Contents

List of Figures

List of Tables

1 Introduction

In recent years, two topics in the field of asset pricing have rapidly gained in significance: on the one hand machine learning and on the other hand trading costs. However, the emergence of these two topics proceeded largely independent of each other.

Modern asset pricing originated from the Capital Asset Pricing Model (CAPM) (Lintner, 1965; Mossin, 1966; Sharpe, 1964) which was developed in an era when data were scarce and computing power was limited. Based on restrictive assumptions such as, in particular, *no trading costs*, the CAPM is an equilibrium model derived from utility and portfolio theory. It implies that the expected return of a stock is a *linear* function in just *one variable*, the stock's sensitivity to the market risk. Hence, despite its theory-based approach, the CAPM suggests a simple linear regression setup to empirically model returns.

In subsequent years, however, several studies have found anomalies, i.e. other stock-specific characteristics that explain variations in stock returns that remain unexplained by a given asset pricing model and hence contradict it.[1] In response, new asset pricing models were constructed by augmenting the CAPM to include additional factors besides the market factor while staying with its linear form (Fama & French, 1993, 2015).

With increasing data availability and computational capacity, the number of studies claiming to have discovered new anomaly characteristics has grown rapidly to over 300 (Harvey, Liu, & Zhu, 2016). This development gave rise to the following fundamental questions: Firstly, what is the after trading cost performance of these anomalies? Until then, most of the research in asset pricing focused on gross returns without considering trading costs (Frazzini, Israel, & Moskowitz, 2012). However, many of the anomaly characteristics imply trading strategies with positive gross returns, but high turnover and trading costs that reduce the net return, in extreme cases even to unprofitability. Secondly, how can the information contained in all the anomaly characteristics be used jointly to model stock returns? In this high-dimensional setting with highly correlated variables, simple linear regression is prone to overfitting and will therefore not yield satisfactory results. Instead, other methods from the domain of supervised machine learning can help overcome these problems.

In response to the first question, several studies have examined the performance *after trading costs* of separate anomalies.[2] To address the second question several authors have applied machine learning techniques allowing for *nonlinearities* and interactions between *multiple variables* to asset pricing.[3]

[1] See e.g. Banz (1981); Basu (1977); Rosenberg, Reid, and Lanstein (1985)
[2] See e.g. A. Y. Chen and Velikov (2017); Novy-Marx and Velikov (2016)
[3] See e.g. Gu, Kelly, and Xiu (2020)

However, the fact that the two topics, trading costs and machine learning, have gained importance largely independently of each other seems surprising, as machine learning based trading strategies appear to generate high gross returns but often involve high turnovers, which diminish net returns. Therefore, I study the after-cost performance of machine learning based trading strategies by combining the machine learning models presented by Gu et al. (2020) and the trading costs provided by A. Y. Chen and Zimmermann (in press).

For this purpose, I proceed as follows: Given a dataset containing 202 firm-level characteristics, I use classical and advanced regression models as well as tree-based methods and neural networks to predict the gross abnormal return of stocks. Next, machine learning portfolios are constructed by means of a value-weighted long-short decile spread strategy formed on the basis of the gross 1-month-ahead out-of-sample predicted abnormal returns. Subsequently, I analyze the predictive power of the various machine learning models and the gross and net out-of-sample performance of the machine learning portfolios. For these analyses a simple linear model based solely on marketcap, book-to-market and 12-month momentum, as recommended by (Lewellen, 2015), is used as benchmark. In addition, I try to further improve the after-cost performance of machine learning portfolios in two different ways: On the one hand, by reducing the turnover and therefore the trading costs of the machine learning portfolios. In order to do so, I adopt a turnover mitigation sorting approach, also known as buy/hold spreads, as proposed by Novy-Marx and Velikov (2016). On the other hand, by further improving the gross performance of the machine learning portfolios. For this purpose, I construct ensemble models which combine individual machine learning models as suggested by Rasekhschaffe and Jones (2019).

Similar to (Gu et al., 2020), I find that in terms of out-of-sample predictive power and out-of-sample gross performance of the machine learning portfolios, some of the advanced regression models deliver medium sized, whereas tree-based methods and neural networks yield substantial improvements over the benchmark. After trading costs, however, only the machine learning portfolios based on neural networks outperform the benchmark. These results also hold after adjusting for risk suggesting that the outperformance is not caused by higher risk. Furthermore, I find that ensemble models seem to be more promising in improving net returns than turnover mitigation sorting.

I contribute to the trading cost literature by studying the trading costs and net returns of trading strategies that consider a large set of characteristics simultaneously similar to DeMiguel et al. (2020). In contrast to DeMiguel et al. (2020), machine learning portfolios do not combine results from traditional portfolio sorts based on one characteristic at a time but rather use all characteristics to predict returns based on which the portfolio is then constructed.

My contribution to the literature on machine learning in asset pricing is twofold: Firstly, by investigating the net returns, I assess the returns investors can expect from machine learning portfolios, which are crucial for their investment decision. This is all the more important as not all machine learning portfolios that have high gross returns generate net returns which are greater than those of classical approaches. Secondly, I demonstrate the potential of ensemble models in enhancing the gross and net performance of machine learning portfolios.

The remainder of this study is structured in the following way: Section 2 briefly reviews the existing literature on machine learning in asset pricing as well as on trading costs. Section 3 describes the data used for the analyses as well as the data preprocessing steps performed. In Section 4 the theoretical basis for the further analyses is explained, in particular the different machine learning methods and how their predictions can be used to construct portfolios. In Section 5, the results on the performance of the different machine learning models are presented, compared with each other and with the literature, and interpreted. Section 6 concludes.

2 Literature Review

Although traditional methods have still dominated publications of the last years (Weigand, 2019), machine learning is a trending topic in asset pricing. As Drobetz and Otto (2021) point out, most existing studies on machine learning in asset pricing examine one type of model at a time. For instance, Feng, Giglio, and Xiu (2020), who apply a factor selection procedure in an attempt to "tame the factor zoo", and Kozak, Nagel, and Santosh (2020) use special types of regularized regressions to study the cross-section of returns. Moritz and Zimmermann (2016) and Hanauer, Kononova, and Rapp (2022) demonstrate how tree-based methods can be utilized to construct trading strategies. The former use lagged returns to estimate future returns, upon which the trading strategy rests. The latter predict monthly peer-implied fair values of European stocks based on a set of accounting variables, then use the prediction to calculate a mispricing signal, upon which the trading strategy is then built. Others employ neural networks to predict stock returns (e.g. L. Chen, Pelger, and Zhu (2019); Feng, He, and Polson (2018); Messmer (2017)).

Besides these, there is a conceptually different approach which makes use of dimensionality reduction techniques such as Instrumented Principal Component Analysis (IPCA) (Kelly, Pruitt, & Su, 2020) or autoencoders (Gu, Kelly, & Xiu, 2021).[4] Instead of directly modelling stock returns as a function of stock characteristics, this approach models stock returns as a function of latent common factors. The stock-specific exposure to these latent common factors are in turn modelled as linear function (IPCA) or as flexible nonlinear function (autoencoder) of the stock characteristics. Unlike the others, this approach does not just allow to predict stock returns but also to extract latent common factors from a large set of characteristics.

Apart from these studies that investigate one type of model at a time, there are just a few that cover and compare multiple types of models at the same time, most notably Gu et al. (2020).[5] Their analyses include classical and advanced regression models, tree-based methods and neural networks. They use these machine learning models to predict gross stock returns given a large set of characteristics. They find that compared to classical and advanced regression models tree-based methods as well as neural network have a considerably higher out-of-sample predictive power and attribute this to their ability to capture nonlinearities and interactions between features. Moreover, they demonstrate that portfolios formed on the basis of the gross 1-month-ahead out-of-sample returns predicted by the different machine learning models achieve substantially higher gross returns than those based on classical regression models,

[4] Windmüller (2021) applies IPCA to a global sample.

[5] Drobetz and Otto (2021) and Cakici and Zaremba (2022) replicate the analyses of Gu et al. (2020) on a European and global sample, respectively.

even after adjusting for risk. Although they also report the turnovers of these portfolios, which turn out to be high suggesting that the trading costs are high too, they do not examine the after-cost performance of machine learning portfolios. Furthermore, despite studying a broad set of machine learning models, they do not analyze the performance of an ensemble or composite model that combines multiple models.

Rasekhschaffe and Jones (2019) also study different types of machine learning models for predicting stock returns. They observe that neural networks are the best performing single models. Additionally, they also analyze an ensemble model combining all single models. They find that this ensemble model outperforms even the best performing single model. This highlights the potential of ensemble models to further improve the already high gross performance of single machine learning models. However, Rasekhschaffe and Jones (2019) do not take turnovers and trading costs into account.

Most of the existing literature on trading costs focuses on the costs of trading strategies based on single characteristics (DeMiguel et al., 2020). A. Y. Chen and Velikov (2017), who examine 120 anomalies, state that despite impressive gross returns after trading costs, a large proportion of anomalies yield only small positive returns, if any at all. Thus, they conclude that trading costs have a massive effect. However, DeMiguel et al. (2020), who in contrast to most studies evaluate the trading costs of strategies based on multiple characteristics, show that the effect of trading costs is lower when using a trading strategy that considers multiple characteristics simultaneously. Comparing several techniques, which aim at reducing trading costs, for trading strategies based on single characteristics, Novy-Marx and Velikov (2016, 2019) find buy/hold spreads, which I will present later in more detail, to be most effective.

Since the literature suggests that many trading strategies based on single characteristics earn only small positive returns, if any at all, it is even more important to investigate the after-cost performance of machine learning portfolios since they usually have high turnovers resulting in high trading costs. Based on the literature, two promising approaches for improving the after-cost performance of machine learning portfolios seem to be the construction of ensembles models and buy/hold spreads.

3 Data

Having motivated the research approach of this study by means of a short literature review, in this section I will briefly explain from which sources the data used in this analysis originate as well as the data preprocessing steps that were performed before applying the machine learning models.

3.1 Data Sources

The data cover monthly information about individual US equities coming from two different sources. Firstly, I obtain price and gross performance data from the Center for Research in Security Prices (CRSP) provided via Wharton Research Database (WRDS) (2021). Secondly, I retrieve data on 202 firm-level characteristics from A. Y. Chen and Zimmermann (in press).[6] Monthly, quarterly and yearly updated characteristics are already lagged by 1, 4 and 6 months, respectively. After merging the data from both sources, the resulting data can be used for training and evaluating various machine learning models for predicting a stock's gross performance. In addition, A. Y. Chen and Zimmermann (in press) also provide data on stock-level trading costs as measured by half of the spread. These trading cost data allows to assess the after-cost performance of portfolios formed based on the gross performance predicted by various machine learning models.

3.2 Data Preprocessing

Before using the data for training various machine learning models, I conducted several data filtering and preprocessing steps. I filter for common shares (share-code 10 and 11) of firms listed in the NYSE, AMEX, and NASDAQ (exchange code 1, 2 and 3) excluding financial firms (SIC code between 6000 and 6999) as proposed by Hou, Xue, and Zhang (2020). Moreover, I exclude microcaps, i.e. stocks with a market capitalization smaller than the 20th percentile of the market capitalization of NYSE stocks. Microcaps are only 3% of of the total market capitalization but they account for 60% of the number of stocks (Fama & French, 2008). Therefore, not excluding microcaps would result in training machine learning models on a dataset that is dominated by micro-caps even though their economic releveance is very low. Finally, I restrict my further analyses to the period between 1970 and 2017 for two reasons: on the one hand, before 1970 most characteristics in the dataset have low coverage.[7] On the other hand, A. Y. Chen and Zimmermann (in press) provide trading

[6] For definitions of all the characteristics, the reader may refer to Table 6 in the Appendix.
[7] For a plot visualizing the number of characteristics with annual coverage $\geq 90\%$, see Figure 17 in the Appendix.

cost data only up until 2017. The resulting dataset consists of 7,828 unique stocks and 809,180 stock-month observations with an average of 1,405 stocks per month.

Apart from these filterings, I adjust monthly stock returns and scale characteristics. In order to avoid delisting bias Shumway (1997), holding period returns are adjusted for delisting by compounding holding period returns in the month prior to delisting with delisting returns (Beaver, McNichols, & Price, 2007; Hou et al., 2020). Furthermore, to limit the negative effect that outliers in the target variable have on training machine learning models, the returns used for training the models are cross-sectionally winsorized at the 1% and 99% level. However, the evaluation of machine learning portfolios is conducted on unwinsorized returns. In order to limit the negative effect of outliers in the characteristics on training machine learning models, characteristics are cross-sectionally standardized using rank-based scaling (Kelly, Pruitt, & Su, 2019).[8] For rank-based scaling, I calculate in each month the ranks of the stocks for each characteristic separately. Then, I divide the ranks by the number of non-missing observations of that characteristic in that month and subtract 0.5. As a result, all characteristics take only values in the interval $[-0.5, +0.5]$. Lastly, missing characteristic values are imputed using the cross-sectional median after rank-based scaling, i.e. 0.

[8]For descriptive statistics of the unstandardized characteristics, the reader may refer to Table 7 in the Appendix.

4 Methodology

In this section, the general framework for predicting returns via machine learning is described first. Next, the machine learning models used in the further analyses are explained. Afterwards, I will outline how the sample is splitted to train and tune machine learning models as well as to assess their out-of-sample prediction performance. To analyze the hidden structures of machine learning models the concept of Shapley-value based feature importance will be briefly introduced. Finally, details on the construction of machine learning portfolios will be presented.

4.1 Return Prediction Using Machine Learning

The abnormal return of a stock i, $i = 1, \ldots, N$, in month t, $t = 1, \ldots, T$, can be calculated as

$$r_{i,t}^{abn} = r_{i,t} - r_{M,t}, \tag{1}$$

where $r_{i,t}$ is the return of stock i in month t and $r_{M,t}$ is the value-weighted market return in month t. Similar to Gu et al. (2020), using an additive prediction error model the 1-month-ahead abnormal return of a stock, $r_{i,t+1}^{abn}$, can be written as

$$r_{i,t+1}^{abn} = E_t[r_{i,t+1}^{abn}|x_{i,t}] + \epsilon_{i,t+1}, \tag{2}$$

where $E_t[r_{i,t+1}^{abn}|x_{i,t}]$ is the conditional expectation in month t of stock i's abnormal return in month $t+1$ given a p-dimensional column vector of stock-specific characteristics known at t, $x_{i,t} \in \mathbb{R}^p$. $\epsilon_{i,t+1}$ is an additive prediction error term. In turn, the conditional expectation can be modeled as an unknown function f^*, $f^* : \mathbb{R}^p \to \mathbb{R}$, of the stock-specific characteristics, i.e.

$$E_t[r_{i,t+1}^{abn}|x_{i,t}] = f^*(x_{i,t}). \tag{3}$$

Machine learning models from the domain of supervised learning can be trained to approximate this unkown function $f^*(\cdot)$ by some function $f(\cdot, \theta)$ which is parametrized by a vector of coefficients θ. To learn θ from a given dataset, some loss function $\mathcal{L}$, which measures the distance between the observed and the predicted values, needs to be minimized with respect to θ. Besides the coefficients, most machine learning models contain hyperparameters which cannot be learned from the dataset but need to be tuned by the user. The functional form of f as well as the loss function vary across different machine learning models. Given the stock-specific characteristics of a stock known at the end of month t, a trained and tuned machine learning model can be used to make

a prediction, $\hat{r}^{abn}_{i,t+1}$, of the abnormal return of that stock for the next month (predicted values are indicated by $\hat{}$).

In order to construct a machine learning portfolio, Gu et al. (2020) sort all stocks according to their 1-month-ahead out-of-sample predicted return into deciles and then buy the stocks with the highest predicted returns (top decile) and sell those with the lowest predicted returns (bottom decile). However, this sorting does not depend on the general level of predicted returns across all stocks, i.e. the value-weighted predicted market return, but rather on a stock's predicted return relative to the predicted return of other stocks. For this reason, sorting based on the predicted return of stocks yields identical result as sorting based on the predicted abnormal return of stocks, which is defined as the predicted return of a stock minus the value-weighted predicted market return (i.e. it removes the general level of predicted returns across all stocks).

Based on this, I decided to use for the prediction as well as the sorting the abnormal return instead of the return for two reasons:

Firstly, since I focus on examining machine learning portfolios, I am not interested in the exact value of the predicted return of a single stock or portfolio except the machine learning based long-short portfolio. Instead, I am just interested to know for each stock whether it is predicted to perform substantially better or worse than the average (i.e. abnormal return) to then allocate it to the long or short portfolio accordingly. Put differently, I am solely interested in the cross-section of stock returns, not the equity premium. The predicted return of a long-short portfolio, however, can directly be calculated from the predicted abnormal return of the long and of the short portfolio by subtracting both, since the value-weighted predicted market return cancels out. This can be seen from the following equations:

$$\begin{aligned} \hat{r}^{abn}_{L,t+1} - \hat{r}^{abn}_{S,t+1} &= (\hat{r}_{L,t+1} - \hat{r}_{M,t+1}) - (\hat{r}_{S,t+1} - \hat{r}_{M,t+1}) \\ &= \hat{r}_{L,t+1} - \hat{r}_{S,t+1} = \hat{r}_{LS,t+1}, \end{aligned} \tag{4}$$

where $\hat{r}_{\cdot,t+1}$ and $\hat{r}^{abn}_{\cdot,t+1}$ are the predicted returns and the predicted abnormal returns of the long- (L), the short- (S), the long-short (LS) and the value-weighted market portfolio (M) for month $t+1$, respectively.

Secondly, the abnormal return prediction has a higher signal-to-noise ratio than that of the return, which makes it easier for machine learning models to learn the patterns. Subtracting the return of the value-weighted market portfolio removes noise, i.e. variance in the data that contains no information with respect to the goal of the prediction, namely which stocks perform substantially better or worse than the average. Otherwise, a large part of the variance in the target variable would be due to market movements, although

I am not interested in these, which, however, make forecasting more difficult. Moreover, calculating abnormal returns can be considered to be some kind of cross-sectional centering. Therefore, using abnormal returns makes observations from different months with different market returns more comparable (Rasekhschaffe & Jones, 2019).

4.2 Machine Learning Models

Having shortly outlined the general framework for using machine learning to predict a stock's abnormal return, I will now describe all machine learning models I applied in this context. For each model, the description is structured in the following way: First, I will elaborate on the basic intuition behind the respective model. Then, I will state its functional form, i.e. the structure of the specific function $f(\cdot, \theta)$, as well as its loss function and, based on this, explain how it is trained. Ultimately, I will compare it to the other models and explain its advantages and disadvantages. For reasons of convenience, I shall refer to $r_{i,t+1}^{abn}$ as target variable and to $x_{i,t}$ as feature vector or features.

4.2.1 Simple Linear Regression

Simple linear regression assumes that the conditional expectation of the target variable is linear in the features. Hence, simple linear regression has the following functional form:

$$f(x_{i,t}, \theta) = \theta^\top x_{i,t}, \tag{5}$$

where θ, $\theta^\top = (\theta_1, \theta_2, \ldots, \theta_p) \in \mathbb{R}^p$, is a column vector of coefficients. These coefficients can be estimated using Ordinary Least Squares (OLS). It is based on the following loss function:

$$\mathcal{L}_{MSE}(\theta) = \frac{1}{NT} \sum_{i=1}^{N} \sum_{t=1}^{T} \left(r_{i,t+1}^{abn} - f(x_{i,t}, \theta) \right)^2 \tag{6}$$

which is also known as the Mean Squared Error (MSE). OLS returns the vector of coefficients, which minimizes MSE.

Since this minimization problem has an analytical solution, estimating the coefficients of the simple linear regression model is computationally not expensive. Moreover, in contrast to all other models in this analysis the above stated model does not contain any hyperparameters which would need to be tuned. Another advantage of simple linear regression models is that they are easy to interpret as the coefficients directly describe how a change in the features affects the target variable. Furthermore, if there is just a small number

of training samples or sparse data, simple linear regression can sometimes perform better than more sophisticated nonlinear models (Hastie et al., 2009).

Unfortunately, simple linear regression does not inherently take interactions and nonlinearities into account. Even though simple linear regression can be applied to nonlinear transformations of and interactions between features, those need to be specified and included explicitly by the user upfront. Another shortcoming is that if features are highly correlated the estimated coefficients as well as their statistical significance get undermined leading to fallacious interpretations which is also known as the multicollinearity problem.[9] Probably the most serious disadvantage of simple linear regression models is that they are prone to overfitting. With high-dimensional data, they struggle to distinguish between signal and noise. As a consequence, they also model noise which results in a higher in-sample performance but a substantially lower out-of-sample performance.[10]

4.2.2 Regularized Regression Methods

In order to avoid the above described problem of overfitting, one may resort to regularized regression techniques such as ridge regression, lasso regression or elastic net. The simple linear regression model and these regularized regression techniques share the same functional form (see Equation 5) but they differ with respect to the loss function. Each regularized regression method adds a different penalty to the MSE loss function. These penalties force the regularized regression techniques to not just fit coefficients as best as possible to the data, but also to keep the coefficients as small as possible. Thus, the coefficients of noisy features get shrunk effectively reducing overfitting. Nevertheless, since θ is multidimensional, it is not immediately clear what small means in this context. Each of the regularized regression methods has a different approach to measuring the size of θ (ℓ_1-norm, ℓ_2-norm or a mixture of both) resulting in different loss functions and therefore they return different coefficients.

Ridge regression augments the MSE loss function by adding as a penalty the ℓ_2-norm of the vector of coefficients multiplied by a non-negative scalar λ (Hoerl & Kennard, 1970):

$$\begin{aligned} \mathcal{L}_{Ridge}(\theta, \lambda) &= \mathcal{L}_{MSE}(\theta) + \lambda \sum_{j=1}^{p} \theta_j^2 \\ &= \frac{1}{NT} \sum_{i=1}^{N} \sum_{t=1}^{T} \left(r_{i,t+1}^{abn} - f(x_{i,t}, \theta) \right)^2 + \lambda \sum_{j=1}^{p} \theta_j^2. \end{aligned} \tag{7}$$

[9] For further details on multicollinearity, see Allen (1997).
[10] For further details on simple linear regression models, please refer to Fahrmeir et al. (2013).

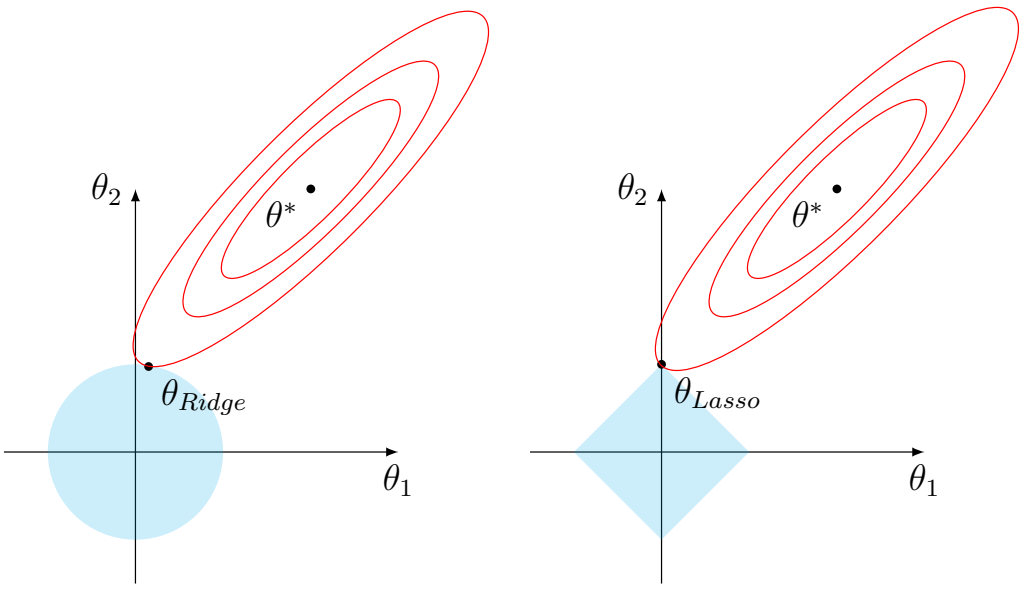

Figure 1: Comparison between ridge regression and lasso regression
This figure depicts the optimal coefficients under OLS (θ^*), the contour lines of the MSE loss function (red ellipses) as well as the constraint regions (blue areas) and the optimal coefficients under ridge regression (left), θ_{Rdige}, and lasso regression (right), θ_{Lasso}, based on Hastie et al. (2009).

Instead of the ℓ_2-norm, lasso regression adds the ℓ_1-norm of the vector of coefficients multiplied by a non-negative scalar λ as a penalty to the MSE loss function (Tibshirani, 1996):

$$\begin{aligned} \mathcal{L}_{Lasso}(\theta, \lambda) &= \mathcal{L}_{MSE}(\theta) + \lambda \sum_{j=1}^{p} |\theta_j| \\ &= \frac{1}{NT} \sum_{i=1}^{N} \sum_{t=1}^{T} \left(r_{i,t+1}^{abn} - f(x_{i,t}, \theta)\right)^2 + \lambda \sum_{j=1}^{p} |\theta_j|. \end{aligned} \tag{8}$$

The fact that in contrast to ridge regression lasso regression uses the ℓ_1-norm instead of the ℓ_2-norm for the penalty term might at first sight seem to be a minor difference. However, this seemlingly minor difference has important implications on the coefficients and thereby also on the estimated models.

The loss function minimization problems of ridge and lasso regression can equivalently be formulated as constrained minimization problems with MSE as objective function, respectively. The constraint states that the respective norm of the vector of coefficients has to be smaller than or equal to some constant. For the 2-dimensional case, this can visually be exemplified by Figure 1. θ^* is the solution of the unconstrained minimization of MSE, i.e. the optimal solution under OLS. Each of the red contour lines indicates a set of combinations of coefficients $((\theta_1, \theta_2))$ for which the MSE is constant. The further a contours line is away from θ^*, the larger is the corresponding value of the MSE. The light blue areas represent the set of combinations of coefficients which satisfy the respective constraint. Obviously, the shape of the set of combinations

of coefficients which satisfy the respective constraint heavily depends on the norm. Under the constrained minimization problem, the optimal combination of coefficients is graphically given by the tangency point of the blue area and the red contour lines (θ_{Ridge} and θ_{Lasso}, respectively).

Ceteris paribus, the optimal solution heavily depends on the shape of the constraint area and therefore also on the used norm. Under the ℓ_1-norm, the constraint area has a square shape. Therefore, the optimal solution is often at one of the corners (i.e. on one of the coordinate axis), hence one of the two coefficients is zero. In contrast, under the ℓ_2-norm, the constraint area has a round shape. Thus, the optimal solution is usually not on one of the coordinate axes. As a consequence, both coefficients are unequal to zero. In summary, due to the geometry of the ℓ_2-norm and ℓ_1-norm, ridge regression shrinks the coefficients of all features more or less equally, whereas lasso regression shrinks the coefficients of certain features to zero while leaving the coefficient of others rather unchanged. For that reason, ridge regression is considered to be a shrinkage method and lasso regression to be a feature selection method.

