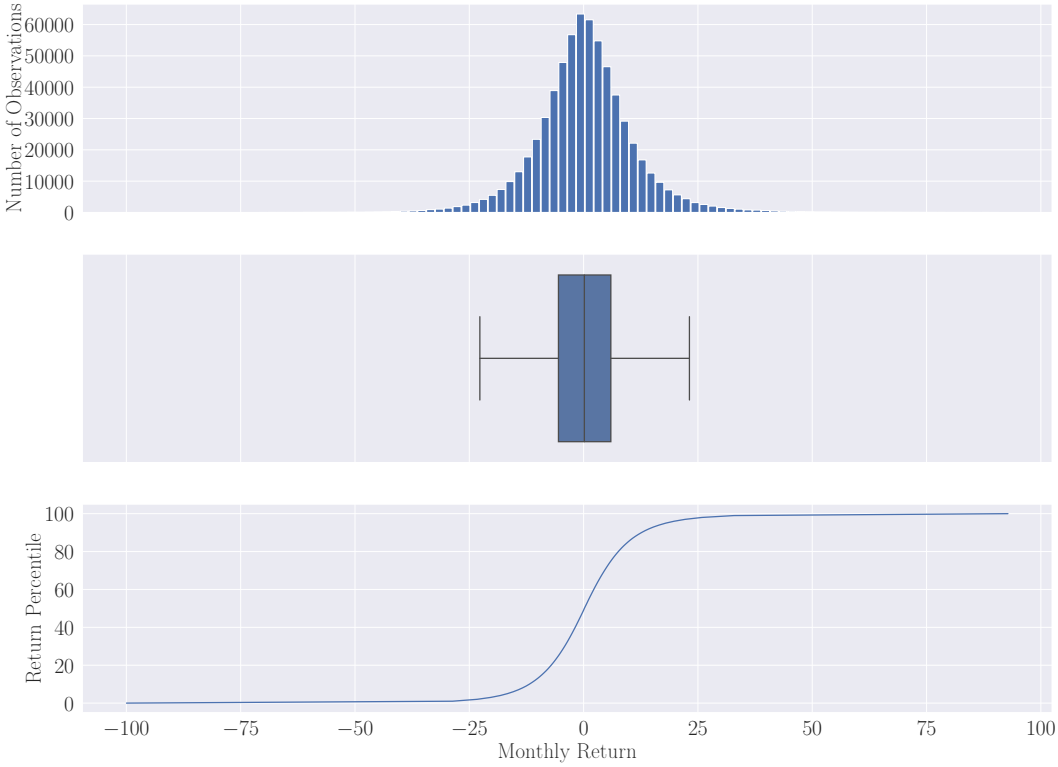
This section describes the methodological issues related to the architecture, training, performance evaluation and interpretation of the neural networks employed in this thesis. The architecture and training of the networks follows closely Gu *et al.* (2020), who study the same problem as this thesis of stock return prediction from characteristics. The predictive performance is also compared to their paper. Finally, construction of the measures used here for neural network interpretation is discussed.

3.2.1 The Prediction Task

The task is to predict stocks' returns in the following month based on a set of the stocks' characteristics calculated as of the current month. In equation:

$$E_t(r^i_{t+1}) = f(x_{1,i,t}, x_{2,i,t}, \ldots, x_{K,i,t}) \tag{3.1}$$

These characteristics are time-variant, for example, if net income is used as a characteristic, its values for a given company change as often as the firm in question issues its financial reports. To produce a return prediction for stock i for the next month $(t+1)$, the currently available characteristics for stock i enter the prediction function f (neural network), which outputs the predicted return. In other words, the K characteristics $x_{1,i,t}, x_{2,i,t}, \ldots, x_{K,i,t}$ are referred to as predictors or, in ML terminology, *features*, and represent the inputs to the neural network, while the predicted return constitutes the network's *output*. As explained in the Literature Review, the network is able to combine the



(a) Histogram, Boxplot and Percentiles

	Mean	Std	Min	25%	50%	75%	Max
Return	0.384	11.193	-99.966	-5.504	0.161	5.955	92.801

(b) Summary Statistics

	10%	20%	30%	40%	50%	60%	70%	80%	90%
Return	-11.862	-7.1	-4.145	-1.844	0.161	2.219	4.572	7.59	12.639

(c) Deciles

Figure 3.5: Descriptive Statistics of Monthly Returns

All three panels show summary statistics of the predicted variable: monthly return (in percentage points). Note that three figures in Panel (a) share the same horizontal axis. As is usual, the boxplot shows the 25% and 75% percentile as the "box", and values two standard deviations from these quantiles as "whiskers". While the histogram and boxplot can be thought of as the sample analogue of probability density, the last figure in Panel (a) can be thought of as sample analogue of cumulative distribution function.

characteristics in a non-linear fashion, creating their interactions and general functional forms, to produce the prediction.

3.2.2 Architecture of the Neural Networks

The neural networks' architecture employed in this thesis follows closely Gu *et al.* (2020), which studies the very same prediction task. In brief summary: all networks are feed-forward with the input dimension 30 and the output dimension 1; models of 4 different depths are used, with 1, 2, 3, and hidden layers consisting of 32, 16, 8, and 4 neurons each respectively; all layers are fully connected, with batch-normalization (Ioffe & Szegedy 2015) and ReLU activations on all hidden layers. As this is a regression problem, there is no activation on the output layer. The weights are regularized using L1 penalty. The following explains and motivates these choices in detail.

First, let us describe a feed-forward neural network in general terms (e.g., Goodfellow *et al.* (2016)). A feed-forward neural network can be represented as directed graph, consisting of several *layers*. An example of such a graph is given in Figure 3.6. The input layer consists of several *neurons* (nodes in the graph). This layer is connected to the next layer of neurons (the first hidden layer) by edges going from each input neuron to each hidden layer neuron. When each neuron of a layer is connected to each neuron in the next layer, we say that the two layers are fully connected. Each edge connecting two neurons is parametrized by a single trainable weight. More hidden layers can be connected to the previous hidden layer in the same fashion. Finally, the last hidden layer is connected, again, fully, to the output layer, which is the final prediction.

This directed graph is a representation of the computation performed by the neural network to get from the input to the output. The values of a given hidden layer, $\boldsymbol{h}$, are computed using the previous layer's values $\boldsymbol{x}$, and the matrix of weights on the edges $\boldsymbol{W}$ using sum of products:

$$\boldsymbol{h} = a(\boldsymbol{W}\boldsymbol{x}), \tag{3.2}$$

or written element-wise, the value of neuron h_i in the hidden layer is computed as

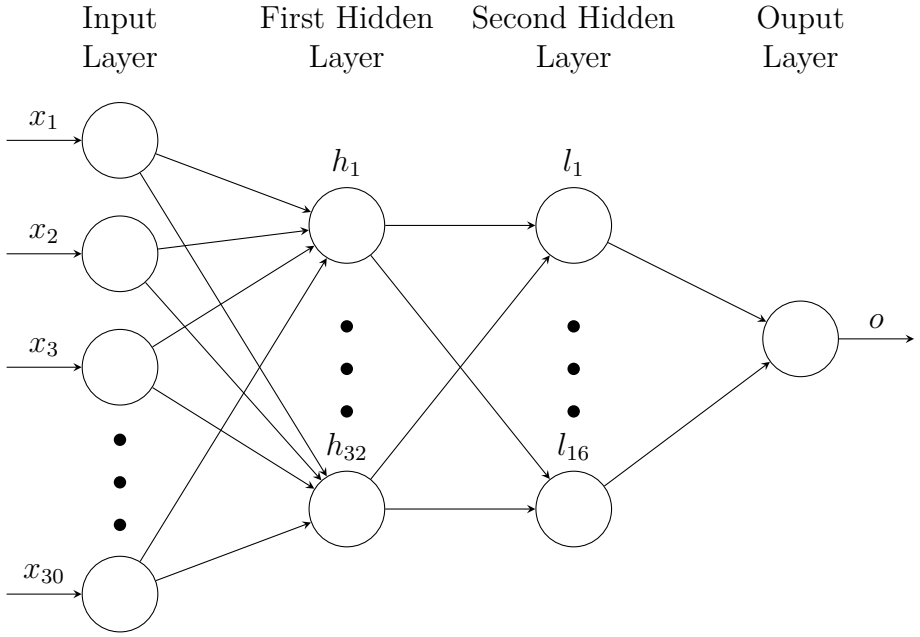


Figure 3.6: Computation Graph of the Neural Network NN2

$$h_i = a\left(\sum_j w_{i,j} x_j\right),$$

where the function a is a non-linear activation function, such as the rectified linear unit (ReLU) used in this thesis:

$$a(z) = \mathrm{ReLU}(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{otherwise.} \end{cases}$$

.

The output layer is computed using the last hidden layer in the same manner, except there is no activation function (in case of a regression problem at least). That is, denoting $\boldsymbol{v}$ the weights on the last edges and $\boldsymbol{l}$ the output of the last hidden layer, the output of the neural network o (here, the prediction of return of stock i at time $t+1$) is

$$o = \sum_j v_j l_j. \tag{3.3}$$

In the case of neural network without hidden layers, the model simplifies to linear regression. Adding a hidden layer, which is a non-linear interaction of the previous layer's neurons, the input features are allowed to interact in any manner. Essentially, a hidden layer represents the input features in a sparser

manner, generating more abstract information from them. This information then enters the next hidden layer and the information is made yet more concentrated. This continues until the output layer, which produces the most high-level information: the prediction. In this way, the network learns to find relationships between the features such that they predict the target (here, the return) well. The network's depth corresponds to the complexity of the model: the model with a single hidden layer can be considered the least complex one, as the inputs only enter one non-linear interaction, and the complexity increases up to 4 hidden layers, where the most abstract or high-level information is extracted.

In this thesis, a neural network takes input of 30 real numbers (the stock's characteristics at given time point) and propagates it through the series of hidden layers to produce the return prediction. The are displayed as the 30 x nodes in the input layer in Figure 3.6. A choice must be made as to the number of hidden layers and number of neurons in each layer. While on optimal architecture can be searched for, here it is sufficient to use architectures from prior literature, as the goal of this thesis is not to devise a new, superior model. Instead, I follow Gu *et al.* (2020) and use the same numbers of layers and neurons. I train models of 4 different depths: 1, 2, 3, and 4 hidden layers. The first hidden layer comprises of 32 neurons. Subsequent layers, if used, contains 16, 8 and 4 neurons for the second, third and fourth layer respectively. Note that the decreasing number of neurons is why we say the information is becoming more and more concentrated the deeper the hidden layer. Throughout the thesis, these 4 distinct neural networks are referred to as NN1, NN2, NN3 and NN4, the number standing for the respective number of hidden layers in the network. Figure 3.6 shows the computation graph of NN2.

In addition, I add a batch-normalization layer (Ioffe & Szegedy 2015) after every hidden layer, again in line with Gu *et al.* (2020). This changes the formula for hidden layer values from 3.2 to to:

$$\boldsymbol{h} = \mathrm{ReLU}(\mathrm{BN}(\boldsymbol{W}\boldsymbol{x})) \tag{3.4}$$

where BN represents the batch-normalization operation. The operation simply normalizes its input data (subtracts sample mean and divides by sample variance). The data is split to batches to improve computing speed of the operation. The operation helps with multiple aspects of training the neural networks, as it prevents the values coming out of a hidden layer from being

extreme. This helps to regularize the network and cuts the training time. Importantly, it helps with the problem of "internal covariate shift", where the distributions of inputs to hidden layers are shifted relative to their validation and testing counterparts, which harms the predictive power and the convergence. By demeaning and standardizing the data, batch normalization preserves the information contained in the hidden layer while preventing the covariate shift (Ioffe & Szegedy 2015).

3.2.3 Training, Regularization and Hyperparameter Tuning

In general, training any neural network amounts to searching for such weights on its edges such that the loss on the training data (typically reflecting the prediction error) is small. The optimization is done numerically, by iteratively adjusting the weights to gradually decrease the resulting loss value. At the beginning of training, the model is initialized with small random weights, where the random state is determined using the so-called random seed. Prediction is computed using these weights, which are then adjusted using the negative gradient of the loss (Kingma & Ba 2014). This is applied iteratively until a stopping criterion is met and the final weights are extracted. This thesis uses mean squared error as the loss.

The data is fed to the network in so-called batches, meaning that several inputs are processed together and the weight update is calculated across the whole batch. When all training data-points have been fed to the network exactly once, we say one epoch has passed. This is repeated for a number of epochs. I use batch size of 5,000, which is enough for processing the data reasonably fast while not overflowing the memory. I use 100 epochs in line with Gu *et al.* (2020), but this number is seldom reached during the training (it works just as an upper limit), due to early stopping, as explained below.

There are several optimizers one can use to descend the slopes of the loss function. In line with Gu *et al.* (2020), I use Adam optimizer (Kingma & Ba 2014). The advantage of Adam is two-fold. First, it essentially lowers the learning rate as the training progresses. The learning rate governs how much the weights are adjusted in a given direction (in the direction of negative gradient of the loss in the simple case of Stochastic Gradient Descent optimizer). Shrinking the learning rate gradually allows faster learning at the beginning of the training and a more nuanced convergence near the optimum.

Neural networks are non-parametric way of modeling the relationship be-

tween the predictors and the predicted variables, in the sense that there is no a priori assumption made about the functional form of the relationship. In fact, the Universal Approximation Theorem shows that with already a single hidden layer, the network outlined above is able to approximate any "well-behaved" function arbitrarily well. This is directly opposed to the linear approach, where we fit the relationship between inputs and outputs assuming a linear functional form.

This necessarily means that neural networks are prone to overfitting the training data. An overfitted model performs well on the training data and poorly on previously unseen data. This is a problem, as the point of having a model in the first place is to be able to draw general conclusions and predict from new data. This is why the models are evaluated solely on held-out data (testing sample), which is disjoint from both training and validation sample, both for purposes of measuring performance and interpreting the models. In addition, there is a number of methods that can be used to prevent overfitting on the training data in the first place, commonly called regularization. The goal is to limit the training process to prevent overfitting while leaving enough room for learning. I use the same regularization techniques as Gu *et al.* (2020), namely, learning rate shrinkage, batch normalization, early stopping, weight regularization, and ensembling. Batch normalization and learning rate shrinkage is discussed in Subsection 3.2.2, ensembling is described in 3.2.4 (it can be considered a regularization technique, as different models fit the training data differently, their average takes out the possible over-fitting of any single model (averages out the overfitting error), and thus is able to generalize better.) The remaining techniques, early stopping and weight regularization, are discussed here:

Early stopping amounts to ceasing the training at some earlier point than reaching the pre-specified number of epochs. To determine that point, one can evaluate the model's predictive power after each epoch. A subsample of the data disjoint from training data (called validation sample) is used for the evaluation. This simulates the out-of-sample performance. When the validation loss increases for k consecutive epochs (k is called *patience*), training is stopped. In line with Gu *et al.* (2020), I use patience of 5.

Weight regularization allows to punish the model for finding too big weights, as large weights are a symptom of overfitting. The size of weights can be measured as their L2 or L1 norm; L1 norm is used here. The strength of the weight regularization is chosen as a hyperparameter on validation data and in

line with Gu *et al.* (2020). The hyperparameter tuning itself is detailed in the following paragraph.

The remaining modeling choices (starting learning rate and strength of L1 regularization) are done using hyperparameter tuning. The hyperparameters discussed so far are not very data-specific, so it suffices to choose them using prior literature (Gu *et al.* (2020) is used as it is closest to this thesis). However, there are parameters that are very data-specific, and must therefore be chosen using the data. Choosing the parameters on the training data would lead to overfitting, which is why they are selected using a sample disjoint from training data, called validation sample. This is called hyperparameter tuning. In this thesis, the learning rate and the strength of L1 regularization are tuned. Each model (each of its 10 random seeds) is run 10 independent times, each time sampling the learning rate and the L1 hyperparameters randomly from pre-specified intervals using logarithmic distribution. The intervals are $[1e{-}3, 1e{-}2]$ and $[1e{-}5, 1e{-}3]$ respectively, again in line with Gu *et al.* (2020). A single best instance (of each random seed) is then selected, as determined by predictive performance of the model on the validation set.

The train-validation-test split is done as follows. There are 29 years of data in the entire dataset, with rather evenly distributed observations from 1990 to 2018. Multiple train-validation-test splits are performed. The first split uses the first 6 years of data are used as training set. The next 4 years are used as the validation set, and the year after that as the test set. The remaining splits are done similarly, each year, the training data is rolled forward using expanding window (that is, taking 7 years from the beginning of the dataset, 8, 9 etc.). Validation set is rolled forward using fixed window (always consisting of 4 years) and starts directly at the end of the corresponding training set, and the test set is rolled forward also using fixed window (always consisting of 1 year) and starts directly at the end of the corresponding validation set. The train-validation-test split can be therefore summarized as 6–4–1, 7–4–1, . . . , 24–4–1, where the last split uses all years in the dataset. Each time a different split is taken, the model is retrained from scratch. This approach reflects that of Gu *et al.* (2020), adjusted appropriately to my shorter dataset (30 years instead of 60) and is used instead of cross-validation so that the temporal ordering of the data is preserved (the model is trained on the past and evaluated on the future, never in the reverse).

3.2.4 Ensembling

Ensembling in general refers to the use of multiple models at the same time to make a single prediction. It is a popular technique in current ML, due to the fact that it tends to diminish the generalization error considerably (improves prediction accuracy out-of-sample) (Zhang & Ma 2012), and stock returns prediction is no exception (Gu *et al.* 2020). The process consists of training several models in the same prediction task and then *assembling* them into a single model: the ensemble prediction is typically a result of individual models majority vote (classification tasks) or average (regression tasks); the latter is the case of this thesis. Intuitively, as the errors of individual models average out, ensembling decreases the variance of the prediction (and thereby increases accuracy).

Since the optimization algorithm of neural networks must be initialized with some small weights (initial values of the parameters being optimized), a natural approach of constructing an ensemble model is to perform the optimization multiple times, each time with a different random initial weights, save all thus optimized models, and finally join them into a single model using average. Implementation-wise, sampling different initial weights is done by setting a different *random seed* at the beginning of the optimization, thus, the term *random seeds* is used as a short way to refer to the multiple instances of the same model with different initial weights.

This is precisely the approach taken in Gu *et al.* (2020) in stock returns prediction task: ensembles are constructed by averaging several random seeds. This thesis also follows this approach, using 10 random seeds for each model. As models with different random seeds can be trained in parallel, this does not place such a burden on the computation time: for example, if a single model needs 2 CPUs (or 1 GPU) to train and the resources available are 10 CPUs (or 5 GPUs), it is possible to parallelize 5 seeds. These numbers are precisely the resources used in training models in this thesis, and it follows that the number 10 was selected to reflect the trade-off between computation intensity and sufficient number of seeds, given these technical constraints.

3.2.5 Model Evaluation

The quality of the models is evaluated using two common approaches. First, the accuracy of the models' predictions is measured using three out-of-sample metrics R^2, root-mean-squared error (RMSE) and mean-absolute error (MAE). Second, long-short portfolios are constructed based on model's out-of-sample

predictions, and the returns of the trading portfolios are studied. Note that the former approach focuses on the *predictability* (whether the model predicts cross-section of stocks well enough), while the latter focuses on the *profitability* of a trading strategy based on the predictions. The latter approach is known as backtesting, and approximates the situation of using the model in financial practice to trade equity. This section describes the methodological details of the two approaches in turn.

