WORKING PAPER
N. 233
OCTOBER 2024

Machine Learning in Portfolio Decisions

Massimo Guidolin

Giulia Panzeri

Manuela Pedio

Università Bocconi
BAFFI
Centre on Economics, Finance and Regulation

Via Röntgen 1 | 20136 Milano – Italia | Tel +39 02 5836.5908/5564
baffi@unibocconi.it | www.baffi.unibocconi.eu

Machine Learning in Portfolio Decisions *

Massimo Guidolin † Giulia Panzeri ‡ Manuela Pedio §

This draft: October 13, 2024

Abstract

Machine learning is significantly shaping the advancement of various fields, and among them, notably, finance, where its range of applications and efficiency impacts are seemingly boundless. Contemporary techniques, particularly in reinforcement learning, have prompted both practitioners and academics to contemplate the potential of an artificial intelligence revolution in portfolio management. In this paper, we provide an overview of the primary methods in machine learning currently utilized in portfolio decision-making. We delve into discussions surrounding the existing limitations of machine learning algorithms and explore prevailing hypotheses regarding their future expansions. Specifically, we categorize and analyze the applications of machine learning in systematic trading strategies, portfolio weight optimization, smart beta and passive investment strategies, textual analysis, and trade execution, each separately surveyed for a comprehensive understanding.

Keywords: Machine learning; portfolio choice; artificial intelligence; neural language processing; stock return predictions, market timing, mean-variance asset allocation.

JEL Classification: C45, C61, G10, G11, G17.

1 Introduction

Machine learning (henceforth, ML) is a field of artificial intelligence that uses statistical techniques to give computer systems the ability to learn (i.e., progressively improve performance on a specific task) with data without being explicitly programmed. As explained by Carbonell et al. (1983), ML facilitates the development of "intelligent" computer programs capable of pattern recognition, decision-making, and innovative interaction with the external world. Given its potency, ML holds

*We would like to thank Max Goebel for his help on various parts of the paper. All errors remain ours only.

†Department of Finance, Bocconi University and Baffi CAREFIN, Milan, IT. E-mail: massimo.guidolin@unibocconi.it

‡Department of Finance, Bocconi University and Baffi CAREFIN, Milan, IT. E-mail: giulia.panzeri@unibocconi.it

§University of Bristol, UK and Baffi CAREFIN Centre (Bocconi University), Milan. Email: manuela.pedio@bristol.ac.uk

the promise of revolutionizing various sectors, including finance. This paper will specifically delve into ML applications in financial decision-making, defined as the methodologies employed to guide optimal decisions by individual investors or portfolio managers with an interest in asset trading and allocation within diversified investment portfolios.

In the realm of finance, the integration of ML has triggered a recent surge in the utilization of high-dimensional statistical models. These models incorporate optimization algorithms, dimensionality-reduction techniques, and regularization methods, all aimed at selecting an optimal model specification and mitigating overfitting in forecasting (refer to Gu et al. (2020) for empirical evidence). Asset managers are increasingly embracing these ML models due to their efficacy in making precise predictions and uncovering patterns within financial data. In fact, with the anticipated advent of quantum computing, poised to significantly enhance computational power, there exists a potential for the widespread adoption of computationally intensive ML models (see Egger et al. (2020)).[1]

In practical terms, ML encompasses a continuously expanding array of algorithms that empower machines to discern patterns in data, often without the need for *explicit*, problem-specific programming instructions, or at the very least, without a stringent requirement for a direct correlation between labeled outcomes and inputs. In finance, ML finds widespread application in analyzing data concerning various drivers of asset returns to generate predictions. These drivers may encompass macroeconomic indicators, industry trends, and company-specific data. Through the utilization of ML algorithms, analysts can discern patterns and correlations among these factors at a given point in time and subsequent returns. Consequently, composite investment signals can be formulated, often outperforming individual sources of information. A key advantage of ML algorithms lies in their power to uncover intricate patterns and relationships, including nonlinear and contemporaneous ones, which may prove challenging or even impossible to identify using traditional, often linear, algorithms typical of *standard* econometrics, such as simple regression analysis. The financial sector, inherently data-rich, is well-positioned to benefit from ML due to its extensive records of investment assets and customer interactions; in finance, ML has demonstrated successful applications in areas such as fraud detection, risk assessment, and portfolio management, among others (see the discussion in Lee et al. (2023)).