Elastic net adds as a penalty a combination of the ℓ_1- and the ℓ_2-norm of the vector of coefficients multiplied by a non-negative scalar λ to MSE (see Equation 9). Hence, elastic net is a mixture of ridge and lasso regression. The exact mixture is determined by the mixing hyperparameter α, where $\alpha = 0$ corresponds to lasso regression and $\alpha = 1$ to ridge regression (Zou & Hastie, 2005):

$$\begin{aligned}\mathcal{L}_{Elast}(\theta, \lambda, \alpha) &= \mathcal{L}_{MSE}(\theta) + \lambda \sum_{j=1}^{p} (\alpha\theta_j^2 + (1-\alpha)|\theta_j|) \\ &= \frac{1}{NT} \sum_{i=1}^{N} \sum_{t=1}^{T} \left(r_{i,t+1}^{abn} - f(x_{i,t}, \theta)\right)^2 + \lambda \sum_{j=1}^{p} (\alpha\theta_j^2 + (1-\alpha)|\theta_j|).\end{aligned} \tag{9}$$

As outlined above, the geometric shape of the constraint region has a crucial influence on the coefficients. Figure 2 shows the geometric shape of the constraint region for elastic net with $\alpha = 0.5$ as well as for ridge and lasso regression for the 2-dimensional case. It becomes visually evident that elastic net lies between ridge and lasso regression. Thus, elastic net combines shrinkage as well as feature selection.

Ridge regression, lasso regression and elastic net share the hyperparameter λ. The efffect that λ has on the coefficients is similar across these three models: For $\lambda = 0$, the regularized regression methods are identical to simple linear regression. As λ is increased the regularized regression techniques shrink the coefficients closer to zero, however, as described above, each method does this in its own way.

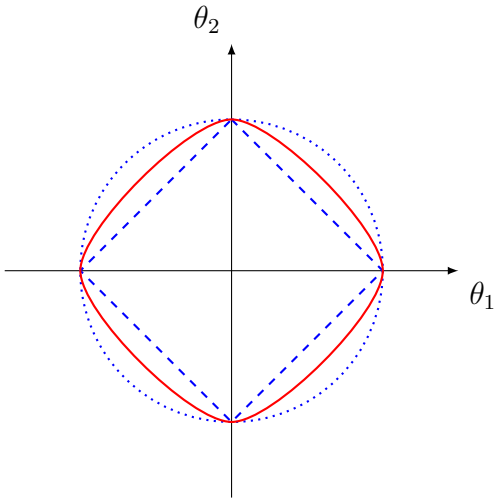

Figure 2: Shape of the constraint region under elastic net
This figure depicts the shape of the constraint region under elastic net for $\alpha = 0.5$ (——) as well as for $\alpha = 0$ (- - -), i.e. lasso regression, and $\alpha = 1$ (·····), i.e. ridge regression, based on Zou and Hastie (2005).

For ridge regression, the problem of finding the vector of coefficients which minimizes $\mathcal{L}_{Ridge}$ has an analytical solution, hence it is not computationally expensive. In contrast, for lasso regression and elastic net the minimization of the their loss functions with respect to the vector of coefficients has no analytical solution. Furthermore, since their loss functions are not differentiable, one cannot resort to classical gradient descent. Nevertheless, coefficients can be computed using algorithms such as proximal gradient descent or least-angle regression, which come at slightly higher computational costs.

In contrast to the vector of coefficients, the hyperparameter λ cannot be directly estimated from the data, but needs to be tuned by the user. This comes with an additional computational effort as compared to OLS. Moreover, for elastic net in addition to λ also α needs to be tuned.

Since the regularized regression methods have the same functional form as simple linear regression, they also share the advantages and disadvantages arising from it. To be precise, regularized regression models are interpretable but they do not account for nonlinearities or interactions between features. But unlike OLS, the regularized regression methods do not tend to overfit in the presence of high-dimensional data, because they add different penalties to the loss function used by OLS.

4.2.3 Principal Component Regression

Principal component regression (PCR) is another method to avoid overfitting when applying linear regression to highdimensional data. In contrast to linear regression and regularized regression methods, PCR does not directly fit a linear model to the data. Instead, it first reduces the dimension of the feature

space before training a simple linear regression.

To be precise, PCR performs the following two steps: Firstly, a principal component analysis (PCA) is applied solely to the features. PCA aims at reducing the dimensionality of the data while retaining as much information, measured by the variance, as possible. It projects each observation in th p-dimensional feature space, x_{it}, via a linear transformation Φ, $\Phi \in \mathbb{R}^{q \times p}$, onto a point in a q-dimensional subspace, z_{it}, where $q < p$.

$$z_{i,t} = \Phi x_{i,t} \tag{10}$$

Secondly, the target variable is regressed against the projected observations of the features using MSE as a loss function. Consequently, PCR has the following functional form:

$$f(x_{i,t}, \theta) = \theta^\top z_{i,t} = \theta^\top \Phi x_{i,t} \tag{11}$$

where $\theta \in \mathbb{R}^q$.[11]

The estimation of Φ has an analytical solution that is based on the eigenvalues of the correlation matrix of the features. Since the estimation of θ via OLS also has an analytical solution, PCR is computationally not expensive. Nevertheless, q, the dimension of the subspace onto which PCA projects, is a hyperparameter which requires tuning.

At first sight, the coefficients returned by PCR might seem to be hardly interpretable due to the projection of the observations onto a lower dimensional subspace. But, the effective coefficients of all features, which can directly be interpreted as in the case of simple linear regression, can be obtained by calculating $\theta^\top \Phi$. Another advantage of PCR is that it helps to circumvene the problem of overfitting occurring under OLS in the context of high-dimensional data. Moreover, it is also a remedy against the multicollinearity problem since the principal components generated by PCA, which summarize the information contained in different groups of highly correlated features, are uncorrelated to each other.

A disadvantage which PCR shares with the other models discussed so far is that it does not inherently incorporate nonlinearities and interactions between features, since the functional form of a linear model applied to a linear transformation of the features is still linear. By definition, PCA, which is solely applied to the features, tries to capture the largest fraction of the variance contained in the features while neglecting the target variable. As a consequence, PCA cannot distinguish between noise, which is variance unrelated to the target variable, and signal, which is variance related to the target variable

[11] For details on PCR and PCA, please refer to Izenman (2008).

and thus useful for prediction. This is probably the most serious disadvantage of PCR.

4.2.4 Tree-Based Methods

Since the functional forms of all the machine learning models presented so far are linear, they all share the advantages and disadvantages arising from this. In particular, these models struggle when interactions between features are predictive of the target variable or the relationship between the features and the target variable is nonlinear. In response to these two shortcomings, one may resort to tree-based methods.

A regression tree of depth D with K leaves partitions the feature space $\mathbb{R}^p$ into a set of K cuboid regions $\theta = \{\theta_1, \theta_2, \ldots, \theta_K\}$. The edges of those regions are either parallel or orthogonal to the coordinate axes. Inside each of these regions $\theta_k, k = 1, 2, \ldots, K$, the target variable is modelled to be constant, c_k. Obviously, setting this constant for each region equal to the mean of the target variable of the observations belonging to that region minimizes MSE (Bishop & Nasrabadi, 2006). In essence, the functional form of a regression tree is given by:

$$f(x_{i,t}, \theta) = \sum_{l=1}^{K} c_k \mathbb{1}_{\{x_{i,t} \in \theta_k\}} \tag{12}$$

$$c_k = \frac{1}{N_k} \sum_{x_{i,t} \in \theta_k} r_{i,t+1}, \tag{13}$$

where N_k is the number of observations belonging to region θ_k. Figures 3 and 4 depict the diagram of, the partition induced by and the predicted values of the same exemplary simplified regression tree for a 2-dimensional feature space.

In order to fit a regression tree and to obtain the partition θ, recursive binary splitting is used. It proceeds as follows: Starting with the complete feature space at the root node, every distinct value of every feature is searched to find the split variable and split value that partitions the data into two regions such that the MSE after the split is minimized. Then, the same splitting procedure is applied separately to each of the newly created regions. This process is repeated until either a specified depth is reached or the improvement in MSE becomes smaller than some tolerance (Kuhn & Johnson, 2013).

Unfortunately, the out-of-sample performance of single regression trees is usually poor. One reason for this is that single regression trees are prone to overfitting. There are, however, tree-based methods such as random forest (Breiman, 2001) or gradient boosted regression trees (Friedman, 2001) that are less prone to overfitting. A random forest fits a large number, B, of regression trees and then creates a so called ensemble model by taking the average

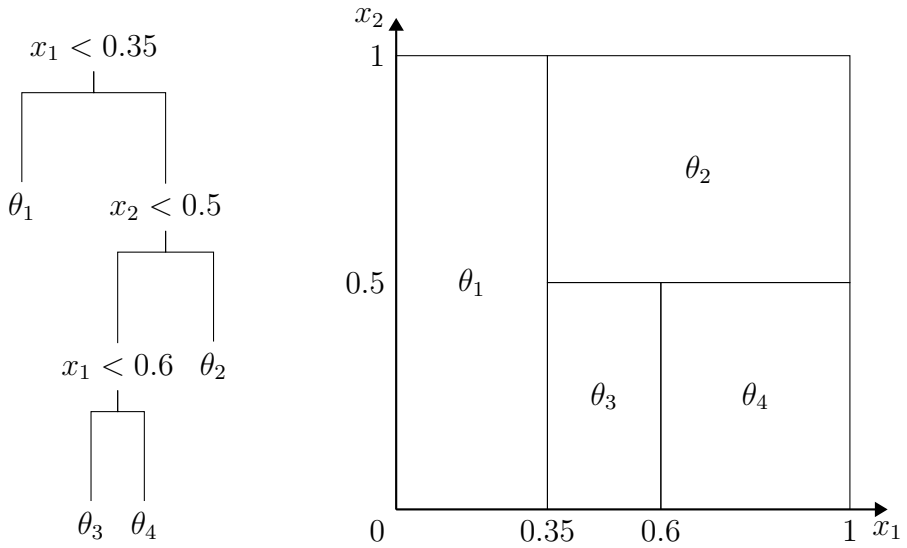

Figure 3: Exemplary regression tree

This figure depicts an exemplary regression tree of depth 3 with 4 leaves (left) as well as the partition of the 2-dimensional feature space into 4 regions induced by the exemplary regression tree (right) based on Hastie et al. (2009).

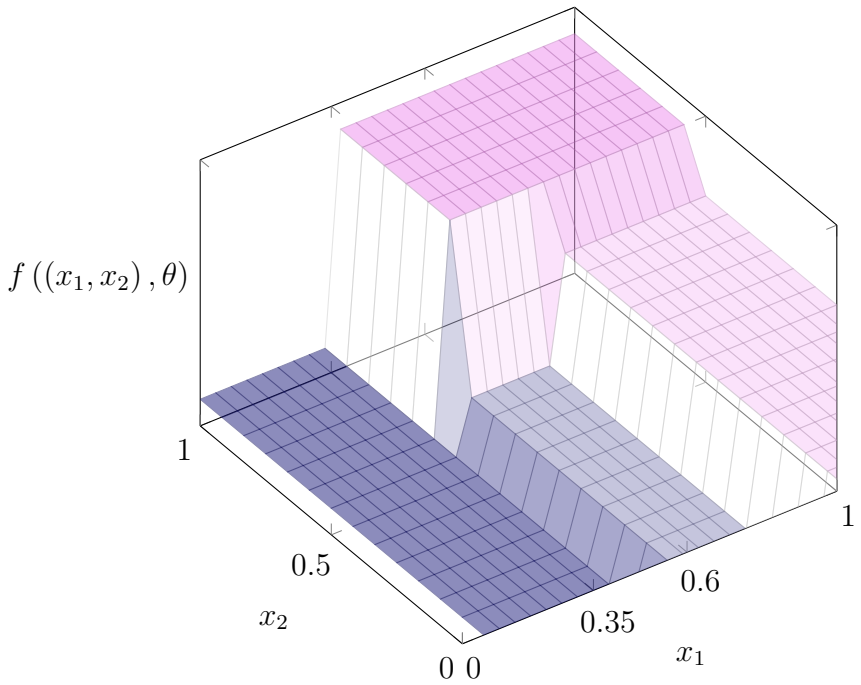

Figure 4: Exemplary regression tree (cont.)

This figure depicts a perspective plot corresponding to the exemplary regression tree displayed Figure 3 based on Hastie et al. (2009).

across those single regression trees. More specifically, B bootstrap samples (i.e. sampling with replacement) are drawn from the original data. For each of these bootstrap samples a regression tree is trained, however at each node only a random subset of the p features is considered for splitting. To generate a prediction from a trained random forest for given x_{it}, each of the B regression trees makes a prediction, these predictions are added together and divided by B to obtain the prediction of the random forest (Kuhn & Johnson, 2013).

Similarly to random forest, gradient boosted regression trees also fit a large number, B, of regression trees. However, in contrast to random forests these single trees are neither trained independently of each other nor do they equally contribute to the overall model. For the case of the MSE loss function, a gradient boosted regression tree is trained in the following way: A shallow regression tree is trained based on the complete training sample. Subsequently, the residuals of the previously trained model are calculated and a new tree is trained onto these residuals. Then, the predictions of the newly trained tree are scaled by the so called learning rate LR, $LR \in (0, 1)$, and added to the predictions of the previous model creating a new ensemble model. This process is repeated until $B - 1$ regression trees have been trained onto the respective residuals (James et al., 2013).

Since opposed to all other models discussed here tree-based methods are completely non-parametric, they are characterized by a high degree of flexibilty which allows them to also account for nonlinearities as well as interactions. Nevertheless, this comes at a cost: Firstly, random forests and gradient boosted regression trees are computationally very expensive. This is due to the fact that both methods fit a large number of single regression trees to build an ensemble model and both have hyperparameters that need to be tuned. Secondly, random forest and gradient boosted regression trees cannot directly be interpreted. Single regression trees are directly visually interpretable but usually have a poor out-of-sample performance due to overfitting. In contrast random forests and gradient boosted regression trees, are less prone to overfitting and thus clearly outperform single regression trees, since they combine predictions from a large number of regression trees, however for the same reason they also suffer from low interpretability.

4.2.5 Neural Networks

Neural networks are another type of highly flexible, yet parametric models. Even though there is a broad variety of different kinds of neural networks, I will limit myself to feedforward neural networks for reasons which I will explain later.

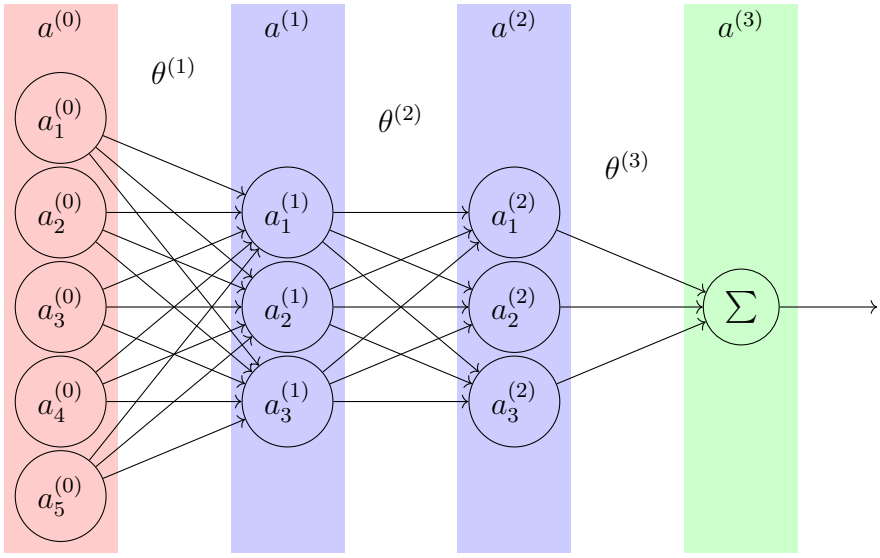

Figure 5: Exemplary feedforward neural network

This figure depicts an exemplary feedforward network consisting of one input layer (■), $a^{(0)}$, containing 5 nodes, two hidden layers (■), $a^{(1)}$ and $a^{(2)}$, each containing 3 nodes and one output layer (■), $a^{(3)}$. The arrows represent the weights stored in $\theta^{(1)}, \theta^{(2)}$ and $\theta^{(3)}$, respectively, which are required to calculate the value of the nodes in the subsequent layer.

Figure 5 depicts an exemplary feedforward neural network. A (deep) feedforward neural networks consists of an input layer (■), at least one hidden layer (■) and one output layer (■). Roughly speaking, for a given observation the information contained in the feature vector enters the feedforward neural network via the input layer, then the information is processed via various computations sequentially through a sequence of hidden layers before the predicted value for the target variable given the feature is eventually returned via the output layer. This mechanism of generating a prediction from a feature vector using a trained feedforward neural network is known as forward-propagation.

Similarly to Michelucci (2018), feedforward neural networks can be defined more formally as follows: A feedforward neural network consists of subsequent layers $l = 0, 1, \ldots, L$, one input layer ($l = 0$), $L - 1$ hidden layers ($l = 1, 2, \ldots, L - 1$) and one output layer $l = L$. Layer l contains $n^{(l)}$ nodes. In particular, $n^{(0)} = p$, i.e. the number of nodes in the input layer is equal to the dimension of the vector of features, and $n^{(L)} = 1$ due to the regression setup. The column vector of the values of the nodes in layer l is denoted by $a^{(l)} = (a_1^{(l)}, a_2^{(l)}, \ldots, a_{n^{(l)}}^{(l)})$. Formally, forward-propagation proceeds as follows: First, the values of the nodes in the input layer are set to the vector of characteristics, i.e. $a^{(0)} = x_{i,t}$. In particular, the value of the j-th node in the input layer is set to the j-th entry in $x_{i,t}$. Next, for each node in the first hidden layer, the weighted sum of all nodes in the previous layer plus some constant, the so called bias, needs to be computed. Subsequently, for each node the activation function is applied to the result of the previous calculation. Then, forward-propagation steps to the next hidden layer and starts over. It does so

until it reaches the output layer. In the output layer, just the weighted sum of all nodes in the previous layer and the bias is returned without applying the activation function. The complete procedure is described by the following two equations:

$$a^{[l]} = g\left(\theta^{[l]} a^{[l-1]} + \theta_0^{[l]}\right), l = 1, 2, \ldots, L-1 \tag{14}$$

$$a^{[L]} = \theta^{[L]} a^{[L-1]} + \theta_0^{[L]}. \tag{15}$$

$\theta_0^{[l]} \in \mathbb{R}^{n^{[l]}}, l = 1, 2, \ldots, L$, is a vector, whose j-th entry is the bias used to calculate the value of the j-th node in layer l. $\theta^{[l]} \in \mathbb{R}^{n^{[l]} \times n^{[l-1]}}, l = 1, 2, \ldots, L$, is a matrix, whose j-th row contains the weights to be multipied with the values of the nodes in layer $l-1$ in order to calculate the value of the j-th node in layer l. In total, the feedforward neural network comprises of

$$n_\theta = \sum_{l=1}^{L} n^{(l)}(n^{(l-1)} + 1) \tag{16}$$

coefficients that need to be estimated and let $\theta, \theta = (\theta_1, \theta_2, \ldots, \theta_{n_\theta})$, denote a vector containing all coefficients of the feedforward neural network. g, $g : \mathbb{R} \to \mathbb{R}$, is the activation function, which is applied elementwise. There are various activation functions available, for instance the identity, sigmoid, tanh or Rectified Linear Unit (ReLU) function. Nevertheless, for reasons that will become apparent later, I will only make use of the ReLU activation function, which is defined as follows:

$$\text{ReLU}(x) = \max(0, x). \tag{17}$$

Based on these elaborations, the functional form of a feedforward neural network used for regression can be written as:

$$f(x_{i,t}, \theta) = a^{[L]} = \theta^{[L]} a^{[L-1]} + \theta_0^{[L]} \tag{18}$$

It should be noted that in order to calculate $f(x_{i,t}, \theta)$ (i.e. $a^{[L]}$), $a^{[L-1]}, \ldots, a^{[1]}$ need to be computed beforehand.

$$\begin{aligned} \mathcal{L}_{NN}(\theta, \lambda) &= \mathcal{L}_{MSE}(\theta) + \lambda \sum_{j=1}^{n_\theta} \theta_j^2 \\ &= \frac{1}{NT} \sum_{i=1}^{N} \sum_{t=1}^{T} \left(r_{i,t+1}^{abn} - f(x_{i,t}, \theta)\right)^2 + \lambda \sum_{j=1}^{n_\theta} \theta_j^2 \end{aligned} \tag{19}$$

As for ridge regression, adding the ℓ_2-norm of the vector of coefficients multiplied by a non-negative scalar λ to MSE can prevent overfitting (see Equation 19). Since the problem of minimizing $\mathcal{L}_{NN}$ has no analytical solution, gradient based methods need to be used. Backpropagation, allows to determine the

gradient of f with respect to θ, which is non-trivial due to the highly nested structure of f. It computes partial derivatives by applying the chain rule while iterating backwards from the output layer to the input layer. However, since the starting values for the minimization of $\mathcal{L}_{NN}$ are randomly intialized, using different seeds yields different results. To mitigate this undesired randomness, similar to (Gu et al., 2020), I construct ensembles consisting of five feedforward neural networks trained with different seeds by averaging predictions over them.

Unfortunately, the estimation of the coefficients of a neural network is computationally very expensive, since a large amount of parameters need to be iteratively estimated. In addition to the estimation of the coefficients, λ, the hyperparameter controlling the degree of regularization, needs to be tuned.

Despite the high computational costs, it is also the large amount of coefficients which makes neural networks highly flexible. By means of nonlinear activation functions they model nonlinearities between the features and the target variable. Moreover, via the nodes inside the hidden layers neural networks can capture even high order interactions between features. However, the capabilities of neural networks go even beyond that: According to the universal approximation theorem, feedforward networks with a linear output layer, at least one hidden layer, a sufficient number nodes inside the hidden layers and ReLU activtion functions can approximate any continuous function (Leshno et al., 1993). Against this background, it seems justified that I limit my analyses to feedforward neural networks with ReLU activation functions. Even though from a theoretical perspective neural networks are the most powerful machine learning models, choosing an appropriate neural network architecture, i.e. the number of hidden layers and number of nodes inside each hidden layer, is crucial to exploit the potential of neural networks in the best possible way. Unfortunately, there are no rules on how to determine the optimal architecture for a neural network. An approach similar to hyperparameter tuning, adaptively searchs different architectures and opts for the one with the best out-of-sample predictive power. A major disadvantage of neural networks besides their high computational cost, is that due to their highly nested structure they are rather intransparent and cannot be directly interpreted.[12]

4.3 Training, Hyperparameter Tuning and Prediction

Since a lot of the characteristics are updated once per year and since training machine learning models is computationally expensive, I recalculate the models just once a year at the end of June. At each recalculation date (t),

[12]For further details on neural networks, the interested reader is referred to Goodfellow, Bengio, and Courville (2016).

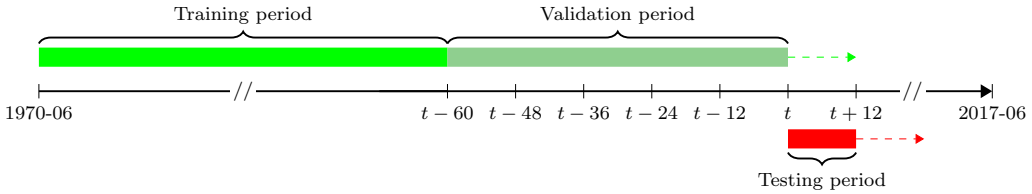

Figure 6: Sample splitting

This figure visualizes the sample splitting. At each recalculation date t, three subsets of the data are defined: The train set covers the period from July 1970 until the recalculation date except the last five years (■), i.e. $[1970\text{-}07, t-60]$. The validation set covers the last five years before the recalculation date (■), i.e. $(t-60, t]$. The test set covers the year after the recalculation date (■), i.e. $(t, t+12]$.

I define three disjoint subsets of the data: The train set covers the period from July 1970 until the recalculation date except the last five years (i.e. $[1970\text{-}07, t-60]$). The validation set contains five years preceding the recalculation date (i.e. $(t-60, t]$). The test set covers the year after the recalculation date (i.e. $(t, t+12]$). In particular, this kind of sample splitting retains the temporal ordering of the data. As I move from one recalculation date to the next, the train set increases by one year, whereas the validation and the test set are rolled forward by one year maintaining their size. The sample splitting is visualized in Figure 6.

These three subsets of the data are used for distinctively different purposes. Based on the train set, a given model is trained multiple times for different hyperparameter values yielding different trained versions of the same model. The predictive performance of those different trained versions of the same model is then assessed on the validation set. Subsequently, just the trained version of the model (and its corresponding hyperparameter values) is kept, which achieved the best performance on the validation set. This process is known as hyperparameter tuning.[13] Finally, the tuned model is used to make out-of-sample predictions on the test set, which has neither been seen during training the model nor during tuning its hyperparameters. These out-of-sample predictions allow to examine the out-of-sample predictive power of the tuned model. In total, the above described procedure allows for 37 years of out-of-sample testing, since the one year test set is annually rolled forward from 1980 to 2017.

Unfortunately, for several characteristics the coverage is low or changes substantially over sample period.[14] Consequently, in order to ensure high data quality, a dynamic feature selection is applied: At each recalculation date only those features that had a mean coverage of at least 90% during the five years preceding the recalculation date are included for training the machine learning models.

[13] For hyperparameter tuning, I largely follow Gu et al. (2020). For a table reporting the values used for hyperparameter tuning, please refer to Table 8 in the Appendix.

[14] For the average coverage of each characteristic over the entire period, please refer to Table 7 in the Appendix.

4.4 Feature Importance Using Shapley Values

As outlined above, some machine learning models are hardly interpretable, especially those that come with a high degree of flexibility. For this reason, random forests, gradient boosted regression trees and neural networks are considered to be black box models. Nevertheless, not only the predictions generated by a machine learning model are of interest but also how these predictions came about and in particular how much each feature has contributed to them. The notion of feature importance tries to answer the latter question. One approach to measuring feature importance is based on the Shapley-value.

The Shapley-value is a concept originating from the domain of coalitional game theory (Shapley, 1953). In coalitional game theory a characteristic function assigns a coalition of p players a certain payoff. This characteristic function is usually a non-additive black box function. Thus, the question of how much each player contributed to the payoff of the coaltion arises. The Shapley-value distributes the payoff of a coalition among the players proportional to the their marginal contribution. For details on the computation of the Shapley-value the reader may refer to Lundberg and Lee (2017). Interpreting a model based on p features as a coalition of players and a prediction made from a model as a payoff of a coalition, the Shapley-value can be used to determine how much each feature contributes to the predictions Štrumbelj and Kononenko (2014).

4.5 Machine Learning Portfolios

The trained and tuned machine learning models can be used to construct portfolios. For a given machine learning model, I proceed as follows: First, at the end of each month t, I calculate the 1-month-ahead out-of-sample predicted abnormal return, i.e. $\hat{r}^{abn}_{i,t+1}$. Then, using NYSE breakpoints I assign all stocks into decile portfolios (1=Low, 10=High) based on their predicted abnormal return. Inside each portfolio, stocks are weighted according to their marketcap (i.e. value-weighting). As recommended by Hou et al. (2020), using NYSE breakpoints as well as value-weighting allows to limit the effect that small stocks have on the results. Finally, I construct a zero-net investment portfolio that goes long in the decile 10 portfolio and short in the decile 1 portfolio (LS). For all the portfolios, reassignment and rebalancing takes place at the end of each month.