Both evaluation approaches are executed on the held-out (testing) sample, so that the network does not see the testing observations while learning. As described in Subsection 3.2.3, each model (NN1 to NN4) is completely re-trained at the beginning of each testing year to make sure that the model is up-to-date with the current state of the world. As this re-training does not start from the year 1 of the dataset (models need more than 1 year of data to train and validate meaningfully), we are left with 19 testing years end of the data for performance evaluation and backtesting: 2000 to 2018, both inclusive. (The model is updated yearly and not monthly due to the high computational intensity of re-training. In other words, the model itself (the weights) remains unchanged for the entire year, but it receives up-to-date data each month. At the beginning of the next testing year, the re-training takes place and the evaluation procedure is repeated.) This is the same approach as in Gu *et al.* (2020); Tobek & Hronec (2020).

Predictive Ability

The predictive accuracy of the models is measured using three metrics R^2, root-mean-squared error (RMSE) and mean-absolute error (MAE). As all the models are evaluated on the test set, whose data were never used during training and hyperparameter tuning, we call all these metrics out-of-sample (OOS), as opposed to in-sample. It is important to use the out-of-sample metrics to be sure that the results are not due to overfitting.

R^2 is here defined slightly differently than usual:

$$R^2_{OOS} = 1 - \frac{\sum_{(i,t)\in T_3} \left(r_{i,t+1} - \hat{r}_{i,t+1}\right)^2}{\sum_{(i,t)\in T_3} r^2_{i,t+1}},$$

where T_3 indicates the testing sample. This is also the definition of R^2 used by Gu *et al.* (2020). As the usual R^2, it measures how much of the variance in the predicted variable (here, return) is explained by the model, but it is distinct

from the usual R^2 in that there is no demeaning in the denominator. This means that instead of comparing the forecasts to the naive forecast of average return, the metric compares predictions to the naive forecast of zero returns. This is because in the task of predicting individual stock returns, forecasts using global average often underperfom those using just zero, so using the usual R^2 definition would be too low a hurdle (Gu *et al.* 2020).

The RMSE is defined in the usual manner:

$$RMSE_{OOS} = \sqrt{\frac{1}{|T_3|} \sum_{(i,t) \in T_3} (r_{i,t+1} - \hat{r}_{i,t+1})^2},$$

and represents the average squared error of the prediction after taking a root of it, which is done so that the final measure is directly comparable in scale to the original returns. Note that it is very similar to the numerator of R^2.

The MAE is defined as usual as well:

$$MAE_{OOS} = \frac{1}{|T_3|} \sum_{(i,t) \in T_3} |r_{i,t+1} - \hat{r}_{i,t+1}|$$

and differs from RMSE just in taking absolute value of the error instead of square, which gives same weight to large errors as to the small ones, as opposed to RMSE where large errors have larger weights (rising with their square).

Profitability of Trading Strategy (Backtest)

Another approach to evaluate the quality of models' predictions is to construct portfolios based on the predicted returns. A the beginning of each month, the network makes prediction of returns for all currently available stocks. The stocks are then sorted on the predicted return and top $l\%$ of them are considered as bought (long position) and bottom $s\%$ are shorted. A typical value for l and s is 10, meaning that the top decile is the long position and the bottom decile the short one. At the end of the month, the true return is revealed. If the model is any good, the true return on the long (short) positions should be high (low), relative to the return on the entire universe of stocks. The performance of the long-short portfolio is then the return on all long positions minus the return on all short positions. In other words, the model should be able to predict winners and loosers and thus generate profits if acted upon. To see if this is the case, the properties of the returns on long-short portfolio can be

evaluated using the usual financial metrics, such as mean, standard deviation, Sharpe Ratio, or Maximum Drawdown.

This approach is generally known as *backtesting* and it approximates the situation of using the model of financial practice to trade equity. Note that since the backtest is performed on held-out (testing) data, which comes from the future that the model has not yet seen, this approximation may actually be considered quite faithful. A limitation, however, is that the backtesting return assumes no transaction costs, such as fees to the broker and shorting costs, which makes the backtesting return artificially slightly higher. Another limitation is that it may may be simply impossible to execute some of the trades in practice due to illiquidity. However, both these limitations can be considered rather small in the case of this thesis, because all the stocks in the dataset are highly liquid and tradable thanks to filters by Tobek & Hronec (2020).

3.2.6 Interpretation

Possibly the most straightforward way of interpreting any model is to show how individual predictors contribute to the prediction, in other words, which explanatory variables matter the most and which are less important. This notion is called *feature importance*.[3] Feature importance can be measured locally or globally. The former approach studies why a particular observation is assigned its prediction: how can the prediction be attributed to individual predictors? The latter explains how the model decides overall, across all observations. This thesis investigates both, using Integrated Gradients (Sundararajan *et al.* 2017) as both local and global measure. The reasons behind choosing this metric as well as its theoretical underpinnings are explained in detail in the Literature Review. This section describes the technical aspects of the measure's calculation.

In addition, this thesis proposes a novel metric, called Portfolio Reliance, to uncover the sources behind the superior ability of networks to form long-short portfolios (predict winners and losers among the stocks in the market). This is in contrast to the Integrated Gradients in that it focuses on the tails of the returns distribution only (prediction of the very high and very low returns), while Integrated Gradients considers the entire distribution. The metric shows how return of the long-short portfolio decreases when a given feature is dis-

[3]There is as of now no existing approach to extract statistical significance in ML, although Fisher *et al.* (2019) provide some development in this direction.

torted. Intuitively, if the portfolio relies heavily on a feature, the long-short return decreases notably when this feature is rendered uninformative. Conversely, if distorting a feature leaves the long-short return intact, the feature can be considered unimportant for the portfolio. The distortion is performed by swapping halves of data for that feature while leaving the rest of the dataset intact, which renders the feature completely uninformative of return while it does *not* change its marginal distribution. Preserving the marginal distribution is an advantage, since the values are still reasonable, just uninformative. This distortion method is developed by Fisher *et al.* (2019) in their Model Reliance measure.[4]

For purposes of completeness, Model Reliance is calculated as an additional global feature importance metric. However, since it measures mean squared error decreases, which are very noisy in this prediction task, it is much less informative than Integrated Gradients. Nevertheless, the methodology is described here in detail as well, as it forms the basis for Portfolio Reliance method.

(Local) Integrated Gradients

Integrated Gradients (Sundararajan *et al.* 2017) is a local measure of feature importance, i.e., it is computed for each observation separately. It considers the gradient of the prediction with respect to the predictors. If a small change in a predictor's value has large effect on the prediction, the value of the gradient is large.

Again, denote $f : \mathbb{R}^k \to \mathbb{R}$ the prediction function (here, the NN model). Further, denote $\boldsymbol{z} \in \mathbb{R}^k$ a single input with k scalar elements, where k denotes the total number of features (here, 30). Let $\boldsymbol{z'} \in \mathbb{R}^k$ be a *baseline input*, an observation that can be considered as a point from which the other observations depart. (For example, in case of inputs being images, the baseline can be a black image. In this thesis, it is a vector of zeros, as motivated below.)

For simplicity of the notation, assume that $f : \mathbb{R}^k \to [0, 1]$. The Integrated Gradient of i^{th} feature of the input $\boldsymbol{z}$ is defined as

$$IG_i(\boldsymbol{z}) := (z_i - z_i') \int_{\alpha=0}^{1} \frac{\partial f(\boldsymbol{z'} + \alpha(\boldsymbol{z} - \boldsymbol{z'}))}{\partial z_i} d\alpha,$$

where $\frac{\partial f(z)}{\partial z_i}$ stands for the gradient of $f(\boldsymbol{z})$ along the i^{th} dimension. This means that the measure considers a straight path in $\mathbb{R}^k$ from the baseline input

[4]Model Reliance, like Integrated Gradients, focuses on the entire distribution of returns, and is therefore likewise unsuitable for the purpose of focusing on the tails.

$\boldsymbol{z'}$ to the given input $\boldsymbol{z}$ and calculates the gradient of prediction at all points along that path. The Integrated Gradient measure cumulates these gradients by taking the integral across the path.

To calculate the Integrated Gradient of i^{th} feature of the input $\boldsymbol{z}$ empirically, it is necessary to approximate the integral by summation across several points along the straightline path (Sundararajan *et al.* 2017). That is:

$$IG_i^{\text{approx}}(\boldsymbol{z}) := (z_i - z_i') \sum_{k=1}^{m} \frac{\partial f(\boldsymbol{z'} + \frac{k}{m}(\boldsymbol{z} - \boldsymbol{z'}))}{\partial z_i} \frac{1}{m},$$

where m, called the *step size*, is the number of the steps in the Reimman approximation of the integral, here, the number of points along the straightline path at which evaluate the gradient.

Two implementation decisions must be made when calculating the Integrated Gradients: first, choosing the right baseline input, and second, choosing the right the step size. First, the baseline input should be chosen such that it can be interpreted as "no signal". In this thesis, there are two possible baselines: random noise and all-zero input. The latter was chosen, as it lends itself well to the "no signal" interpretation: all the features are normalized between -1 and 1 and have a mean of 0, thus, an all-zero input can be considered a natural point of departure. Additionally, the authors recommend to check that the model's prediction at baseline is around zero. This is important because it allows to use Integrated Gradient to explain the prediction as a function of the input, and not as of the input and the baseline. Indeed, the predictions for the all-zero input are around zero, which confirms that the all-zero input is the correct baseline here. Second, the step size should be chosen such that the approximation is sufficiently accurate: the authors recommend to check that the attributions approximately sum up to the difference between the prediction at $\boldsymbol{z}$ and at $\boldsymbol{z'}$. Given that the latter is around zero, it remains to check that the attributions for $\boldsymbol{z}$ sum up to the prediction at $\boldsymbol{z}$ (note that this is a verification of passing the Completeness Axiom, see Literature Review). This thesis uses the step size of 50, which passes this summation check comfortably.

Global Integrated Gradients

Integrated Gradients, which is a local measure of feature importance by nature, are also used in this thesis as a global measure. (Recall that local measures answer the question of why was a particular prediction made, while global measures answer the question of which features are important for the model

overall.) The measure is turned global simply by averaging across all predictions. However, as both large negative and large positive local effects mean that the feature is important, it is necessary to take the average out of *absolute* local values. The measure then says what is the average absolute size of the contribution of a feature to the predicted return. Thus, the features with large Global Integrated Gradient can be considered important for the model overall and used as a global feature importance measure.

Model Reliance

Model Reliance (Fisher *et al.* 2019) is a global measure of feature importance. It is based on the idea that if a feature is important for making prediction, then distorting the feature by adding noise to it will damage prediction performance. Conversely, if distorting a feature leaves the prediction performance relatively intact, that feature can be considered unimportant for the prediction. Thus, denoting the prediction function (such as an ML model) f, Model Reliance of f on random variable X can be *informally* defined as

$$MR(f) := \frac{\text{Expected loss of } f \text{ under noise}}{\text{Expected loss of } f \text{ without noise}},$$

where the noise renders the random variable X completely uninformative of the predicted variable while it at the same time does *not* change the marginal distribution of X. More specifically, denote the predicted random variable as Y and consider X_1 and X_2 as two explanatory random variables. The expected loss of f can be then denoted as

$$e_{\text{orig}}(f) := \mathbb{E}L(f, (Y, X_1, X_2)).$$

Further denote X_1^s as random variable following the same marginal distribution as X_1, but independent of Y.

Then the expected loss of f under noise, which distorts X_1 to X_1^s, can be denoted as

$$e_{\text{switch}}(f) := \mathbb{E}L(f, (Y, X_1^s, X_2)).$$

Finally, Model Reliance can be *formally* defined as

$$MR(f) := \frac{e_{\text{switch}}(f)}{e_{\text{orig}}(f)}.$$

To see the intuition behind this formula, consider first the case that X is very informative of Y. Then, e_{switch} is much larger than e_{orig} and Model Reliance is larger than 1. The more X is informative of Y, the larger the Model Reliance of f on X — the more the model *relies* on the feature to make the prediction. In the case when Model Reliance is exactly 1, the feature can be considered unimportant, as completely distorting it does not change the model's loss. Model Reliance can also be less than one, in the case that the distorted feature actually perform better than the original feature.

The sample estimate of $e_{\text{orig}}(f)$ is the loss used in the model's training. Denote the target as $\boldsymbol{y} \in \mathbb{R}^N$ and consider two features $\boldsymbol{x_1}, \boldsymbol{x_2} \in \mathbb{R}^N$:

$$\hat{e}_{\text{orig}}(f) := \frac{1}{N} \sum_{i=1}^{N} L\left(f, (y_i, x_{1_i}, x_{2_i})\right).$$

The sample estimate of $e_{\text{switch}}(f)$ can be done in two different ways, either by randomly permuting the values of the given feature or by dividing the feature's values in halves and then switching the halves. This thesis uses the latter, as it is computationally less demanding. (In both cases, the rest of the features and the predicted variables are intact.) It follows that the sample estimate of $e_{\text{switch}}(f)$ can be calculated as

$$\begin{aligned}\hat{e}_{\text{switch}}(f) :=& \frac{1}{2\lfloor N/2 \rfloor} \sum_{i=1}^{\lfloor N/2 \rfloor} L\left(f, \left(y_i, x_{1_{i+\lfloor N/2 \rfloor}}, x_{2_i}\right)\right) + \\ &+ L\left(f, \left(y_{i+\lfloor N/2 \rfloor}, x_{1_i}, x_{2_{i+\lfloor N/2 \rfloor}}\right)\right).\end{aligned}$$

This is simply the loss of the prediction using the same data as original, but with values in the feature $\boldsymbol{x_1}$ split in half and the halves swapped, leaving the other features (here, $\boldsymbol{x_2}$) and the predicted variable y unchanged. (If N is even, the split in half can be performed exactly, and if N is odd, the last observation in the dataset is not used and the split is then performed in the same way as if N is even.) The extension to the case with more than two features is straightforward: when calculating Model Reliance of feature $\boldsymbol{x_1}$, we disturbed the values in $\boldsymbol{x}_1$, leaving $\boldsymbol{y}$ and $\boldsymbol{x_2}$ unchanged. If we additionally have, say, $\boldsymbol{x_3}$ and $\boldsymbol{x_4}$, they are treated in the same way as $\boldsymbol{x_2}$, that is, unchanged.

Finally, the sample estimate of Model Reliance on a given feature is:

$$\widehat{MR}(f) := \frac{\hat{e}_{\text{switch}}(f)}{\hat{e}_{\text{orig}}(f)}.$$

That is, when calculating reliance of the model on a given feature, two predictions are made using the model: the usual one, with all features as well as the target undisturbed, and the other one, identical in all aspects except for the disturbance in the assessed feature. The loss of both predictions is then computed as usual. Model Reliance is then the ratio of these two losses, the disturbed loss divided by the undisturbed one.

Portfolio Reliance

Portfolio Reliance, metric proposed by this thesis, is calculated in the same manner as Model Reliance, but the metric used to evaluate the difference in the performance is not the model's loss, but the change in the mean return on the long-short portfolios generated by the networks. The metric preserves the idea of permuting the feature by swapping its halves and seeing how the performance is worsened:

$$\widehat{PR}(f) := \hat{r}_{\text{orig}}(f) - \hat{r}_{\text{switch}}(f).$$

Note that the metric is not a ratio, but a difference, which allows to interpret the metric directly as the decrease in the profitability of the trading strategy.

This means that $\hat{e}_{\text{orig}}$ and $\hat{e}_{\text{switch}}$ are calculated slightly differently, denote them $\hat{r}_{\text{orig}}$ and $\hat{r}_{\text{switch}}$:

$$\hat{r}_{\text{orig}}(f) := \frac{1}{N} \sum_{i \in L_{orig}} y_i - \frac{1}{N} \sum_{i \in S_{orig}} y_i,$$

$$\hat{r}_{\text{switch}}(f) := \frac{1}{N} \sum_{i \in L_{switch}} y_i - \frac{1}{N} \sum_{i \in S_{switch}} y_i,$$

where L_{orig} (S_{orig}) is the long (short) portfolio constructed using predictions from the undisturbed data and L_{switch} (S_{switch}) is the long (short) portfolio constructed using predictions from the disturbed data. The data distortion is done in the same manner as in Model Reliance, i.e., values of the particular features are swapped across halves, which makes the feature uninformative of the return while preserving its marginal distribution. The portfolios are constructed as described in 3.2.5, using top 10% of stocks for long and bottom 10% for short, as ordered by out-of-sample predictions.

Chapter 4

Results

This chapter presents the results in two distinct halves. First, Section 4.1.1 reviews shows how the performance of the networks compares to the state of the art. Both accuracy of the predictions and ability to form profitable long-short portfolios used for trading is considered. This assessment forms a necessary background for the main analysis: the neural network interpretation, which is presented in the second half, Sections 4.2 and 4.3. While the first half merely establishes that the performance of the networks is comparable to prior literature, the second half presents the original contribution of this thesis.

4.1 Performance Evaluation

4.1.1 Predictive Ability

Table 4.1a shows the predictive ability of the networks. All four nets (NN1 to NN4) are rather similar in the mean errors of their predictions. The Mean Absolute Error is around 0.075, which is a rather high portion of the variability in monthly return (mean monthly return is 0.005 with 0.051 standard deviation). However, these relatively high prediction errors are a commonplace in stock returns prediction and are due to the low signal-to noise ratio of financial data in general. The R^2 shows the fraction of variability in data explained by the model, above that of the naive forecast of zero return. The values range from 0.022% for NN4 across 0.039% for NN3 to 0.21% for NN1 and finally 0.25% for NN2, which is a commonplace in the stock return prediction task, albeit somewhat lower than in Gu *et al.* (2020), reporting R^2 of around 0.35%.

Figure 4.1b shows a decomposition of R^2 into deciles by return: the stocks are grouped into 10 groups, corresponding to the decile of their return, and R^2

is calculated separately for each decile. The results show that the predictive ability of the networks is higher the higher the return: put simply, the networks are good at predicting returns of stocks that end up performing very well and bad at predicting mediocre returns. In the top 10% of stocks (decile 100), the networks explain as much as 0.5% to 1% of returns variability above naive forecast of 0. The performance is similarly good in the 80th and the 90th decile, but seems to be decreasing. For the 60th, 50th, 40th decile, the performance ranges around 0, meaning that the models explain as much variance as the naive forecast of 0 around the center of the returns distribution. For 30th decile, all models actually under-perform the naive 0 forecast (negative values of R^2). Some models (NN1 and NN3) seem to improve in the lowest decile, (barely) climbing out of the negative R^2 territory. This generally indicates that the models may be better at predicting returns of winners than mediocre and poorly-performing stocks, at least in terms of the explained variance of the data.

4.1.2 Profitability of Trading Strategy (Backtest)

Table 4.3a Figure 4.3b and describe the distribution of monthly returns earned on long-short portfolios generated by the individual networks. As described in 3.2.5, the portfolios are constructed by letting a network make out-of-sample prediction of returns in the next month, ordering the stocks by predicted return from highest to lowest, buying the top 10% of stocks and short-selling the bottom 10%. The figures also show the market return, which serves as a benchmark. All three figures show that all the networks beat the market returns by a large margin, which gives a clear sign of their ability to identify winners and losers ex ante, resulting in a profitable long-short trading strategy. The two figures are now discussed in turn in more detail.