Although the applications of ML to portfolio decisions are not limited to this purpose, ML approaches can easily capture non-linear patterns (see Schmidhuber (2015)), including interactions between input variables. This property may improve the construction of single- and multi-factor signals by capturing higher-order relationships and complex information in the inputs. ML techniques can also capture regimes and structural instability in the relations among financial and economic variables, making them particularly useful in risk management (see, e.g., Uysal & Mulvey (2021)). Meanwhile, the tools from the natural language processing (NLP) branch of ML have enabled the creation of predictors and asset pricing factors based on textual information, such as, but not limited to, corporate

[1]Quantum computing is an innovative technology leveraging quantum mechanical phenomena to perform complex calculations. It uses quantum bits (also known as *qubits*), which can exist simultaneously in multiple states, known as superposition, to process and store information. In contrast to classical computing, this distinctive trait enables quantum computers to execute specific types of calculations significantly faster and more efficiently. This is particularly evident in applications requiring high computational power, such as large-scale optimizations, cryptography, and the design of extensive simulations.

reports, news articles, social media posts, and conference call transcripts (see Pedio (2024)).

The integration of ML into finance, and particularly into asset management practices, did not occur abruptly. For instance, Richard Dennis and William Eckhardt pioneered rule-based trading and investing already in the 1980s.[2] In the 1990s, statistical arbitrage emerged as a popular trading strategy, focusing on capitalizing on market inefficiencies by exploiting price discrepancies among related financial instruments. This strategy entailed the utilization of sophisticated quantitative techniques, including principal component analysis and cointegration, which arguably belong to the domain of classical econometrics. These methods were deployed to identify and leverage profit opportunities with minimal investment and risk. By the early 2000s, high-frequency trading (HFT) started to emerge as a predominant trading strategy. HFT entails employing robust computer algorithms to execute trades swiftly and in large volumes. These strategies heavily depend on advanced technology, including low-latency trading systems and co-location services, to gain a competitive advantage in the markets. Within this landscape, ML has been introduced to augment prediction accuracy through supervised learning, enhance decision-making quality via reinforcement learning, and refine problem-discovery skills using unsupervised learning techniques.

Recently, there has been significant interest among finance practitioners and academics in what are known as "deep learning " algorithms. Deep learning refers to a collection of algorithms that utilize artificial neural networks with multiple layers to model and analyze complex, high-dimensional data. These algorithms have the ability to extract features from raw data without the need for manual feature engineering (e.g., the selection of relevant predictors, see, e.g., Heaton et al. (2016)), achieving accuracy levels comparable to that of the human brain in certain fields, such as image classification.[3] While a milestone in the history of ML was the victory of the Deep Blue algorithm over Garry Kasparov, the reigning human chess champion, in 1997, the real shift took place with AlphaZero, a deep learning neural network that clinched the world chess championship in 2017 by teaching itself through reinforcement learning and pattern recognition techniques in just four hours of self-play.[4] This event has sparked excitement among traders and portfolio managers, who perceive ML as offering nearly limitless possibilities in the realm of asset management applications.

Ultimately, while one must consider the potential for hindsight bias, asset allocation stands out as an ideal sub-field within finance where ML is poised to make significant strides. This assertion is reinforced by the limitations inherent in classical portfolio optimization methods, exemplified by

[2]They introduced the Turtle Trader program, designed to impart trading skills rather than relying solely on innate abilities. This initiative involved training a group of traders known as "the Turtles," who implemented a trend-following strategy and achieved substantial operational success (see Covel (2007))

[3]However Masini et al. (2023) assess the classical analogy between ANNs and the functioning of the human brain as misguided: "Contrary to what has been boasted in the early literature, the empirical success of NN models comes from a mathematical fact that a linear combination of sufficiently many simple basis functions is able to approximate very complicated functions arbitrarily well in some specific choice of metric." (p. 77), which they connect to the power of, for instance, sieve estimation techniques.