In order to define the portfolio turnover as well as the trading costs formally some notations need to be introduced. Let $w^{(p)}_{i,t}$ denote the weight of stock i inside portfolio p, $p = 1, 2, \ldots, 10, LS$, at the beginning of month t, i.e. just after portfolio reassignment and rebalancing at the end of month $t-1$. Similarly, let $\widetilde{w}^{(p)}_{i,t}$ denote the weight of stock i inside portfolio p at the end of

month t, i.e. just before portfolio reassignment and rebalancing at the end of month t. The weights just before portfolio reassignment and rebalancing can be calculated as follows:

$$\widetilde{w}_{i,t}^{(p)} = \frac{w_{i,t}^{(p)}(1 + r_{i,t})}{\sum_{j=1}^{N} w_{j,t}^{(p)}(1 + r_{j,t})}. \tag{20}$$

Based on the weights before and after portfolio reassignment and rebalancing the one-sided turnover of portfolio p incurred at the end of month t, $TO_t^{(p)}$, is given by:

$$TO_t^{(p)} = \frac{1}{2} \sum_{i_1}^{N} \left| w_{i,t+1}^{(p)} - \widetilde{w}_{i,t}^{(p)} \right|. \tag{21}$$

Dividing by two in the above formula ensures that buying and selling is not counted twice. Taking also the trading costs of stock i at the end of month t into account, $tc_{i,t}$, the trading costs of portfolio p incurred at the end of month t, $TC_t^{(p)}$, can be calculated as:

$$TC_t^{(p)} = \sum_{i_1}^{N} \left| w_{i,t+1}^{(p)} - \widetilde{w}_{i,t}^{(p)} \right| tc_{i,t}. \tag{22}$$

Reassigning and rebalancing portfolios at the end of each month results in high portfolio turnover and trading costs, which in turn shrinks net returns. In order to reduce this negative effect on the net returns of the long-short portfolio, one may use a turnover mitigation strategy known as banding or buy/hold spread, which Novy-Marx and Velikov (2019) found to be a promising approach. Applied in the context of the above described decile sorting, banding proceeds at the end of each month as follows: First, as before using NYSE breakpoints all stocks are assigned into decile portfolios based on their 1-month-ahead out-of-sample predicted abnormal return. Second, if a stock is assigned to decile portfolio 1 (10) or if it was in the short-position (long-position) in the previous month and its current decile portfolio assignment does not differ by more than a given sorting tolerance from 1 (10), it is assigned to the short-position (long-position).

However, the effect of turnover mitigation sorting on the net returns is ambiguous. Keeping stocks inside the long-position (short-position) after leaving decile 1 (10) as long as their new decile portfolio assignment does not deviate from decile 1 (10) by more than the respective sorting tolerance, obviously reduces the number of selling (and subsequent buying) transactions. This in turn decreases the turnover as well as the trading costs of the long-short portfolio. However, keeping stocks inside the long position (short position) even though their predicted return is smaller (larger) than the one of the stocks in decile

10 (decile 1), narrows down the gross return of the long-short portfolio. The magnitude of these effects highly depends on the sorting tolerance. Ideally, the sorting tolerance is sufficiently large to induce a substantial reduction in turnover and trading costs and at the same time sufficiently small to prevent a significant decrease in gross returns. Due to these countervailing effects the overall effect on net returns is ex ante unclear.

5 Empirical Results

Based on the methods described in the previous section, I will now analyze the out-of-sample performance of fourteen trained machine learning models. Following Gu et al. (2020), a simple linear regression model, which is based just on marketcap, book-to-market and 12-month momentum, will act as a benchmark (OLS-3), as proposed by Lewellen (2015). The other thirteen machine learning models, which make use of all available characteristics, include, simple linear regression (OLS-all), ridge regression (Ridge), elastic net (ENet), lasso regression (lasso), PCR, random forest (RF), a special implementation of a gradient boosted trees (XGBoost) as well as six ensembles of feedforward neural networks. Since, as described earlier, choosing appropriate neural network architectures poses a potential problem I will adopt those of (Gu et al., 2020). NN1 (NN2, NN3, NN4, NN5) represents an ensemble consisting of five neural networks each with 1 (2, 3, 4, 5) hidden layer(s) and (32) ((32,16), (32,16,8), (32,16,8,4), (32,16,8,4,2)) nodes inside the corresponding hidden layer(s). Besides, I also trained an ensemble of five neural networks with 1, 2, 3, 4 and 5 hidden layers, respectively, but architectures analog to those described above (NN1-5).

For each of these models, the out-of-sample predictive power and the feature importances will be discussed. Moreover, I will examine the gross and net performance of the corresponding machine learning portfolios as well as the impact of turnover mitigation sorting and the construction of ensemble models on their performance. Besides from comparing the models to each other, I will also relate my results to the literature, namely Gu et al. (2020) and (Drobetz & Otto, 2021), along the way.

5.1 Predictive Power

Figure 7 displays the predictive power of the various machine learning models as measured by the out-of-sample R^2. OLS-3 has an out-of-sample R^2 of 0.24%. Even though OLS-all makes use of all available information contained in the characteristics including all those contained in OLS-3 it does not have a noticeably higher predictive power than OLS-3. This is due to the aforementioned risk of OLS overfitting to the training sample when applied to high-dimensional data resulting in a lower out-of-sample predictive power.

As outlined in Sections 4.2.2 and 4.2.3, regularized regression techniques as well as PCR can help to avoid this problem. Lasso regression and elastic net improve the predictive power considerably by 0.17% resulting in an out-of-sample R^2 of 0.41%, whereas ridge regression leaves the predictive power unchanged. The predictive power of lasso regression relative to the one of

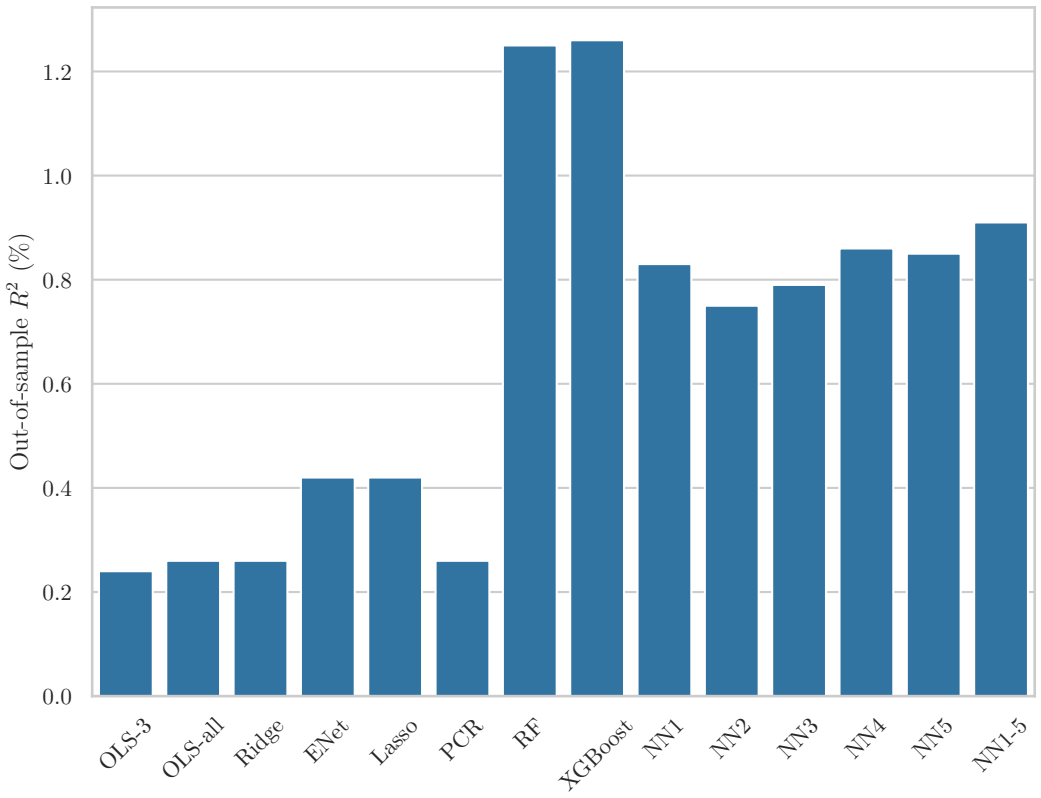

Figure 7: Out-of-sample predictive performance (percentage R^2)
This figure displays the out-of-sample predictive performance for each machine learning model as measured by its out-of-sample percentage R^2. The models include OLS using only marketcap, book-to-market and 12-month momentum (OLS-3), and the following models using all available features: OLS (OLS-all), ridge regression (Ridge), elastic net (ENet), lasso regression (Lasso), principal component regression (PCR), random forest (RF), gradient boosted tree (XGBoost) and ensembles of neural networks with 1 to 5 hidden layers (NN1, NN2, NN3, NN4, NN5) as well as one ensemble consisting of 5 neural networks with 1, 2, 3, 4 and 5 hidden layers, respectively.

ridge regression might suggest that for this application feature selection is more successful at preventing overfitting and improving predictive power than just shrinking the model coefficients as ridge regression does. Similarly to ridge regression, PCR does not improve the predictive power. Even though, PCA tries to capture the largest fraction of the variance in the features, it cannot distinguish between signal and noise with respect to the prediction of returns. This might explain, why PCR does not yield a better predictive power than OLS-all, especially since in return prediction the signal-to-noise ratio is typically very low.

In comparison to the linear regression methods discussed so far, the various neural networks have a substantially higher out-of-sample R^2 ranging between 0.75% and 0.91%. Among those, the ensemble combining neural networks with different numbers of hidden layers, NN1-5, performs best. Having an out-of-sample R^2 of 1.25%, tree-based methods, namely random forests and boosted trees, show the highest predictive power among all models. In summary, neural

networks and tree-based methods achieve by far a higher predictive power than linear regression methods. This suggests that nonlinearities and interactions between features are essential to the prediction of stock returns, since the tree-based methods and neural networks are the only models considering them.

Similarly, Gu et al. (2020) report that linear methods deliver limited improvements over OLS-3, whereas tree-based models and neural networks possess of a substantially higher predictive power. Overall, also the results of (Drobetz & Otto, 2021) on the European stock market point into the same direction. Nevertheless, in contrast to my findings, Gu et al. (2020) find that neural networks perform slightly better than tree-based methods. Although the overall pattern of the predictive power across the various models shown by Gu et al. (2020) is similar to those in Figure 7, their exact values of R^2 are considerably smaller. This is most likely due to the fact that Gu et al. (2020) predict returns instead abnormal returns. As explained in the second part of Section 4.1, the prediction of returns has a lower signal-to-noise ratio than the prediction of abnormal returns, which makes it harder for machine learning models to detect patterns resulting in a worse out-of-sample predictive performance

5.2 Feature Importance

In order to calculate the relative importance of individual features inside a given machine learning model, I proceed as follows: First, for each out-of-sample test period I calculate the Shapley-value for each feature and normalize it to sum to one.[15] Then, for each feature I compute the average of the normalized Shapley-values across all out-of-sample test periods, from now on referred to as relative importance. To determine the relative importance of an individual feature across all models, I rank the features according to the sum of the relative importance across all models. The results for the top 30 features are visualized in Figure 8. As the relative importance of a specific feature inside a given machine learning model increases, the colour of the corresponding cell in the figure gradually changes from white or light blue to dark blue.

All the models except for PCR seem to agree on the three most important features, namely marketcap, short-term reversal (retadj) and book-to-market (BM). Apart from that, the 30 most important features across all models can be divided into four categories, two major and two minor. The first category, which contains 9 out of the 30 most important features, reflects recent price trends: short-term reversal (STreversal/retadj), 12 month momentum (Mom12m), intermediate momentum (IntMom), return seasonality years 2 to 5 (MomSeason), medium-run reversal (MRreversal), return seasonality last year (MomSeasonShort), 6 month momentum (Mom6m), off season

[15] For a description of Shapley-values, see Section 4.4.

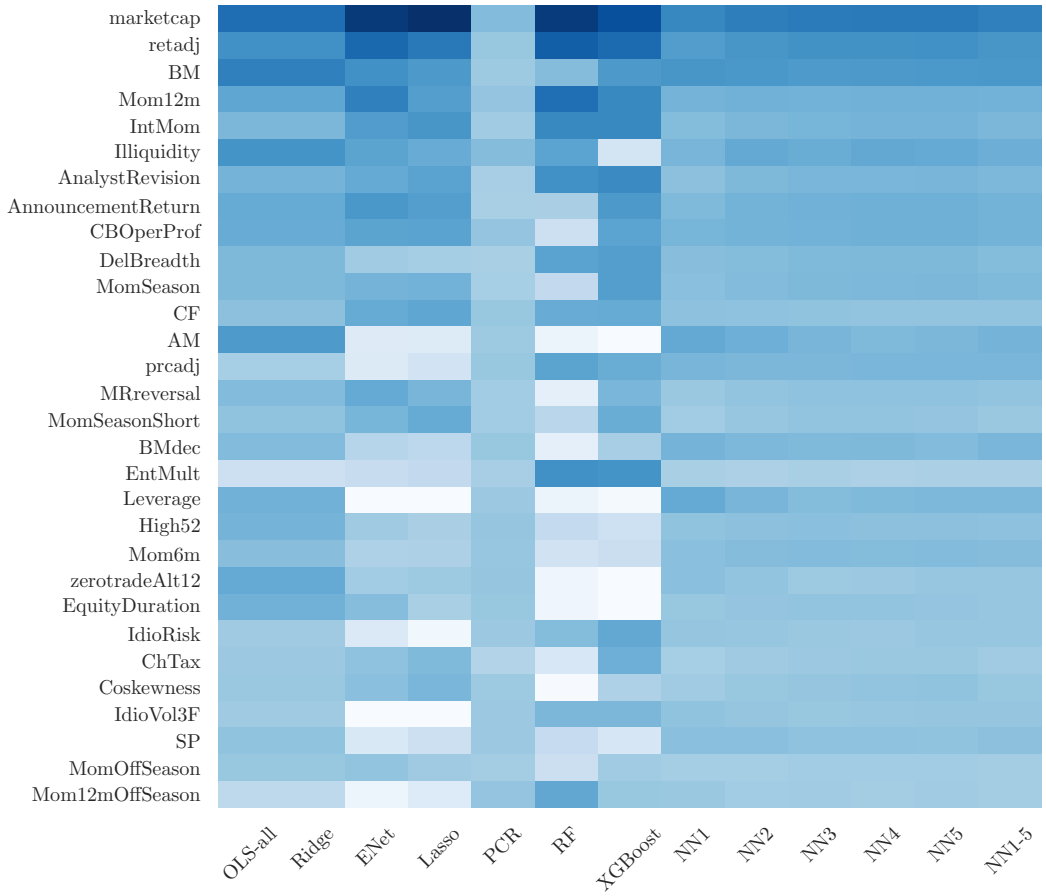

Figure 8: Feature importance

This figure displays the relative feature importance of the 30 most important features across all models as measured by the average normalized Shapley-value. The rows (columns) correspond to the different features (models). Features are ranked in decreasing order according to the sum of the relative importance across all models. With increasing relative importance of a feature inside a model, the colour of the corresponding cell changes gradually from white/light blue to dark blue. For the abbreviations of the models, please refer to the description of Figure 7. For the abbreviations of the features, see Table 6 in the Appendix.

long-term reversal (MomOffSeason) and momentum without the seasonal part (Mom12mOffSeason). The second major category is based on valuation ratios or fundamental signals and consists of 8 of the most important features: book-to-market using recent market equity (BM), cash-based operating profitability (CBOperProf), cash flow to market (CF), total assets to market (AM), book-to-market using December market equity (BMdec), enterprice multiple (EntMult) and leverage. Apart from these two major categories there are also two minor categories of features. The one represents price or trading based characteristics, such as the adjusted price (prcadj), the 52 week high (High52), market capitalization (marketcap), Amihud's illiquidity (Illiquidity) and days with zero trades (zerotradeAlt12). The other subsumes risk related characteristics: idiosyncratic risk with respect to the CAPM (IdioRisk), idiosyncratic risk with respect to the Fama-French three factor model(IdioRisk) and coskewness (Coskewness).

Gu et al. (2020) find the same categories among the most important features accross all models. Unfortunately, on a feature level the results of Drobetz and Otto (2021) on feature importance are hardly comparable since they only use in total 22 features for their prediction.

Besides these more general considerations on the importance of individual features across all models, one can also inspect and interpret the models separately in detail as follows: Firstly, if two models assign the same relative importance to each given feature as indicated by identical colours, these two models seem to resemble each other closely. Secondly, a machine learning model for which most of the displayed features are colored in light blue tends to assign more or less equal importance to all depicted features. In contrast, a machine learning model for which some features are colored in dark blue and some in white assigns high importance to the first ones and low importance to the latter ones and thereby effectively conducts some kind of feature selection.

Based on this, OLS-all as well as ridge regression seem to resemble each other very closely and both seem to consider rather a broad set of features. This is an explanation, why OLS-all and ridge regression have comparable predictive power as dicussed in Section 5.1. A deeper look into the results of hyperparameter tuning for ridge regression at each recalculation date reveals the returned optimal value of λ is most of the time rather small. Consequently, ridge regression yields most of the time results that are close to those of OLS. Lasso regression and elastic net also appear to be quite similar based on the relative importances they assign to the features. However, in contrast to OLS-all and ridge regression, lasso regression and elastic net do feature selection. PCR distributes relative importance more or less uniformly across all features. This observation is in line with the explanation for PCR's poor predictive performance mentioned in Section 5.1: Trying to capture most of the variance in the features, PCA does not distinguish between signal and noise with respect to return prediction. Therefore, PCR struggles to identify features that are predictive for returns, i.e. signal. The tree-based methods, random forests and boosted trees, are similar to each other and they both select features. However, the tree-based methods do not resemble each other as closely as lasso regression and elastic net do, since there are some features for which the respective relative importance differ visibly as indicated by different colours. Interestingly, in contrast to all other models, random forest and boosted trees attribute high relative importance to enterprise multiple (EntMult). Finally, also the different neural networks are very closely related. They all assign more or less equal importance to all features instead of selecting a few. A potential explanation for this is that the neural networks might focus more on the interactions between features inside the hidden layers instead of feature selection. An indication for this is also the fact that the more hidden layers a

neural network has and therefore the more it takes interactions into account, the more uniformly it distributes relative importance accross all features since the colour of a fature tends to go more towards light blue as the number of hidden layers increases.

Overall, Gu et al. (2020) come to similar conclusions, in particular with respect to elastic net (and lasso regression) as well as the neural networks. However, they find the tree-based methods to perform less feature selection. This is in contrast to Drobetz and Otto (2021) which, in support of my results, observe that tree-based methods assign large relative importance only to a few features, whereas other methods, such as PCR, consider a larger set of features to be important.

5.3 Machine Learning Portfolios

Table 1 summarizes for each machine learning model the performance of the value-weighted decile portfolios and the value-weighted long-short portfolio formed monthly based on the gross 1-month-ahead out-of-sample predicted abnormal returns.[16] The same analysis for equal weighted returns can be found in Table 9 in the Appendix. Note that for the long-short portfolios the returns are shown whereas for the decile portfolios the abnormal returns are reported. In general, the average realized gross abnormal returns (Avg(g)) increase monotonically in decile portfolios (with just few exceptions) from negative to positive. Consequently, across all models, the average realized gross return of the long-short portfolio is positive, statistically significant and economically large. This suggests that all models are, to varying degrees, able to capture cross-sectional variation in realized gross abnormal returns as well as to identify high- and low-performing stocks. For all models, the average gross return in the long position (i.e. decile portfolio 10) is larger than the one in the short position (i.e. negative of decile portfolio 1). The standard deviation of gross abnormal returns (SD(g)), which is a measure of risk, roughly follows a parabola taking the highest values in the extreme deciles and lower values in the decile portfolios in between. The t-statistics (t(g) and (t(n)) can be used for two purposes: Firstly, they allow to assess the statistical significance of the mean return. Secondly, they put risk, as measured by the standard deviation, and return, as measured by the mean return, into relation, since $t = \text{Avg/SD} \times \sqrt{T}$ (T is the number of months which is constant across all models). Since higher returns could be due to higher risk, it is important to also look at the return-risk ratio. The t-statistics of the gross abnormal return increase in decile portfolios from negative to positive. The largest absolute values of the t-statistic are in the extreme deciles with absolute values

[16]For details on portfolio construction, see Section 4.5.

	OLS-3								OLS-all							
	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)
Low (L)	-0.71	-0.15	1.77	-1.81	20.43	0.10	-0.25	-2.98	-1.38	-0.55	2.01	-5.78	60.24	0.32	-0.87	-9.02
2	-0.34	-0.06	1.44	-0.93	47.75	0.23	-0.29	-4.27	-0.71	-0.28	1.79	-3.34	80.94	0.40	-0.68	-8.06
3	-0.14	0.01	1.65	0.13	54.38	0.27	-0.26	-3.39	-0.37	-0.18	1.69	-2.25	84.97	0.41	-0.59	-7.35
4	0.04	0.22	1.96	2.42	56.31	0.30	-0.08	-0.83	-0.10	-0.05	1.55	-0.64	86.82	0.42	-0.47	-6.36
5	0.21	0.04	2.15	0.35	58.08	0.33	-0.29	-2.90	0.14	0.12	1.54	1.66	87.11	0.42	-0.30	-4.19
6	0.39	0.11	2.01	1.19	57.94	0.35	-0.24	-2.51	0.37	0.17	1.66	2.14	87.18	0.43	-0.26	-3.30
7	0.57	0.23	2.30	2.09	57.25	0.38	-0.16	-1.46	0.62	0.24	1.78	2.84	87.20	0.44	-0.20	-2.44
8	0.77	0.36	2.58	2.98	54.90	0.41	-0.04	-0.37	0.90	0.22	1.91	2.47	85.79	0.45	-0.23	-2.55
9	1.03	0.57	2.95	4.10	47.79	0.39	0.18	1.32	1.26	0.35	2.27	3.28	83.71	0.47	-0.12	-1.13
High (H)	1.44	1.28	3.33	8.15	28.13	0.25	1.03	6.58	1.93	0.81	3.25	5.27	72.34	0.48	0.33	2.17
H-L	2.15	1.43	4.49	6.77	48.57	0.35	1.08	5.13	3.31	1.35	4.20	6.84	132.58	0.80	0.56	2.84
	Ridge								ENet							
	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)
Low (L)	-1.38	-0.55	2.01	-5.78	60.24	0.32	-0.87	-9.02	-0.93	-0.45	1.77	-5.41	52.30	0.27	-0.72	-8.50
2	-0.71	-0.28	1.79	-3.34	80.94	0.40	-0.68	-8.06	-0.48	-0.18	1.51	-2.47	77.55	0.38	-0.55	-7.64
3	-0.37	-0.18	1.69	-2.25	84.96	0.41	-0.59	-7.35	-0.23	-0.11	1.71	-1.37	81.35	0.40	-0.51	-6.33
4	-0.10	-0.05	1.55	-0.64	86.82	0.42	-0.47	-6.36	-0.02	-0.00	1.65	-0.04	82.61	0.42	-0.42	-5.41
5	0.14	0.12	1.54	1.66	87.11	0.42	-0.30	-4.19	0.16	0.03	1.83	0.37	82.54	0.43	-0.40	-4.55
6	0.37	0.17	1.66	2.13	87.18	0.43	-0.26	-3.30	0.35	0.23	2.00	2.43	83.60	0.45	-0.22	-2.35
7	0.62	0.24	1.78	2.84	87.20	0.44	-0.20	-2.43	0.55	0.14	2.12	1.39	83.41	0.47	-0.33	-3.35
8	0.90	0.22	1.91	2.47	85.79	0.45	-0.23	-2.55	0.77	0.21	2.38	1.92	81.97	0.49	-0.28	-2.48
9	1.26	0.35	2.27	3.28	83.71	0.47	-0.12	-1.13	1.06	0.48	2.68	3.82	79.40	0.52	-0.04	-0.31
High (H)	1.93	0.81	3.25	5.26	72.34	0.48	0.33	2.16	1.57	1.11	3.45	6.81	63.92	0.48	0.62	3.84
H-L	3.31	1.35	4.20	6.83	132.58	0.80	0.56	2.83	2.50	1.56	4.34	7.62	116.22	0.76	0.80	3.95
	Lasso								PCR							
	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)
Low (L)	-0.87	-0.49	1.75	-5.89	45.96	0.25	-0.73	-8.66	-0.86	-0.41	1.70	-5.08	42.74	0.21	-0.62	-7.53
2	-0.42	-0.29	1.61	-3.79	62.79	0.32	-0.61	-7.83	-0.41	-0.25	1.56	-3.38	68.08	0.33	-0.57	-7.71
3	-0.18	-0.08	1.69	-1.00	66.96	0.34	-0.42	-5.20	-0.18	-0.03	1.47	-0.44	74.21	0.36	-0.39	-5.73
4	0.01	0.01	1.65	0.07	68.91	0.35	-0.35	-4.39	0.01	-0.09	1.57	-1.19	76.76	0.39	-0.48	-6.46
5	0.19	0.08	1.88	0.95	70.13	0.36	-0.28	-3.17	0.19	0.13	1.76	1.53	78.82	0.42	-0.29	-3.47
6	0.36	0.22	1.95	2.39	71.72	0.39	-0.17	-1.85	0.37	0.22	1.91	2.48	79.48	0.44	-0.22	-2.42
7	0.55	0.33	2.11	3.32	71.83	0.40	-0.07	-0.74	0.55	0.15	2.14	1.50	78.03	0.45	-0.30	-2.96
8	0.76	0.20	2.44	1.76	71.44	0.42	-0.22	-1.94	0.76	0.43	2.70	3.35	75.76	0.47	-0.05	-0.37
9	1.03	0.51	2.78	3.90	70.27	0.46	0.05	0.38	1.02	0.54	2.98	3.83	71.24	0.49	0.05	0.38
High (H)	1.50	1.26	3.52	7.58	58.80	0.45	0.81	4.91	1.52	0.79	3.50	4.82	54.03	0.41	0.38	2.33
H-L	2.37	1.74	4.43	8.36	104.76	0.69	1.05	5.07	2.38	1.20	4.54	5.62	96.78	0.62	0.58	2.73
	RF								XGBoost							
	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)
Low (L)	-0.35	-0.68	2.29	-6.33	68.13	0.38	-1.06	-9.71	-0.75	-0.71	2.24	-6.74	67.79	0.36	-1.07	-9.99
2	-0.13	-0.16	2.04	-1.69	80.45	0.40	-0.56	-5.78	-0.37	-0.16	1.90	-1.77	82.43	0.40	-0.55	-6.20
3	-0.02	-0.10	1.85	-1.17	82.62	0.40	-0.50	-5.70	-0.18	-0.15	1.84	-1.73	86.08	0.41	-0.56	-6.39
4	0.08	-0.12	1.79	-1.43	84.05	0.40	-0.52	-6.11	-0.03	-0.09	1.79	-1.06	86.79	0.41	-0.50	-5.93
5	0.17	0.06	1.75	0.68	83.91	0.40	-0.34	-4.12	0.10	0.01	1.60	0.20	87.69	0.42	-0.40	-5.34
6	0.27	0.03	1.78	0.39	82.28	0.39	-0.36	-4.29	0.24	0.08	1.60	1.09	87.58	0.43	-0.34	-4.59
7	0.37	0.01	1.97	0.12	83.03	0.41	-0.39	-4.27	0.39	0.16	1.84	1.88	86.94	0.44	-0.27	-3.16
8	0.48	0.32	2.14	3.22	82.30	0.44	-0.11	-1.11	0.56	0.21	1.95	2.27	86.74	0.46	-0.25	-2.69
9	0.65	0.44	2.80	3.35	73.14	0.45	-0.01	-0.10	0.82	0.38	2.39	3.35	83.84	0.48	-0.10	-0.90
High (H)	1.39	1.20	3.96	6.45	56.38	0.43	0.77	4.13	1.62	1.03	3.52	6.22	70.22	0.48	0.55	3.32
H-L	1.74	1.89	5.07	7.90	124.51	0.81	1.07	4.52	2.36	1.74	4.75	7.79	138.01	0.84	0.90	4.05
	NN1								NN2							
	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)
Low (L)	-1.85	-0.83	1.99	-8.86	59.65	0.34	-1.16	-12.28	-1.60	-0.85	2.21	-8.15	61.41	0.35	-1.20	-11.41
2	-0.92	-0.39	1.75	-4.76	79.06	0.39	-0.79	-9.43	-0.76	-0.40	1.62	-5.21	80.58	0.40	-0.80	-10.35
3	-0.52	-0.25	1.78	-2.99	84.55	0.40	-0.66	-7.77	-0.41	-0.12	1.75	-1.40	83.35	0.40	-0.51	-6.16
4	-0.22	-0.15	1.88	-1.65	85.58	0.40	-0.55	-6.17	-0.15	-0.17	1.52	-2.43	86.04	0.40	-0.58	-8.02
5	0.05	-0.09	1.54	-1.27	85.62	0.41	-0.50	-6.79	0.09	-0.02	1.54	-0.34	86.01	0.41	-0.43	-5.99
6	0.32	0.09	1.62	1.24	86.82	0.42	-0.32	-4.21	0.32	0.07	1.72	0.86	87.01	0.41	-0.34	-4.26
7	0.60	0.28	1.80	3.33	86.37	0.43	-0.14	-1.72	0.57	0.35	1.73	4.27	85.69	0.42	-0.07	-0.89
8	0.93	0.29	1.87	3.26	84.15	0.43	-0.14	-1.63	0.86	0.26	1.83	3.03	85.46	0.44	-0.17	-2.02
9	1.38	0.45	2.19	4.36	81.26	0.45	-0.00	-0.01	1.27	0.58	2.19	5.59	82.38	0.46	0.12	1.15
High (H)	2.41	1.16	3.24	7.58	67.94	0.44	0.72	4.70	2.21	1.00	3.37	6.30	70.15	0.47	0.53	3.35
H-L	4.26	1.98	3.84	10.99	127.59	0.78	1.21	6.75	3.81	1.85	3.99	9.83	131.56	0.82	1.03	5.49
	NN3								NN4							
	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)
Low (L)	-1.53	-0.86	2.46	-7.46	61.64	0.36	-1.22	-10.36	-1.40	-0.86	2.23	-8.16	61.08	0.35	-1.21	-11.38
2	-0.72	-0.39	1.66	-4.99	80.59	0.40	-0.79	-10.03	-0.66	-0.34	1.71	-4.25	80.99	0.41	-0.75	-9.16
3	-0.39	-0.26	1.64	-3.40	84.10	0.40	-0.66	-8.46	-0.35	-0.28	1.69	-3.56	83.93	0.40	-0.68	-8.45
4	-0.14	-0.05	1.71	-0.64	86.27	0.40	-0.46	-5.68	-0.11	-0.22	1.78	-2.63	85.65	0.40	-0.62	-7.44
5	0.09	-0.07	1.65	-0.96	85.83	0.40	-0.48	-6.11	0.10	0.01	1.60	0.14	86.74	0.41	-0.40	-5.32
6	0.31	0.12	1.63	1.63	86.47	0.41	-0.29	-3.78	0.31	0.18	1.63	2.35	86.02	0.41	-0.23	-3.02
7	0.55	0.34	1.69	4.29	86.18	0.42	-0.08	-1.04	0.54	0.19	1.74	2.37	85.53	0.42	-0.22	-2.75
8	0.84	0.36	1.87	4.15	85.22	0.44	-0.08	-0.91	0.81	0.39	1.94	4.25	84.00	0.43	-0.05	-0.50
9	1.23	0.51	2.22	4.84	82.20	0.46	0.04	0.42	1.18	0.48	2.15	4.75	82.08	0.46	0.02	0.22
High (H)	2.16	1.19	3.55	7.14	70.42	0.48	0.72	4.31	2.06	1.16	3.44	7.16	69.76	0.47	0.69	4.25
H-L	3.69	2.05	4.27	10.22	132.06	0.83	1.22	6.17	3.46	2.02	3.98	10.75	130.84	0.82	1.19	6.41
	NN5								NN1-5							
	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)
Low (L)	-0.88	-0.89	2.34	-8.08	60.44	0.35	-1.24	-11.12	-1.40	-0.88	2.10	-8.86	61.20	0.35	-1.23	-12.22
2	-0.33	-0.31	1.64	-4.07	80.55	0.40	-0.72	-9.16	-0.65	-0.42	1.83	-4.93	80.44	0.40	-0.83	-9.46
3	-0.11	-0.27	1.86	-3.11	83.39	0.40	-0.67	-7.53	-0.34	-0.22	1.58	-2.95	84.03	0.40	-0.62	-8.31
4	0.07	-0.14	1.64	-1.82	85.87	0.40	-0.54	-6.99	-0.10	-0.09	1.68	-1.16	85.76	0.40	-0.50	-6.24
5	0.22	0.02	1.60	0.24	85.68	0.40	-0.38	-5.15	0.11	0.02	1.53	0.25	86.30	0.41	-0.39	-5.40
6	0.37	-0.01	1.79	-0.16	86.27	0.41	-0.43	-5.06	0.32	0.09	1.67	1.21	86.51	0.41	-0.32	-4.04
7	0.53	0.24	1.88	2.69	85.22	0.42	-0.18	-2.04	0.55	0.22	1.74	2.73	86.26	0.42	-0.20	-2.47
8	0.72	0.38	1.75	4.62	84.74	0.44	-0.06	-0.68	0.81	0.29	2.02	3.07	84.55	0.44	-0.14	-1.52
9	0.98	0.38	2.08	3.92	81.41	0.46	-0.08	-0.78	1.18	0.50	2.15	4.90	82.57	0.46	0.04	0.37
High (H)	1.59	1.25	3.34	7.96	67.94	0.46	0.80	5.09	2.02	1.14	3.27	7.41	69.65	0.46	0.68	4.44
H-L	2.47	2.14	4.02	11.31	128.38	0.81	1.34	7.13	3.42	2.02	4.01	10.67	130.86	0.81	1.20	6.45