Table 4.3a shows the descriptive statistics of monthly returns on the long-short portfolio. While the market earns 0.5% in an average months, the networks earn more than double: 0.9% for the worst-performing net (NN3) and 1.4% for the best networks (NN1 and NN4). This translates to yearly return of 12, 15, 18 and 19% (in NN3, NN2, NN4 and NN1 respectively), which comfortably beats the 7% earned by the market. While beating the market in average return, the networks exhibit *less* return volatility at the same time: the standard deviation of the market return is 0.05, while the networks' is around 0.04. This favorable risk-return trade-off is summarized in Sharpe Ratio of around 1

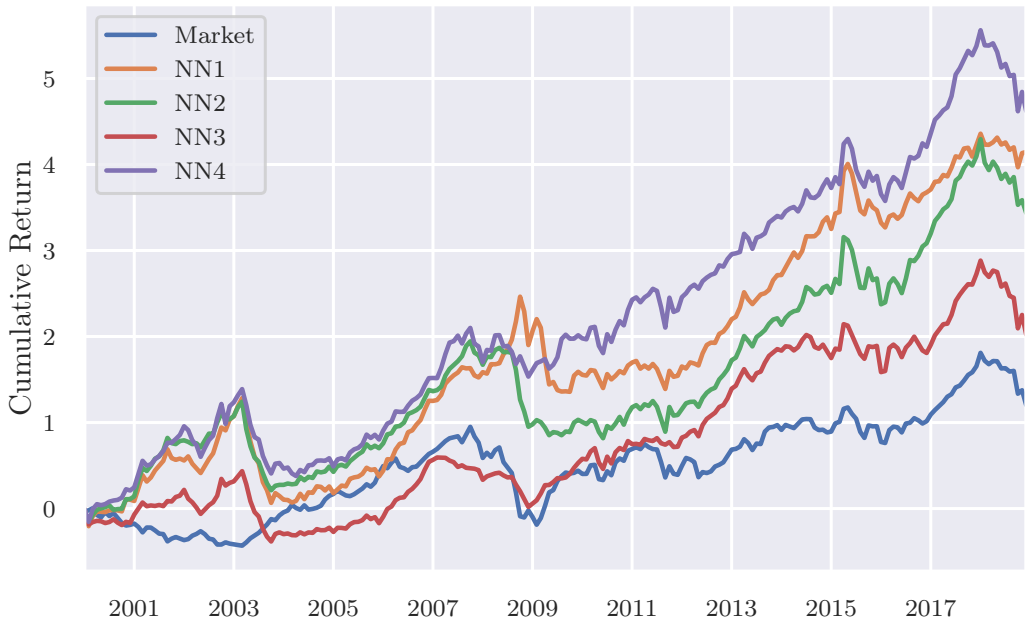
	NN1	NN2	NN3	NN4
Mean Absolute Error	0.075	0.077	0.078	0.075
Mean Squared Error	0.016	0.018	0.019	0.020
Root Mean Squared Error	0.105	0.106	0.108	0.107
R Square	0.212	0.252	0.039	0.022

(a) All Out-of-Sample Measures of Predictive Ability

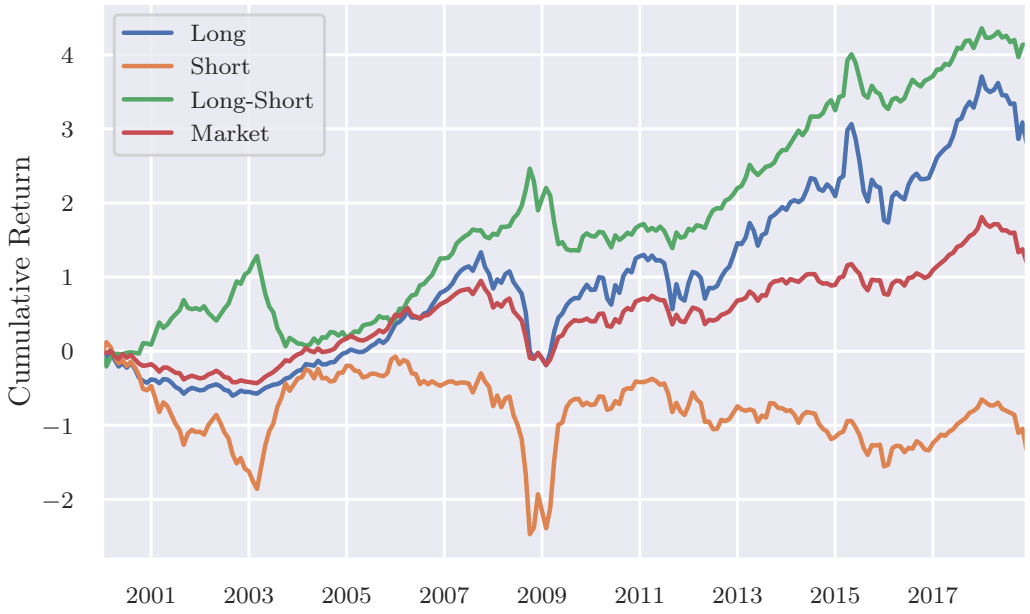
(b) Decomposition of Out-of-Sample R^2 into Deciles

Figure 4.1: Out-of-Sample Predictive Ability of the Networks

NN1 to NN4 stand for neural networks of respective depths. Panel (a) shows predictive ability of all models. All metrics are calculated out-of-sample and are defined in Section 3.2.5. R^2 is in percentage points. Panel (b) decomposes the last row in Panel (a), R^2, into deciles. Specifically, out-of-sample predictions of each network are split into 10 subsets based on decile of the actual return. R^2 of the prediction is then calculated on the individual subsets. For example, decile 10 (100) denotes the bottom (top) 10% of observations by return (all deciles are mutually exclusive by construction). The figure shows that all networks are able to predict high returns well (decile 80, 90 and 100), but they under-perform the naive prediction of 0 (negative R^2) in the low to medium return deciles (30 to 60).



(a) Cumulative Returns on the Long-Short Portfolios



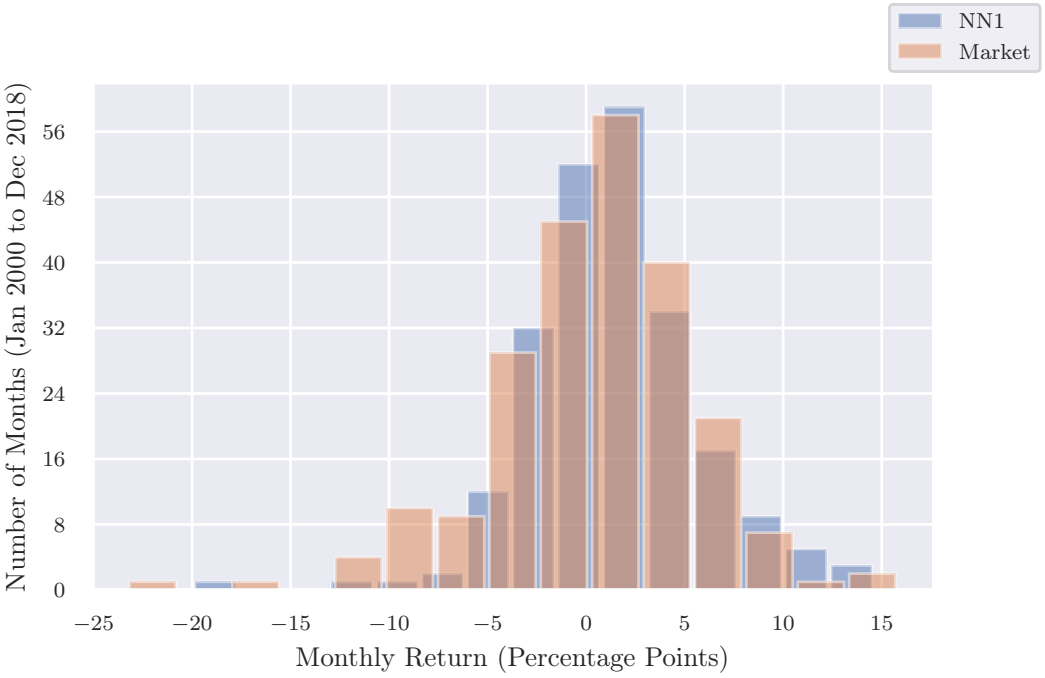
(b) Long and Short Legs of the Portfolio (NN1)

Figure 4.2: Cumulative Returns on the Long-Short Portfolio

NN1 to NN4 stand for neural networks of respective depths. As described in 3.2.5, the portfolios are constructed by letting each network make out-of-sample prediction of returns in the next month, ordering the stocks by predicted return from highest to lowest, buying the top 10% of stocks and short-selling the bottom 10%. The Market portfolio is constructed by buying the entire universe of stocks available in the given month. All portfolios are equal-weighted. Panel (a) shows the long-short portfolios of all models, while Panel (b) decomposes the long-short return of NN1 into the long and short legs.

	Market	NN1	NN2	NN3	NN4
Mean	0.005	0.014	0.012	0.009	0.014
Mean (Yearly)	0.072	0.191	0.153	0.122	0.183
Standard Deviation	0.051	0.043	0.039	0.042	0.040
Sharpe Ratio	0.325	1.160	1.047	0.712	1.229
Skewness	-0.633	-0.148	-0.133	0.571	0.574
Kurtosis	2.118	2.640	2.504	2.850	3.020
Max Drawdown	-0.584	-0.225	-0.324	-0.272	-0.172

(a) Descriptive Statistics of Returns on Long-Short Portfolios



(b) Histogram of Monthly Returns on the Long-Short Portfolio Generated by NN1

Figure 4.3: Descriptive Statistics of the Returns on the Long-Short Portfolios

As described in 3.2.5, the portfolios are constructed by letting each network make out-of-sample prediction of returns in the next month, ordering the stocks by predicted return from highest to lowest, buying the top 10% of stocks and short-selling the bottom 10%. The Market portfolio is constructed by buying the entire universe of stocks available in the given month. All portfolios are equal-weighted. Panel (a) describes the distribution of the returns earn on the long-short portfolios based on the networks' out-of-sample predictions. Panel (b) visualizes the same information in a histogram, focusing on NN1 and Market return.

(compared to market's 0.3). This gives very similar performance to the prior literature: compare the Sharpe Ratios to Gu *et al.* (2020) (in brackets): 1.16 (1.16) for NN1, 1.05 (1.15) for NN2 and 1.23 (1.35) for NN4. The only exception is NN3, which has a considerably poorer Sharpe Ratio of 0.7 (1.20). Tobek & Hronec (2020) report Sharpe Ratios of the networks from 0.88 to 1.58, which further confirms that the performance of the networks in this thesis is state-of-the-art. The remaining rows in Table 4.3a summarize the skewness and kurtosis of the distributions of monthly returns: NN1 and NN3 are skewed to the right, NN3 and NN4 to the left, and the tails are generally only somewhat lighter than that of standard normal distribution for all models (kurtosis from 2.5 to 3.0). Maximum Drawdown shows the loss a trader would suffer if she started trading in 2000 and exited in the least favorable moment possible: again, the networks beat the market (-0.27 to -0.32 compared to markets' -0.58), and are close to state-of-the-art (-0.15 to -0.26 in Gu *et al.* (2020) and -0.19 to -0.38 in Tobek & Hronec (2020).)

Figure 4.3b further illustrates the more favorable mean, standard deviation, skewness and kurtosis of the networks' return distributions than that of market's: the mean (and also all the deciles) are more to the right, the left tail is less pronounced and the right tail is also more favorable (meaning that investors experience less terrible months and more great months than when investing to the market). The somewhat lower standard deviation is also visible in the figure. Most months see a return of 0 to 2.5%.

Figure 4.2a shows the returns on the very same portfolios from a time-series perspective: it shows cumulative return earned at any time point since 2000 (beginning of the out-of-sample data). Again, market return is plotted as a benchmark. All networks consistently outperform the market (again, with a slight exception of the rather poorly performing NN3 in the early years). If trading from 2000 and exiting in 2018, an investor would pocket around 400% of the initial investment (200% from the poorly performing NN3 and 120% for the market). The recessions are also visible: 2003, 2008 and 2018 saw negative returns, showing as decreasing cumulative return in the figure.

Figure 4.2b shows the decomposition of the cumulative return on the long-short portfolio[1] to its long and short legs. The figure also shows the market return, which serves as a benchmark. IT shows that the short-sold stocks are consistently below the market and in negative numbers, which translates to profits for the short position, while the long position is at par or better than

[1]The portfolio is generated by NN1. Other architectures give similar results.

the market. The short and long position combined result in a consistent outperformance. Also note that the short position helps to protect the portfolio in the recessions: The network was able to predict the losers if 2002 and 2008 recessions, which meant that the return of the long-short portfolio was partially insulated from the economic downturn.

All the backtesting performance described to this point assumes that the long (short) position is created as top (bottom) 10% of the stocks, as ordered by predicted return. Table 4.1 describes how the results change if the focus of the positions would be more or less narrow (resulting in lower or higher number of firms in both positions: 8, 17, 88, 177, 177, and 355 from right to left).[2] The results reviewed until now are in column *10–10* (meaning 10% of the market is bought and 10% short-sold). Column *20–20* represents the situation if the positions would be distributed into more firms (broader portfolios) and columns *5–5*, *1–1* and *0.5–0.5* represent progressively narrower positions (less firms bought and sold). As is expected, the mean returns are higher for the narrower portfolios (as much as 44% yearly for 1–1 allocation), but this comes at the cost of higher standard deviation (more volatile returns) and worse Maximum Drawdown. It appears that the optimum risk-return trade-off is with 5–5 allocation, which gives the highest Sharpe Ratio of 1.2. While all variants still beat the market in mean return, only 10–10 and 20–20 allocations are less volatile than the market, which is because there are too little firms in the smaller portfolios to ensure a decrease in volatility (e.g., 17 firms in the long and 17 in the short position for the 1–1 allocation).

4.2 Local Feature Importance And Sign of Effects

Panel (a) of Figure 4.4 illustrates how individual predictions of the networks can be interpreted. Particularly, any single prediction can be decomposed in an additive manner into the effects of individual features. The measure used for this decomposition is called Integrated Gradient; its use its motivated in Section 2.4 and the methodological details are discussed in Section 3.2.6. The Panel shows the values of Integrated Gradient for a single out-of-sample prediction of NN1. The predicted return is the sum of the values of Integrated Gradient across features. In other words, the Integrated Gradient of a single feature is exactly the change in the prediction (return) that is due to (or attributed to)

[2]The portfolio is generated by NN1. Other architectures give similar results.

	Market	0.5-0.5	1-1	5-5	10-10	20-20
Mean	0.005	0.025	0.028	0.019	0.014	0.011
Mean (Yearly)	0.072	0.408	0.440	0.262	0.191	0.145
Standard Deviation	0.051	0.111	0.082	0.053	0.043	0.033
Sharpe Ratio	0.325	0.773	1.196	1.226	1.160	1.171
Skewness	-0.633	-0.085	-0.144	-0.192	-0.148	-0.215
Kurtosis	2.118	0.262	0.982	2.273	2.640	4.265
Max Drawdown	-0.584	-0.826	-0.567	-0.263	-0.225	-0.181

Table 4.1: Descriptive Statistics of Returns on Long-Short Portfolios Generated by NN1 in Different Capital Allocations

As described in 3.2.5, the NN1 portfolio is constructed by letting a network make out-of-sample prediction of returns in the next month, ordering the stocks by predicted return from highest to lowest, buying the top $m\%$ of stocks and short-selling the bottom $n\%$. Here, $m = n$ and takes values 0.5, 1, 5, 10 and 20, which gives respectively 8, 17, 88, 177, 177, and 355 firms in the long postiion and the same number of firms in the short position. All portfolios are equal-weighted.

that feature. For example, in the case of the particular prediction analyzed in Panel (a), *Whited-Wu Index* increases the predicted return by 0.4 percentage points, *52-Week High* decreases it by 0.4 percentage points (i.e., back to 0), *Earnings Forecast-to-Price* decrease it further by 0.1 percentage points, etc. for all features.

Panel (b) of the same figure shows the same, but for all out-of-sample predictions of NN1. For each feature, the boxplot shows the distribution of the values of Integrated Gradient across predictions. A striking pattern emerges: some features clearly have more influence on the predictions than others. The most influential features appear on the top and include *Whited-Wu Index*, *52-Week High*, and *Earnings Forecast-to-Price*, while the least influential features appear on the bottom, and their influence on the predicted return is usually 0. The next section, 4.3, investigates this notion in detail.

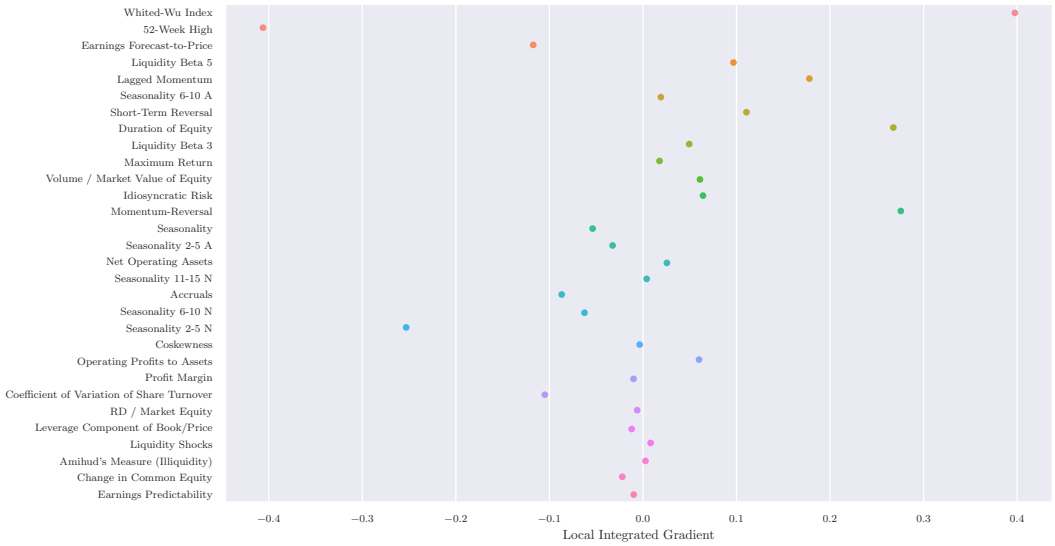
It is important to understand why a feature can contribute both negatively and positively to the predicted return. This is not to be confused with the sign of the effect as we are used to it from linear regression: the values are not coefficients, but additive portions of the prediction. Say that the effect of *Whited-Wu Index* is strictly linear and positive. Then, large positive returns would be due to large positive *Whited-Wu Index* (the feature contributes *positively* to the predicted return in the sense of Figure 4.4) and, *at the same time*, large negative returns would be due to large negative *Whited-Wu Index* (the

feature contributes *negatively* to predicted return in the sense of Figure 4.4), while the association is positive in the classical linear regression sense.