[4]Deep Blue relied on brute force computational speed to evaluate moves. AlphaZero utilizes a Monte Carlo tree search technique in conjunction with deep neural networks to assess board positions and determine optimal moves. Its victory over the reigning world chess-specific champion algorithm, Stockfish, in December 2017 garnered significant attention and emphasized the potential of deep learning to develop autonomous intelligent systems (see McGrath et al. (2022)). Stockfish is a free and open-source chess engine that functions as a hybrid ML computer program capable of playing chess at a high level. AlphaZero's success has spurred further exploration into the use of deep learning across various domains and games.

Markowitz's model and the related literature, which grapple with rigid structures and challenges in estimating mean and variance-covariance inputs (see Michaud (1989)). ML tools offer a promising solution by generating more precise estimates of expected returns and substituting the variance-covariance matrix with more robust forecasts. Furthermore, evolutionary and deep learning algorithms hold the potential to incorporate additional and inherently complex constraints into portfolio optimization problems. ML methodologies, such as reinforcement learning, also offer the possibility of constructing portfolios directly, eliminating unnatural divisions between the inferential stage of input acquisition and the portfolio optimization phase.

Figure 1 provides a visual map to guide a Reader through some of the notions and topics covered in this paper. In the diagram, the key distinction is between the analysis of ML deployed to support systematic trading strategies (typical of banks' proprietary trading desks but also many hedge funds) covered in Section 2 vs. decision systems used in portfolio optimization, covered in Section 3. In the process, Section 2 touches upon the use of ML to implement a variety of trading strategies, including statistical arbitrage, risk parity, and momentum trading. Importantly, like in Bagnara (2024), in this survey, we do not cover papers that primarily use Bayesian methods. Although there is a clear link between Bayesian approaches and regularization in ML, these papers do not fall directly within the domain of ML. Bayesian inference is based on learning from the data through the updating of beliefs through prior and posterior probability distributions. ML encompasses computational algorithms designed to identify complex patterns in largely unstructured data.[5]

Section 3 is mostly inspired by Markowitz's mean-variance (henceforth, MV) framework but also offers some space to its multi-faceted implementations (e.g., minimum variance analysis, Sharpe ratio maximization, etc.). This means that we shall devote limited attention to an alternative type of asset allocation system—in fact, less popular in practice—that pursues a more classical (micro-founded) expected utility maximization approach (see Guidolin (2013) for a comparison of alternative asset allocation models). Sections 2 and 3 also provide the opportunity to review the key classifications of different types of ML algorithms we call upon in the paper, such as supervised vs. unsupervised ML, shrinkage and regularization vs. unsupervised dimensionality reduction techniques, reinforcement, and deep learning, etc. Ensembling and model stacking are also discussed. Section 4 briefly surveys the asset management literature that has examined the potential for total, outright automatization of the portfolio selection process through the adoption of robo-advisers. Section 5 examines recent papers on the use of ML in devising multi-factor, model-based portfolio strategies, often called "smart beta ". Section 6 briefly touches upon the use of textual analysis as a driver of portfolio decisions and in the construction of smart beta risk premia. One subsection is devoted to the recent surge of interest in large-scale language models. Section 7 covers the flourishing literature on the usage of ML in sustainable and green investing, with special reference to the problem of automatic ESG scoring (rating). Section 8 surveys the growing use of ML in trade execution that implements both the systematic strategies discussed in Section 2 and the MV and smart investment approaches in Sections 3 and 4. Section 9 injects a healthy dose of skepticism into our treatment by discussing the limitations

[5]While Bayesian methods are not inherently opposed to or in competition with ML, they represent an alternative inferential approach that can be integrated into various ML techniques, such as Bayesian neural networks (see, e.g., Neal (2012)). A detailed discussion of these intersections is beyond the scope of our review.