Table 1: Performance of machine learning decile portfolios (value-weighted)

This table reports the value-weighted performance of decile portfolios sorted on the 1-month-ahead out-of-sample predicted abnormal return using NYSE breakpoints. Note that for the long-short portfolios the returns are shown whereas for the decile portfolios the abnormal returns are reported. For the column names, see the description of Table 2. For the abbreviations of the models, see the description of Figure 7.

greater than 1.96 indicating statistically significant gross abnormal returns. The decile portfolios in the middle (usually 4 to 6) do not earn statistically significant gross abnormal returns. In contrast, the turnover (TO) is lowest in the extreme deciles and typically highest in deciles 5 and 6. Despite this, the total trading costs (TC) tend to increase slightly in decile portfolios with just minor exceptions. Since size as measured by the marketcap is one of the most important features across all models, large cap stocks tend to be assigned to decile portfolios 1 to 5 whereas small cap stocks tend to be in decile portfolios 6 to 10. However, because large cap stocks usually have lower trading costs than small cap stocks, this could be an explanation why trading costs increase in decile portfolios. Average realized net returns (Avg(n)) and their corresponding t-statistic tend to increase in decile portfolios from negative to positive. Across all models, after costs decile portfolios 1 to (atleast) 5 have statistically significantly negative average returns and just decile portfolio 10 has a statistically significantly positive average return.

By definition of the abnormal return, the value weighted abnormal return across all stocks is zero. At first sight, it might therefore seem surprising that for most of the machine learning models in Table 1 the mean of the average realized return across all deciles is not equal to zero. However, as explained above large cap stock tend to get assigned to decile portfolios 1 to 5 whereas small cap stocks tend to get assigned to decile portfolios 6 to 10. Consequently, decile portfolios 1 to 5 have a larger aggregated marketcaps than decile portfolios 6 to 10. For this reason, taking the mean (i.e. equal weight) of the average realized return across all deciles is not equivalent to calculating the value-weighted abnormal return across all stocks and thus does not need to yield zero as result.

In order to facilitate comparisons between the long-short portfolios of different machine learning models, Table 2 reports the performance just for the value-weighted long-short portfolios. The same analysis for equal weighted returns can be found in Table 10 in the Appendix. With an average realized monthly gross return of 2.14% (28.93% annualized) and a monthly volatility of 4.02% (13.93% annualized), the neural network with five hidden layers (NN5) achieves the best performance. Overall, the gross performance of the long-short portfolios of the different machine learning models aligns more or less with their respective predictive power. OLS-all, ridge regression and PCR underperform OLS-3 (1.43%), whereas lasso regression and elastic net slightly outperform OLS-3 in terms of gross returns (1.56% and 1.74%). Tree-based methods (1.74% and 1.89%) as well as neural networks (between 1.85% and 2.14%) have substantially higher average realized gross returns than OLS-3. However, even though neural networks have a lower out-of-sample R^2 than the tree-based methods, they earn clearly higher gross returns (except for NN2).

One explanation is that compared to tree-based methods the predictions of the neural networks are overshooting especially for the extreme decile portfolios as indicated by the gap between predicted and realized returns (see Table 1). This behaviour results in a lower out-of-sample R^2 but has no negative effect neither on sorting nor on the performance of the long-short portfolio. Another explanation is that R^2 weights each observation whereas the returns of the long-short portfolios are value-weighted. If tree-based methods are more accurate at predicting the gross abnormal returns of the numerically many small-caps, but neural networks perform better at predicting the gross abnormal returns of the numerically fewer large caps, then the tree-based methods will have a higher R^2. Nevertheless, this does not translate into higher value-weighted returns in the long-short portfolio since the returns of small cap stocks are outweighed by the returns of large cap stocks. Therefore, this can also lead to a misalignment of the predictive power and the performance of the long-short portfolio.

	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)
OLS-3	2.15	1.43	4.49	6.77	48.57	0.35	1.08	5.13
OLS-all	3.31	1.35	4.20	6.84	132.58	0.80	0.56	2.84
Ridge	3.31	1.35	4.20	6.83	132.58	0.80	0.56	2.83
ENet	2.50	1.56	4.34	7.62	116.22	0.76	0.80	3.95
Lasso	2.37	1.74	4.43	8.36	104.76	0.69	1.05	5.07
PCR	2.38	1.20	4.54	5.62	96.78	0.62	0.58	2.73
RF	1.74	1.89	5.07	7.90	124.51	0.81	1.07	4.52
XGBoost	2.36	1.74	4.75	7.79	138.01	0.84	0.90	4.05
NN1	4.26	1.98	3.84	10.99	127.59	0.78	1.21	6.75
NN2	3.81	1.85	3.99	9.83	131.56	0.82	1.03	5.49
NN3	3.69	2.05	4.27	10.22	132.06	0.83	1.22	6.17
NN4	3.46	2.02	3.98	10.75	130.84	0.82	1.19	6.41
NN5	2.47	2.14	4.02	11.31	128.38	0.81	1.34	7.13
NN1-5	3.42	2.02	4.01	10.67	130.86	0.81	1.20	6.45

Table 2: Performance of machine learning portfolios (value-weighted)
This table reports the value-weighted performance of long-short portfolios sorted on the 1-month-ahead out-of-sample predicted abnormal return using NYSE breakpoints (long = top decile, short = bottom decile). Columns Pred, Avg, SD, t, TO, TC refer to the avg. predicted return, avg. realized return, their standard deviation, their t-statistic, the turnover and the transaction costs. g (n) indicates gross (net) returns. For the abbreviations of the models, see the description of Figure 7.

Except PCR and the tree-based methods, all machine learning models have a lower standard deviation of gross abnormal returns than OLS-3. As outlined above, the t-statistic can be used as measure of statistical significance and as a measure of the relationship between return and risk. The long-short portfolios of all machine learning models earn highly statistically gross abnormal returns. Taking risk into account PCR is the only machine learnig which underperforms OLS-3. OLS-all and ridge regression perform similar to OLS-3,

whereas elastic net, lasso and the tree-based methods slightly outperform it. A clearly more favorable risk-return combination than OLS-3 is achieved by all neural networks.

Based on turnover as well as trading costs all machine learning models can be divided into three groups, low, medium and high. The first consists only of OLS-3 having by far the lowest turnover (48.57%) and trading costs (0.35%). The second group contains PCR, lasso regression and elastic net with turnover and trading costs between 96.78% and 116.22% and 0.62% and 0.76%, respectively. The third group includes OLS-all, ridge regression, the tree-based methods and all neural networks having turnovers and trading costs ranging from 124.51% to 138.01% and from 0.78% to 0.84%, respectively.

After trading costs, the neural network with 5 hidden layers still has the best performance with an average net return of 1.34% per month (17.32% annualized). Since OLS-3 has the lowest trading costs across all machine learning models, OLS-all, ridge regression and PCR, which underperform OLS-3 before trading costs, also underperform OLS-3 in terms of net returns (1.08%). Also lasso regression and elastic net as well as the tree-based methods, which have higher gross returns than OLS-3 but trading costs more than twice the one of OLS-3, have lower net returns than OLS-3. Just the neural networks (except for NN2) outperform OLS-3 in terms of net returns (between 1.19% and 1.34%) despite having high turnover and trading costs. This also holds true when considering the net return-risk ratio as measured by the t-statistic of net returns (t(n)).

Overall, the average realized returns and the t-statistics, i.e. return-risk ratio, are closely related to each other, which holds true before and after trading costs. Thus, the positive average realized returns of the machine learning portfolios do not seem to be caused by higher risk.

Overall, Gu et al. (2020), which do not study trading costs or net returns, find similar patterns considering the gross performance of the long-short portfolio across the various machine learning models. Beyond that, for both the tree-based methods and the neural networks, they report average realized gross returns and turnovers that match quite closely with those I find. In contrast, regarding the performance of the long-short portfolios compared to each other Drobetz and Otto (2021) arrive at results that differ from mine and those of Gu et al. (2020). For instance, they find PCR to have larger average gross realized returns than tree-based methods and neural networks or OLS-3 to have the most favorable gross return-risk ratio as measured by the t-statistic across all models.

Having extensively discussed the performance summary of the machine learning portfolios, Figure 9 depicts the cumulative gross performance of the long-

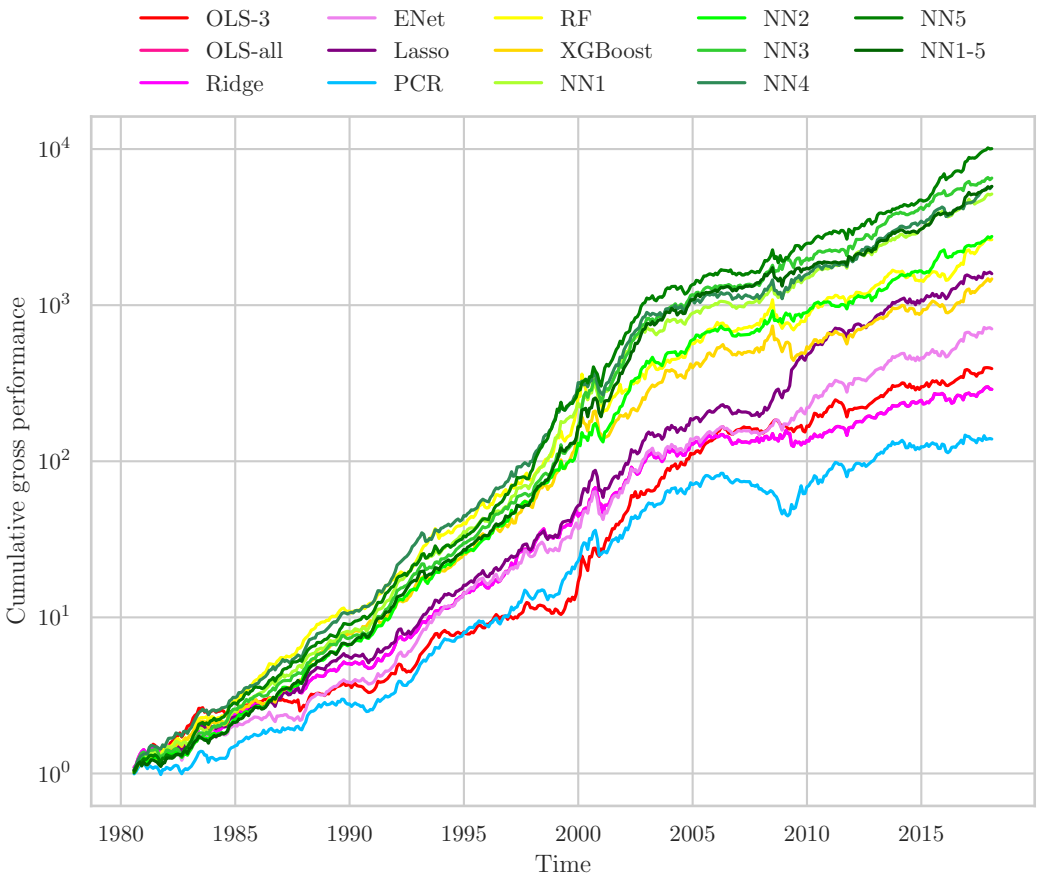

Figure 9: Cumulative gross performance of machine learning portfolios
This figure depicts the value-weighted cumulative gross performance of long-short portfolios sorted on the 1-month-ahead out-of-sample predicted abnormal return using NYSE breakpoints over time (long = top decile, short = bottom decile). The vertical axis is on log-scale. For the abbreviations of the models, please refer to the description of Figure 7.

short portfolio for all machine learning models over time with a log-scaled vertical axis. Over the entire period, neural networks followed by tree-based methods, lasso regression and elastic net outperform OLS-3. This ordering is largely preserved over time. All neural networks except NN2 comove very closely. Similarly, also the tree-based methods comove. Elastic net, which technically is a mixture of lasso and ridge regression, moves between those two. The comovement of the returns of the long-short portfolios of different machine learning models will be studied later in more detail.

Apparently, for all machine learning portfolios there is a kink in the performance plot in 2003. Before 2003, the slopes inside the performance plot are steeper, whereas after 2003 the slopes are flatter. This pattern can also be observed in the performance plots shown by Gu et al. (2020) as well as in the other performance plots below. It implies that the returns of the machine learning portfolios were higher before 2003 than after. This is in line with Green, Hand, and Zhang (2017) stating that in 2003 there is a considerable persistent drop in returns of characteristics based long-short portfolios.

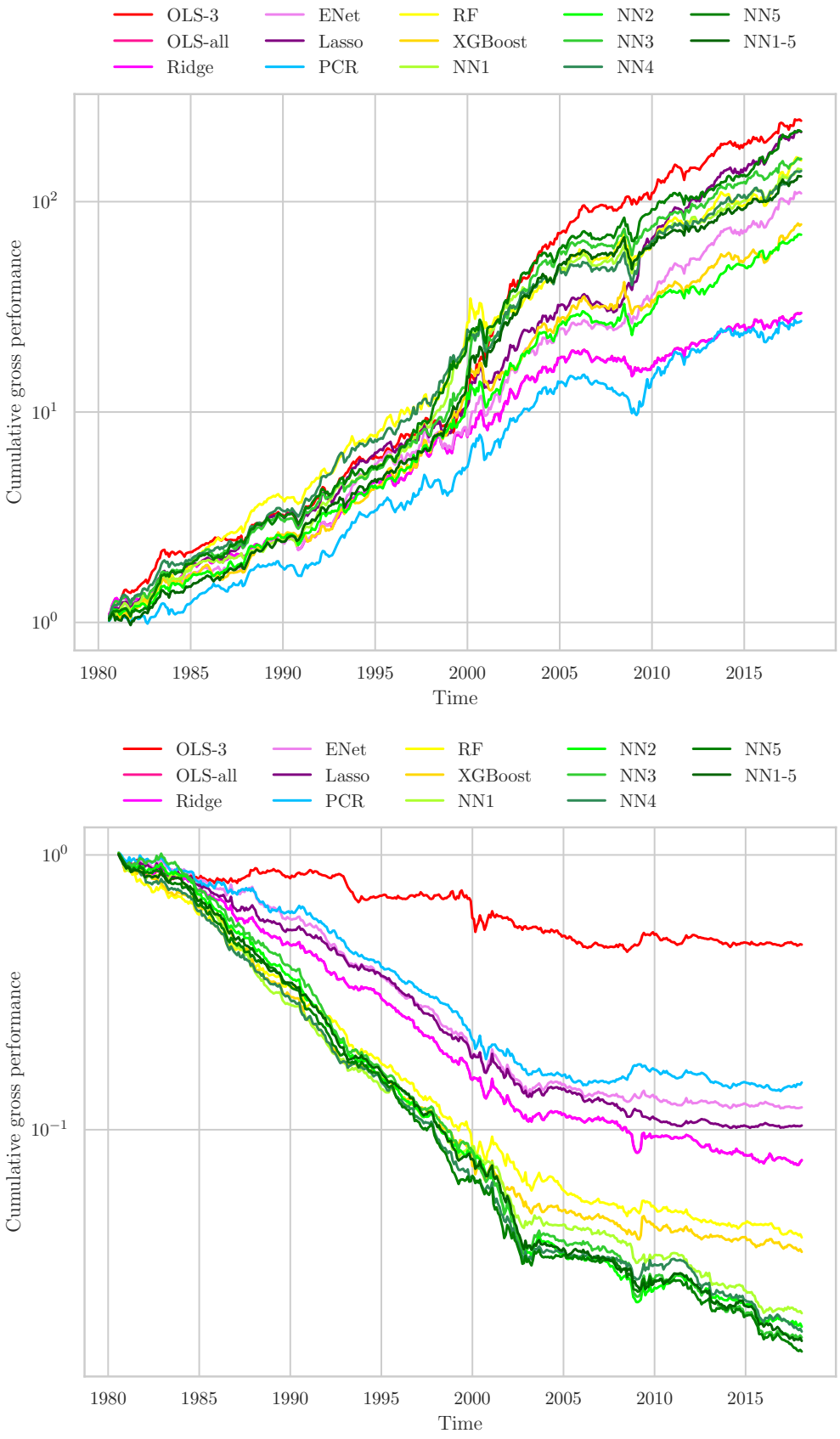

Figure 10: Cumulative gross performance of machine learning based long-only and short-only portfolios

This figure depicts in the top (bottom) pannel the value-weighted cumulative gross performance of long-only (short-only) portfolios sorted on the 1-month-ahead out-of-sample predicted abnormal return using NYSE breakpoints over time (long = top decile, short = bottom decile). The vertical axis is on log-scale. For the abbreviations of the models, please refer to the description of Figure 7.

The top panel in Figure 10 visualizes the cumulative gross performance of the long positions for all machine learning models over time with a log-scaled vertical axis. Most of the models have a rather similar cumulative gross performance in the long position. Furthermore, since there are a lot of intersections between lines corresponding to different models, there is no clear ordering of the machine learning models with respect to the cumulative gross performance of the long position preserved over time (e.g. random forest performs best between 1987 and 1997, but is mediocre from 2000 onwards). Interestingly, in the long positions OLS-3 outperforms all other models, which can also be seen from Table 1. In terms of cumulative performance of the long positions, OLS-3 only starts to outperform the other models from 2003 onwards. This might be related to the above mentioned drop in returns of characteristics based long-short portfolios around 2003.

The bottom panel in Figure 10 shows the cumulative gross performance of the short positions for all machine learning models over time with a log-scaled vertical axis. In contrast to the long positions, there are clear differences in the cumulative gross performance of the short position across different models. Moreover, since there are just a few intersections between lines corresponding to different models, the ordering of the machine learning models with respect to the cumulative gross performance of the short position seems to be preserved over time. Compared to other models, OLS-3 clearly underperforms in the short position.

Taking top and bottom panel of Figure 10 together, differences between models in the performance of the long-short portfolio are mainly due to differences in the performance in the short position. In addition, all machine learning models except OLS-3 derive a large fraction of their performance and all of their outperformance against OLS-3 from the short positions.

Figure 11 visualizes the cumulative net performance of the long-short portfolios for all machine learning models over time with a log-scaled vertical axis. In contrast to neural networks and linear models (OLS-all, Ridge, ENet, Lasso, PCR), which either constantly out- or underperform OLS-3 in terms of cumulative net performance, tree-based methods outperform OLS-3 until 2003 and then underperform. Except for OLS-3, lasso regression and elastic net, the net returns of the long-short portfolios of all models experience a substantial drop in 2003, causing cumulative net performance to nearly stagnate in some cases.

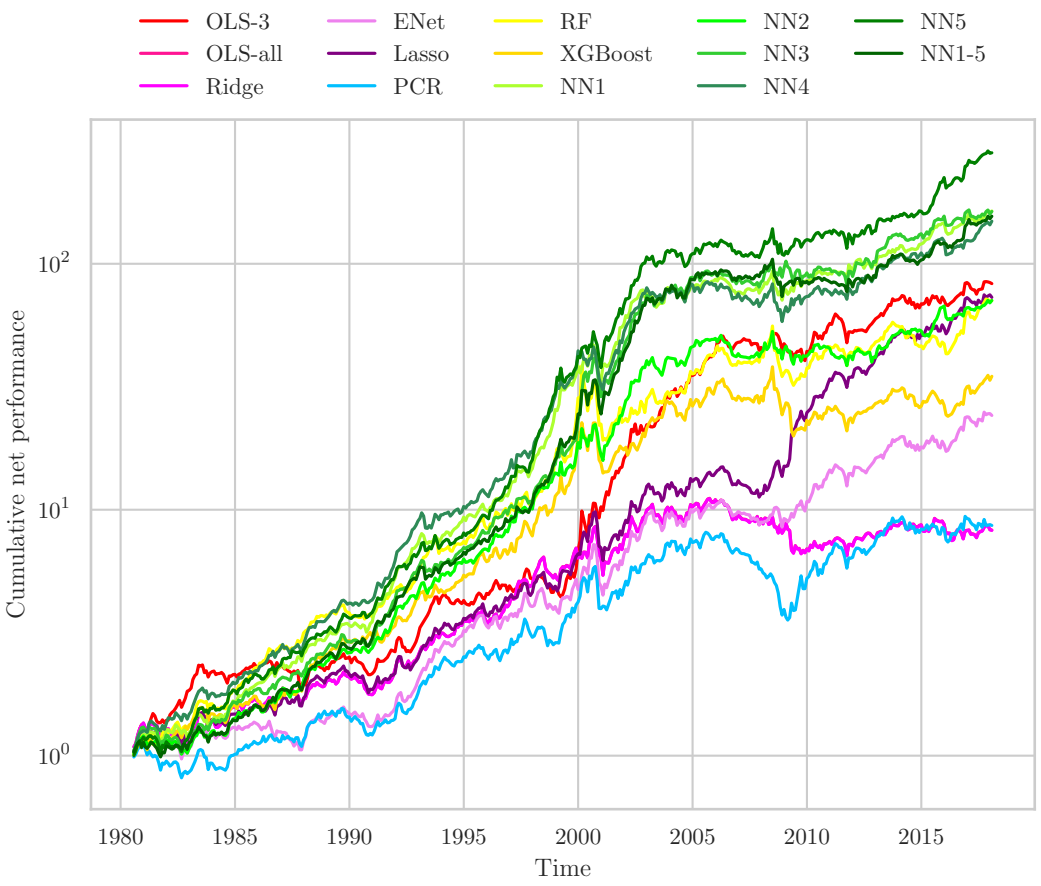

Figure 11: Cumulative net performance of machine learning portfolios
This figure depicts the value-weighted cumulative net performance of long-short portfolios sorted on the 1-month-ahead out-of-sample predicted abnormal return using NYSE breakpoints over time (long = top decile, short = bottom decile). The vertical axis is on log-scale. For the abbreviations of the models, please refer to the description of Figure 7.