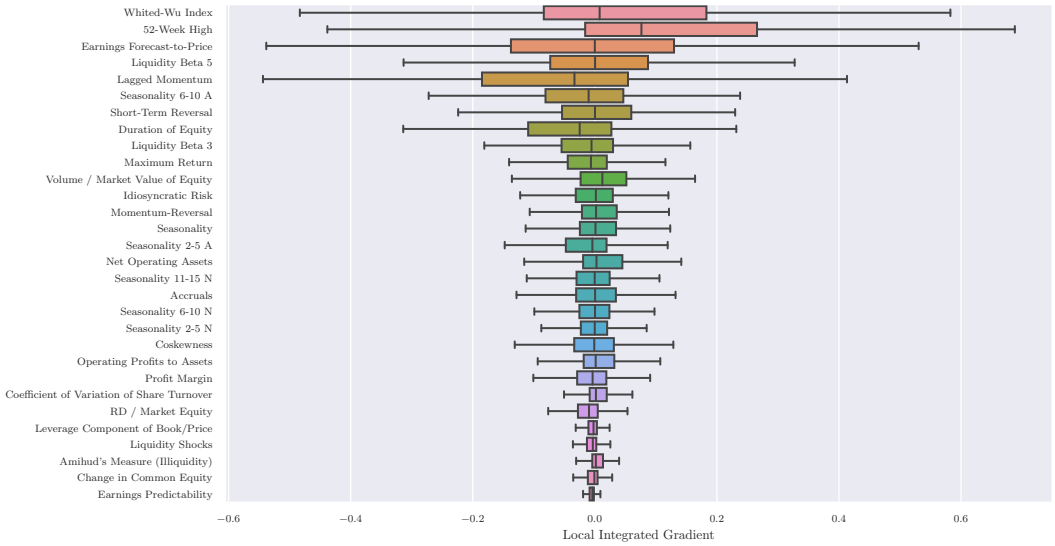
Figure 4.5 and Table 4.2 show again the contribution of features to the predicted return, but this time in terms of multiples of feature values: a feature contributes x times its value to the predicted return (in other words, the feature value times x equals the Local Integrated Gradient). The figure reports the distribution of these values across observations, while columns *Mean* and *Std* report the average and standard deviation of these distributions.

Crudely speaking, this is an analogue of the "coefficients" from linear regression. However, there are two crucial differences. First, the "coefficient" is different across observations, and not the same for all like in linear regression. (The distribution of the values is seen in Figure 4.5.) Second, the interpretation is slightly different as well. In linear regression, we can say that prediction changes by x units in response to unit increase in value of a feature. Here, we do not operate with unit increases in features. Rather, we can say that on average (across observations), a feature contributes x times the feature value to the predicted return. (It is useful to become familiar with the interpretation of Local Integrated Gradient first – above – then, this rephrasing in terms of multiples of feature values becomes more natural.) Also note that there are absolutely no established statistical properties of the *Mean / Std* column in Table 4.2, so the column cannot be interpreted as is usual in linear regression in terms of statistical significance of the effect. It merely says how much the mean is away from zero in terms of standard deviations.

Keeping these two differences in mind, Figure 4.5 and Table 4.2 allow to see the typical sign of the effect of a feature. For example, *52-Week High* contributes on average 0.457 times its value to the predicted return (column *Mean*), and the effect is usually positive across all observations (second boxplot from top in the figure). Indeed, the effect of *52-Week High* is expected to be positive according to prior literature (column *Sign (Original)* of the table). Other features can be interpreted similarly. It is useful to compare the distribution of the effects in Figure 4.5 to the column *Sign (Original)* of the table 4.2. In some cases, the direction suggested by prior literature is overwhelmingly confirmed (for example, see the notably positive impact of *Earnings Forecast-to-Price* or the negative impact of *Duration of Equity*, visible as boxplots clearly to the right (left) of 0 in the figure). Many similar examples can be found in the figure. In addition, there are several interesting cases where the variable appears to influence returns in both positive and negative direction, both with large



(a) Single Prediction



(b) All Predictions

Figure 4.4: Attribution of Predictions to Individual Features

Panel (a) shows values of Integrated Gradient for a single (randomly chosen) out-of-sample prediction of NN1. The predicted return is the sum of the shown Integrated Gradient values. All predictions can be decomposed in the similar manner – Panel (b) summarizes all out-of-sample predictions of NN1 using boxplots. As usual, the "box" of a boxplot marks the 25%, 50% and 75% percentiles, and the "whiskers" show distance of 2 standard deviations from the box. The outliers are not plotted for of a clearer display.

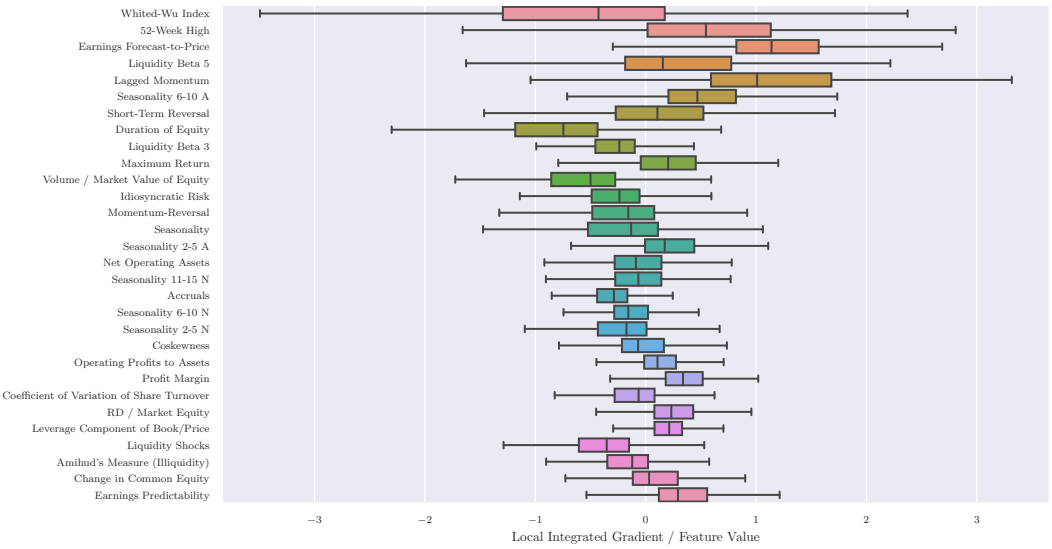


Figure 4.5: Distribution of Feature Effects

For each feature and each observation, the feature's contribution to the prediction (Local Integrated Gradient) is divided by the value of the feature. For each feature, the figure shows the distribution of these values. The figure allows to see the typical sign of the effect of a feature, for example, the effect of *Earnings Forecast-to-Price* on returns is typically positive, and the effect of *Duration of Equity* is typically negative, both as expected by prior literature (4.2). The numerical values of the mean and standard deviation of the distributions are reported in Table 4.2. The results are calculated using all out-of-sample predictions of NN1.

magnitudes: notably, *Whited-Wu Index*, *Short-Term Reversal*, and *Maximum Return*. While these variables clearly influence the predictions to a large extent, the effect appears to be non-monotonous. This is further supported in 4.3, which shows that these variables are precisely those which the networks consider important and linear regression unimportant. This offers additional evidence of their non-linear (e.g., U-shaped, or interaction-intensive) relationship to returns. This offers some indication that these variables might have a more complex relationship to returns than previously thought.

Columns *Sign (Original)* and *Sign (Here)* of the table give direction of the effect in the original paper that published the feature and here. The original papers all assume linear relationship, so the sign is that of the coefficient from linear regression of returns on the feature and control variables. The sign found in this thesis is simply the sign of the *Mean* column, i.e., the direction in which the feature influences predicted return on average in network NN1.

In 22 cases out of 30, the sign found in this thesis agrees with sign published in the respective original paper, which is quite notable. First, it confirms that

	Mean	Std	Mean / Std	Sign (Original)	Sign (Here)
Whited-Wu Index	-0.830	1.951	-0.425	1	-1
52-Week High	0.457	1.392	0.328	1	1
Earnings Forecast-to-Price	1.222	0.617	1.980	1	1
Liquidity Beta 5	0.528	1.397	0.378	1	1
Lagged Momentum	1.222	0.889	1.374	1	1
Seasonality 6-10 A	0.488	0.630	0.774	1	1
Short-Term Reversal	0.093	0.787	0.118	-1	1
Duration of Equity	-0.850	0.692	-1.228	-1	-1
Liquidity Beta 3	-0.294	0.377	-0.782	-1	-1
Maximum Return	0.235	0.582	0.404	-1	1
Volume / Market Value of Equity	-0.634	0.649	-0.977	-1	-1
Idiosyncratic Risk	-0.295	0.422	-0.699	-1	-1
Momentum-Reversal	-0.235	0.503	-0.466	-1	-1
Seasonality	-0.321	0.776	-0.413	1	-1
Seasonality 2-5 A	0.254	0.576	0.441	1	1
Net Operating Assets	-0.036	0.512	-0.070	-1	-1
Seasonality 11-15 N	-0.063	0.490	-0.129	-1	-1
Accruals	-0.313	0.274	-1.143	-1	-1
Seasonality 6-10 N	-0.117	0.337	-0.348	-1	-1
Seasonality 2-5 N	-0.259	0.422	-0.614	-1	-1
Coskewness	-0.020	0.413	-0.049	-1	-1
Operating Profits to Assets	0.134	0.290	0.462	1	1
Profit Margin	0.374	0.284	1.319	1	1
Coefficient of Variation of Share Turnover	-0.157	0.426	-0.369	1	-1
RD / Market Equity	0.286	0.329	0.867	1	1
Leverage Component of Book/Price	0.209	0.216	0.967	1	1
Liquidity Shocks	-0.391	0.341	-1.147	-1	-1
Amihud's Measure (Illiquidity)	-0.219	0.425	-0.515	1	-1
Change in Common Equity	0.065	0.370	0.175	-1	1
Earnings Predictability	0.374	0.400	0.933	-1	1

Table 4.2: Descriptive Statistics of Feature Effects

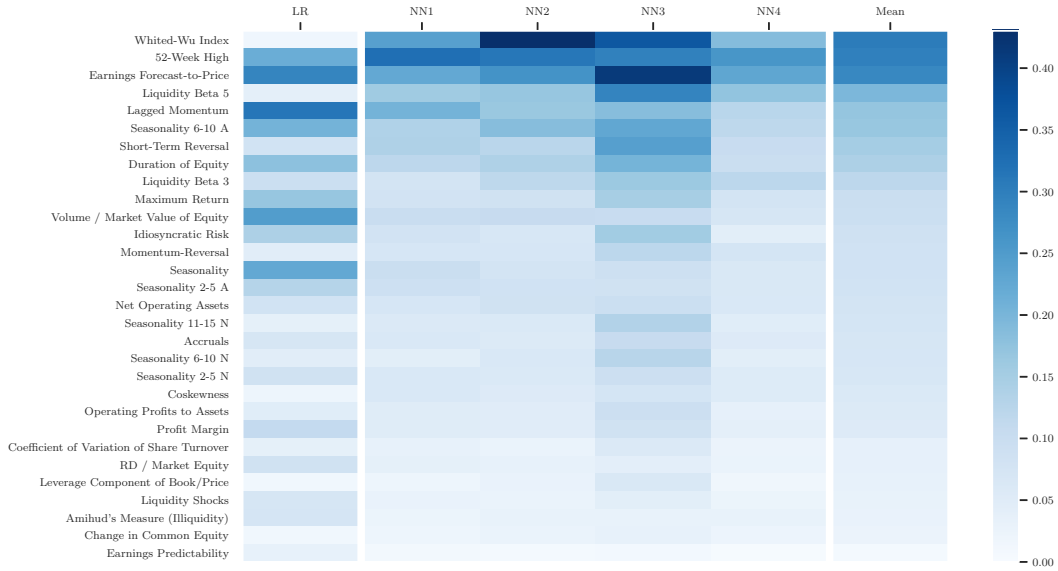
Column *Mean* shows how on average (across observations) a feature contributes to the predicted return in terms of multiple of the values of the observation. For each feature, the value is obtained by first calculating, for each observation, the feature's contribution to the prediction (Local Integrated Gradient) divided by the value of the feature, and second, averaging across all observations. Column *Std* is calculated in the same manner, but standard deviation is computed instead of average in the second step. Column *Sign (Original)* shows the sign of the feature as per its original paper (all original papers assume linear relationship between the feature and the return, and the sign is the sign of the coefficient in linear regression). Column *Sign (Here)* shows simply the sign of the *Mean* column. The results are calculated using all out-of-sample predictions of NN1.

the results are not off, but rather follow the economic intuitions and findings of prior literature (Table 2.1). Second, it appears that the relationships between the predictors and returns are monotonous (albeit possibly non-linear), in other words, it seems that predictors' effects have a general direction, either positive, or negative, and moreover, that this direction agrees with economic theory. To the best of my knowledge, this work is the first to offer this insight: it appears that return-predicting neural networks overwhelmingly agree with the economic mechanisms proposed over the past 50 years of asset-pricing research.

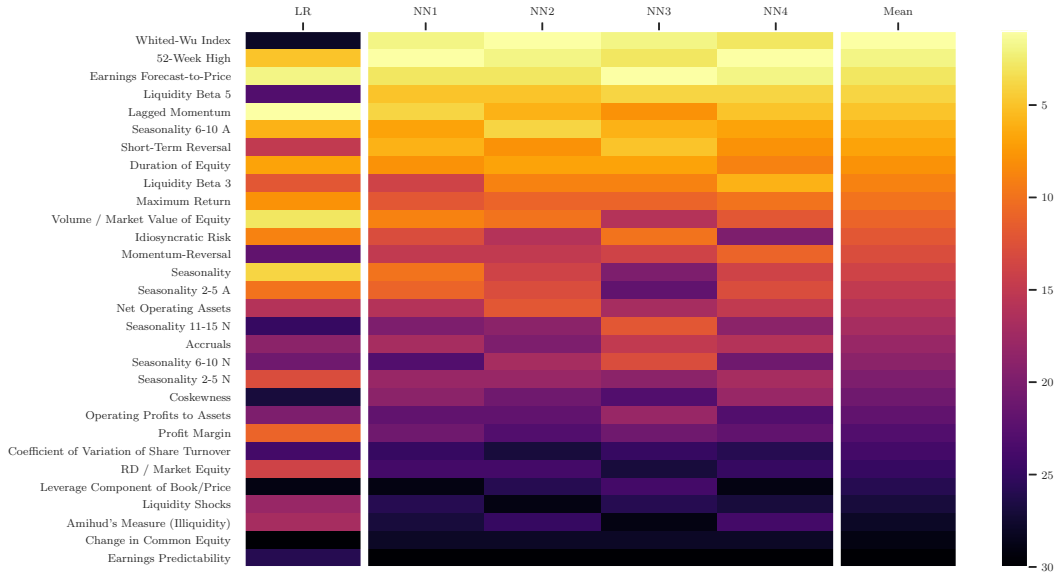
In only 8 cases out of 30, the signs do not agree with the original paper. 4 of these cases are the least important variables (4 bottom rows of the table), so this discrepancy can be considered of little import, since the effect of the variables on returns is close to 0 anyway. The remaining 4 discrepancies are more interesting: *Whited-Wu Index*, *Short-Term Reversal*, *Maximum Return*, and *Coefficient of Variation of Share Turnover*: we already noted that these variables appear to have a non-monotonous relationship to returns according to Figure 4.5. As a result, summarizing these relationships' direction with a sign may not be meaningful, which can explain their discrepancy in sign between the original paper and the neural network employed in this thesis.

4.3 Global Feature Importance

This section offers insight into what variables the neural networks consider important for stock return prediction. The insight is called global, as it focuses on the overall workings of the model, rather than on explaining individual predictions. Figures 4.6 and 4.7 shows the values of feature importance for all features, as measured respectively by Global Integrated Gradients and Portfolio Reliance. The former shows feature importance across the entire returns' distribution and offers new insights into the old asset-pricing question of which variables explain differences across stock returns, while the latter focuses on the long-short profitability in the tails of the returns' distribution, which offers implications for financial practice. First two subsections now discuss the results for the two measures in turn, the third subsection compares the results of the two measures, and the following subsections discuss stability of the results in time and across random seeds.



(a) Values of Global Integrated Gradient



(b) Order of Features by Importance

Figure 4.6: Feature Importance Measured with Global Integrated Gradient

Column *LR* shows linear regression and columns *NN1* to *NN4* the neural networks of respective depths. Column *Mean* gives the average value across the neural networks. Panel (a) shows values of the Global Integrated Gradient for all features, for all models and shows an average share of predicted return attributed to the feature (in same scale as the return itself, i.e., percentage points). Numerical values for Panel (a) are given in the Appendix B in Figure B.1. Panel (b) shows ordering of features by their importance, with bright colors corresponding to high order. Label 1 — bright yellow (30 — black) corresponds to most (least) important feature in given model, as measured by highest (lowest) Global Integrated Gradient.

4.3.1 Integrated Gradient

Figure 4.6 shows importance of all features, for all models as measured by Global Integrated Gradients. In both panels, the most (least) important features appear on top (bottom) of the figure. The measure is the average of absolute values of Local Integrated Gradient in Section 4.2 and shows how much the predicted return changes on average.[3] Intuitively, if the output changes a lot due to the feature, the value of Global Integrated Gradient is high and the feature is considered important for the prediction. Panel (a) shows the values of the measure, while Panel (b) shows the order of the features from 1 to 30 (1 for most important).

Panel (a) shows the values of the Integrated Gradient for all features (the exact numerical values are given in Appendix B). *Whited-Wu Index*, the most important feature, on average changes the predicted return by around 0.3 percentage points. The same holds for *52-Week High* and *Earnings Forecast-to-Price*, the second and third most important features. The value 0.3 is quite high, considering that the standard deviation of return is 11.2 percentage points. The next 5 features in importance are *Liquidity Beta 5*, *Seasonality 6-10 A*, *Short-Term Reversal*, *Duration of Equity* and *Liquidity Beta 3*, which on average change the predicted return by 0.2–0.1 percentage points. Next 14 features (*Maximum Return* to *Profit Margin*) change the prediction n by 0.1–0.05 percentage points on average, and the remaining 8 change it by less than 0.05 percentage points.

In light of economic motivation behind the features (Table 2.1), the most important categories seem to be financial constraints (*Whited-Wu Index*), limited attention and behavioral biases of investors (*Earnings Forecast-to-Price*, *52-Week High and Short-Term Reversal*), risk of illiquidity (*Liquidity Beta 3 and Liquidity Beta 5*), seasonality (*Seasonality 6-10 A*), and value effect (*Duration of Equity*).

The figure also shows that all neural networks agree on the importance to a great extent, both in values of the measure and in the implied order of the features. This is visible in both panels as same color across a row. Additionally, the figure shows that linear regression (column *LR*) does not agree with the neural networks on the importance of some features, most notably, *Whited-Wu Index*, *Liquidity Beta 5* and *Short-Term Reversal* are considered important by

[3] Absolute values of Local Integrated Gradient are considered so the effects do not average out.