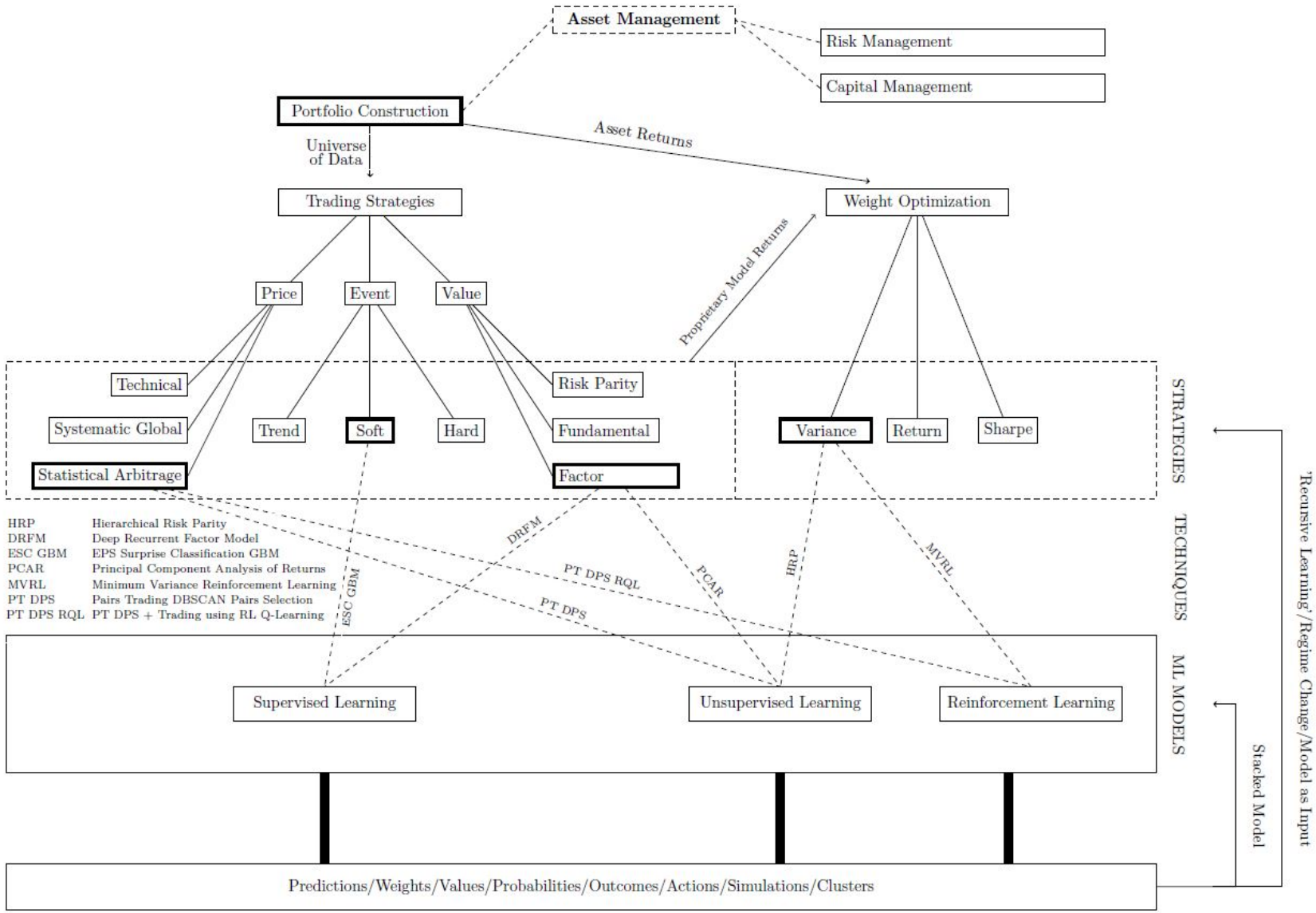

Figure 1: Classifying different types of ML algorithms and their applications to portfolio decision-making.

that ML poses to practitioners for the time being. Yet, we also emphasize how academic research may contribute to overcoming such limitations. Section 10 provides a sketch of the main findings of a novel literature that has systematically back-tested and compared the realized performance of alternative ML methods on live data sets. Section 11 concludes.