Figure 12 displays the correlations between the gross returns of the long-short portfolios for all machine learning models. The plot suggests that, except OLS-3, there are four groups of machine-learning models whose returns are highly correlated ($\rho \geq 0.7$) with returns of other machine learning models belonging to the same group and show medium ($0.5 \leq \rho < 0.7$) to low correlation ($\rho < 0.5$) with returns of machine learning models that belong to a different group. The first group consists of OLS-all and ridge regression. This is in agreement with the findings in Sections 5.1 and 5.2 suggesting that these two models behave very similarly. The next group of models contains elastic net, lasso regression and PCR. The third group comprises of random forest and XGBoost, i.e. the tree-based methods. Finally, the neural networks form their own group of models. These groups are in line with the previous observations on the comovement of the returns of machine learning models made based on the cumulative gross performance plot. For net returns, these results are not notably different (see Figure 18 in the Appendix).

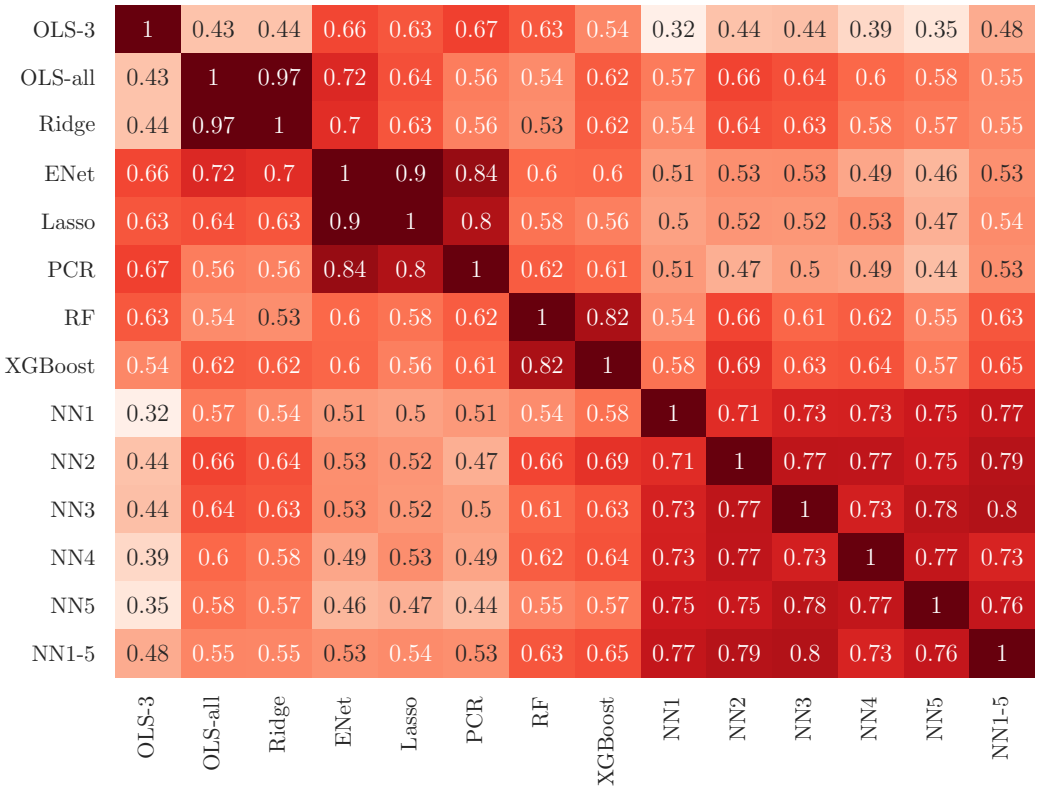

Figure 12: Correlation of gross returns of machine learning portfolios
This figure depicts the correlations between the value-weighted gross returns of long-short portfolios sorted on the 1-month-ahead out-of-sample predicted abnormal return using NYSE breakpoints (long = top decile, short = bottom decile). The numbers report the coefficients of correlation. As the coefficient of correlation increases from 0 to +1, the colour of the corresponding cell changes gradually from white to red. For the abbreviations of the models, please refer to the description of Figure 7.

After examining how the performance develops over time, I will now present for each machine learning model the results of the regression of the long-short portfolio's gross return against the gross factor returns of the Fama-French five-factor model (Fama & French, 2015) augmented by the momentum factor (Carhart, 1997) (FF5FM+Mom) (see Table 3).[17]

This allows for studying two aspects for each machine learning portfolio: On the one hand, how much does a given long-short portfolio load against each of the six factors, or to what extent can the long-short portfolio returns be explained by increased exposure to the six factors? This question can be answered by looking at the coefficients of the regression for the six factors. On the other hand, since higher returns could be due to higher risk, i.e. more exposure towards the six factors, what is the risk-adjusted performance of a given long-short portfolio? This question can be answered by looking at the intercept of the regression, i.e. α. Table 3 also includes the intercept ($\alpha(n)$) obtained when regressing the net returns of the machine learning portfolios

[17]Factor returns are obtained from Wharton Research Database (WRDS) (2021) and Kenneth French's website (see `http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html`).

against the FF5FM+Mom. Except for the intercept, the coefficients obtained from the regression against the gross and the regression against the net returns differ by less than 0.02. Detailed results of the regression for the net returns can be found in Table 11 in the Appendix.

In the following, only coefficients which are statistically significant as indicated by the an absolute value of the t-statistic larger than 1.96 will be considered. All models have positive exposures towards the market factor, the size factor and the momentum factor. Interestingly, all neural networks load less towards the market than the other models, which might partially explain why their gross returns have a lower standard deviation (SD(g)) as mentioned during the discussion of Table 2. Across all models, size and momentum are the most important factors since each model puts largest weight either on size (OLS-3, elast net, PCR) or on momentum (OLS-all, ridge regression, tree-based methods) or equally on both (neural networks). This is in line with the results in 5.2 suggesting that across all models marketcap and momentum are among the features with the highest relative importance. However, compared to other models as well as to other factors OLS-3 overtly loads towards size.

Apart from the common feature that all models have positive exposures towards market, size and momentum, each model has specific exposures to value, profitability and investment except neural networks. All in all, neural networks do not have substantial loadings towards factors other than market, size and momentum. OLS-all and ridge regression are the only models that have a positive exposure towards profitability. OLS-3 and the tree-based methods load positively on value and negatively on profitability. Additionally, the tree-based methods have negative exposure towards investment. Elastic net, lasso regression and PCR load positively on value, but negatively on investment. These observations are consistent with the groups discovered in the correlation plot.

Before costs, all models outperform the FF5FM+Mom after adjusting for risk as indicated by statistically significantly positive alphas (α), including OLS-3. This might seem surprising since one might mistakenly think OLS-3 was nested in the FF5FM+Mom. However, the value factor of the Fama-French five-factor model uses the annually updated book-to-market ratio given the book equity of the previous fiscal year and the market equity as of previous December Lewellen (2015). In contrast, the book-to-market ratio inside OLS-3 is calculated monthly based on the latest available observations of book and market equity. Therefore, OLS-3 also incorporates recent changes in prices and valuation, which might explain its outperformance over the FF5FM+Mom.

OLS-all, ridge regression and PCR have an gross alpha comparable to the one of OLS-3 (0.46%), whereas elastic net and lasso regression have larger gross alphas (0.78% and 0.94%). Tree-based methods (1.24% and 1.11%) as well as neural networks (between 1.30% and 1.58%) substantially outperform the

	α(n)	α	mktrf	smb	hml	rmw	cma	umd	R^2 (%)
OLS-3	0.11	0.46	0.22	1.17	0.49	-0.12	-0.03	0.47	85.04
	(1.3)	(5.21)	(10.09)	(36.11)	(11.8)	(-2.84)	(-0.55)	(23.77)	
OLS-all	-0.28	0.52	0.25	0.32	0.07	0.18	-0.04	0.37	23.81
	(-1.5)	(2.76)	(5.34)	(4.7)	(0.76)	(2.07)	(-0.28)	(8.77)	
Ridge	-0.28	0.52	0.25	0.32	0.07	0.18	-0.04	0.37	23.80
	(-1.51)	(2.76)	(5.34)	(4.7)	(0.76)	(2.07)	(-0.28)	(8.77)	
ENet	0.02	0.78	0.28	0.77	0.38	0.07	-0.29	0.21	38.21
	(0.13)	(4.43)	(6.47)	(11.99)	(4.59)	(0.84)	(-2.39)	(5.3)	
Lasso	0.25	0.94	0.32	0.76	0.42	0.09	-0.25	0.15	36.15
	(1.41)	(5.17)	(7.09)	(11.47)	(4.89)	(1.04)	(-1.96)	(3.66)	
PCR	-0.19	0.43	0.36	0.80	0.43	-0.09	-0.29	0.16	46.79
	(-1.12)	(2.55)	(8.7)	(12.91)	(5.4)	(-1.16)	(-2.5)	(4.2)	
RF	0.43	1.24	0.17	0.56	0.24	-0.37	-0.52	0.60	51.04
	(2.42)	(6.86)	(3.77)	(8.51)	(2.83)	(-4.42)	(-4.19)	(14.94)	
XGBoost	0.27	1.11	0.21	0.36	0.11	-0.32	-0.34	0.50	39.52
	(1.45)	(5.92)	(4.44)	(5.17)	(1.2)	(-3.68)	(-2.59)	(12.03)	
NN1	0.77	1.55	0.14	0.14	0.03	-0.03	-0.35	0.14	12.08
	(4.23)	(8.43)	(3.15)	(2.05)	(0.38)	(-0.36)	(-2.77)	(3.34)	
NN2	0.47	1.30	0.09	0.29	0.01	-0.07	-0.17	0.33	22.43
	(2.62)	(7.19)	(1.94)	(4.43)	(0.15)	(-0.88)	(-1.34)	(8.25)	
NN3	0.64	1.48	0.12	0.30	0.00	-0.01	-0.18	0.30	18.80
	(3.29)	(7.49)	(2.51)	(4.15)	(0.02)	(-0.16)	(-1.34)	(6.95)	
NN4	0.59	1.42	0.17	0.28	0.06	0.05	-0.23	0.24	17.08
	(3.24)	(7.65)	(3.82)	(4.2)	(0.71)	(0.52)	(-1.83)	(5.72)	
NN5	0.77	1.58	0.15	0.22	-0.02	0.04	-0.13	0.22	13.05
	(4.04)	(8.21)	(3.12)	(3.18)	(-0.23)	(0.48)	(-1.01)	(5.08)	
NN1-5	0.70	1.52	0.11	0.35	0.01	-0.17	-0.12	0.24	21.16
	(3.92)	(8.33)	(2.48)	(5.19)	(0.07)	(-2.04)	(-0.94)	(5.82)	

Table 3: Regression of machine learning portfolio returns against the FF5FM+Mom

This table summarizes the regressions of gross machine learning portfolio returns against the returns of the factors of the Fama-French five-factor model (Fama & French, 2015) augmented by the momentum factor (Carhart, 1997). mktrf, smb, hml, rmw, cma and umd represent the estimated regression coefficients of the six factor. α denotes the intercept. t-statistics are indicated in parentheses below. In addition, $\alpha(n)$ reports the intercept of the regression of net returns of the same machine learning portfolio against the six factors. For the abbreviations of the models, see the description of Figure 7.

FF5FM+Mom as measured by gross alpha. Overall, the gross risk-adjusted performance closely aligns with the gross return of the respective machine-learning based long-short portfolios as reported in Table 2. This suggests that the outperformance in terms of gross returns of the various machine learning models against OLS-3 is not driven by higher risk.

After costs, just the neural networks (α(n) between 0.47% and 0.77%) as well as the random forest (0.43%) achieve statistically significant outperformance against the FF5FM+Mom. All other models have a net alpha that is not statistically significantly different from zero and thus neither statistically significantly negative. All in all, except for the random random forest the net outperformance aligns with the net returns of the respective machine-learning based long-short portfolios as reported in Table 2.

The FF5FM+Mom allows to explain 85% of the variation in the gross returns of the long-short portfolio based on OLS-3. This is not surprising, since apart from OLS-3 using a different version of the book-to-market ratio, the FF5FM+Mom incorporates all predictors of OLS-3. Besides that, the FF5FM+Mom can on average explain 37% of the variation in the gross return of the long-short portfolios of all other models except the neural networks. However, for neural network based long-short portfolios it can only explain on average 17% in the variation of gross returns. This is partly due to the fact that neural networks inherently model nonlinearities and interaction effects which are not captured by linear models such as the FF5FM+Mom.

Drobetz and Otto (2021) and Gu et al. (2020) (except for elastic net) report for all machine learning models under consideration significantly positive gross α-s relative to the FF5FM+Momm, also for OLS-3. Also regarding the R^2, the results of Gu et al. (2020) follow a similar pattern as mine (except for elastic net). Beyond that, for both tree-based methods and neural networks, they report gross α-s that match quite closely with those I find. As for the average realized gross returns, the numerical results of Drobetz and Otto (2021) differ from mine and those of Gu et al. (2020). For example, they find PCR to achieve a larger gross α than those of tree-based methods and neural networks.

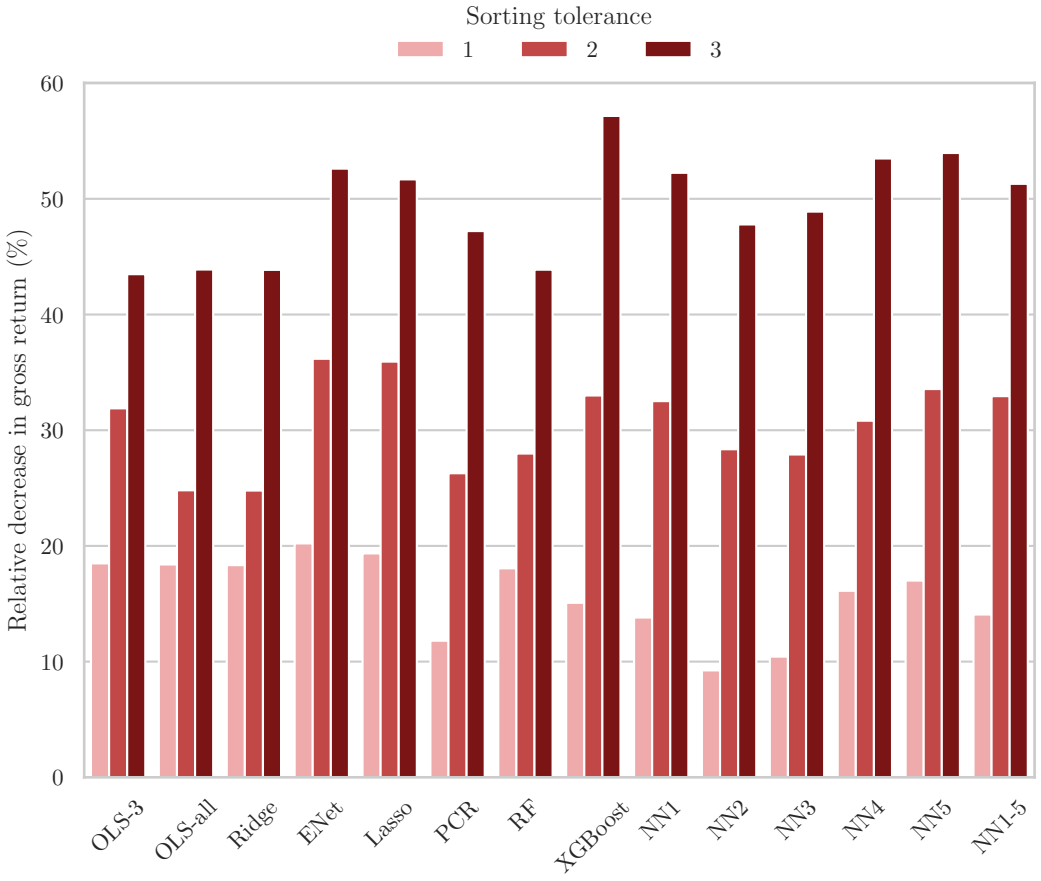

Figure 13: Effect of turnover mitigation sorting on the gross returns of machine learning portfolios

This table reports the effect turnover mitigation sorting has on the value-weighted performance of long-short portfolios sorted on the 1-month-ahead out-of-sample predicted abnormal return using NYSE breakpoints (long = top decile, short = bottom decile) as measured by the relative decrease in gross returns (%). For details on turnover mitigation sorting, see the second part of Section 4.5. For the abbreviations of the models, see the description of Figure 7.

5.4 Turnover Mitigation

Having analyzed the gross and net performance of machine learning models, I will now investigate whether turnover mitigation sorting is a suitable approach to further enhance the net performance of machine learning portfolios. Table 4 reports for each machine learning model the performance of the long-short portfolio when allowing for different sorting tolerances.[18] Figure 13 visualizes for different sorting tolerances the effect of turnover mitigation sorting on gross returns of machine learning portfolios as measured by the relative decrease in gross returns compared to standard sorting, i.e. sorting with 0 tolerance. As expected, across all machine learning models gross returns decrease as the sorting tolerance increases. However, the size of the reduction in gross returns varies between machine learning models. For a sorting tolerance of 1, the gross return decreases in relative terms on average by 15%. Nevertheless, for

[18]For details on turnover mitigation sorting, see the second part of Section 4.5.

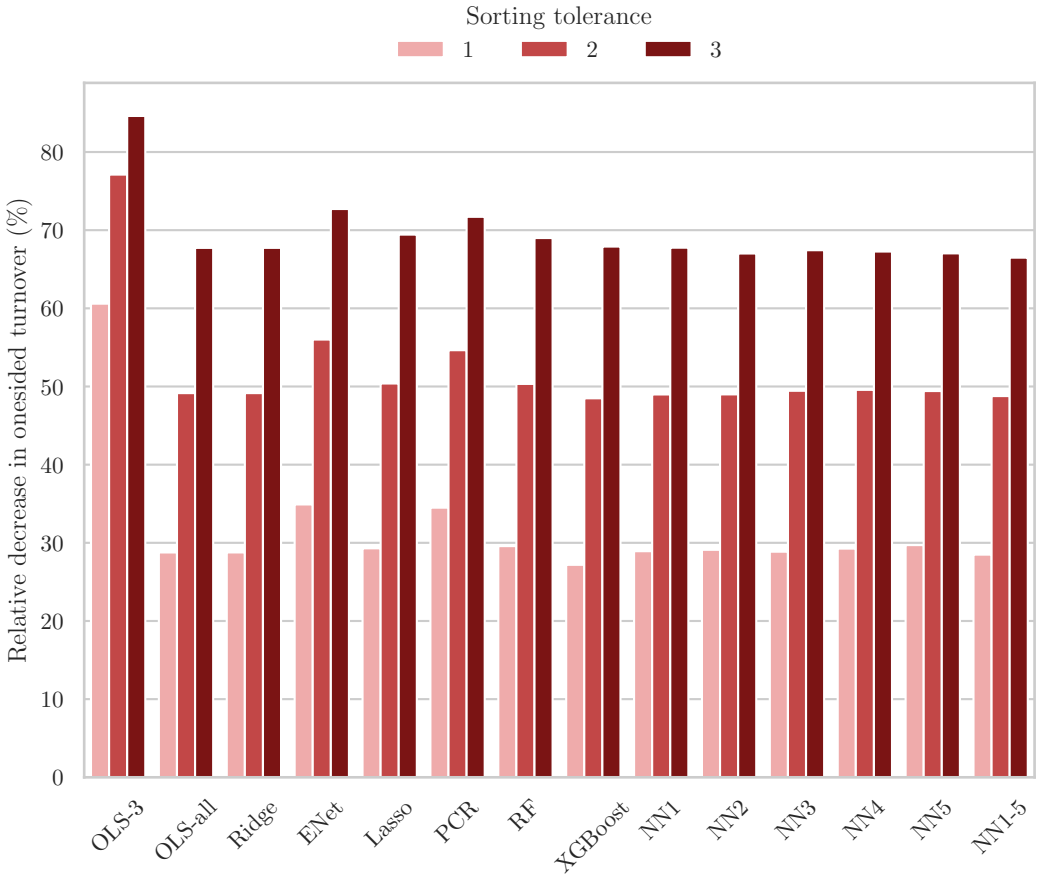

Figure 14: Effect of turnover mitigation sorting on the turnover of machine learning portfolios

This table reports the effect turnover mitigation sorting has on the turnover of long-short portfolios sorted on the 1-month-ahead out-of-sample predicted abnormal return using NYSE breakpoints (long = top decile, short = bottom decile) as measured by the relative decrease in onsided turnover (%). For details on turnover mitigation sorting, see the second part of Section 4.5. For the abbreviations of the models, see the description of Figure 7.

PCR, NN1 and NN2 the decrease in gross returns is clearly smaller than the average. With a sorting tolerance of 2, the gross returns of OLS-all and ridge regression decline by less than the average, which is 30%. Finally, compared to standard sorting for a sorting tolerance of 3 gross returns drop on average by 50%, however for OLS-3, OLS-all, ridge regression and random forest the reduction in gross returns is visibly lower.

Figure 14 depicts for different sorting tolerances the effect of turnover mitigation sorting on turnovers of machine learning portfolios in terms of the relative decrease in turnovers compared to standard sorting. Unsurprisingly, turnovers decline considerably as the sorting tolerance increases. Except for OLS-3, the effect of turnover mitigation sorting on turnovers is the same across all models. In particular, sorting with a tolerance of 1, 2 and 3 leads to a reduction in turnover by approximately 30%, 50% and 68% respectively. For OLS-3, the decrease in turnover is even larger. The effects of turnover mitigation sorting on trading costs are nearly identical. For the corresponding figure, the reader

Model	Tol	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)
OLS-3	0	2.15	1.43	4.49	6.77	48.57	0.35	1.08	5.13
	1	1.94	1.16	4.29	5.76	19.14	0.15	1.02	5.05
	2	1.77	0.97	4.05	5.10	11.12	0.09	0.88	4.64
	3	1.61	0.81	3.78	4.54	7.47	0.06	0.75	4.19
OLS-all	0	3.31	1.35	4.20	6.84	132.58	0.80	0.56	2.84
	1	2.95	1.10	3.87	6.05	94.44	0.56	0.54	2.99
	2	2.51	1.02	3.32	6.51	67.41	0.40	0.62	3.99
	3	1.94	0.76	2.68	6.01	42.78	0.25	0.51	4.07
Ridge	0	3.31	1.35	4.20	6.83	132.58	0.80	0.56	2.83
	1	2.95	1.10	3.87	6.05	94.44	0.56	0.54	2.99
	2	2.51	1.02	3.32	6.51	67.41	0.40	0.62	3.99
	3	1.94	0.76	2.68	6.01	42.77	0.25	0.51	4.07
ENet	0	2.50	1.56	4.34	7.62	116.22	0.76	0.80	3.95
	1	2.25	1.24	3.95	6.68	75.64	0.50	0.74	4.04
	2	1.93	0.99	3.46	6.09	51.13	0.34	0.66	4.06
	3	1.53	0.74	3.14	4.99	31.72	0.21	0.53	3.61
Lasso	0	2.37	1.74	4.43	8.36	104.76	0.69	1.05	5.07
	1	2.13	1.41	4.04	7.40	74.07	0.49	0.92	4.88
	2	1.82	1.12	3.54	6.70	51.98	0.34	0.78	4.70
	3	1.42	0.84	3.03	5.91	32.02	0.21	0.64	4.48
PCR	0	2.38	1.20	4.54	5.62	96.78	0.62	0.58	2.73
	1	2.14	1.06	4.23	5.33	63.39	0.41	0.65	3.27
	2	1.86	0.89	3.98	4.73	43.89	0.29	0.60	3.21
	3	1.52	0.63	3.75	3.60	27.38	0.18	0.45	2.58
RF	0	1.74	1.89	5.07	7.90	124.51	0.81	1.07	4.52
	1	1.47	1.55	4.84	6.79	87.68	0.57	0.98	4.32
	2	1.22	1.36	4.60	6.26	61.84	0.40	0.96	4.44
	3	0.96	1.06	4.01	5.60	38.61	0.25	0.81	4.29
XGBoost	0	2.36	1.74	4.75	7.79	138.01	0.84	0.90	4.05
	1	2.05	1.48	4.49	6.99	100.49	0.61	0.87	4.14
	2	1.68	1.17	4.14	5.98	71.09	0.43	0.74	3.80
	3	1.26	0.75	3.57	4.44	44.30	0.26	0.48	2.87
NN1	0	4.26	1.98	3.84	10.99	127.59	0.78	1.21	6.75
	1	3.72	1.71	3.51	10.35	90.68	0.55	1.16	7.09
	2	3.13	1.34	3.19	8.92	65.10	0.39	0.95	6.35
	3	2.36	0.95	2.73	7.37	41.14	0.24	0.71	5.51
NN2	0	3.81	1.85	3.99	9.83	131.56	0.82	1.03	5.49
	1	3.31	1.68	3.63	9.82	93.27	0.58	1.10	6.49
	2	2.78	1.33	3.21	8.76	67.11	0.41	0.91	6.05
	3	2.11	0.97	2.75	7.46	43.40	0.26	0.70	5.44
NN3	0	3.69	2.05	4.27	10.22	132.06	0.83	1.22	6.17
	1	3.20	1.84	3.85	10.16	93.94	0.58	1.26	7.01
	2	2.68	1.48	3.40	9.25	66.76	0.41	1.07	6.74
	3	2.05	1.05	2.85	7.82	43.00	0.26	0.79	5.92
NN4	0	3.46	2.02	3.98	10.75	130.84	0.82	1.19	6.41
	1	3.00	1.69	3.51	10.23	92.55	0.57	1.12	6.81
	2	2.51	1.39	3.24	9.16	65.98	0.41	0.99	6.53
	3	1.92	0.94	2.77	7.19	42.83	0.26	0.68	5.21
NN5	0	2.47	2.14	4.02	11.31	128.38	0.81	1.34	7.13
	1	2.14	1.78	3.59	10.53	90.24	0.56	1.22	7.29
	2	1.80	1.42	3.16	9.57	64.94	0.40	1.03	6.94
	3	1.39	0.99	2.88	7.27	42.33	0.25	0.73	5.39
NN1-5	0	3.42	2.02	4.01	10.67	130.86	0.81	1.20	6.45
	1	2.98	1.73	3.79	9.72	93.57	0.58	1.16	6.57
	2	2.50	1.35	3.21	8.94	67.03	0.41	0.95	6.29
	3	1.91	0.98	2.78	7.50	43.86	0.26	0.72	5.51

Table 4: Performance of machine learning portfolios under turnover mitigation sorting

This table reports the value-weighted performance of long-short portfolios sorted on the 1-month-ahead out-of-sample predicted abnormal return using NYSE breakpoints (long = top decile, short = bottom decile) under turnover mitigation sorting for various sorting tolerances. For details on turnover mitigation sorting, see the second part of Section 4.5. For the column names, see the description of Table 2. For the abbreviations of the models, see the description of Figure 7.