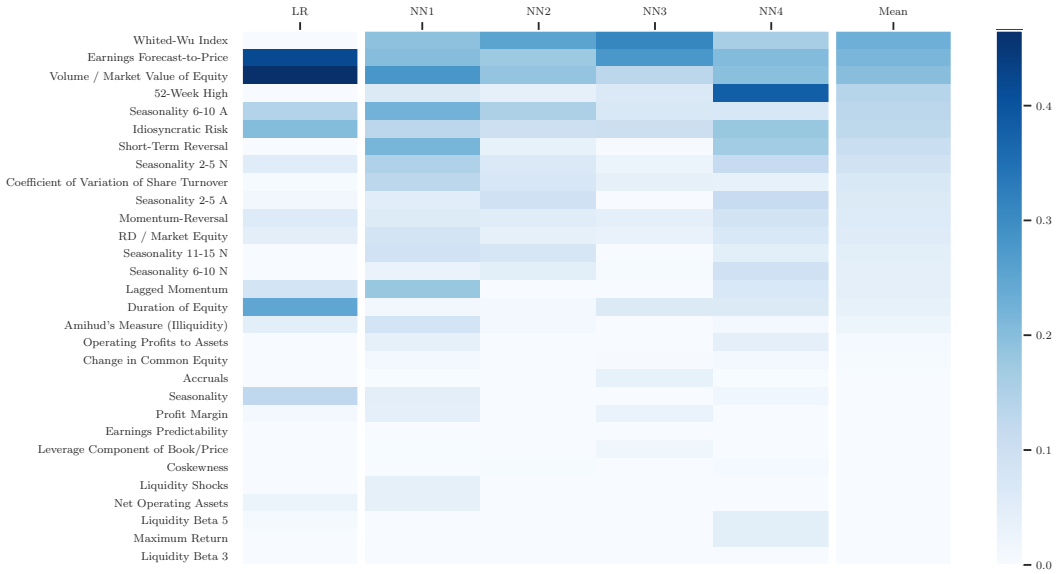
the networks, but very marginal by the linear regression. This is most likely because the impact of the features is not linear: either, their relationship with returns is not well-fitted by linear function, or they only impact the return through an interaction with other features. (Both the non-linearities and interaction terms are captured by the hidden layers of the networks but not by the linear regression due to the architecture of the respective models.)

4.3.2 Portfolio Reliance

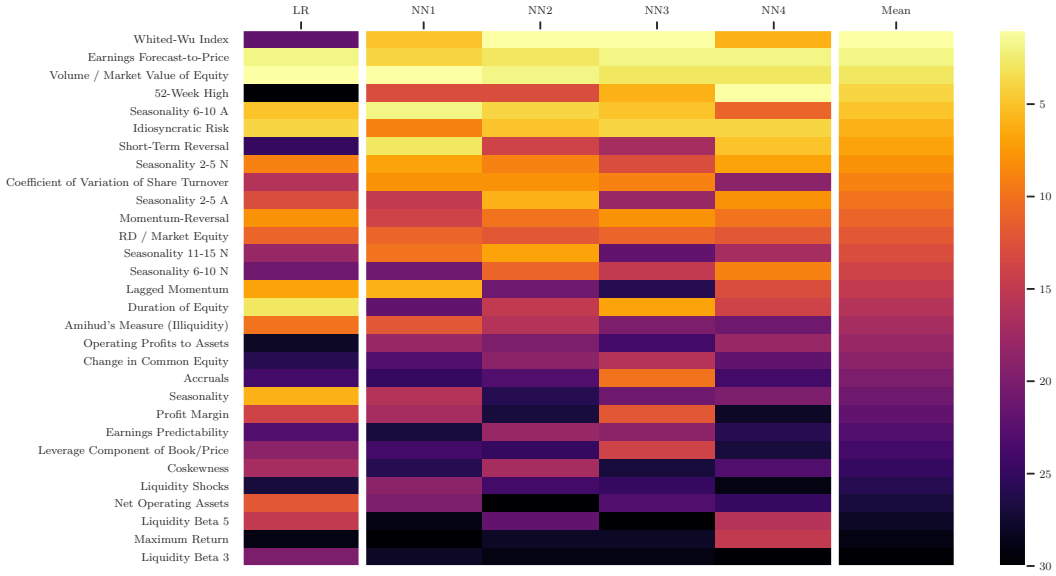
Figure 4.7 shows importance of all features, for all models as measured by Portfolio Reliance. The measure shows how important a feature is for performance of the long-short portfolios (see 4.1.2). Specifically, values represent the decrease in return on the long-short portfolio resulting from corrupting the feature (rendering it completely uninformative by permutation of its values; the methodology behind is discussed in Chapter 3. Intuitively, if the mean return on the long-short portfolio decreases a lot when the feature is corrupted, the value of Portfolio Reliance is high and the feature is considered important. Same as with Figure 4.6, the most (least) important features appear on top (bottom) of the figure. Again, Panel (a) shows the values of the measure, while Panel (b) shows the order of the features from 1 to 30 (1 for most important).

Panel (a) shows the values of Portfolio Reliance for all features (the exact numerical values are given in Appendix B). *Whited-Wu Index* is again the most important feature. Its Portfolio Reliance value (mean across the networks) is 0.23, which means that the mean return on long-short portfolio decreases by 0.23 percentage points when *Whited-Wu Index* is corrupted. Considering that the mean monthly return is around 1 percentge point (1.4 for NN1, 0.9 for NN3, other in between), it means distorting the feature takes away almost quarter of the performance. Two other features follow closely: *Earnings Forecast-to-Price* (0.215) and *Volume to Market Value of Equity* (0.199). Next are four features with values between 0.138 and 0.1: *52-Week High*, *Seasonality 6-10 A*, *Idiosyncratic Risk* and *Short-Term Reversal*. Six other features have Portfolio Reliance above 0.05: *Coefficient of Variation of Share Turnover*, *Momentum-Reversal*, *RD to Market Equity* and three seasonality measures. The rest of the variables have values close to 0 (or even slightly negative), which means they are unimportant or slighlty detrimental to the performance of the portfolios.

In terms of the economic motivation of the features (Table 2.1), the most important groups appear to be financial constraints (*Whited-Wu Index*), limited



(a) Values of Portfolio Reliance



(b) Order of Features by Importance

Figure 4.7: Feature Importance Measured with Portfolio Reliance.

Column *LR* shows linear regression and columns *NN1* to *NN4* the neural networks of respective depths. Column *Mean* gives the average value across the neural networks. Panel (a) shows values of the Portfolio Reliance for all features, for all models. Panel (b) shows ordering of features by their importance, with bright colors corresponding to high order. Label 1 — bright yellow (30 — black) corresponds to most (least) important feature in given model, as measured by highest (lowest) Portfolio Reliance.

attention and behavioral biases of investors (*Earnings Forecast-to-Price*, *52-Week High*, *Short-Term Reversal*, their attitude to risk (*Idiosyncratic Risk*) and illiquidity (*Volume to Market Value of Equity*).

Again, the networks quite agree on the importance of individual features (albeit to a somewhat lesser extent than in Integrated Gradients, 4.6). Same as with Integrated Gradients, linear regression (column *LR*) disagrees on the importance of *Whited-Wu Index* and *Short-Term Reversal*, and additionally, *52-Week-High*. Again, the reasons for this are likely the same as discussed with the Integrated Gradient: these features likely have non-linear (e.g., U-shaped) or interaction-intensive effect that the linear regression fails to capture.

4.3.3 Comparison of Feature Importance Measures

Figure 4.8 compares the results of the previous two subsections, namely, feature importance as measured by Integrated Gradient and by Portfolio Reliance. There is a crucial semantical difference between the two measures. The former captures the importane of the features across the entire universe of stocks, while the latter focuses only on the long-short portfolios constructed using the networks, i.e., on prediction of the very high and very low returns. Both approaches have a merit: the former closer to the academician's interest in explaining the cross-section of stock returns, the latter answers the practicioner's question of which features the networks rely on to predict winners and loosers correctly and thus outperform the market. As shown in 4.1.1, the networks are not particularly good at explaining the entire cross-section of returns (negative R^2 in the medium deciles), so Portfolio Reliance may be more insightful than Integrated Gradients in terms of answering the question of what contributes to the neural networks outstanding performance.

The figure shows that the two measures seem to agree on important features to a large extent. This is quite reassuring, since the measures are constructed completely differently, and yet their results support each other. *Whited-Wu Index*, *52-Week High*, *Earnings Forecast-to-Price*, *Seasonality 6-10 A*, and *Short-Term Reversal* are among the most important features for both measures.

However, there are two features that are quite important in Integrated Gradients and unimportant in Portfolio Reliance: *Liquidity Beta 5* and *Lagged Momentum*. Considering the just-discussed semantical differences between the two measures, these two variables seem more important for predicting the cross-section rather than the tails of the return. Vice versa, two features are very

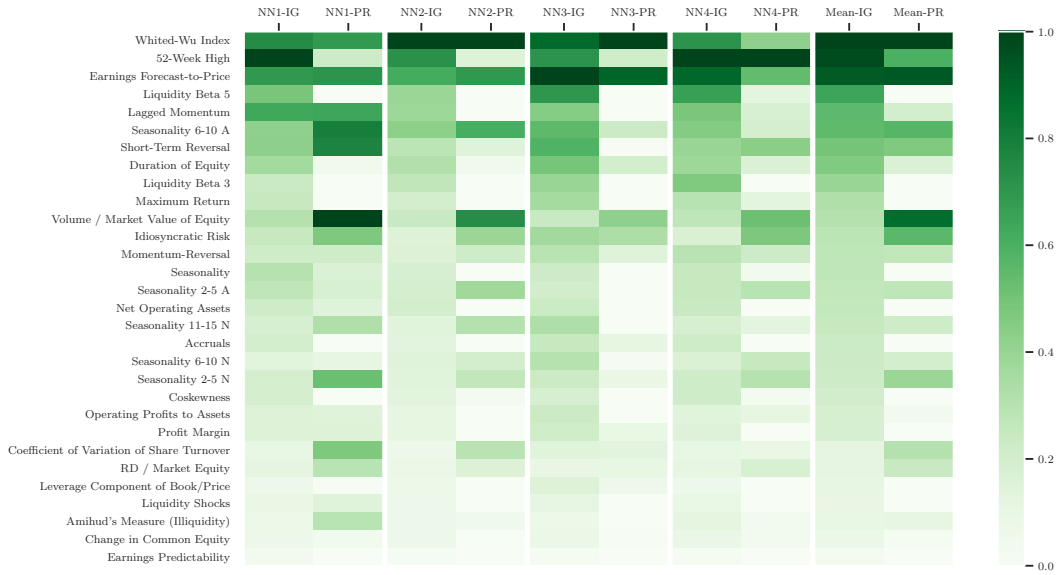


Figure 4.8: Comparison of Feature Importance Measured with Integrated Gradients and with Portfolio Reliance

Column *LR* shows linear regression and columns *NN1* to *NN4* the neural networks of respective depths. The values of Integrated Gradients are the same as in Panel (a) of Figure 4.6, the values of Portfolio Reliance are the same as in Panel (a) of Figure 4.7. The only difference is that all columns are divided by their maximum value so as to unify the scale across the different measures. Values 1 correspond to the most important feature in given column, values of 0 to the least important feature.

important in Portfolio Reliance byt quite mediocre in Integrated Gradients: *Volume to Market Value of Equity* and *Idiosyncratic Risk*. These two variables apprear to be more important for predicting the tails of returns rather then the entire cross-section.

4.3.4 Feature Importance In Time

This section decomposes the main results (Figures 4.6 and 4.7) into different time periods. Recall from Section 3.2.3 that any model (e.g., NN1) is completely re-trained every year, as new data arrives and the training set expands (and validation and testing set rolls forward accordingly). The main results are average across all these independent instances of models in time. This section offers a decomposition. Decomposing the results in time is interesting as it offers some insight to whether the relationship between returns and firm characteristics is stable in time.

Figures 4.9 and 4.10 show a decomposition of Integrated Gradient and Port-

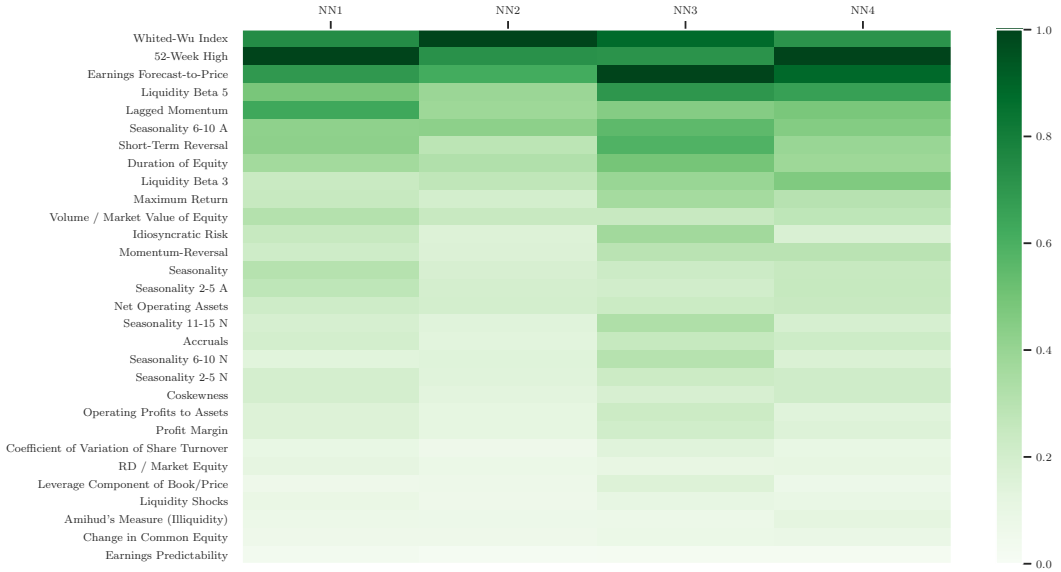
folio Reliance measures in time for all models, while Figure 4.11 shows the same, but averaged across the four models. Integrated Gradient seems very stable across time: it seems that the cross-sectional relationships learned by the neural networks are rather stable in time. On the other hand, Portfolio Reliance is quite unstable in time, meaning that the ability of the models to predict future winners and loosers depends on different variables year after year.

4.3.5 Feature Importance Across Random Seeds

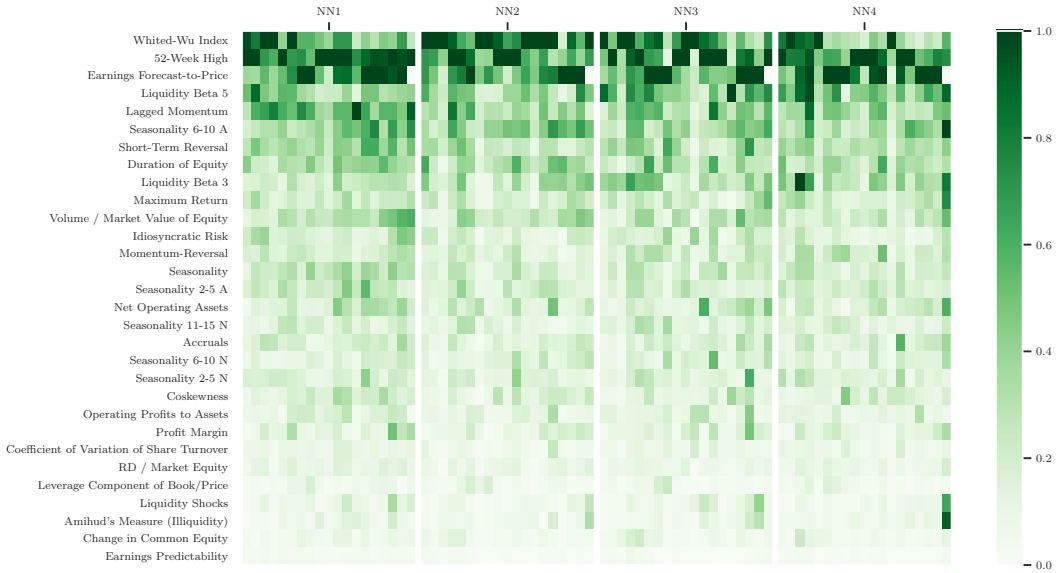
This section decomposes the main results (Figures 4.6 4.7) into the component models (random seeds) of the ensemble models. This provides a more nuanced understanding of how the ensemble models produce the predictions under the hood: a single prediction is made by each component model individually and the individual predictions are then averaged. These decompositions thus allow to see if the component models are in agreement about important variables or rather supplement one another.

Figure 4.12 shows the decomposition for Global Integrated Gradients, that is, the decision-making of the model overall. It shows that the component models are in agreement about importance of the variables to a large extent: the values of Integrated Gradient for a single variable are similar across the component models. This is true across the four model architectures. There is an interesting implication of this result: recall from Section 3.2.4 that the component models are identical to each other, except for small differences in the initial weights at the beginning of the training. From this perspective, the results indicate that these small changes in values at the beginning of the optimization have little influence on the learned parameters.