2 ML in trading strategies

Systematic trading strategies represented an early and straightforward application of ML to portfolio decisions. Under such strategies, a money manager or proprietary trader employs capital to purchase or short-sell securities based on quantitative signals derived from ML algorithms. Such signals are based on predefined rules or algorithms designed to eliminate emotions and biases from the decision-making process. The rules are created by quantitative analyses of market data and seek to exploit market inefficiencies to generate consistent returns over time. The problem of automatizing the trading rules has a long history in financial economics, straddling the fields of asset pricing and portfolio management. Over the years, the literature has witnessed a proliferation of trading strategies (often referred to as anomalies) not explained by fundamental risk as more and more stock characteristics have been shown to yield persistent predictive power for the cross-section of stock returns (see Novy-Marx & Velikov (2016)). The set of available candidate signals is, in fact, so large that a recent strand of literature has cast doubts on the robustness of the traditional tests used to identify useful predictors (see Harvey et al. (2016)). ML algorithms offer the advantage of processing large amounts of data quickly and being able to identify complex patterns, which may be challenging for human traders to recognize. This advantage allows for more informed decisions, potentially leading to better trading strategies and higher realized returns.

Price strategies include technical trading, systematic global macro strategies, and statistical arbitrage. *Technical trading* uses market data and its transformations to predict the future price of an asset. There are several types of systematic technical trading rules, including trend-following strategies, mean reversion strategies, and momentum strategies. *Systematic global macro strategies* rely on macroeconomic indicators to trade across asset classes and countries. *Statistical arbitrage* seeks to identify mispricings by detecting asset relationships and/or potential anomalies under the assumption that the anomaly will be re-absorbed by "normal " market dynamics. All the strategies above give a central role to the absolute level of prices to be compared to some reference point as part of the input data in generating the predicted outcomes.

Event strategies refer to a class of strategies that focus on trading around significant events that may impact the value of assets, such as earnings reports, mergers and acquisitions, macroeconomic announcements, and political events. The idea behind event-driven investing is that the market may not always fully price in the impact of these events, creating opportunities for smart investors to take advantage of pricing inefficiencies. Soft-event and hard-event strategies are the two types of event-driven strategies that require predicting changes caused by events. *Soft-event strategies* involve predicting the direction of a trend based on relatively weak signals in terms of concrete and immediate changes. For example, a company may be undergoing a change in management, which

is not considered a hard event but may still have an impact on the value of its stock. In contrast, *hard-event strategies* require predicting changes that have a clear and immediate impact on the value of an asset. An example of a hard-event strategy might involve predicting the outcome of a merger or acquisition and taking a position in the target company based on the latter.

Value strategies refer to investment strategies that focus on buying assets that are undervalued in the market, with the expectation that their prices will eventually increase to reflect their true value. Value strategies can be applied to a range of assets, including stocks, bonds, and real estate. Value strategies encompass several investment approaches that seek to identify undervalued assets and generate long-term returns. These strategies can be broadly categorized into three groups: risk parity, factor investing, and fundamental themes.

Given this short classification, we now use the analysis of applications of ML to systematic trading strategies to briefly review the main methodologies in ML and their key categorizations.