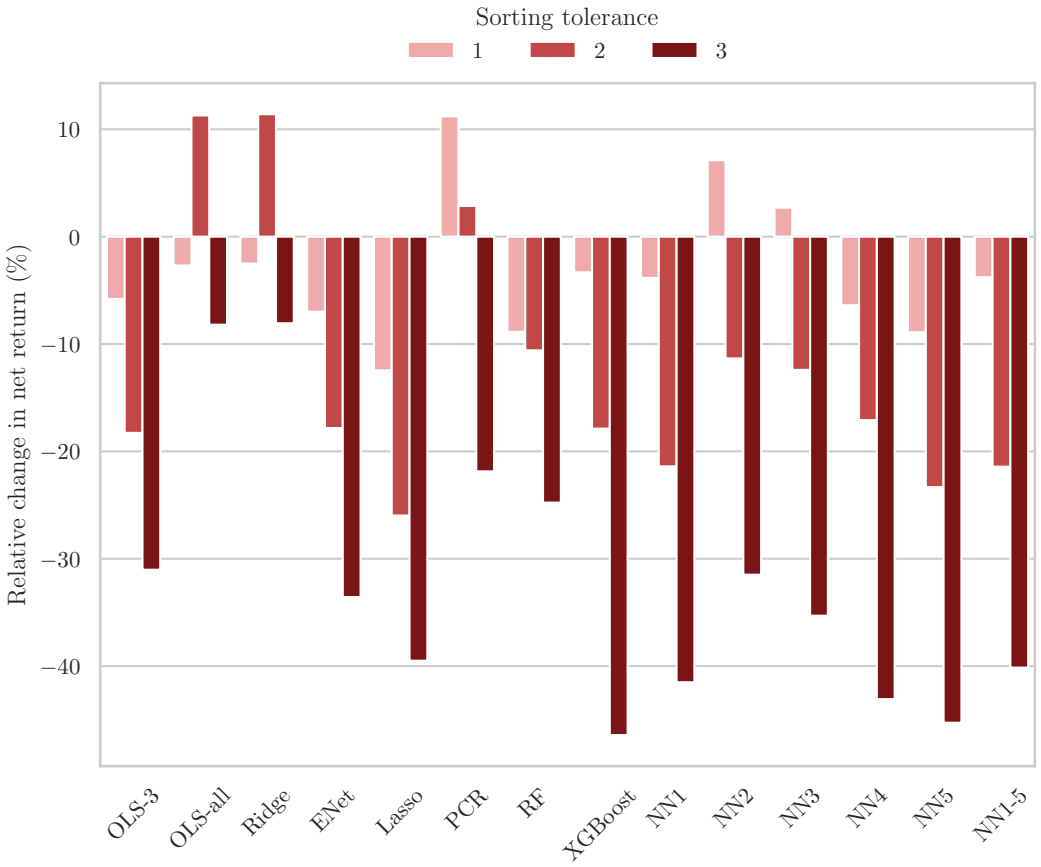

Figure 15: Effect of turnover mitigation sorting on the net returns of machine learning portfolios

This table reports the effect turnover mitigation sorting has on the net returns of long-short portfolios sorted on the 1-month-ahead out-of-sample predicted abnormal return using NYSE breakpoints (long = top decile, short = bottom decile) as measured by the relative increase in net returns (%). For details on turnover mitigation sorting, see the second part of Section 4.5. For the abbreviations of the models, see the description of Figure 7.

may refer to Figure 19 in the Appendix. In summary, Figures 13 and 14 demonstrate the above described countervailing effects of turnover mitigation sorting on net returns.

Figure 15 shows for different sorting tolerances the overall effect of turnover mitigation sorting on net returns of machine learning portfolios as measured by the relative change in net returns relative to standard sorting. Turnover mitigation sorting increases the net returns of PCR, NN2 and NN3 for a sorting tolerance of 1 and of OLS-all and ridge regression for a sorting tolerance of 2. For all other models, turnover mitigation sorting does not improve net returns. This is in line with the observations above: Since turnover mitigation sorting affects the turnover and trading costs of all models (except OLS-3) identically, those models that suffer the lowest decrease in gross returns due to turnover mitigation sorting, profit from it the most in terms of net return. In fact, the models that suffer the lowest decrease in gross returns are PCR, NN2 and NN3 for a sorting tolerance of 1 and OLS-all and ridge regression for a sorting tolerance of 2 (see Figure 13). However, despite these improvements NN5 still achieves under standard sorting the highest net return across all models and sorting tolerances, since OLS-all, ridge regression, PCR, NN2 and NN3

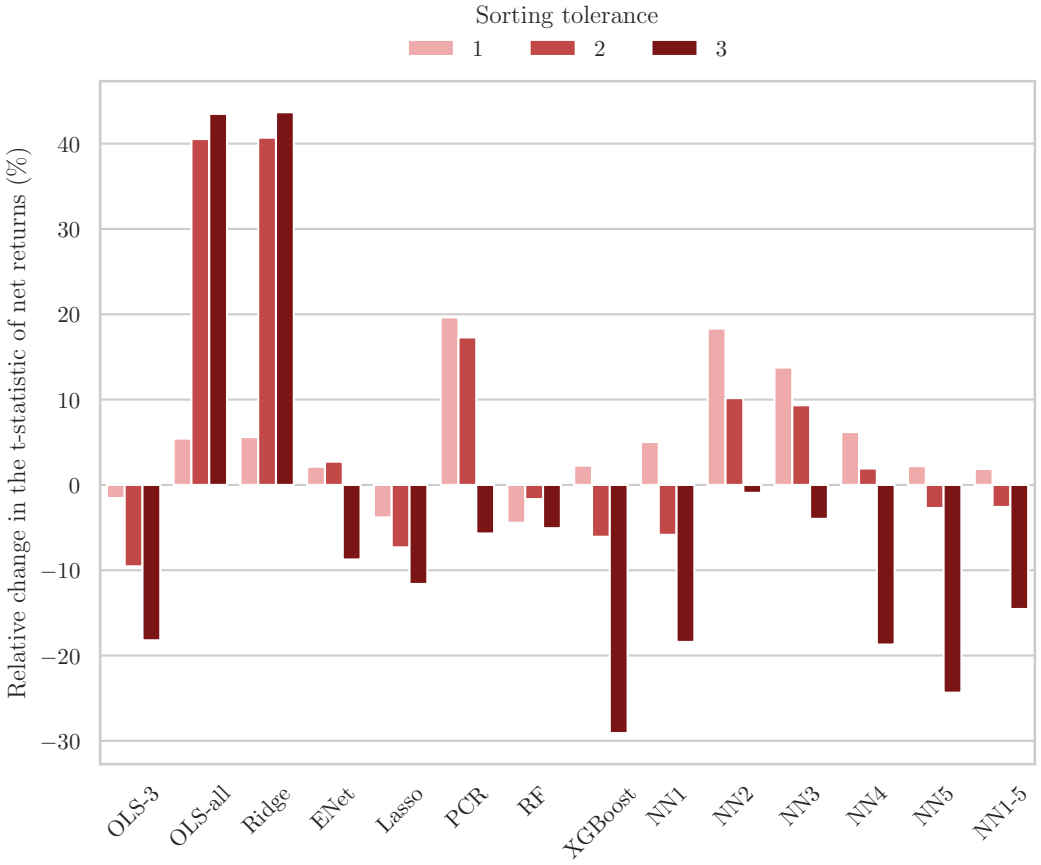

Figure 16: Effect of turnover mitigation sorting on the t-statistic of net returns of machine learning portfolios

This table reports the effect turnover mitigation sorting has on the t-statistic of net returns of long-short portfolios sorted on the 1-month-ahead out-of-sample predicted abnormal return using NYSE breakpoints (long = top decile, short = bottom decile) as measured by the relative increase in the t-statistic (%). For details on turnover mitigation sorting, see the second part of Section 4.5. For the abbreviations of the models, see the description of Figure 7.

previously clearly underperformed NN5 (see Table 4).

Nevertheless, besides having an impact on the net returns turnover mitigation sorting affects the standard deviation of returns and thereby also the combination of risk and net return as measured by the t-statistic of net returns. Figure 16 visualizes for different sorting tolerances the effect of turnover mitigation sorting on the t-statistics of net returns of machine learning portfolios as measured by the relative change in t-statistics compared to standard sorting. For a sorting tolerance of 1, turnover mitigation sorting increases the t-statistics of net returns for all models except OLS-3, lasso regression and random forest. But, the improvements in t-statistics are rather limited except for the models for which turnover mitigation sorting increases net returns (i.e. OLS-all, ridge regression, PCR, NN2 and NN3). However, OLS-all, ridge regression and PCR previously clearly underperformed in terms of the t-statistic of net returns. Therefore, it is not surprising that NN5, whose t-statistic of net returns is slightly higher for a sorting tolerance of 1 than under standard sorting, still outperforms across all models and sorting tolerances (see Table 4).

	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)
NN5	2.47	2.14	4.02	11.31	128.38	0.81	1.34	7.13
BoC	1.99	2.29	4.94	9.85	127.53	0.83	1.47	6.37
NN-all	3.31	2.16	4.03	11.38	130.16	0.82	1.34	7.12
NN-all+RF	3.01	2.31	4.23	11.58	129.71	0.83	1.48	7.50
NN-all+RF+OLS-3	2.77	2.25	4.37	10.94	128.03	0.82	1.43	6.98

Table 5: Performance of machine learning ensembles portfolios
This table reports the value-weighted performance of long-short portfolios sorted on the 1-month-ahead out-of-sample abnormal return predicted by an ensemble model using NYSE breakpoints. A prediction of the ensemble model is the average of the predictions of its constituting models. The ensembles are defined as: BoC = Lasso + RF + NN5; NN-all = NN1 + NN2 + NN3 + NN4 + NN5 + NN1-5; For the abbreviations of the constituting models, see the description of Figure 7. For the column names, see the description of Table 2.

5.5 Ensembles

In the following, four special ensembles of machine learning models will be dicussed. These ensemble models generate predictions by simply taking the average across the predictions of their constituting machine learning models. The ensemble models include a "Best-of-Class" (BoC)-ensemble which combines the best performing model among the linear models (i.e. lasso regression), the tree-based methods (i.e. random forest) and the neural networks (i.e. NN5). The ensemble comprising of all models outperforming OLS-3 after costs, i.e. all neural network models presented so far, is referred to as NN-all. NN-all+RF represents all models that have a net return higher than (i.e. all neural networks) or equal to the one of OLS-3 (i.e. random forest). Finally, augmenting the latter to also include OLS-3 yields NN-all+RF+OLS-3. Since NN5 is the best performing baseline machine learning model, its performance will be compared to the one of the ensembles of machine learning models, in order to examine, whether and if so to what degree ensembles of machine learning models outperform baseline machine learning models. Table 5 shows the performance of the machine learning portfolio for NN5 as well as for the above mentioned ensembles.

In terms of average realized gross return, standard deviation of gross returns and the t-statistic of gross returns, NN-all's performance is nearly the same as the one of NN-5. All other ensembles have substantially higher gross returns than NN-5 but also higher risk as measured by the standard deviation of gross returns. As a result of the disproportionately higher risk as compared to the higher gross returns, BoC and NN-all+RF+OLS-3 underperform NN5 in terms of the t-statistic of gross returns. Hence, just NN-all+RF achieves a more favorable combination of gross returns and risk than NN5.

The turnover as well as the trading costs are rather identical across all models under consideration. Thus all models are equally affected by trading costs.

Therefore, the performance of all models relative to NN5 in terms of net returns is the same as in terms of gross returns: NN-all performs similar to NN5, whereas all other models earn higher net returns than NN5. However, due to the disproportionately higher risk of BoC and NN-all+RF+OLS-3, just NN-all+RF outperforms NN5 in terms of its combination of net return and risk.

6 Conclusion

Although several machine learning portfolios have a partly substantially higher gross performance than traditional benchmarks, only machine learning portfolios based on neural networks also outperform in terms of net returns. This also holds after adjusting for risk. The results from the attempt to further improve the net performance of machine learning portfolios suggest that ensemble models seem to have a higher potential than turnover mitigation sorting in this respect.

Based on these results, future research aimed at further improving the after-cost performance of machine learning portfolios may examine on the one hand cost mitigation strategies other than turnover mitigation sorting and on the other hand more sophisticated types of ensemble models.

While Novy-Marx and Velikov (2016) state that turnover mitigation sorting, to which they refer as buy/hold spreads, is the most successful cost mitigation strategy for portfolios based on traditional sorts on one characteristic, this does not necessarily imply that the same holds true for machine learning portfolios which make use of a large set of characteristics. Therefore, it might be insightful to analyze and compare different cost mitigation strategies for machine learning portfolios, such as trading only low-expected-trading-cost stocks or reducing the rebalancing frequency from monthly to quarterly (Novy-Marx & Velikov, 2019).

The type of ensemble used in the above analyses generates predictions by averaging over the prediction of its constituting machine learning models. However, there are more sophisticated types of ensembles that do not somehow arbitrarily weight all constituting models equally such as stacking ensembles. Roughly speaking, stacking ensembles fit a model on top of the predictions of the constituting models to obtain the optimal weights with which the predictions of each constituting model enter the ensemble.

Appendices

Variable	Description	Data category	Economic category
AbnormalAccruals	Abnormal Accruals	Accounting	Accruals
Accruals	Accruals	Accounting	Accruals
AccrualsBM	Book-to-market and accruals	Accounting	Valuation
Activism1	Takeover vulnerability	13F	Other
Activism2	Active shareholders	13F	Ownership
AdExp	Advertising Expense	Accounting	R&d
AgeIPO	IPO and age	Event	Other
AM	Total assets to market	Accounting	Valuation
AnalystRevision	EPS forecast revision	Analyst	Earnings forecast
AnalystValue	Analyst Value	Analyst	Valuation
AnnouncementReturn	Earnings announcement return	Price	Earnings event
AOP	Analyst Optimism	Analyst	Other
AssetGrowth	Asset growth	Accounting	Investment
Beta	CAPM beta	Price	Risk
BetaFP	Frazzini-Pedersen Beta	Price	Other
BetaLiquidityPS	Pastor-Stambaugh liquidity beta	Price	Liquidity
BetaTailRisk	Tail risk beta	Price	Risk
betaVIX	Systematic volatility	Price	Volatility
BidAskSpread	Bid-ask spread	Trading	Liquidity
BM	Book to market using most recent ME	Accounting	Valuation
BMdec	Book to market using December ME	Accounting	Valuation
BookLeverage	Book leverage (annual)	Accounting	Leverage
BPEBM	Leverage component of BM	Accounting	Leverage
BrandInvest	Brand capital investment	Accounting	Investment alt
Cash	Cash to assets	Accounting	Asset composition
CashProd	Cash Productivity	Accounting	Profitability alt
CBOperProf	Cash-based operating profitability	Accounting	Profitability
CF	Cash flow to market	Accounting	Valuation
cfp	Operating Cash flows to price	Accounting	Valuation
ChangeInRecommendation	Change in recommendation	Analyst	Recommendation
ChAssetTurnover	Change in Asset Turnover	Accounting	Sales growth
ChEQ	Growth in book equity	Accounting	Investment
ChForecastAccrual	Change in Forecast and Accrual	Analyst	Earnings forecast
ChInv	Inventory Growth	Accounting	Investment alt
ChInvIA	Change in capital inv (ind adj)	Accounting	Investment growth
ChNAnalyst	Decline in Analyst Coverage	Analyst	Earnings event
ChNNCOA	Change in Net Noncurrent Op Assets	Accounting	Investment alt
ChNWC	Change in Net Working Capital	Accounting	Investment alt
ChTax	Change in Taxes	Accounting	Other
CitationsRD	Citations to RD expenses	Other	Profitability alt
CompEquIss	Composite equity issuance	Accounting	External financing
CompositeDebtIssuance	Composite debt issuance	Accounting	External financing
ConsRecomm	Consensus Recommendation	Analyst	Recommendation
ConvDebt	Convertible debt indicator	Event	External financing
CoskewACX	Coskewness using daily returns	Price	Risk
Coskewness	Coskewness	Price	Risk
CredRatDG	Credit Rating Downgrade	Event	Other
CustomerMomentum	Customer momentum	Other	Lead lag
DebtIssuance	Debt Issuance	Event	External financing
DelBreadth	Breadth of ownership	13F	Ownership
DelCOA	Change in current operating assets	Accounting	Investment alt
DelCOL	Change in current operating liabilities	Accounting	External financing
DelDRC	Deferred Revenue	Accounting	Investment alt
DelEqu	Change in equity to assets	Accounting	Investment
DelFINL	Change in financial liabilities	Accounting	External financing
DelLTI	Change in long-term investment	Accounting	Investment
DelNetFin	Change in net financial assets	Accounting	Investment alt
DivInit	Dividend Initiation	Event	Payout indicator
DivOmit	Dividend Omission	Event	Payout indicator
DivSeason	Dividend seasonality	Event	Payout indicator
DivYieldST	Predicted div yield next month	Accounting	Valuation
dNoa	change in net operating assets	Accounting	Investment
DolVol	Past trading volume	Trading	Volume
DownRecomm	Down forecast EPS	Analyst	Earnings forecast
EarningsConsistency	Earnings consistency	Accounting	Earnings growth
EarningsForecastDisparity	Long-vs-short EPS forecasts	Analyst	Earnings forecast
EarningsStreak	Earnings surprise streak	Accounting	Earnings growth
EarningsSurprise	Earnings Surprise	Analyst	Earnings growth
EarnSupBig	Earnings surprise of big firms	Accounting	Lead lag
EBM	Enterprise component of BM	Accounting	Valuation
EntMult	Enterprise Multiple	Accounting	Valuation
EP	Earnings-to-Price Ratio	Price	Valuation
EquityDuration	Equity Duration	Price	Valuation
ExchSwitch	Exchange Switch	Event	Other
ExclExp	Excluded Expenses	Analyst	Composite accounting
FEPS	Analyst earnings per share	Analyst	Profitability
fgr5yrLag	Long-term EPS forecast	Analyst	Earnings forecast
FirmAge	Firm age based on CRSP	Other	Info proxy
FirmAgeMom	Firm Age - Momentum	Price	Momentum

ForecastDispersion	EPS Forecast Dispersion	Analyst	Volatility
FR	Pension Funding Status	Accounting	Composite accounting
Frontier	Efficient frontier index	Accounting	Valuation
Governance	Governance Index	Other	Other
GP	gross profits / total assets	Accounting	Profitability
GrAdExp	Growth in advertising expenses	Accounting	Investment alt
grcapx	Change in capex (two years)	Accounting	Investment growth
grcapx3y	Change in capex (three years)	Accounting	Investment growth
GrLTNOA	Growth in long term operating assets	Accounting	Investment
GrSaleToGrInv	Sales growth over inventory growth	Accounting	Sales growth
GrSaleToGrOverhead	Sales growth over overhead growth	Accounting	Sales growth
Herf	Industry concentration (sales)	Other	Other
HerfAsset	Industry concentration (assets)	Other	Other
HerfBE	Industry concentration (equity)	Other	Other
High52	52 week high	Price	Momentum
hire	Employment growth	Other	Investment alt
IdioRisk	Idiosyncratic risk	Price	Volatility
IdioVol3F	Idiosyncratic risk (3 factor)	Price	Volatility
IdioVolAHT	Idiosyncratic risk (AHT)	Price	Volatility
Illiquidity	Amihud's illiquidity	Trading	Liquidity
IndIPO	Initial Public Offerings	Event	External financing
IndMom	Industry Momentum	Price	Momentum
IndRetBig	Industry return of big firms	Price	Lead lag
IntanBM	Intangible return using BM	Accounting	Long term reversal
IntanCFP	Intangible return using CFtoP	Accounting	Long term reversal
IntanEP	Intangible return using EP	Accounting	Long term reversal
IntanSP	Intangible return using Sale2P	Accounting	Long term reversal
IntMom	Intermediate Momentum	Price	Momentum
Investment	Investment to revenue	Accounting	Investment
InvestPPEInv	change in ppe and inv/assets	Accounting	Investment
InvGrowth	Inventory Growth	Accounting	Profitability
IO_ShortInterest	Inst own among high short interest	13F	Ownership
iomom_cust	Customers momentum	Other	Lead lag
iomom_supp	Suppliers momentum	Other	Lead lag
Leverage	Market leverage	Price	Leverage
LRreversal	Long-run reversal	Price	Long term reversal
marketcap	Market capitalization	Price	Valuation
MaxRet	Maximum return over month	Price	Volatility
MeanRankRevGrowth	Revenue Growth Rank	Accounting	Sales growth
Mom12m	Momentum (12 month)	Price	Momentum
Mom12mOffSeason	Momentum without the seasonal part	Price	Other
Mom6m	Momentum (6 month)	Price	Momentum
Mom6mJunk	Junk Stock Momentum	Price	Momentum
MomOffSeason	Off season long-term reversal	Price	Other
MomOffSeason06YrPlus	Off season reversal years 6 to 10	Price	Other
MomOffSeason11YrPlus	Off season reversal years 11 to 15	Price	Other
MomOffSeason16YrPlus	Off season reversal years 16 to 20	Price	Other
MomRev	Momentum and LT Reversal	Price	Momentum
MomSeason	Return seasonality years 2 to 5	Price	Other
MomSeason06YrPlus	Return seasonality years 6 to 10	Price	Other
MomSeason11YrPlus	Return seasonality years 11 to 15	Price	Other
MomSeason16YrPlus	Return seasonality years 16 to 20	Price	Other
MomSeasonShort	Return seasonality last year	Price	Other
MomVol	Momentum in high volume stocks	Price	Momentum
MRreversal	Medium-run reversal	Price	Long term reversal
MS	Mohanram G-score	Accounting	Composite accounting
NetDebtFinance	Net debt financing	Accounting	External financing
NetDebtPrice	Net debt to price	Accounting	Leverage
NetEquityFinance	Net equity financing	Accounting	External financing
NetPayoutYield	Net Payout Yield	Price	Valuation
NOA	Net Operating Assets	Accounting	Asset composition
NumEarnIncrease	Earnings streak length	Accounting	Earnings growth
OperProf	operating profits / book equity	Accounting	Profitability
OperProfRD	Operating profitability R&D adjusted	Accounting	Profitability
OPLeverage	Operating leverage	Accounting	Other
OptionVolume1	Option to stock volume	Trading	Volume
OptionVolume2	Option volume to average	Trading	Volume
OrderBacklog	Order backlog	Accounting	Sales growth
OrderBacklogChg	Change in order backlog	Accounting	Accruals
OrgCap	Organizational capital	Accounting	R&d
OScore	O Score	Accounting	Default risk
PatentsRD	Patents to RD expenses	Other	Profitability alt
PayoutYield	Payout Yield	Price	Valuation
PctAcc	Percent Operating Accruals	Accounting	Accruals
PctTotAcc	Percent Total Accruals	Accounting	Accruals
prcadj	Adjusted stock price	Price	Valuation
PredictedFE	Predicted Analyst forecast error	Accounting	Earnings forecast
PriceDelayRsq	Price delay r square	Price	Lead lag
PriceDelaySlope	Price delay coeff	Price	Lead lag
PriceDelayTstat	Price delay SE adjusted	Price	Lead lag
ProbInformedTrading	Probability of Informed Trading	Trading	Liquidity
PS	Piotroski F-score	Accounting	Composite accounting
RD	R&D over market cap	Accounting	R&d
RDAbility	R&D ability	Accounting	Other
RDcap	R&D capital-to-assets	Accounting	Asset composition
RDIPO	IPO and no R&D spending	Event	R&d
RDS	Real dirty surplus	Accounting	Composite accounting

realestate	Real estate holdings	Accounting	Asset composition
ResidualMomentum	Momentum based on FF3 residuals	Price	Momentum
retConglomerate	Conglomerate return	Price	Lead lag
ReturnSkew	Return skewness	Price	Risk
ReturnSkew3F	Idiosyncratic skewness (3F model)	Price	Risk
REV6	Earnings forecast revisions	Analyst	Earnings forecast
RevenueSurprise	Revenue Surprise	Accounting	Sales growth
RIO_Disp	Inst Own and Forecast Dispersion	13F	Short sale constraints
RIO_MB	Inst Own and Market to Book	13F	Short sale constraints
RIO_Turnover	Inst Own and Turnover	13F	Short sale constraints
RIO_Volatility	Inst Own and Idio Vol	13F	Short sale constraints
roaq	Return on assets (qtrly)	Accounting	Profitability
RoE	net income / book equity	Accounting	Profitability
sfe	Earnings Forecast to price	Analyst	Valuation
ShareIss1Y	Share issuance (1 year)	Accounting	External financing
ShareIss5Y	Share issuance (5 year)	Accounting	External financing
ShareRepurchase	Share repurchases	Event	Payout indicator
ShareVol	Share Volume	Trading	Volume
ShortInterest	Short Interest	Other	Short sale constraints
sinAlgo	Sin Stock (selection criteria)	Other	Other
skew1	Volatility smirk near the money	Options	Optionrisk
SmileSlope	Put volatility minus call volatility	Options	Optionrisk
SP	Sales-to-price	Accounting	Valuation
Spinoff	Spinoffs	Event	Other
std_turn	Share turnover volatility	Trading	Liquidity
STreversal	Short-term reversal	Price	Short-term reversal
SurpriseRD	Unexpected R&D increase	Event	R&d
tang	Tangibility	Accounting	Asset composition
Tax	Taxable income to income	Accounting	Other
TotalAccruals	Total accruals	Accounting	Investment alt
UpRecomm	Up Forecast	Analyst	Earnings forecast
VarCF	Cash-flow to price variance	Accounting	Cash flow risk
VolMkt	Volume to market equity	Trading	Volume
VolSD	Volume Variance	Trading	Liquidity
VolumeTrend	Volume Trend	Trading	Volume
XFIN	Net external financing	Accounting	External financing
zerotrade	Days with zero trades	Trading	Liquidity
zerotradeAlt1	Days with zero trades	Trading	Liquidity
zerotradeAlt12	Days with zero trades	Trading	Liquidity

Table 6: Definitions of the characteristics
This table reports the definitions of all the characteristics as provided by A. Y. Chen and Zimmermann (in press).

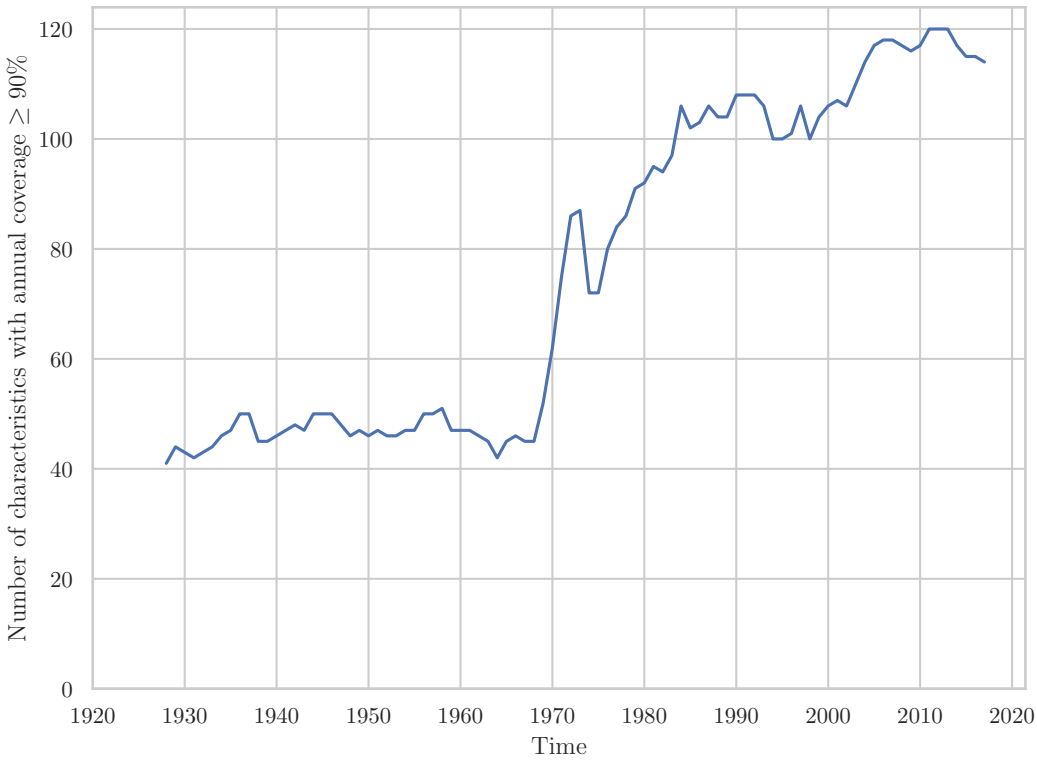

Figure 17: Number of high-coverage characteristics over time
This figure depicts the number of characteristics with an annual coverage $\geq 90\%$ over time.