Figure 4.13 shows the decomposition for Portfolio Reliance measure, and thus offers insights into how variable importance at the tails of returns' distribution differs across component models. The component models still appear to agree, albeit to a lesser extent than in the case of Integrated Gradients, and in some cases, the component models rather appear to supplement each other. It can be speculated that this offers a peek into why ensemble models work well: small errors made by the component models average out and the truth arises in between – as Fisher *et al.* (2019) write in the title of their paper: "All models are wrong, but many are useful." From this perspective, multiple patterns may offer equally good explanation of the data. As a result, different random seeds



(a) Main Result

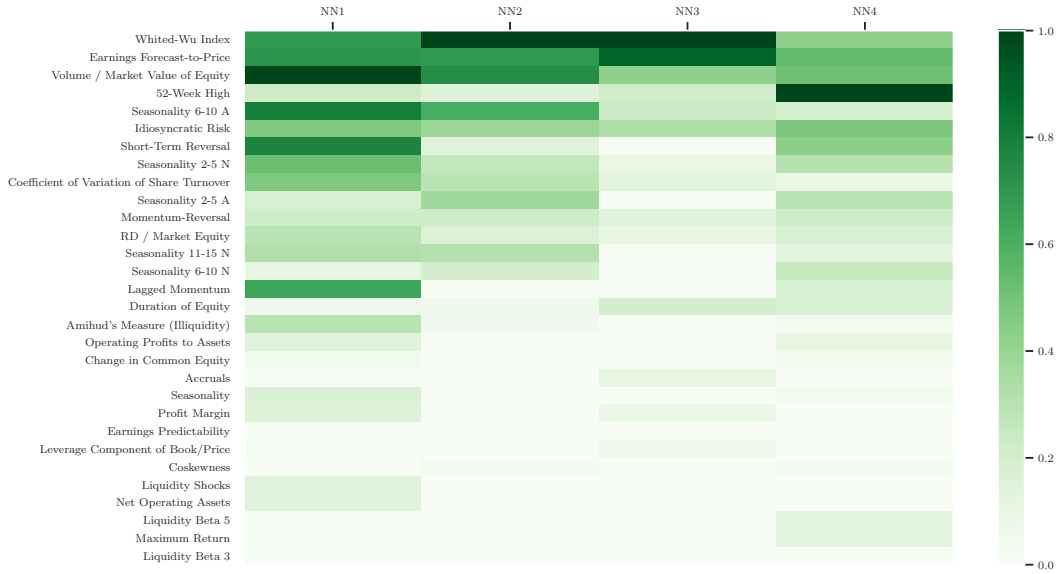


(b) Time Decomposition

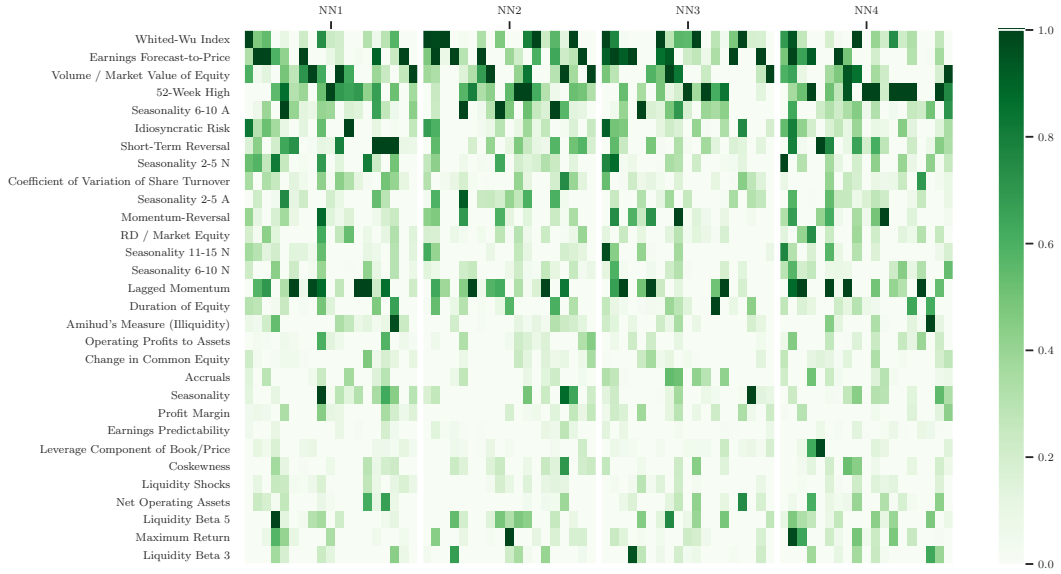
Figure 4.9: Feature Importance in Time as Measured by Global Integrated Gradients

NN1 to NN4 denote neural networks of respective depths. Both panels show relative values of Global Integrated Gradient: all values are divided by the value for the most important feature, so as to unify the scale across the different models and time periods. While Panel (a) shows the same (only rescaled) result as Figure 4.6a, Panel (b) offers its decomposition into the 19 testing years, 2000 to 2018, for each model.

can find different relationships and when the ensemble model combines this knowledge, it gains an edge over any of its sub-components.



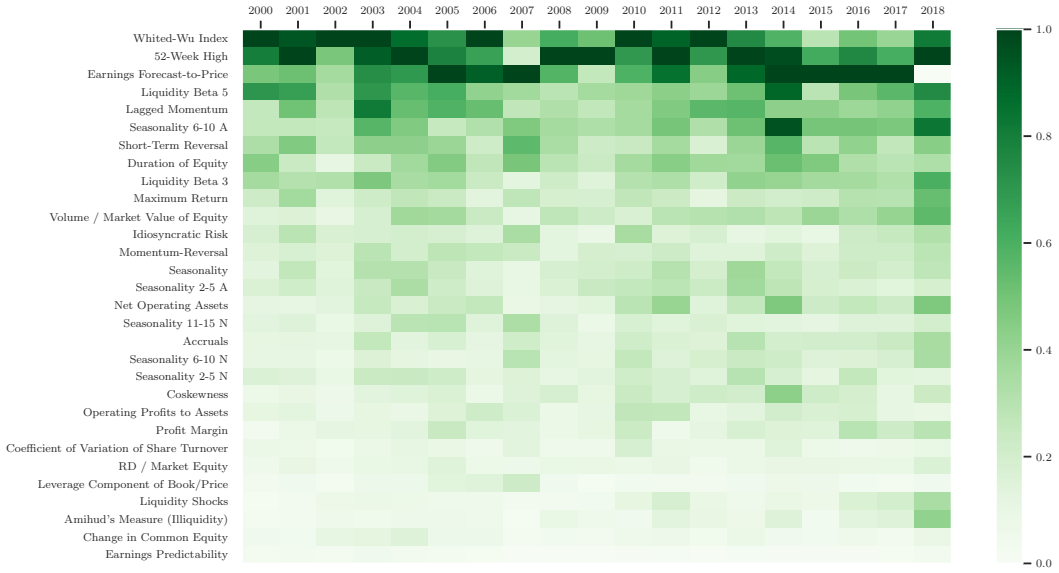
(a) Main Result



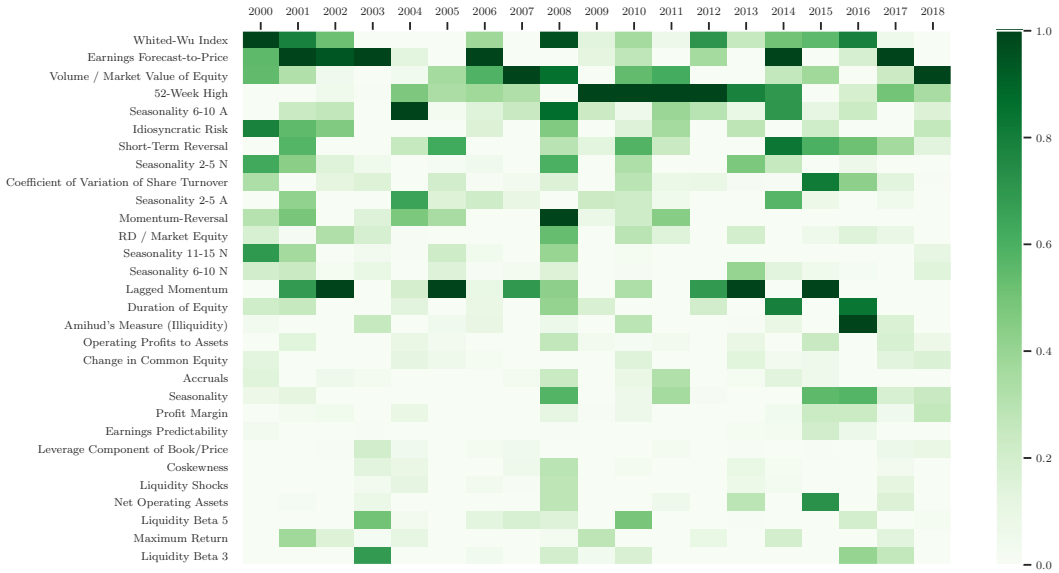
(b) Time Decomposition

Figure 4.10: Feature Importance in Time as Measured by Portfolio Reliance

NN1 to NN4 denote neural networks of respective depths. Both panels show relative values of Portfolio Reliance: all values are divided by the value for the most important feature, so as to unify the scale across the different models and time periods. While Panel (a) shows the same (only rescaled) result as Figure 4.7a, Panel (b) offers its decomposition into the 19 testing years, 2000 to 2018, for each model.



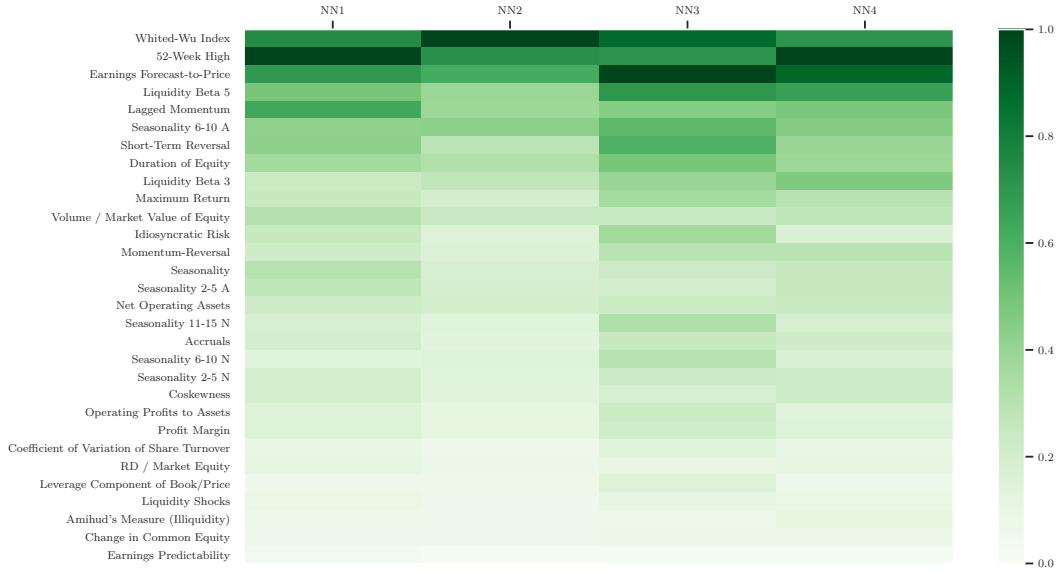
(a) Integrated Gradients



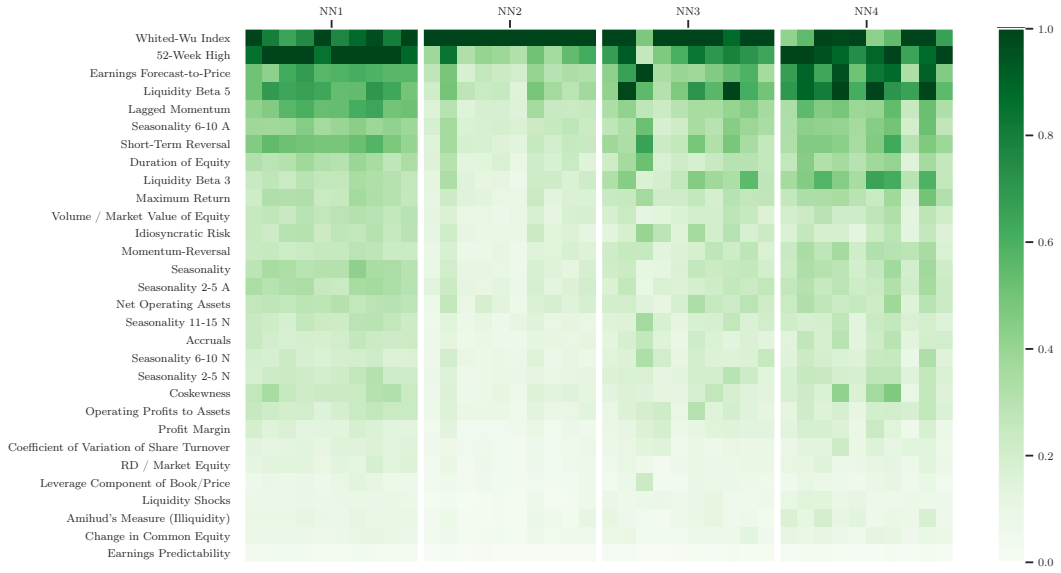
(b) Portfolio Reliance

Figure 4.11: Feature Importance in Time: Mean Across Models

The figure shows mean across all models (NN1 to NN4) of (a) Integrated Gradients and (b) Portfolio Reliance. All values are relative: divided by the value for the most important feature, so as to unify the scale across the different measures and time periods. As discussed in 3.2.3, all models are re-fitted at the beginning of each year using the up-to-date data and the out-of-sample prediction is then made for the next year. The window moves forward, creating 19 independent models with 19 corresponding testing years (2000 to 2019, both included) for each architecture (NN1 to NN4). This figure takes average across the four architectures and shows a single value for a given testing year and given feature.



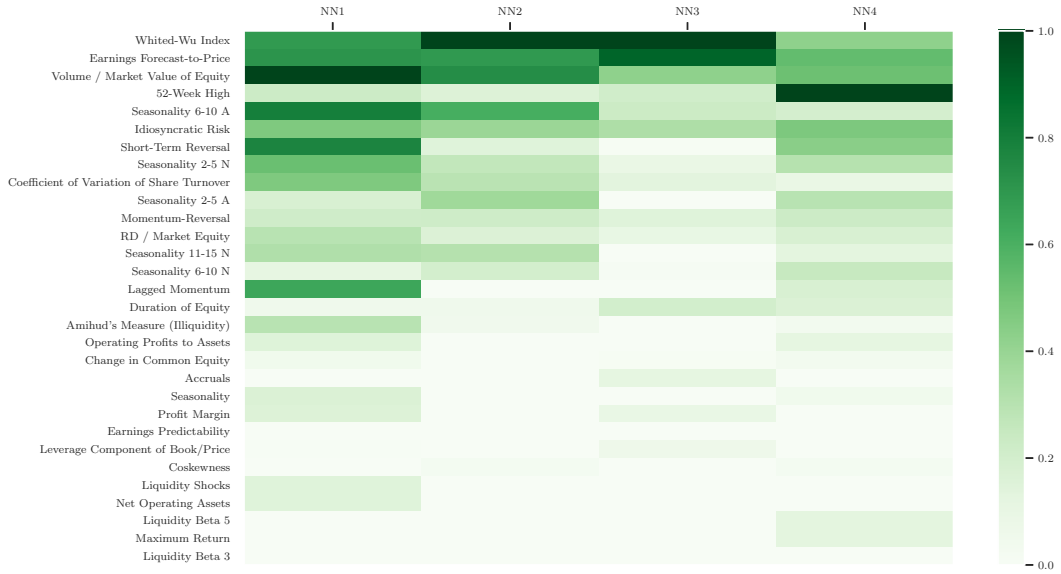
(a) Main Result



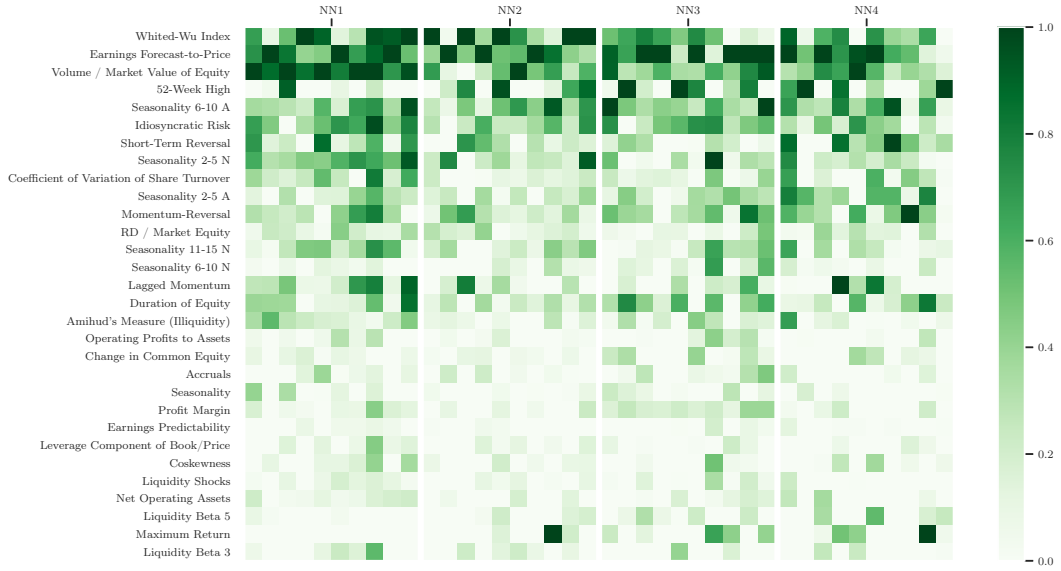
(b) Decomposition into Random Seeds

Figure 4.12: Feature Importance across Random Seeds as Measured by Integrated Gradients

NN1 to NN4 denote neural networks of respective depths. Both panels show relative values of Global Integrated Gradient: all values are divided by the value for the most important feature, so as to unify the scale across the different models. Panel (a) shows the same (only rescaled) result as Figure 4.6a (the ensemble model), while Panel (b) offers its decomposition into the individual components of the ensemble.



(a) Main Result



(b) Decomposition Into Random Seeds

Figure 4.13: Feature Importance across Random Seeds as Measured by Portfolio Reliance

NN1 to NN4 denote neural networks of respective depths. Both panels show relative values of Portfolio Reliance: all values are divided by the value for the most important feature, so as to unify the scale across the different models. Panel (a) shows the same (only rescaled) result as Figure 4.7a (the ensemble model), while Panel (b) offers its decomposition into the individual components of the ensemble.

Chapter 5

Conclusion

The recent explosion in ML interpretability methods offers new opportunities to explain stock returns within the framework of the most powerful approach to modeling them: the neural network. This thesis pioneers the exploration of two of these newly opened opportunities.

First, the thesis extracts new insights about the problem that has been occupying finance for over five decades: which variables explain differences in the cross-section of stock returns? Prior research offers preliminary findings of which variables are important according to neural networks (Gu *et al.* 2020; Tobek & Hronec 2020). This thesis investigates 30 predictors that appear most important in existing research more deeply. In particular, it measures the typical sign of the effects that the predictors have on the return in neural networks. This is important, as it allows to see whether neural networks go *with*, or *against*, the economically motivated theoretical signs of the predictors. If we (safely) ignore the 4 least important variables, the networks agree in the sign in as much as 22 out of 26 cases, which means that they overwhelmingly support the economic mechanisms suggested by the asset-pricing literature. Additionally, the thesis investigates more deeply than prior literature the overall importance of the variables in the network, by choosing an interpretable metric well grounded in ML theory and studying the results in various decompositions. Financial constraints, behavioral biases of investors, risk of illiquidity and value effect appear to be the most important drivers of stock returns uncovered by the networks. At the same time, there are several pieces of evidence that *Whited-Wu Index*, *Short-Term Reversal* and *Maximum Return* may have a more complex relationship to returns then previously thought. The results are stable across model architectures, ensemble components as well as time periods

and narrow down the search for important determinants of stock returns.

Second, this thesis explores the sources of the unparalleled ability of neural networks to construct profitable long-short portfolios. This offers first insights into these state-of-the-art models that are increasingly used in the financial practice. The thesis shows how any single return prediction can be decomposed into its drivers and also computes a novel metric, Portfolio Reliance, allowing to see which variables the networks consider important to identify future winners and losers among the stocks in the market. A decomposition in time further uncovers that these variables change from year to year, unlike the overall, or cross-sectional, importance of the features. The networks constructed in this thesis offer comparable profitability to state-of-the-art models both in terms of mean return and Sharpe ratio, which makes them very relevant to a finance practitioner. The use of a highly liquid universe of stocks further increases the relevance of the results for trading.

The areas for further research are numerous. First, running all results again on simulated data would bring a greater insight into the soundness of the interpretability measures: when the researcher has complete knowledge of the data generating process, she can perfectly asses whether the selected measures perform well. Second, it could be investigated what is the exact *form* of the non-linearities: which interactions of variables are important and what are the exact functional forms of the relationship between the predictors and stock return found by the networks. Third, it should be studied whether the interpretation of the networks in asset-pricing is susceptible to adversarial attacks, as is the case of much more complex networks used for image recognition (Ghorbani *et al.* 2019). Finally, and this is likely the most thrilling area of future research, the statistical properties of feature importance measures in ML are currently quite unclear. More research – e.g. in the direction of Fisher *et al.* (2019) – could bring about an exciting union of statistics and interpretable machine learning.