2.1 Supervised learning

Supervised learning (SL) techniques are used to learn the relationship between a set of exogenous attributes (features) and a designated, dependent attribute (outcome or target variable) (see Masini et al. (2023) for a detailed review of applications of SL to time series forecasting in economics and finance). Specifically, with reference to a typical financial application, SL refers to the mathematical structure describing how to make a prediction of an asset return $r_{i,t+1}$ given a vector of p predictors, $\mathbf{x}_t$. Instead of simply learning from the general environment of the task, like reinforcement learning does (see Section 3.3), SL methods learn the relationships in the data according to a precise modeling framework. SL tasks can be divided into classification and regression tasks.

- *Classification methods* in supervised learning are techniques used to predict the class or category of a given data point based on a set of features or attributes (e.g., binary 1, 0; multi-class 1, 2, 3, ..., K). These methods rely on labeled data, in which the correct output or class is known for each data point in the training set. Some commonly used classification methods include decision trees, logistic regression, naive Bayes, support vector machines (SVM), and k-nearest neighbors (KNN). Very briefly, decision trees split the data into subsets based on features, while logistic regression models the probability of a binary outcome. Naive Bayes uses probability theory to classify data, SVM finds the best hyperplane to separate data in clusters, and KNN assigns the class of the closest data points to a new data point. Choosing the appropriate classification method depends on the nature of the data and the problem at hand.
- *Regression methods* are commonly used to fit the relationship between input features and a continuous target variable. These methods aim to learn a function that can accurately predict the target variable based on the input features. Linear regression is a widely used method that models the relationship between the input features and the target variable using a linear function. Other regression methods include polynomial regressions, ridge regressions, and LASSO regressions. Of course, the field of classical econometrics illustrates a number of models for the conditional mean function that are not necessarily linear.[6]

[6]However, in no way traditional econometric models and ML can be seen as being in opposition. In fact, Tang et al.

When applied to trading strategies, the basic SL task involves some form of price (assuming its stationarity) or return prediction. This includes regressors used to predict the price level or returns and classifiers that predict the direction of price movements or, equivalently, the sign of subsequent returns. SL models, especially neural networks (to be reviewed below), can keep up with changing market regimes as long as they are able to perform *online training* (also known as sequential learning), i.e., when a model is allowed to learn from incoming data continuously, updating its parameters incrementally as new examples become available.[7]

To provide some intuition on the nature of the problem, consider a baseline portfolio problem that may be written as follows. Let $\mathbf{r}$ denote a $T \times 1$ vector of returns on one specific asset, $\mathbf{x}_{t-1}$ the $1 \times p$ vector of p candidate predictors so that $\mathbf{X} \equiv [\mathbf{x}_0' \ \mathbf{x}_1' \ ... \ \mathbf{x}_{T-1}']'$ is the $T \times p$ matrix of predictors, and T is the size of the available sample of data. The equation we use to model the returns on an asset takes the following general form:

$$r_t = f(x_{t-1}) + \epsilon_t. \tag{1}$$

Once the model has been estimated, the expected return of an asset at time $t+1$ using data available through t is

$$E_t[r_{t+1}] = \hat{f}(x_t), \tag{2}$$

where $\hat{f}(x_t)$ refers to some estimator of the functional form relating predictors to the asset return. This prediction can then be used in asset allocation or to devise a systematic trading strategy. A commonly used method is the *classical linear regression model*, estimated by ordinary least squares (OLS). In this case, the form approximating f is the following linear function:

$$\hat{f}(\mathbf{X}; \boldsymbol{\theta}) = f(\mathbf{X}; \hat{\boldsymbol{\theta}}) = \hat{\mu} + \boldsymbol{X}\hat{\boldsymbol{\beta}}. \tag{3}$$