Variable	Cvrg(%)	Mean	SD	Min	25%	50%	75%	Max
AbnormalAccruals	88.6	-7.7e-05	0.11	-1.6	-0.038	-0.0051	0.03	7.6
Accruals	96.1	-0.016	0.073	-1.6	-0.039	-0.008	0.013	5.8
AccrualsBM	6.2	0.13	0.33	0	0	0	0	1
Activism1	9.2	15	2.7	6	13	15	17	23
Activism2	2.2	8.6	9.9	0	0	7.5	11	87
AdExp	35.0	0.044	0.078	8.4e-07	0.0057	0.019	0.051	4.7
AgeIPO	4.3	22	25	2	7	12	26	1.7e+02
AM	98.1	1.6	2.3	0	0.51	0.99	1.9	1.3e+02
AnalystRevision	83.2	1	1.6	-1.8e+02	0.99	1	1	5.7e+02
AnalystValue	59.9	0.8	0.56	-6.2	0.52	0.71	0.98	51
AnnouncementReturn	93.2	0.005	0.073	-0.91	-0.029	0.0028	0.037	2.9
AOP	59.9	-7.5	7e+02	-1.2e+05	-1	-0.32	-0.069	2.4e+04
AssetGrowth	96.3	-0.2	0.77	-64	-0.21	-0.096	-0.022	1
Beta	100.0	0.97	0.61	-6	0.56	0.89	1.3	10
BetaFP	93.5	1.1	0.54	9.2e-05	0.71	1	1.4	6.9
BetaLiquidityPS	93.5	-0.0034	0.34	-6.4	-0.17	-0.0023	0.17	6.8
BetaTailRisk	81.0	0.68	0.48	-11	0.35	0.62	0.92	5.5
betaVIX	70.1	-0.0006	0.0098	-0.48	-0.0046	-0.0004	0.0036	0.25
BidAskSpread	96.4	0.0081	0.0059	0	0.0044	0.0065	0.0099	0.29
BM	96.2	-0.9	0.89	-9.4	-1.4	-0.81	-0.31	2.9
BMdec	95.9	0.67	0.72	-85	0.3	0.54	0.9	24
BookLeverage	98.1	-2.1	51	-4e+03	-2.6	-1.9	-1.5	9.4e+03
BPEBM	97.7	-0.18	2.2e+02	-1.2e+05	-0.22	-0.038	0.039	1.2e+05
BrandInvest	18.4	-4.5e+03	1.2e+05	-1.4e+07	-1.6e+03	-4.9e+02	-1.4e+02	0
Cash	79.1	0.13	0.17	-0.03	0.019	0.06	0.17	1
CashProd	97.1	-10	2.1e+03	-4.3e+05	-8.7	-0.22	11	2.1e+05
CBOperProf	96.1	0.16	0.13	-11	0.097	0.15	0.22	5.9
CF	98.1	0.11	0.19	-29	0.051	0.093	0.16	22
cfp	94.8	0.058	0.2	-16	0.0078	0.054	0.11	21
ChangeInRecommendation	24.8	-0.011	1.1	-4	-1	0	1	4
ChAssetTurnover	88.2	0.67	1.7e+02	-1.1e+04	-0.16	0.005	0.15	3.9e+04
ChEQ	93.9	-1.3	3.2	-7.6e+02	-1.2	-1.1	-1	-0.0014
ChForecastAccrual	29.6	0.48	0.5	0	0	0	1	1
ChInv	96.3	-0.016	0.049	-1.3	-0.024	-0.0031	0.0006	1.2
ChInvIA	95.9	-7.2e+10	1.3e+13	-5.3e+14	-0.13	0.27	0.69	4.7e+15
ChNAnalyst	0.0	-0.043	0.21	-1	0	0	0	0
ChNNCOA	95.8	-0.0034	0.21	-22	-0.035	0.0017	0.035	1.1e+02
ChNWC	96.1	0.0031	0.059	-1.7	-0.02	0.0012	0.024	3.3
ChTax	92.4	0.0054	7.5	-9.9e+02	-0.0017	0.00096	0.0048	3.4e+03
CitationsRD	15.5	0.36	0.48	0	0	0	1	1
CompEquIss	75.7	0.9	3.3	-6.2	0.097	0.33	0.83	4.5e+02
CompositeDebtIssuance	70.0	-0.6	1.3	-12	-1	-0.45	-0.0043	11
ConsRecomm	7.2	-0.27	0.44	-1	-1	0	0	0
ConvDebt	98.1	-0.2	0.4	-1	0	0	0	0
CoskewACX	99.1	0.15	0.43	-1.5	-0.03	0.089	0.23	6
Coskewness	100.0	0.2	0.4	-2	-0.039	0.17	0.4	2.9
CredRatDG	80.8	-0.029	0.17	-1	0	0	0	0
CustomerMomentum	11.7	0.012	0.1	-0.98	-0.038	0.011	0.06	4.8
DebtIssuance	96.2	-0.62	0.49	-1	-1	-1	0	0
DelBreadth	80.9	0.13	0.86	-40	-0.21	0.088	0.45	40
DelCOA	96.3	-0.038	0.09	-1.5	-0.065	-0.022	0.0015	1.4
DelCOL	96.1	-0.024	0.062	-5.7	-0.042	-0.015	0.0022	1.5
DelDRC	11.6	0.01	0.032	-0.51	-0.00046	0.0026	0.013	0.93
DelEqu	96.2	-0.062	0.96	-22	-0.095	-0.045	-0.01	2.4e+02
DelFINL	95.8	-0.031	0.12	-2.3	-0.054	-0.0017	0.014	2.3
DelLTI	96.3	-0.0039	0.046	-1.7	0	0	0	1.2
DelNetFin	95.8	-0.018	0.15	-2.2	-0.062	-0.0045	0.036	2.2
DivInit	100.0	0.0086	0.092	0	0	0	0	1
DivOmit	100.0	-0.004	0.063	-1	0	0	0	0
DivSeason	63.1	0.41	0.49	0	0	0	1	1
DivYieldST	62.4	0.63	1	0	0	0	1	3
dNoa	96.2	-0.11	0.7	-51	-0.14	-0.052	0.0045	1.2e+02
DolVol	96.6	-4.2	2.4	-13	-5.9	-4.3	-2.4	6.7
DownRecomm	25.0	-0.39	0.49	-1	-1	0	0	0
EarningsConsistency	44.7	0.16	0.95	-26	-0.018	0.1	0.28	78
EarningsForecastDisparity	55.0	31	4.5e+02	-2.4e+04	-9.1	0.47	14	5.8e+04
EarningsStreak	52.3	-0.001	0.05	-7.8	-0.00065	0.00052	0.0018	1
EarningsSurprise	90.6	5.3e+10	2.7e+13	-1e+15	-0.65	0.04	0.67	1.3e+16
EarnSupBig	26.5	-3e+08	1.2e+11	-5.4e+13	-0.38	-0.043	0.28	53
EBM	97.7	0.37	2.2e+02	-1.2e+05	0.11	0.31	0.65	1.2e+05
EntMult	91.5	-21	8.1e+02	-2.1e+05	-13	-8.4	-5.8	1.1e+02
EP	86.3	0.076	0.07	0	0.039	0.062	0.096	10
EquityDuration	96.0	-3e+04	7.7e+06	-2e+09	-18	-16	-15	1.4e+04
ExchSwitch	100.0	-0.014	0.12	-1	0	0	0	0
ExclExp	69.3	-0.047	0.34	-2.5	-0.08	0	0.02	1.6
FEPS	83.6	2.1	2.6	-45	0.94	1.8	2.8	1.6e+02
fgr5yrLag	58.7	-16	9.6	-3.3e+02	-19	-14	-10	1.8e+02
FirmAge	65.5	-3.3e+02	2.4e+02	-1.1e+03	-5e+02	-2.8e+02	-1.4e+02	-1
FirmAgeMom	7.0	0.14	0.46	-0.94	-0.08	0.081	0.28	18
ForecastDispersion	64.8	-0.16	1.2	-2.1e+02	-0.092	-0.04	-0.02	0
FR	36.7	-0.013	0.18	-22	-0.025	-0.0028	0.015	4.3
Frontier	51.4	-0.4	0.77	-7.9	-0.82	-0.35	0.094	4.2
Governance	27.2	-9.2	2.6	-14	-11	-9	-7	-5
GP	95.8	0.37	0.26	-6.3	0.19	0.33	0.5	3.6

GrAdExp	31.9	-0.11	0.39	-7.4	-0.23	-0.093	0.022	4.3
grcapx	91.1	-1.1	32	-5.2e+03	-0.82	-0.23	0.15	5.7e+02
grcapx3y	86.4	-1.5	20	-4.8e+03	-1.6	-1.2	-0.86	2.7e+02
GrLTNOA	96.2	0.03	0.21	-1.1e+02	-0.01	0.029	0.069	22
GrSaleToGrInv	83.9	-0.13	33	-7.7e+03	-0.11	0.022	0.15	1.1e+03
GrSaleToGrOverhead	80.9	0.013	9.5	-1.5e+03	-0.067	0.00072	0.074	1.2e+03
Herf	91.0	-0.38	0.28	-1.4	-0.53	-0.31	-0.16	0
HerfAsset	88.0	-0.39	0.28	-1	-0.54	-0.32	-0.16	-0.022
HerfBE	88.0	-60	6.1e+03	-8.6e+05	-0.57	-0.33	-0.17	0
High52	100.0	0.83	0.19	0.0085	0.73	0.87	0.96	9.4
hire	98.1	-0.069	0.22	-2	-0.12	-0.032	0.015	2
IdioRisk	100.0	-0.021	0.014	-1.7	-0.025	-0.017	-0.012	0
IdioVol3F	100.0	-0.018	0.012	-1.4	-0.023	-0.015	-0.011	-9.1e-20
IdioVolAHT	100.0	-0.022	0.012	-0.53	-0.027	-0.02	-0.015	-0.0017
Illiquidity	96.1	1.2e-07	8.9e-07	1.2e-12	1.1e-09	7.4e-09	5.7e-08	0.00023
IndIPO	100.0	-0.043	0.2	-1	0	0	0	0
IndMom	100.0	0.087	0.16	-0.76	-0.0038	0.078	0.17	2.5
IndRetBig	26.5	0.019	0.066	-0.37	-0.018	0.017	0.052	0.63
IntanBM	74.5	-0.21	0.62	-8.2	-0.51	-0.15	0.15	3.4
IntanCFP	77.7	-0.027	0.29	-16	-0.084	0.0097	0.089	4.4
IntanEP	77.7	-0.025	0.33	-18	-0.083	0.038	0.12	12
IntanSP	77.7	-0.37	1.5	-10	-0.79	-0.026	0.5	7.7
IntMom	99.9	0.11	0.4	-0.98	-0.082	0.069	0.24	31
Investment	88.8	-1	0.48	-36	-1.2	-0.97	-0.77	42
InvestPPEInv	95.5	-0.11	0.24	-18	-0.14	-0.066	-0.02	2.6
InvGrowth	70.3	-0.18	3.1	-4.8e+02	-0.2	-0.05	0.058	1
IO_ShortInterest	0.7	74	43	0	39	82	1.1e+02	3.4e+02
iomom_cust	21.2	0.76	0.42	0	1	1	1	1
iomom_supp	22.6	0.78	0.41	0	1	1	1	1
Leverage	97.9	0.97	2	0	0.2	0.5	1.1	1.6e+02
LRreversal	93.2	-0.51	1.3	-1.3e+02	-0.71	-0.27	0.055	1
marketcap	100.0	4.4e+06	1.9e+07	1.7e+04	2.7e+05	7.9e+05	2.3e+06	8.8e+08
MaxRet	100.0	-0.052	0.046	-7.2	-0.063	-0.041	-0.027	0
MeanRankRevGrowth	74.6	2.2e+03	9.9e+02	34	1.5e+03	2.1e+03	2.9e+03	6.5e+03
Mom12m	99.9	0.22	0.67	-0.99	-0.082	0.13	0.37	50
Mom12mOffSeason	100.0	0.018	0.043	-0.33	-0.0045	0.015	0.036	1.3
Mom6m	100.0	0.095	0.36	-0.96	-0.077	0.059	0.21	36
Mom6mJunk	55.3	0.11	0.41	-0.96	-0.084	0.064	0.23	36
MomOffSeason	98.6	-0.018	0.021	-0.38	-0.026	-0.015	-0.0058	0.4
MomOffSeason06YrPlus	81.2	-0.016	0.02	-2.2	-0.024	-0.014	-0.0057	0.59
MomOffSeason11YrPlus	64.2	-0.016	0.019	-0.56	-0.024	-0.014	-0.0059	1.1
MomOffSeason16YrPlus	46.0	-0.015	0.015	-0.17	-0.022	-0.014	-0.0066	0.11
MomRev	5.4	0.44	0.5	0	0	0	1	1
MomSeason	98.7	0.017	0.076	-0.87	-0.021	0.013	0.05	3.8
MomSeason06YrPlus	81.3	0.016	0.07	-0.87	-0.018	0.012	0.045	2.5
MomSeason11YrPlus	64.3	0.016	0.068	-0.82	-0.017	0.012	0.044	2.7
MomSeason16YrPlus	50.8	0.015	0.064	-0.87	-0.015	0.012	0.041	2.4
MomSeasonShort	99.9	0.018	0.13	-0.88	-0.048	0.011	0.074	7.9
MomVol	63.9	6	2.6	1	4	6	8	10
MRreversal	99.9	-0.11	0.39	-31	-0.24	-0.068	0.083	0.97
MS	23.2	3.9	1.6	1	3	4	5	6
NetDebtFinance	86.9	-0.022	0.1	-1	-0.04	0	0.015	0.97
NetDebtPrice	42.5	-0.67	1.2	-67	-0.89	-0.4	-0.093	11
NetEquityFinance	96.0	-0.011	0.1	-1	-0.0081	-0.00015	0.0045	0.99
NetPayoutYield	85.0	0.017	0.082	-8.8	-0.0017	0.018	0.043	6.7
NOA	96.0	-0.68	1.2	-51	-0.8	-0.67	-0.52	2.7e+02
NumEarnIncrease	96.6	1.5	2.1	0	0	0	2	8
OperProf	78.9	0.31	5	-5.5e+02	0.19	0.29	0.39	8.6e+02
OperProfRD	83.0	0.18	0.19	-85	0.12	0.17	0.24	5.8
OPLeverage	98.1	1	0.82	0	0.47	0.84	1.3	85
OptionVolume1	42.2	-8.3e+02	1.6e+03	-1.1e+05	-9.7e+02	-4.1e+02	-1.4e+02	0
OptionVolume2	41.9	-1.2	4.1	-1.2e+03	-1.3	-0.91	-0.59	0
OrderBacklog	21.4	-0.62	0.93	-19	-0.66	-0.32	-0.15	-2.4e-05
OrderBacklogChg	19.8	-0.0038	0.33	-6.3	-0.057	-0.0033	0.045	8.2
OrgCap	51.9	-0.25	0.72	-2.2	-0.72	-0.41	0.042	8.3
OScore	56.6	-0.018	0.13	-1	0	0	0	0
PatentsRD	16.0	0.35	0.48	0	0	0	1	1
PayoutYield	72.5	0.088	0.2	2.8e-07	0.017	0.039	0.081	39
PctAcc	94.8	1.5	51	-4e+03	-0.64	0.28	1.2	7.9e+03
PctTotAcc	63.6	-1.6	93	-1.6e+04	-1.2	-0.74	-0.09	2.3e+03
prcadj	100.0	34	51	0.12	17	27	40	4.5e+03
PredictedFE	34.4	-0.051	0.029	-0.27	-0.067	-0.047	-0.031	0.054
PriceDelayRsq	98.9	0.19	0.21	0.00012	0.045	0.11	0.26	1
PriceDelaySlope	98.9	2.7	7.8e+02	-2.3e+03	-0.36	0.22	0.73	2.8e+05
PriceDelayTstat	98.5	1.5	1.3	-3.1	0.72	1.5	2.2	6.2
ProbInformedTrading	0.0	0.26	0.04	0.19	0.24	0.27	0.27	0.35
PS	5.3	5.3	1.5	0	4	5	6	9
RD	53.2	0.035	0.059	0	0.0055	0.019	0.044	6.9
RDAbility	6.7	0.56	5.2	-42	-0.61	0.4	1.9	64
RDcap	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
RDIPO	98.1	-0.0051	0.071	-1	0	0	0	0
RDS	76.6	-2.7e+02	4.1e+03	-5.4e+05	-40	-3.5	0.14	8.8e+04
realestate	48.0	0.013	0.15	-1.6	-0.084	-0.00043	0.089	1.4
ResidualMomentum	89.1	-0.026	0.32	-3.4	-0.22	-0.018	0.18	2.9
retadj	100.0	0.019	0.13	-0.88	-0.046	0.011	0.074	10
retConglomerate	28.4	0.012	0.081	-0.8	-0.03	0.012	0.051	1.4
ReturnSkew	100.0	-0.19	0.86	-4.5	-0.63	-0.17	0.26	4.5

ReturnSkew3F	100.0	-0.15	0.82	-4.4	-0.57	-0.13	0.28	4.4
REV6	18.6	-0.0098	0.059	-4.8	-0.013	-0.0014	0.0024	4.3
RevenueSurprise	90.2	-0.32	1.8e+02	-8.6e+04	-0.74	0.2	0.88	3.2e+03
RIO_Disp	24.4	3.3	1.1	1	2	3	4	5
RIO_MB	22.5	2.5	1.2	1	1	2	3	5
RIO_Turnover	22.9	2.9	1.2	1	2	3	4	5
RIO_Volatility	13.6	3	1.1	1	2	3	4	5
roaq	89.1	0.013	0.055	-4	0.0061	0.015	0.025	16
RoE	98.0	0.097	4.6	-7.9e+02	0.072	0.13	0.18	4.9e+02
sfe	16.8	0.071	0.13	-8.6	0.046	0.071	0.1	1.8
ShareIss1Y	99.9	-0.4	11	-5.7e+03	-0.073	-0.0083	0	1
ShareIss5Y	84.7	-4.8	61	-9.8e+03	-3.1	-0.38	-0.019	1
ShareRepurchase	98.1	0.44	0.5	0	0	0	1	1
ShareVol	23.8	-0.28	0.45	-1	-1	0	0	0
ShortInterest	64.0	-3.5e+04	5e+04	-1.2e+06	-4.6e+04	-1.7e+04	-4e+03	0
sinAlgo	5.9	0.24	0.43	0	0	0	0	1
skew1	29.2	-0.059	0.065	-1.6	-0.077	-0.047	-0.027	1.1
SmileSlope	42.4	-0.0061	0.078	-2.8	-0.021	-0.0032	0.011	2.2
SP	98.1	1.6	2.4	-0.2	0.44	0.92	1.8	1.5e+02
Spinoff	100.0	0.0056	0.075	0	0	0	0	1
std_turn	1.4	-0.052	0.089	-2.5	-0.054	-0.027	-0.013	-0.00014
SurpriseRD	50.8	0.29	0.45	0	0	0	1	1
tang	48.6	0.66	0.16	0.03	0.57	0.68	0.76	1.5
Tax	92.8	1.5	24	-2.2e+02	0.7	1.2	1.6	4.5e+03
TotalAccruals	95.7	-0.037	0.56	-22	-0.076	-0.026	0.014	1.2e+02
UpRecomm	25.0	0.38	0.48	0	0	0	1	1
VarCF	92.4	-0.068	3.2	-7e+02	-0.0042	-0.0013	-0.0004	-4.4e-07
VolMkt	96.3	-0.14	0.21	-17	-0.17	-0.077	-0.032	-5e-05
VolSD	94.4	-6.8	24	-1.2e+03	-4.8	-1.3	-0.26	-0.00036
VolumeTrend	91.4	-0.011	0.017	-0.065	-0.022	-0.011	-0.00017	0.056
XFIN	88.0	-0.016	0.15	-4.6	-0.039	0.0092	0.044	3.5
zerotrade	96.5	0.1	0.68	2.2e-10	1.3e-08	2.9e-08	7.6e-08	19
zerotradeAlt1	96.7	0.1	0.76	1.1e-10	1.2e-08	2.7e-08	6.9e-08	20
zerotradeAlt12	96.1	0.1	0.67	3.2e-10	1.3e-08	3.1e-08	8.4e-08	18

Table 7: Descriptive statistics of the characteristics This table reports for each characteristic the coverage (Cvrg(%)), the mean, the standard deviation (SD), the minimum (Min) and the maximum (max) as well as the 25%-, 50%- and 75%-quantiles.

	Hyperparameter
Ridge	$\lambda \in \{0.00001, 0.0001, 0.001, 0.01, 0.1\}$
Lasso	$\lambda \in \{0.00001, 0.0001, 0.001, 0.01, 0.1\}$
ENet	$\lambda \in \{0.00001, 0.0001, 0.001, 0.01, 0.1\}$, $\alpha = 0.5$
PCR	$q \in \{2, 4, 8, 12, 16, 24, 32, 40, 48\}$
RF	D = 6, B = 300, Max. features $\in \{3, 5, 10, 20, 30, 50\}$
XGBoost	D = 2, B = 1000, LR $\in \{0.01, 0.1\}$
NNs	ℓ_2-penalty $\lambda \in \{0.00001, 0.001\}$, Batch size = 1000, Max. iter. = 1000, Patience = 50

Table 8: Hyperparameters for all machine learning models

This table reports for each machine learning model the hyperparameters and their potential values used during hyperparameter tuning. The variables are defined as: λ = shrinkage hyperparameter for regularization, α = mixture hyperparameter for elastic net, q = number of principal components, D = max. tree depth, B = number of trees, LR = learning rate.

	OLS-3								OLS-all							
	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)
Low (L)	-0.67	-0.27	2.36	-2.42	25.08	0.15	-0.42	-3.77	-1.40	-0.63	2.68	-4.99	61.62	0.44	-1.07	-8.40
2	-0.34	-0.16	1.86	-1.87	50.33	0.30	-0.47	-5.30	-0.71	-0.25	2.11	-2.46	82.08	0.55	-0.79	-7.95
3	-0.13	-0.09	1.70	-1.15	57.78	0.36	-0.45	-5.67	-0.37	-0.06	1.99	-0.68	85.46	0.57	-0.63	-6.72
4	0.05	0.11	1.77	1.32	60.20	0.40	-0.29	-3.44	-0.10	0.03	1.89	0.32	86.89	0.58	-0.55	-6.21
5	0.22	0.02	2.00	0.20	61.84	0.43	-0.41	-4.38	0.14	0.26	1.91	2.88	87.54	0.59	-0.33	-3.69
6	0.39	0.12	2.19	1.21	61.68	0.45	-0.33	-3.20	0.37	0.36	2.07	3.68	87.49	0.61	-0.25	-2.55
7	0.57	0.35	2.56	2.88	60.24	0.47	-0.13	-1.06	0.62	0.50	2.11	5.08	86.51	0.62	-0.12	-1.17
8	0.78	0.56	2.76	4.27	56.97	0.47	0.08	0.63	0.91	0.64	2.31	5.89	85.08	0.63	0.01	0.07
9	1.03	0.88	2.94	6.34	49.62	0.43	0.45	3.23	1.28	0.95	2.66	7.62	80.70	0.63	0.32	2.56
High (H)	1.47	1.62	3.36	10.22	27.35	0.26	1.36	8.64	2.07	1.70	3.37	10.67	57.31	0.50	1.20	7.61
H-L	2.14	1.88	4.21	9.50	52.43	0.41	1.48	7.49	3.47	2.33	3.42	14.42	118.94	0.94	1.39	8.84

	Ridge								ENet							
	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)
Low (L)	-1.40	-0.63	2.68	-4.99	61.62	0.44	-1.07	-8.40	-0.94	-0.58	2.43	-5.08	58.44	0.40	-0.98	-8.46
2	-0.71	-0.24	2.11	-2.46	82.07	0.55	-0.79	-7.95	-0.47	-0.23	1.88	-2.61	78.16	0.51	-0.74	-8.24
3	-0.37	-0.06	1.99	-0.67	85.46	0.57	-0.63	-6.72	-0.23	-0.04	1.83	-0.52	81.30	0.53	-0.58	-6.63
4	-0.10	0.03	1.89	0.31	86.89	0.58	-0.55	-6.22	-0.02	-0.01	1.81	-0.08	82.49	0.55	-0.55	-6.50
5	0.14	0.26	1.91	2.88	87.54	0.59	-0.33	-3.70	0.16	0.21	1.97	2.23	83.07	0.57	-0.36	-3.88
6	0.37	0.36	2.07	3.69	87.50	0.61	-0.25	-2.54	0.35	0.27	2.14	2.70	83.11	0.59	-0.31	-3.14
7	0.62	0.50	2.11	5.07	86.51	0.62	-0.12	-1.17	0.55	0.40	2.35	3.59	82.53	0.61	-0.21	-1.93
8	0.91	0.64	2.31	5.89	85.08	0.63	0.01	0.07	0.78	0.58	2.56	4.82	80.63	0.62	-0.04	-0.35
9	1.28	0.95	2.66	7.62	80.70	0.63	0.32	2.57	1.07	0.86	2.78	6.56	76.12	0.63	0.23	1.79
High (H)	2.07	1.69	3.37	10.67	57.31	0.50	1.20	7.61	1.64	1.82	3.36	11.48	53.79	0.48	1.33	8.50
H-L	3.47	2.33	3.42	14.42	118.94	0.94	1.39	8.84	2.59	2.40	3.68	13.83	112.23	0.89	1.51	8.93