Bibliography

Acharya, V. V. & L. H. Pedersen (2005): "Asset pricing with liquidity risk." *Journal of financial Economics* **77(2)**: pp. 375–410.

Amihud, Y. (2002): "Illiquidity and stock returns: cross-section and time-series effects." *Journal of financial markets* **5(1)**: pp. 31–56.

Ang, A., R. J. Hodrick, Y. Xing, & X. Zhang (2006): "The cross-section of volatility and expected returns." *The Journal of Finance* **61(1)**: pp. 259–299.

Asparouhova, E., H. Bessembinder, & I. Kalcheva (2010): "Liquidity biases in asset pricing tests." *Journal of Financial Economics* **96(2)**: pp. 215–237.

Baehrens, D., T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, & K.-R. Müller (2010): "How to explain individual classification decisions." *The Journal of Machine Learning Research* **11**: pp. 1803–1831.

Bali, T. G., N. Cakici, & R. F. Whitelaw (2011): "Maxing out: Stocks as lotteries and the cross-section of expected returns." *Journal of Financial Economics* **99(2)**: pp. 427–446.

Bali, T. G., L. Peng, Y. Shen, & Y. Tang (2013): "Liquidity shocks and stock market reactions." *The Review of Financial Studies* **27(5)**: pp. 1434–1485.

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

Binder, A., G. Montavon, S. Lapuschkin, K.-R. Müller, & W. Samek (2016): "Layer-wise relevance propagation for neural networks with local renormalization layers." In "International Conference on Artificial Neural Networks," pp. 63–71. Springer.

Bryzgalova, S., M. Pelger, & J. Zhu (2019): "Forest through the trees: Building cross-sections of stock returns." *Available at SSRN 3493458* .

Chan, L. K., J. Lakonishok, & T. Sougiannis (2001): "The stock market valuation of research and development expenditures." *The Journal of Finance* **56(6)**: pp. 2431–2456.

Chordia, T., A. Subrahmanyam, & V. R. Anshuman (2001): "Trading activity and expected stock returns." *Journal of financial Economics* **59(1)**: pp. 3–32.

Cochrane, J. H. (2009): *Asset pricing: Revised edition*. Princeton university press.

Cochrane, J. H. (2011): "Presidential address: Discount rates." *The Journal of finance* **66(4)**: pp. 1047–1108.

De Prado, M. L. (2018): *Advances in financial machine learning*. John Wiley & Sons.

Dechow, P. M., R. G. Sloan, & M. T. Soliman (2004): "Implied equity duration: A new measure of equity risk." *Review of Accounting Studies* **9(2-3)**: pp. 197–228.

Di Persio, L. & O. Honchar (2016): "Artificial neural networks architectures for stock price prediction: Comparisons and applications." *International journal of circuits, systems and signal processing* **10(2016)**: pp. 403–413.

Elgers, P. T., M. H. Lo, & R. J. Pfeiffer Jr (2001): "Delayed security price adjustments to financial analysts' forecasts of annual earnings." *The Accounting Review* **76(4)**: pp. 613–632.

Fadlalla, A. & C.-H. Lin (2001): "An analysis of the applications of neural networks in finance." *Interfaces* **31(4)**: pp. 112–122.

Fama, E. F. & K. R. French (1993): "Common risk factors in the returns on stocks and bonds." *Journal of* .

Fama, E. F. & K. R. French (1996): "Multifactor explanations of asset pricing anomalies." *The journal of finance* **51(1)**: pp. 55–84.

Fama, E. F. & K. R. French (2015): "A five-factor asset pricing model." *Journal of financial economics* **116(1)**: pp. 1–22.

Fisher, A., C. Rudin, & F. Dominici (2019): "All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously." *Journal of Machine Learning Research* **20(177)**: pp. 1–81.

Francis, J., R. LaFond, P. M. Olsson, & K. Schipper (2004): "Costs of equity and earnings attributes." *The accounting review* **79(4)**: pp. 967–1010.

George, T. J. & C.-Y. Hwang (2004): "The 52-week high and momentum investing." *The Journal of Finance* **59(5)**: pp. 2145–2176.

Ghorbani, A., A. Abid, & J. Zou (2019): "Interpretation of neural networks is fragile." In "Proceedings of the AAAI Conference on Artificial Intelligence," volume 33, pp. 3681–3688.

Giles, C. L., S. Lawrence, & A. C. Tsoi (1997): "Rule inference for financial prediction using recurrent neural networks." In "Proceedings of the IEEE/IAFE 1997 Computational Intelligence for Financial Engineering (CIFEr)," pp. 253–259. IEEE.

Goodfellow, I., Y. Bengio, A. Courville, & Y. Bengio (2016): *Deep learning*, volume 1. MIT press Cambridge.

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

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

Harvey, C. R. & A. Siddique (2000): "Conditional skewness in asset pricing tests." *The Journal of finance* **55(3)**: pp. 1263–1295.

Haugen, R. A. & N. L. Baker (1996): "Commonality in the determinants of expected stock returns." *Journal of Financial Economics* **41(3)**: pp. 401–439.

Heston, S. L. & R. Sadka (2008): "Seasonality in the cross-section of stock returns." *Journal of Financial Economics* **87(2)**: pp. 418–445.

Hirshleifer, D., K. Hou, S. H. Teoh, & Y. Zhang (2004): "Do investors overvalue firms with bloated balance sheets?" *Journal of Accounting and Economics* **38**: pp. 297–331.

Ioffe, S. & C. Szegedy (2015): "Batch normalization: Accelerating deep network training by reducing internal covariate shift." *arXiv preprint arXiv:1502.03167* .

Jegadeesh, N. (1990): "Evidence of predictable behavior of security returns." *The Journal of finance* **45(3)**: pp. 881–898.

Jegadeesh, N. & S. Titman (1993): "Returns to buying winners and selling losers: Implications for stock market efficiency." *The Journal of finance* **48(1)**: pp. 65–91.

Kahneman, D. & A. Tversky (2013): "Prospect theory: An analysis of decision under risk." In "Handbook of the fundamentals of financial decision making: Part I," pp. 99–127. World Scientific.

Karpathy, A., J. Johnson, & L. Fei-Fei (2015): "Visualizing and understanding recurrent networks." *arXiv preprint arXiv:1506.02078* .

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

Kingma, D. P. & J. Ba (2014): "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980* .

Li, Y., D. Turkington, & A. Yazdani (2020): "Beyond the black box: an intuitive approach to investment prediction with machine learning." *The Journal of Financial Data Science* **2(1)**: pp. 61–75.

McLean, R. D. & J. Pontiff (2016): "Does academic research destroy stock return predictability?" *The Journal of Finance* **71(1)**: pp. 5–32.

Molnar, C. (2020): *Interpretable Machine Learning*. Lulu. com.

Novy-Marx, R. (2012): "Is momentum really momentum?" *Journal of Financial Economics* **103(3)**: pp. 429–453.

Olah, C., A. Mordvintsev, & L. Schubert (2017): "Feature visualization." *Distill* **2(11)**: p. e7.

Penman, S. H., S. A. Richardson, & I. Tuna (2007): "The book-to-price effect in stock returns: accounting for leverage." *Journal of accounting research* **45(2)**: pp. 427–467.

Richardson, S. A., R. G. Sloan, M. T. Soliman, & I. r. Tuna (2006): "The implications of accounting distortions and growth for accruals and profitability." *The Accounting Review* **81(3)**: pp. 713–743.

Shapley, L. S. & M. Shubik (1971): "The assignment game i: The core." *International Journal of game theory* **1(1)**: pp. 111–130.

Shrikumar, A., P. Greenside, & A. Kundaje (2017): "Learning important features through propagating activation differences." *arXiv preprint arXiv:1704.02685* .

Sloan, A. (1996): "Create an account or log in." *Accounting review* **71(3)**: pp. 289–315.

Soliman, M. T. (2008): "The use of dupont analysis by market participants." *The Accounting Review* **83(3)**: pp. 823–853.

Springenberg, J. T., A. Dosovitskiy, T. Brox, & M. Riedmiller (2014): "Striving for simplicity: The all convolutional net." *arXiv preprint arXiv:1412.6806* .

Štrumbelj, E. & I. Kononenko (2014): "Explaining prediction models and individual predictions with feature contributions." *Knowledge and information systems* **41(3)**: pp. 647–665.

Sundararajan, M., A. Taly, & Q. Yan (2017): "Axiomatic attribution for deep networks." *arXiv preprint arXiv:1703.01365* .

Tobek, O. & M. Hronec (2020): "Does it pay to follow anomalies research? machine learning approach with international evidence." *Journal of Financial Markets* p. 100588.

Whited, T. M. & G. Wu (2006): "Financial constraints risk." *The Review of Financial Studies* **19(2)**: pp. 531–559.

Zhang, C. & Y. Ma (2012): *Ensemble machine learning: methods and applications*. Springer.

Appendix A

Additional Data Descriptives

This Appendix gives additional descriptive statistics of the distributions of the predictors. Histograms of all feature distributions is given in Figure A.1. The same distributions are also summarized numerically in Table A.1. The numerical values of the correlation matrix of the features are available in Table A.2.

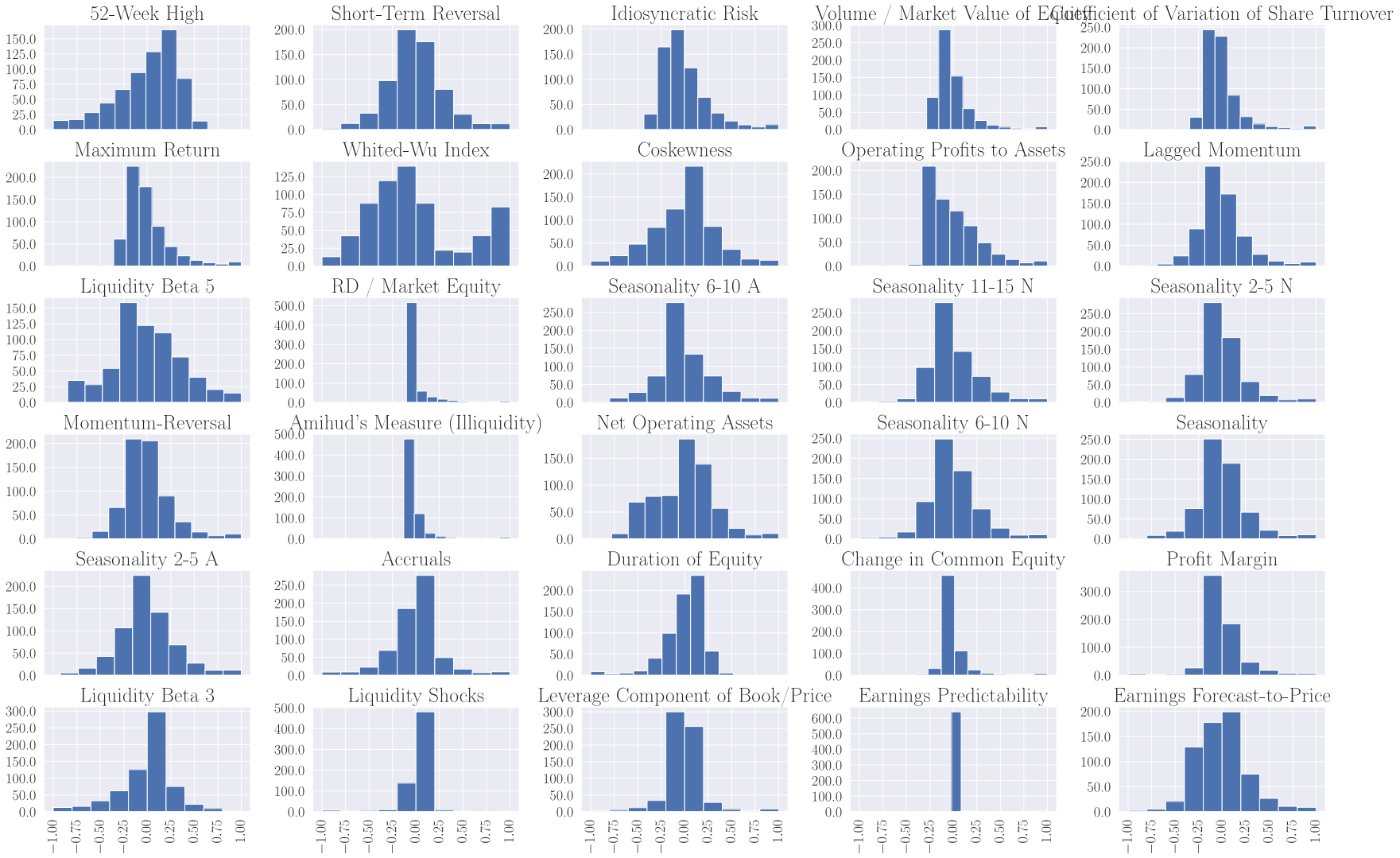


Figure A.1: Histograms of All Features

The figure shows how the values of observations are distributed within each feature. The horizontal axis shows the values of the features (which range between −1 and 1), and the vertical axis shows number of observations (or examples in ML terminology) within each bin, in thousands.

	Mean	Std	Min	25%	50%	75%	Max
52-Week High	0.0	0.3321	-1.0000	-0.1913	0.0728	0.2527	0.6428
Short-Term Reversal	0.0	0.2992	-1.0000	-0.1773	-0.0155	0.1575	1.0000
Idiosyncratic Risk	-0.0	0.2491	-0.4341	-0.1681	-0.0565	0.1030	1.0000
Volume / Market Value of Equity	0.0	0.2019	-0.2846	-0.1173	-0.0533	0.0529	1.0000
Coefficient of Variation of Share Turnover	-0.0	0.2022	-0.3440	-0.1091	-0.0527	0.0497	1.0000
Maximum Return	-0.0	0.2348	-0.3577	-0.1532	-0.0605	0.0821	1.0000
Whited-Wu Index	-0.0	0.5096	-1.0000	-0.3630	-0.1003	0.2134	1.0000
Coskewness	0.0	0.3427	-1.0000	-0.2022	0.0253	0.1821	1.0000
Operating Profits to Assets	0.0	0.2762	-0.4822	-0.2020	-0.0705	0.1476	1.0000
Lagged Momentum	0.0	0.2472	-0.6919	-0.1440	-0.0340	0.1045	1.0000
Liquidity Beta 5	-0.0	0.3695	-0.8467	-0.2089	-0.0271	0.2304	1.0000
RD / Market Equity	0.0	0.1809	-0.1006	-0.0826	-0.0698	-0.0173	1.0000
Seasonality 6-10 A	0.0	0.2904	-1.0000	-0.1313	-0.0425	0.1389	1.0000
Seasonality 11-15 N	0.0	0.2649	-1.0000	-0.1470	-0.0727	0.1322	1.0000
Seasonality 2-5 N	0.0	0.2422	-1.0000	-0.1399	-0.0331	0.1002	1.0000
Momentum-Reversal	-0.0	0.2528	-0.7633	-0.1476	-0.0365	0.1097	1.0000
Amihud's Measure (Illiquidity)	-0.0	0.1483	-0.1302	-0.0552	-0.0346	-0.0115	1.0000
Net Operating Assets	0.0	0.3134	-0.7775	-0.2180	0.0274	0.1836	1.0000
Seasonality 6-10 N	0.0	0.2706	-1.0000	-0.1643	-0.0356	0.1358	1.0000
Seasonality	-0.0	0.2654	-1.0000	-0.1342	-0.0228	0.1212	1.0000
Seasonality 2-5 A	-0.0	0.2986	-0.9264	-0.1648	-0.0313	0.1563	1.0000
Accruals	-0.0	0.2732	-1.0000	-0.1245	0.0297	0.1149	1.0000
Duration of Equity	-0.0	0.2164	-1.0000	-0.0942	0.0439	0.1352	0.5184
Change in Common Equity	-0.0	0.1642	-0.4106	-0.0669	-0.0355	0.0123	1.0000
Profit Margin	-0.0	0.2152	-1.0000	-0.1097	-0.0313	0.0678	1.0000
Liquidity Beta 3	0.0	0.2875	-1.0000	-0.1154	0.0596	0.1450	1.0000
Liquidity Shocks	0.0	0.1506	-1.0000	0.0006	0.0164	0.0289	1.0000
Leverage Component of Book/Price	-0.0	0.2041	-1.0000	-0.0524	-0.0064	0.0502	1.0000
Earnings Predictability	-0.0	0.1132	-0.0191	-0.0182	-0.0163	-0.0150	1.0000
Earnings Forecast-to-Price	0.0	0.2719	-1.0000	-0.1876	-0.0068	0.1381	1.0000

Table A.1: Descriptive Statistics of the Features

	52-Week High	Short-Term Reversal	Idiosyncratic Risk	Volume / Market Value of Equity	Coefficient of Variation of Share Turnover	Maximum Return	Whited-Wu Index	Coskewness	Operating Profits to Assets	Lagged Momentum	Liquidity Beta 5	RD / Market Equity	Seasonality 6-10 A	Seasonality 11-15 N	Seasonality 2-5 N	Momentum-Reversal	Amihud's Measure (Illiquidity)	Net Operating Assets	Seasonality 6-10 N	Seasonality	Seasonality 2-5 A	Accruals	Duration of Equity	Change in Common Equity	Profit Margin	Liquidity Beta 3	Liquidity Shocks	Leverage Component of Book/Price	Earnings Predictability	Earnings Forecast-to-Price
52-Week High	1.000	0.382	-0.345	-0.340	-0.056	-0.224	-0.004	-0.031	-0.001	0.178	-0.144	-0.059	-0.007	0.014	-0.119	-0.060	-0.067	-0.038	-0.037	-0.021	-0.017	-0.052	0.182	-0.073	0.064	0.055	-0.117	-0.060	0.026	0.004
Short-Term Reversal	0.382	1.000	0.105	-0.093	0.025	0.304	0.019	0.001	-0.003	0.013	-0.006	-0.027	0.017	-0.006	-0.011	-0.018	0.041	-0.010	-0.020	0.017	0.003	-0.006	0.070	-0.002	-0.000	-0.010	-0.100	-0.014	0.004	-0.066
Idiosyncratic Risk	-0.345	0.105	1.000	0.230	0.189	0.817	0.093	0.022	0.021	0.055	0.094	-0.013	-0.000	-0.060	0.087	0.030	0.224	0.029	-0.012	0.038	0.013	0.045	-0.020	0.109	-0.090	-0.057	-0.099	0.043	-0.036	-0.114
Volume / Market Value of Equity	-0.340	-0.093	0.230	1.000	0.095	0.178	-0.015	0.022	0.045	0.062	0.301	0.091	0.003	-0.014	0.105	0.073	-0.019	0.030	0.019	0.035	0.013	-0.014	-0.098	0.051	-0.058	-0.211	-0.000	0.037	0.016	0.064
Coefficient of Variation of Share Turnover	-0.056	0.025	0.189	0.095	1.000	0.137	0.100	-0.002	-0.009	0.045	0.018	-0.082	-0.012	-0.059	0.037	0.015	0.405	0.053	-0.025	0.012	0.000	0.039	-0.000	0.094	0.014	-0.007	-0.140	0.013	-0.044	-0.005
Maximum Return	-0.224	0.304	0.817	0.178	0.137	1.000	0.060	0.026	0.002	0.038	0.114	-0.004	0.008	-0.039	0.062	0.013	0.171	0.010	-0.009	0.029	0.008	0.040	-0.007	0.076	-0.080	-0.077	-0.091	0.022	-0.028	-0.114
Whited-Wu Index	-0.004	0.019	0.093	-0.015	0.100	0.060	1.000	0.001	-0.203	0.057	-0.158	-0.226	-0.030	-0.126	0.057	0.044	0.129	-0.313	-0.075	0.006	0.001	0.134	-0.065	-0.028	-0.070	0.034	-0.049	0.039	-0.049	-0.090
Coskewness	-0.031	0.001	0.022	0.022	-0.002	0.026	0.001	1.000	-0.020	-0.018	0.139	-0.029	-0.017	-0.014	-0.006	-0.030	-0.011	-0.032	-0.056	-0.001	-0.002	0.030	-0.004	-0.001	-0.005	0.038	0.005	-0.017	-0.020	-0.060
Operating Profits to Assets	-0.001	-0.003	0.021	0.045	-0.009	0.002	-0.203	-0.020	1.000	-0.007	-0.045	0.243	0.016	0.034	0.097	0.018	-0.023	0.150	0.050	0.033	0.031	-0.110	0.229	0.108	0.184	0.069	0.011	0.013	0.010	0.004
Lagged Momentum	0.178	0.013	0.055	0.062	0.045	0.038	0.057	-0.018	-0.007	1.000	0.037	-0.058	-0.007	-0.040	0.007	-0.004	0.105	-0.003	-0.029	0.150	-0.029	-0.018	0.126	0.043	-0.009	-0.072	-0.136	-0.014	0.001	-0.001
Liquidity Beta 5	-0.144	-0.006	0.094	0.301	0.018	0.114	-0.158	0.139	-0.045	0.037	1.000	0.054	0.041	0.115	0.072	0.030	-0.006	0.072	0.131	0.039	0.036	-0.008	-0.042	-0.051	-0.034	-0.512	-0.017	-0.018	0.013	0.049
RD / Market Equity	-0.059	-0.027	-0.013	0.091	-0.082	-0.004	-0.226	-0.029	0.243	-0.058	0.054	1.000	-0.006	0.037	-0.098	-0.051	-0.070	0.032	-0.026	-0.030	-0.025	-0.096	-0.072	-0.070	-0.128	-0.060	0.042	0.024	0.042	-0.039
Seasonality 6-10 A	-0.007	0.017	-0.000	0.003	-0.012	0.008	-0.030	-0.017	0.016	-0.007	0.041	-0.006	1.000	0.017	-0.030	-0.017	-0.015	0.026	0.018	0.403	0.015	-0.007	0.026	-0.008	0.014	-0.014	0.007	-0.005	0.016	0.002
Seasonality 11-15 N	0.014	-0.006	-0.060	-0.014	-0.059	-0.039	-0.126	-0.014	0.034	-0.040	0.115	0.037	0.017	1.000	-0.066	-0.040	-0.064	0.048	0.048	-0.035	-0.016	-0.024	0.042	-0.066	-0.013	-0.011	0.021	-0.022	0.034	0.005
Seasonality 2-5 N	-0.119	-0.011	0.087	0.105	0.037	0.062	0.057	-0.006	0.097	0.007	0.072	-0.098	-0.030	-0.066	1.000	0.370	0.051	0.081	-0.053	0.009	-0.013	0.045	0.150	0.242	0.090	-0.069	-0.005	0.018	0.006	0.058
Momentum-Reversal	-0.060	-0.018	0.030	0.073	0.015	0.013	0.044	-0.030	0.018	-0.004	0.030	-0.051	-0.017	-0.040	0.370	1.000	0.026	0.014	-0.018	0.011	0.024	-0.010	0.100	0.099	0.030	-0.065	-0.001	-0.003	0.005	0.039
Amihud's Measure (Illiquidity)	-0.067	0.041	0.224	-0.019	0.405	0.171	0.129	-0.011	-0.023	0.105	-0.006	-0.070	-0.015	-0.064	0.051	0.026	1.000	0.051	-0.021	0.013	0.003	0.033	-0.022	0.106	-0.050	0.023	-0.494	0.044	-0.041	-0.013
Net Operating Assets	-0.038	-0.010	0.029	0.030	0.053	0.010	-0.313	-0.032	0.150	-0.003	0.072	0.032	0.026	0.048	0.081	0.014	0.051	1.000	0.092	0.030	0.032	-0.013	0.059	0.353	0.160	0.011	-0.015	-0.151	-0.017	0.046
Seasonality 6-10 N	-0.037	-0.020	-0.012	0.019	-0.025	-0.009	-0.075	-0.056	0.050	-0.029	0.131	-0.026	0.018	0.048	-0.053	-0.018	-0.021	0.092	1.000	-0.030	-0.009	-0.015	0.085	0.007	0.044	-0.049	0.014	-0.015	0.027	0.011
Seasonality	-0.021	0.017	0.038	0.035	0.012	0.029	0.006	-0.001	0.033	0.150	0.039	-0.030	0.403	-0.035	0.009	0.011	0.013	0.030	-0.030	1.000	0.555	0.009	0.058	0.060	0.029	-0.019	-0.016	0.003	-0.001	-0.002
Seasonality 2-5 A	-0.017	0.003	0.013	0.013	0.000	0.008	0.001	-0.002	0.031	-0.029	0.036	-0.025	0.015	-0.016	-0.013	0.024	0.003	0.032	-0.009	0.555	1.000	0.012	0.044	0.056	0.035	-0.022	-0.004	0.004	0.002	0.012
Accruals	-0.052	-0.006	0.045	-0.014	0.039	0.040	0.134	0.030	-0.110	-0.018	-0.008	-0.096	-0.007	-0.024	0.045	-0.010	0.033	-0.013	-0.015	0.009	0.012	1.000	-0.075	0.102	0.028	0.028	0.006	-0.004	-0.033	0.013
Duration of Equity	0.182	0.070	-0.020	-0.098	-0.000	-0.007	-0.065	-0.004	0.229	0.126	-0.042	-0.072	0.026	0.042	0.150	0.100	-0.022	0.059	0.085	0.058	0.044	-0.075	1.000	0.044	0.010	0.049	-0.055	-0.367	0.019	-0.178
Change in Common Equity	-0.073	-0.002	0.109	0.051	0.094	0.076	-0.028	-0.001	0.108	0.043	-0.051	-0.070	-0.008	-0.066	0.242	0.099	0.106	0.353	0.007	0.060	0.056	0.102	0.044	1.000	0.135	0.022	-0.037	0.091	-0.017	0.019
Profit Margin	0.064	-0.000	-0.090	-0.058	0.014	-0.080	-0.070	-0.005	0.184	-0.009	-0.034	-0.128	0.014	-0.013	0.090	0.030	-0.050	0.160	0.044	0.029	0.035	0.028	0.010	0.135	1.000	0.049	0.039	0.038	-0.005	0.217
Liquidity Beta 3	0.055	-0.010	-0.057	-0.211	-0.007	-0.077	0.034	0.038	0.069	-0.072	-0.512	-0.060	-0.014	-0.011	-0.069	-0.065	0.023	0.011	-0.049	-0.019	-0.022	0.028	0.049	0.022	0.049	1.000	0.003	-0.008	-0.037	-0.080
Liquidity Shocks	-0.117	-0.100	-0.099	-0.000	-0.140	-0.091	-0.049	0.005	0.011	-0.136	-0.017	0.042	0.007	0.021	-0.005	-0.001	-0.494	-0.015	0.014	-0.016	-0.004	0.006	-0.055	-0.037	0.039	0.003	1.000	0.003	0.012	0.037
Leverage Component of Book/Price	-0.060	-0.014	0.043	0.037	0.013	0.022	0.039	-0.017	0.013	-0.014	-0.018	0.024	-0.005	-0.022	0.018	-0.003	0.044	-0.151	-0.015	0.003	0.004	-0.004	-0.367	0.091	0.038	-0.008	0.003	1.000	-0.008	0.047
Earnings Predictability	0.026	0.004	-0.036	0.016	-0.044	-0.028	-0.049	-0.020	0.010	0.001	0.013	0.042	0.016	0.034	0.006	0.005	-0.041	-0.017	0.027	-0.001	0.002	-0.033	0.019	-0.017	-0.005	-0.037	0.012	-0.008	1.000	0.020
Earnings Forecast-to-Price	0.004	-0.066	-0.114	0.064	-0.005	-0.114	-0.090	-0.060	0.004	-0.001	0.049	-0.039	0.002	0.005	0.058	0.039	-0.013	0.046	0.011	-0.002	0.012	0.013	-0.178	0.019	0.217	-0.080	0.037	0.047	0.020	1.000

Table A.2: Features Correlation Matrix

The matrix shows pairwise correlation coefficients of all features. A visual version of this matrix is available in the main text in Figure 3.3.

Appendix B

Numerical Versions of Results

This Appendix gives numerical values of the figures in the main results, namely, Figure 4.6a and 4.7a.

	LR	NN1	NN2	NN3	NN4	Mean
Whited-Wu Index	0.017	0.241	0.430	0.364	0.187	0.306
52-Week High	0.216	0.325	0.311	0.297	0.261	0.298
Earnings Forecast-to-Price	0.290	0.226	0.266	0.413	0.231	0.284
Liquidity Beta 5	0.041	0.158	0.169	0.291	0.174	0.198
Lagged Momentum	0.313	0.206	0.165	0.186	0.126	0.171
Seasonality 6-10 A	0.206	0.137	0.185	0.228	0.119	0.167
Short-Term Reversal	0.084	0.138	0.124	0.243	0.103	0.152
Duration of Equity	0.179	0.120	0.139	0.205	0.101	0.141
Liquidity Beta 3	0.097	0.079	0.118	0.164	0.122	0.121
Maximum Return	0.169	0.082	0.086	0.149	0.079	0.099
Volume / Market Value of Equity	0.247	0.102	0.107	0.103	0.073	0.096
Idiosyncratic Risk	0.143	0.081	0.070	0.155	0.046	0.088
Momentum-Reversal	0.047	0.072	0.072	0.121	0.077	0.085
Seasonality	0.227	0.099	0.080	0.094	0.066	0.085
Seasonality 2-5 A	0.130	0.091	0.085	0.087	0.067	0.082
Net Operating Assets	0.083	0.071	0.086	0.097	0.065	0.080
Seasonality 11-15 N	0.039	0.061	0.062	0.136	0.049	0.077
Accruals	0.073	0.065	0.060	0.106	0.059	0.073
Seasonality 6-10 N	0.048	0.046	0.065	0.126	0.045	0.071
Seasonality 2-5 N	0.087	0.064	0.062	0.095	0.057	0.070
Coskewness	0.023	0.064	0.058	0.076	0.056	0.064
Operating Profits to Assets	0.050	0.053	0.051	0.096	0.038	0.059
Profit Margin	0.110	0.053	0.051	0.087	0.041	0.058
Coefficient of Variation of Share Turnover	0.040	0.034	0.030	0.061	0.028	0.038
RD / Market Equity	0.086	0.040	0.037	0.045	0.030	0.038
Leverage Component of Book/Price	0.017	0.022	0.032	0.067	0.020	0.035
Liquidity Shocks	0.073	0.031	0.029	0.046	0.026	0.033
Amihud's Measure (Illiquidity)	0.075	0.026	0.032	0.034	0.033	0.031
Change in Common Equity	0.016	0.022	0.030	0.034	0.024	0.027
Earnings Predictability	0.036	0.013	0.010	0.010	0.007	0.010

Table B.1: Values of Global Integrated Gradient

LR stands for Linear Regression, NN1 to NN4 for neural networks of respective depths. Column *Mean* represents the average value in row, across the neural networks. The features are sorted in descending order by the *Mean* column. The table is the numerical equivalent of Figure 4.6a.

	LR	NN1	NN2	NN3	NN4	Mean
Whited-Wu Index	-0.043	0.194	0.253	0.312	0.161	0.230
Earnings Forecast-to-Price	0.418	0.201	0.175	0.279	0.206	0.215
Volume / Market Value of Equity	0.465	0.281	0.187	0.132	0.197	0.199
52-Week High	-0.473	0.064	0.041	0.066	0.381	0.138
Seasonality 6-10 A	0.145	0.225	0.154	0.072	0.075	0.132
Idiosyncratic Risk	0.204	0.132	0.099	0.104	0.181	0.129
Short-Term Reversal	-0.052	0.219	0.038	0.004	0.168	0.107
Seasonality 2-5 N	0.054	0.148	0.067	0.029	0.118	0.091
Coefficient of Variation of Share Turnover	0.006	0.132	0.074	0.042	0.037	0.071
Seasonality 2-5 A	0.015	0.051	0.095	-0.002	0.114	0.065
Momentum-Reversal	0.063	0.061	0.056	0.046	0.088	0.063
RD / Market Equity	0.049	0.084	0.042	0.033	0.069	0.057
Seasonality 11-15 N	-0.023	0.092	0.079	-0.018	0.049	0.051
Seasonality 6-10 N	-0.043	0.033	0.051	0.006	0.096	0.046
Lagged Momentum	0.084	0.181	-0.013	-0.054	0.069	0.046
Duration of Equity	0.248	0.015	0.014	0.064	0.065	0.039
Amihud's Measure (Illiquidity)	0.050	0.084	0.013	-0.009	0.014	0.025
Operating Profits to Assets	-0.065	0.043	-0.011	-0.040	0.045	0.009
Change in Common Equity	-0.056	0.014	-0.009	0.005	0.013	0.006
Accruals	-0.049	0.002	-0.036	0.037	0.002	0.001
Seasonality	0.126	0.047	-0.054	-0.018	0.019	-0.001
Profit Margin	0.012	0.045	-0.063	0.031	-0.041	-0.007
Earnings Predictability	-0.044	-0.022	0.001	-0.004	-0.002	-0.007
Leverage Component of Book/Price	-0.032	0.003	-0.045	0.021	-0.011	-0.008
Coskewness	-0.003	-0.009	0.007	-0.057	0.011	-0.012
Liquidity Shocks	-0.059	0.042	-0.041	-0.051	-0.047	-0.024
Net Operating Assets	0.028	0.042	-0.112	-0.034	0.001	-0.026
Liquidity Beta 5	0.008	-0.057	-0.020	-0.100	0.050	-0.032
Maximum Return	-0.070	-0.143	-0.085	-0.065	0.050	-0.061
Liquidity Beta 3	-0.037	-0.033	-0.094	-0.079	-0.105	-0.078

Table B.2: Values of Portfolio Reliance

LR stands for Linear Regression, NN1 to NN4 for neural networks of respective depths. Column *Mean* represents the average value in row, across the neural networks. The features are sorted in descending order by the *Mean* column. The table is the numerical equivalent of Figure 4.7a.

Appendix C

Model Reliance: Additional Feature Importance Metric

As described in Chapter 3, Portfolio Reliance metric proposed in this thesis is a close cousin of Model Reliance (Fisher *et al.* 2019). Both measure global importance of a feature by permuting the feature and observing the resulting deterioration in model performance. Intuitively, if a feature is important, making it uninformative should decrease performance. The metrics differ in how performance is defined. Portfolio Reliance looks at the deterioration in mean return on long-short portfolios formed based on the model's prediction, while Model Reliance looks at the decrease in the model's loss (here, the Mean Squared Error). As presented in Section 4, the main predictive ability of the models stems from predicting extreme returns, which motivates Portfolio Reliance as a metric which studies the interpretation of the predictions at the tails of the return distribution. Model Reliance, on the other hand, studies the entire distribution of returns.

The values of Model Reliance have a ready interpretation: recall from Chapter 3 that Model Reliance is a ratio of the Mean Squared Error of the prediction without and with feature distortion. The values of Model Reliance are here above 1, meaning that distorting features harms the prediction. The higher the Model Reliance value, the more the Mean Squared Error is harmed by the distortion and the distorted feature is thus more important.

Since Model Reliance studies global feature importance in the entire returns distribution, its results can be compared to Global Integrated Gradients, the global feature importance measure employed in this thesis. The results for Model Reliance shown in Figure C.1 provide a further support for the main

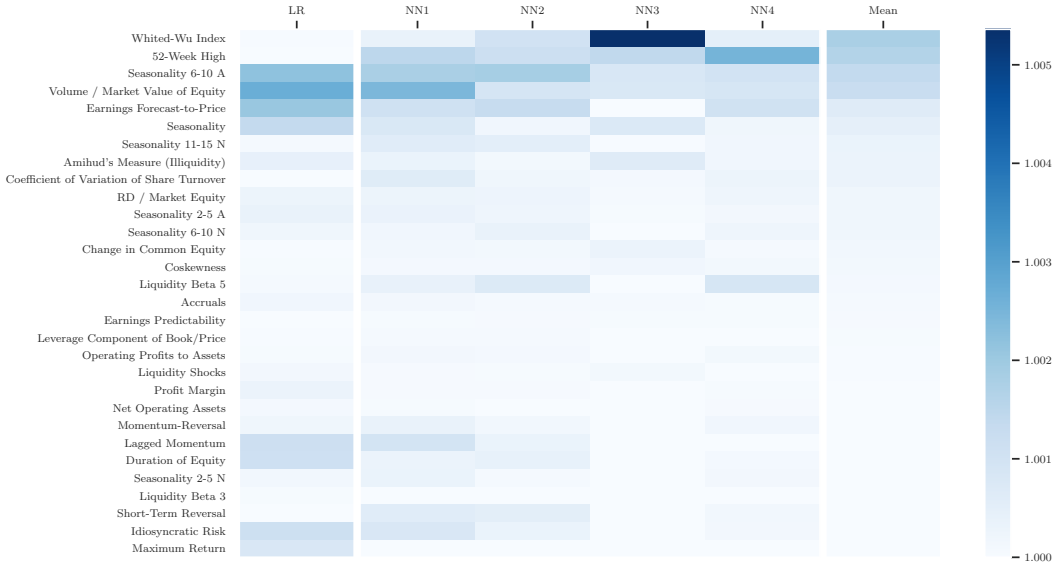


Figure C.1: Global Feature Importance Measured with Model Reliance.

Column *LR* shows feature importance in linear regression and columns *NN1* to *NN4* the neural networks of respective depths. The figure shows values of the Model Reliance for all features. The features are ordered in descending order by mean value computed across the four neural networks (column *Mean*).

result as a robustness check. The order of features by importance is similar if computed with two very different measures (Integrated Gradients and Model Reliance). However, Model Reliance seems to have a lower "sensitivity" in distinguishing feature importance, that is, many features, especially the unimportant ones, have very similar values of Model Reliance. On the other hand, Integrated Gradients appear more sensitive. A possible cause of the Model Reliance's lower "sensitivity" could be that Mean Squared Error of the predictions is quite noisy in the task of forecasting stock returns.

Appendix D

Internet Appendix

This thesis was written using Python. The entire code is available as a git repository in:

`https://github.com/KarolinaChalupova/DiplomaThesis`

Should you like to execute the code yourself, all the necessary packages can be installed from *environment.yml* file, which is provided in the repository as well. The LaTeX code is available at the same address.