	Lasso								PCR							
	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)
Low (L)	-0.87	-0.61	2.27	-5.68	49.68	0.36	-0.97	-8.86	-0.84	-0.50	2.20	-4.84	49.89	0.34	-0.84	-7.95
2	-0.42	-0.30	1.87	-3.37	64.35	0.45	-0.74	-8.28	-0.40	-0.17	1.90	-1.85	69.43	0.45	-0.62	-6.80
3	-0.18	-0.09	1.88	-1.04	67.67	0.47	-0.56	-6.28	-0.17	-0.02	1.90	-0.24	74.92	0.49	-0.51	-5.69
4	0.01	-0.03	1.83	-0.32	69.76	0.49	-0.52	-5.91	0.02	0.01	1.85	0.17	76.98	0.52	-0.50	-5.78
5	0.19	0.21	1.91	2.33	70.94	0.51	-0.30	-3.28	0.20	0.23	1.97	2.48	77.70	0.54	-0.31	-3.36
6	0.36	0.31	2.12	3.05	71.50	0.53	-0.22	-2.22	0.37	0.46	2.18	4.45	77.78	0.56	-0.10	-1.03
7	0.55	0.46	2.27	4.28	71.58	0.54	-0.09	-0.81	0.55	0.43	2.27	4.06	76.52	0.57	-0.14	-1.29
8	0.76	0.54	2.56	4.44	70.26	0.56	-0.02	-0.20	0.76	0.66	2.65	5.32	73.69	0.58	0.09	0.70
9	1.04	0.84	2.84	6.29	66.77	0.56	0.28	2.10	1.03	0.95	2.82	7.16	68.32	0.57	0.38	2.91
High (H)	1.58	1.88	3.50	11.37	49.02	0.45	1.42	8.70	1.58	1.48	3.62	8.70	46.78	0.42	1.06	6.26
H-L	2.46	2.48	3.80	13.88	98.69	0.82	1.67	9.51	2.42	1.98	4.00	10.52	96.67	0.76	1.22	6.60

	RF								XGBoost							
	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)
Low (L)	-0.40	-0.69	3.05	-4.84	63.51	0.50	-1.20	-8.26	-0.82	-0.81	3.02	-5.70	65.62	0.51	-1.32	-9.20
2	-0.13	-0.14	2.04	-1.47	82.09	0.56	-0.71	-7.32	-0.37	-0.23	2.14	-2.23	83.86	0.57	-0.79	-7.83
3	-0.02	-0.00	2.02	-0.05	85.34	0.56	-0.57	-6.01	-0.18	-0.12	1.92	-1.30	86.84	0.57	-0.69	-7.59
4	0.08	0.10	2.01	1.09	85.94	0.56	-0.46	-4.87	-0.03	0.04	1.93	0.44	87.94	0.58	-0.53	-5.87
5	0.17	0.23	1.99	2.40	85.75	0.56	-0.33	-3.57	0.10	0.26	2.00	2.72	88.17	0.58	-0.32	-3.40
6	0.27	0.25	1.98	2.67	85.43	0.55	-0.30	-3.27	0.24	0.28	1.98	2.97	88.34	0.59	-0.31	-3.34
7	0.37	0.29	2.12	2.93	84.71	0.56	-0.27	-2.69	0.39	0.33	2.06	3.36	87.48	0.60	-0.27	-2.79
8	0.49	0.42	2.29	3.90	82.67	0.58	-0.15	-1.44	0.57	0.47	2.13	4.73	86.23	0.61	-0.14	-1.37
9	0.66	0.56	2.64	4.50	75.73	0.58	-0.02	-0.18	0.83	0.61	2.47	5.21	82.18	0.62	-0.01	-0.13
High (H)	2.04	2.27	3.55	13.54	50.09	0.46	1.80	10.84	2.33	2.41	3.51	14.54	53.88	0.49	1.92	11.70
H-L	2.43	2.96	3.20	19.64	113.61	0.96	2.00	13.60	3.15	3.22	3.18	21.49	119.50	1.00	2.22	15.36

	NN1								NN2							
	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)
Low (L)	-2.00	-0.88	3.09	-6.07	57.97	0.46	-1.34	-9.15	-1.71	-0.87	3.11	-5.95	58.72	0.46	-1.33	-9.04
2	-0.92	-0.28	2.09	-2.88	81.71	0.56	-0.85	-8.58	-0.76	-0.30	1.95	-3.22	82.34	0.56	-0.86	-9.23
3	-0.52	-0.15	1.83	-1.76	85.79	0.57	-0.72	-8.34	-0.41	-0.07	1.90	-0.79	85.70	0.56	-0.63	-7.03
4	-0.22	0.00	1.83	0.01	87.13	0.57	-0.57	-6.54	-0.15	0.03	1.82	0.38	87.39	0.56	-0.53	-6.21
5	0.06	0.10	1.91	1.07	87.45	0.57	-0.47	-5.29	0.09	0.14	1.79	1.64	87.98	0.57	-0.43	-5.17
6	0.32	0.35	1.89	3.97	87.46	0.58	-0.22	-2.51	0.32	0.29	1.94	3.17	87.78	0.58	-0.29	-3.17
7	0.61	0.50	2.03	5.26	87.01	0.59	-0.08	-0.89	0.58	0.47	1.96	5.10	87.45	0.59	-0.12	-1.31
8	0.94	0.55	2.13	5.49	85.30	0.60	-0.05	-0.51	0.87	0.63	2.16	6.19	85.82	0.61	0.02	0.24
9	1.40	0.81	2.43	7.08	81.46	0.61	0.20	1.75	1.28	0.88	2.56	7.30	81.76	0.62	0.26	2.17
High (H)	2.98	2.35	3.65	13.71	56.13	0.50	1.85	10.89	2.69	2.17	3.68	12.53	56.04	0.50	1.67	9.71
H-L	4.98	3.24	2.90	23.67	114.10	0.96	2.28	17.32	4.40	3.04	2.91	22.23	114.76	0.96	2.08	15.73

	NN3								NN4							
	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)
Low (L)	-1.64	-0.88	3.27	-5.70	58.51	0.46	-1.34	-8.62	-1.50	-0.92	3.19	-6.14	58.70	0.46	-1.38	-9.13
2	-0.73	-0.23	2.02	-2.40	82.44	0.56	-0.79	-8.29	-0.66	-0.30	1.99	-3.16	82.50	0.56	-0.86	-9.11
3	-0.40	-0.14	1.81	-1.68	85.77	0.56	-0.70	-8.23	-0.35	-0.04	1.88	-0.51	85.83	0.56	-0.60	-6.81
4	-0.14	0.01	1.75	0.07	86.96	0.56	-0.55	-6.69	-0.11	-0.07	1.85	-0.78	87.18	0.56	-0.63	-7.18
5	0.09	0.16	1.90	1.77	87.83	0.57	-0.41	-4.59	0.10	0.17	1.75	2.07	87.75	0.57	-0.40	-4.83
6	0.31	0.34	1.86	3.93	87.64	0.57	-0.23	-2.63	0.31	0.29	1.85	3.31	87.65	0.58	-0.29	-3.29
7	0.56	0.47	2.03	4.91	87.17	0.59	-0.12	-1.29	0.54	0.42	1.98	4.47	87.12	0.59	-0.17	-1.85
8	0.85	0.57	2.17	5.57	85.62	0.61	-0.04	-0.38	0.81	0.58	2.21	5.56	85.50	0.61	-0.03	-0.26
9	1.26	0.87	2.53	7.36	81.56	0.62	0.25	2.16	1.20	0.88	2.44	7.64	81.19	0.62	0.26	2.27
High (H)	2.62	2.21	3.63	12.89	55.96	0.50	1.70	10.03	2.49	2.25	3.79	12.62	55.32	0.50	1.75	9.91
H-L	4.26	3.08	2.94	22.26	114.47	0.96	2.12	15.83	3.99	3.17	3.05	22.04	114.02	0.95	2.22	15.95

	NN5								NN1-5							
	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)
Low (L)	-0.96	-0.89	3.18	-5.92	58.36	0.46	-1.34	-8.93	-1.51	-0.89	3.13	-6.01	58.55	0.46	-1.35	-9.07
2	-0.33	-0.28	1.99	-3.00	82.22	0.56	-0.85	-8.93	-0.66	-0.30	2.03	-3.14	82.35	0.57	-0.87	-9.02
3	-0.11	-0.06	1.85	-0.68	85.75	0.56	-0.62	-7.12	-0.34	-0.07	1.90	-0.77	85.80	0.56	-0.63	-7.05
4	0.07	0.08	1.81	0.95	87.20	0.56	-0.48	-5.66	-0.10	0.01	1.85	0.11	87.18	0.56	-0.55	-6.33
5	0.22	0.19	1.79	2.22	87.42	0.56	-0.38	-4.49	0.11	0.17	1.89	1.97	87.72	0.57	-0.39	-4.43
6	0.37	0.25	1.91	2.80	87.56	0.58	-0.32	-3.60	0.32	0.31	1.91	3.47	87.81	0.58	-0.27	-3.00
7	0.53	0.51	2.01	5.42	86.71	0.58	-0.07	-0.75	0.55	0.41	2.04	4.24	86.95	0.59	-0.18	-1.89
8	0.72	0.59	2.21	5.67	85.46	0.60	-0.01	-0.13	0.82	0.55	2.13	5.53	85.66	0.61	-0.05	-0.52
9	0.99	0.79	2.46	6.83	81.40	0.62	0.17	1.48	1.19	0.88	2.44	7.63	81.70	0.62	0.25	2.24
High (H)	1.89	2.20	3.65	12.78	55.37	0.50	1.70	9.98	2.45	2.25	3.69	12.96	56.27	0.50	1.75	10.16
H-L	2.86	3.08	2.89	22.69	113.73	0.95	2.13	16.21	3.95	3.14	3.01	22.14	114.82	0.96	2.18	15.89

Table 9: Performance of machine learning decile portfolios (equal-weighted)

This table reports the equal-weighted performance of decile portfolios sorted on the 1-month-ahead out-of-sample predicted abnormal return using NYSE breakpoints. Note that for the long-short portfolios the returns are shown whereas for the decile portfolios the abnormal returns are reported. For the column names, see the description of Table 2. For the abbreviations of the models, see the description of Figure 7.

	Pred(g)	Avg(g)	SD(g)	t(g)	TO	TC	Avg(n)	t(n)
OLS-3	2.14	1.88	4.21	9.50	52.43	0.41	1.48	7.49
OLS-all	3.47	2.33	3.42	14.42	118.94	0.94	1.39	8.84
Ridge	3.47	2.33	3.42	14.42	118.94	0.94	1.39	8.84
ENet	2.59	2.40	3.68	13.83	112.23	0.89	1.51	8.93
Lasso	2.46	2.48	3.80	13.88	98.69	0.82	1.67	9.51
PCR	2.42	1.98	4.00	10.52	96.67	0.76	1.22	6.60
RF	2.43	2.96	3.20	19.64	113.61	0.96	2.00	13.60
XGBoost	3.15	3.22	3.18	21.49	119.50	1.00	2.22	15.36
NN1	4.98	3.24	2.90	23.67	114.10	0.96	2.28	17.32
NN2	4.40	3.04	2.91	22.23	114.76	0.96	2.08	15.73
NN3	4.26	3.08	2.94	22.26	114.47	0.96	2.12	15.83
NN4	3.99	3.17	3.05	22.04	114.02	0.95	2.22	15.95
NN5	2.86	3.08	2.89	22.69	113.73	0.95	2.13	16.21
NN1-5	3.95	3.14	3.01	22.14	114.82	0.96	2.18	15.89

Table 10: Performance of machine learning portfolios (equal-weighted)
This table reports the equal-weighted performance of long-short portfolios sorted on the 1-month-ahead out-of-sample predicted abnormal return using NYSE breakpoints (long = top decile, short = bottom decile). Columns Pred, Avg, SD, t, TO, TC refer to the avg. predicted return, avg. realized return, their standard deviation, their t-statistic, the turnover and the transaction costs. g (n) indicates gross (net) returns. For the abbreviations of the models, see the description of Figure 7.

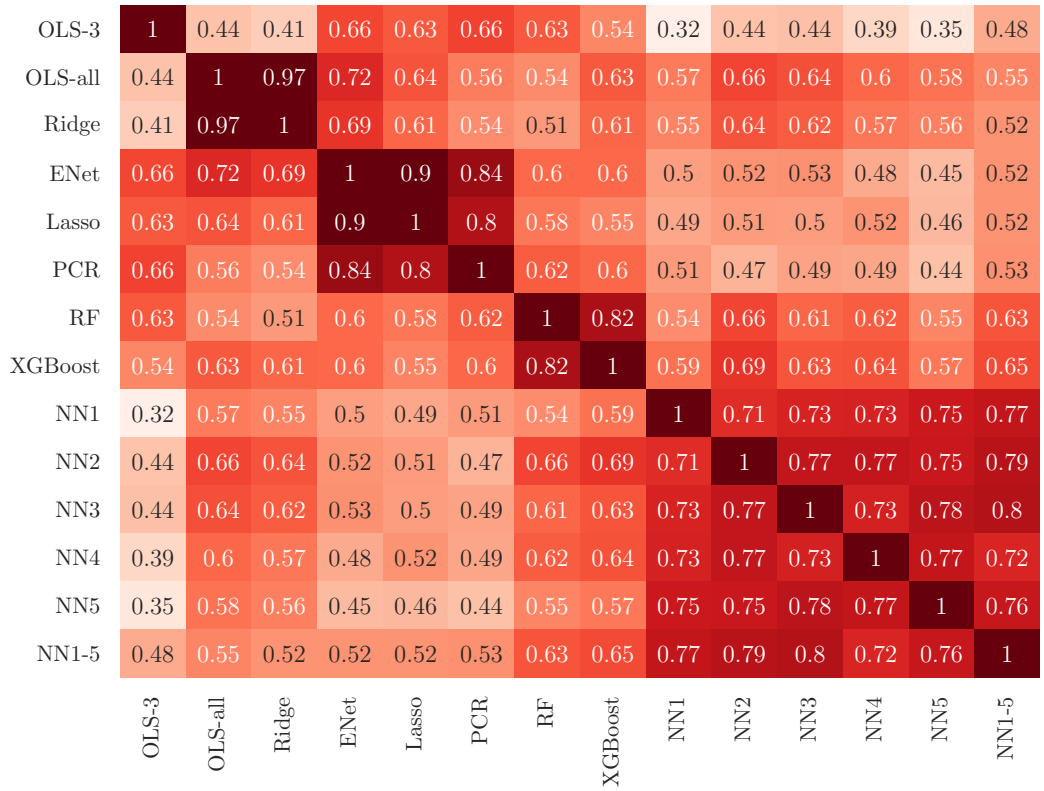

Figure 18: Correlation of net returns of machine learning portfolios
This figure depicts the correlations between the value-weighted net returns of long-short portfolios sorted on the 1-month-ahead out-of-sample predicted abnormal return using NYSE breakpoints (long = top decile, short = bottom decile). The numbers report the coefficients of correlation. As the coefficient of correlation increases from 0 to +1, the colour of the corresponding cell changes gradually from white to red. For the abbreviations of the models, please refer to the description of Figure 7.

	Intercept	mktrf	smb	hml	rmw	cma	umd	R^2 (%)
OLS-3	0.11	0.22	1.17	0.50	-0.12	-0.04	0.47	85.42
	(1.3)	(10.36)	(36.6)	(12.17)	(-2.94)	(-0.74)	(24.15)	
OLS-all	-0.28	0.26	0.32	0.08	0.17	-0.05	0.37	24.48
	(-1.5)	(5.61)	(4.68)	(0.91)	(1.92)	(-0.42)	(8.9)	
Ridge	-0.28	0.26	0.32	0.08	0.17	-0.05	0.37	24.48
	(-1.51)	(5.61)	(4.68)	(0.91)	(1.92)	(-0.42)	(8.9)	
ENet	0.02	0.28	0.76	0.38	0.06	-0.30	0.21	38.88
	(0.13)	(6.68)	(12.01)	(4.68)	(0.73)	(-2.52)	(5.37)	
Lasso	0.25	0.32	0.75	0.42	0.08	-0.26	0.14	36.48
	(1.41)	(7.24)	(11.42)	(4.9)	(0.94)	(-2.06)	(3.52)	
PCR	-0.19	0.37	0.79	0.42	-0.10	-0.29	0.16	47.56
	(-1.12)	(9.04)	(12.91)	(5.39)	(-1.26)	(-2.51)	(4.19)	
RF	0.43	0.17	0.56	0.25	-0.38	-0.53	0.60	51.70
	(2.42)	(3.91)	(8.52)	(2.92)	(-4.59)	(-4.34)	(15.1)	
XGBoost	0.27	0.21	0.35	0.12	-0.33	-0.35	0.51	40.19
	(1.45)	(4.68)	(5.15)	(1.34)	(-3.76)	(-2.74)	(12.17)	
NN1	0.77	0.16	0.13	0.04	-0.05	-0.36	0.14	13.20
	(4.23)	(3.5)	(1.97)	(0.53)	(-0.57)	(-2.89)	(3.53)	
NN2	0.47	0.10	0.28	0.02	-0.09	-0.18	0.34	23.01
	(2.62)	(2.24)	(4.3)	(0.26)	(-1.06)	(-1.44)	(8.43)	
NN3	0.64	0.14	0.29	0.01	-0.02	-0.19	0.31	19.68
	(3.29)	(2.86)	(4.12)	(0.14)	(-0.27)	(-1.44)	(7.15)	
NN4	0.59	0.18	0.27	0.07	0.03	-0.24	0.24	17.78
	(3.24)	(4.08)	(4.06)	(0.82)	(0.32)	(-1.94)	(5.88)	
NN5	0.77	0.16	0.22	-0.01	0.03	-0.14	0.22	13.69
	(4.04)	(3.38)	(3.11)	(-0.13)	(0.37)	(-1.11)	(5.22)	
NN1-5	0.70	0.12	0.34	0.02	-0.19	-0.13	0.24	22.23
	(3.92)	(2.82)	(5.14)	(0.22)	(-2.26)	(-1.05)	(6.02)	

Table 11: Regression of net machine learning portfolio returns against the FF5FM+Mom

This table summarizes the regressions of net machine learning portfolio returns against the returns of the factors of the Fama-French five-factor model (Fama & French, 2015) augmented by the momentum factor (Carhart, 1997). mktrf, smb, hml, rmw, cma and umd represent the estimated regression coefficients of the six factor. α denotes the intercept. t-statistics are indicated in parentheses below. For the abbreviations of the models, see the description of Figure 7.

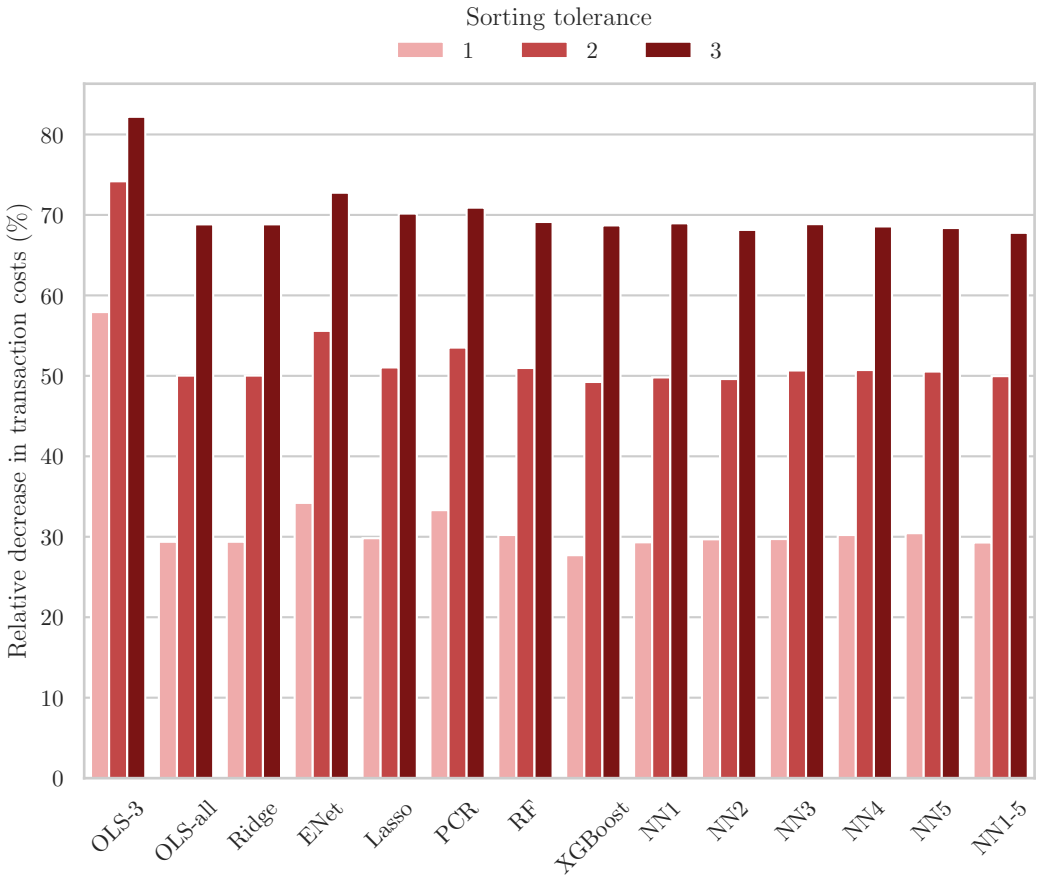

Figure 19: Effect of turnover mitigation sorting on the trading costs of machine learning portfolios

This table reports the effect turnover mitigation sorting has on the turnover of long-short portfolios sorted on the 1-month-ahead out-of-sample predicted abnormal return using NYSE breakpoints (long = top decile, short = bottom decile) as measured by the relative decrease in trading costs (%). For details on turnover mitigation sorting, see the second part of Section 4.5. For the abbreviations of the models, see the description of Figure 7.

References

Allen, M. P. (1997). The problem of multicollinearity. *Understanding regression analysis*, 176–180.

Banz, R. W. (1981). The relationship between return and market value of common stocks. *Journal of Financial Economics*, *9*(1), 3–18.

Basu, S. (1977). Investment performance of common stocks in relation to their price-earnings ratios: A test of the efficient market hypothesis. *The Journal of Finance*, *32*(3), 663–682.

Beaver, W., McNichols, M., & Price, R. (2007). Delisting returns and their effect on accounting-based market anomalies. *Journal of Accounting and Economics*, *43*(2-3), 341–368.

Bishop, C. M., & Nasrabadi, N. M. (2006). *Pattern recognition and machine learning* (Vol. 4) (No. 4). Springer.

Breiman, L. (2001). Random forests. *Machine learning*, *45*(1), 5–32.

Cakici, N., & Zaremba, A. (2022). Empirical asset pricing via machine learning: The global edition. *Available at SSRN 4028525*.

Carhart, M. M. (1997). On persistence in mutual fund performance. *The Journal of Finance*, *52*(1), 57–82.

Chen, A. Y., & Velikov, M. (2017). Accounting for the anomaly zoo: A trading cost perspective. *Available at SSRN*, *3073681*.

Chen, A. Y., & Zimmermann, T. (in press). Open source cross sectional asset pricing. *Critical Finance Review*.

Chen, L., Pelger, M., & Zhu, J. (2019). Deep learning in asset pricing. *Available at SSRN 3350138*.

DeMiguel, V., Martin-Utrera, A., Nogales, F. J., & Uppal, R. (2020). A transaction-cost perspective on the multitude of firm characteristics. *The Review of Financial Studies*, *33*(5), 2180–2222.

Drobetz, W., & Otto, T. (2021). Empirical asset pricing via machine learning: evidence from the european stock market. *Journal of Asset Management*, *22*(7), 507–538.

Fahrmeir, L., Kneib, T., Lang, S., & Marx, B. (2013). *Regression: Models, methods and applications*. Springer.

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. (2008). Dissecting anomalies. *The Journal of Finance*, *63*(4), 1653–1678.

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

Feng, G., Giglio, S., & Xiu, D. (2020). Taming the factor zoo: A test of new factors. *The Journal of Finance*, *75*(3), 1327–1370.

Feng, G., He, J., & Polson, N. G. (2018). Deep learning for predicting asset

returns. *arXiv preprint arXiv:1804.09314*.

Frazzini, A., Israel, R., & Moskowitz, T. J. (2012). Trading costs of asset pricing anomalies. *Fama-Miller Working Paper, Chicago Booth Research Paper*(14-05).

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

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

Green, J., Hand, J. R., & Zhang, X. F. (2017). The characteristics that provide independent information about average us monthly stock returns. *The Review of Financial Studies*, *30*(12), 4389–4436.

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

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

Hanauer, M. X., Kononova, M., & Rapp, M. S. (2022). Boosting agnostic fundamental analysis: Using machine learning to identify mispricing in european stock markets. *Finance Research Letters*, 102856.

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

Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction* (Vol. 2). Springer.

Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, *12*(1), 55–67.

Hou, K., Xue, C., & Zhang, L. (2020). Replicating anomalies. *The Review of Financial Studies*, *33*(5), 2019–2133.

Izenman, A. J. (2008). Modern multivariate statistical techniques. *Regression, classification and manifold learning*, *10*, 978–0.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning* (Vol. 112). Springer.

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

Kelly, B. T., Pruitt, S., & Su, Y. (2020). Instrumented principal component analysis. *Available at SSRN 2983919*.

Kozak, S., Nagel, S., & Santosh, S. (2020). Shrinking the cross-section. *Journal of Financial Economics*, *135*(2), 271–292.

Kuhn, M., & Johnson, K. (2013). *Applied predictive modeling* (Vol. 26). Springer.

Leshno, M., Lin, V. Y., Pinkus, A., & Schocken, S. (1993). Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural networks*, *6*(6), 861–867.

Lewellen, J. (2015). The cross-section of expected stock returns. *Critical Finance Review*, *4*(1), 1–44.

Lintner, J. (1965). Security prices, risk, and maximal gains from diversification. *The Journal of Finance*, *20*(4), 587–615.

Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. In I. Guyon et al. (Eds.), *Advances in neural information processing systems 30* (pp. 4765–4774). Curran Associates, Inc.

Messmer, M. (2017). Deep learning and the cross-section of expected returns. *Available at SSRN 3081555*.

Michelucci, U. (2018). Applied deep learning. *A Case-Based Approach to Understanding Deep Neural Networks*.

Moritz, B., & Zimmermann, T. (2016). Tree-based conditional portfolio sorts: The relation between past and future stock returns. *Available at SSRN 2740751*.

Mossin, J. (1966). Equilibrium in a capital asset market. *Econometrica: Journal of the Econometric Society*, 768–783.

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

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

Rasekhschaffe, K. C., & Jones, R. C. (2019). Machine learning for stock selection. *Financial Analysts Journal*, *75*(3), 70–88.

Rencher, A. C., & Schaalje, G. B. (2008). *Linear models in statistics*. Wiley-Interscience.

Rosenberg, B., Reid, K., & Lanstein, R. (1985). Persuasive evidence of market inefficiency. *The Journal of Portfolio Management*, *11*(3), 9–16.

Shapley, L. S. (1953). Quota solutions op n-person games1. *Edited by Emil Artin and Marston Morse*, 343.

Sharpe, W. F. (1964). Capital asset prices: A theory of market equilibrium under conditions of risk. *The Journal of Finance*, *19*(3), 425–442.

Shumway, T. (1997). The delisting bias in crsp data. *The Journal of Finance*, *52*(1), 327–340.

Štrumbelj, E., & Kononenko, I. (2014). Explaining prediction models and individual predictions with feature contributions. *Knowledge and Information Systems*, *41*(3), 647–665.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, *58*(1), 267–288.

Weigand, A. (2019). Machine learning in empirical asset pricing. *Financial Markets and Portfolio Management*, *33*(1), 93–104.

Wharton Research Database (WRDS). (2021). *Center for research in security prices (crsp).* `https://wrds-www.wharton.upenn.edu/pages/about/`

data-vendors/center-for-research-in-security-prices-crsp/. (Accessed: 2021-12-20)

Windmüller, S. (2021). Firm characteristics and global stock returns: A conditional asset pricing model. *Review of Asset Pricing Studies*.

Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, *67*(2), 301–320.