Here $\hat{\mu}$ is the estimated intercept and $\hat{\boldsymbol{\beta}}$ is the $p \times 1$ estimated coefficient vector. The estimates of the parameters $\boldsymbol{\theta} \equiv [\mu \ \boldsymbol{\beta}']'$ are usually obtained by minimizing the residual sum of squares:

$$\arg\min_{\theta} \mathcal{L}_2(\boldsymbol{\theta}) = \arg\min_{\boldsymbol{\theta}} \|\mathbf{r} - f(\mathbf{X}; \boldsymbol{\theta})\|^2, \tag{4}$$

where $\mathcal{L}_2(\boldsymbol{\theta})$ indicates the least squares loss and $\|\cdot\|$ denotes the l_2 norm.[8] Classical empirical finance has been traditionally based on what is now known as the kitchen sink (KS) model, which is a multivariate prediction model utilizing all p predictors together (see the discussion in Welch & Goyal (2008)). It is well known that this model tends to lead to poor forecasting performance, as the estimated parameters tend to display low bias but high variance. This problem becomes more acute

(2022) reviews the applications of hybrid models, which combine traditional statistical techniques with ML methods to leverage the strengths of both approaches to financial time series forecasting. These hybrid methods have demonstrated improved forecasting performance by addressing the limitations of single models and enhancing generalization capabilities.

[7] In traditional batch learning, a model is trained on a fixed set of data, and after training, it is deployed for inference and forecasting on new data.

[8] Similarly to Gu et al. (2020), one may also consider a Huber loss function which is a hybrid of the least squares loss for relatively small errors and the absolute loss for relatively large errors and that is less sensitive to outliers (see Gupta et al. (2020)). Moreover, as we shall see below, one may expand the least square loss to include shrinkage terms with convex penalties (Ridge, LASSO, and Elastic Net) and boosting ensembles (Gradient Boosting and Regularized Gradient Boosting Machines).

as the number of predictors increases. Another very simple model that has been commonly considered in the asset management industry is the historical average (HA), where the forecast is estimated as $f(\mathbf{X};\hat{\theta}) = \hat{\mu}$, as $\beta = \mathbf{0}$.

2.1.1 SL dimensionality reduction algorithms. The supervised methods for dimensionality reduction incorporate the information of a large set of economic variables in a predictive regression framework using latent factors, which are estimated using information in both $\mathbf{r}$ and $\mathbf{X}$. In this case, the function f takes the form

$$f(\mathbf{X};\boldsymbol{\theta}) = \mu + (\mathbf{XA})\,\boldsymbol{\beta} = \mu + \boldsymbol{Z\beta}, \tag{5}$$

where $\mathbf{A} \equiv [\mathbf{a}_1' \;\; \mathbf{a}_2' \;\; ... \;\; \mathbf{a}_k']'$ is a $p \times K$ matrix of weights, $\mathbf{Z} \equiv [\mathbf{z}_0' \;\; \mathbf{z}_1' \;\; ... \;\; \mathbf{z}_{T-1}']'$ is a $T \times K$ matrix of K latent factors, with $\mathbf{z}_t \equiv [z_{1,t} \;\; z_{2,t} \;\; ... \;\; z_{K,t}]$ and $K << p$. The latent factor models we use differ based on how the matrix $\mathbf{A}$ is derived. Partial Least Squares (PLS), introduced by Wold (1980) and more recently popularized by Kelly & Pruitt (2015), decomposes the matrix of predictors $\mathbf{X}$ and the zero-mean vector of asset returns $\mathbf{r}$ into the form

$$\mathbf{X} = \mathbf{ZP}' + \mathbf{E} \qquad \mathbf{r} = \mathbf{Zq}' + \mathbf{e}, \tag{6}$$

where the matrix $\mathbf{P}$ and the vector $\mathbf{q}$ are the loadings, while $\mathbf{E}$ and $\mathbf{e}$ are the residuals. In order to find the PLS component matrix $\mathbf{Z}$, the columns of the weight matrix $\mathbf{A}$ need to be obtained by solving successive optimization problems. The columns of $\mathbf{A}$ represent the direction of maximum covariance between the predictor variables and the response variable. The optimization problem involves maximizing the covariance between the predictor variables and the weighted response variable, subject to the constraint that the weight vector is of unit length. The criterion to find the kth estimated weight vector $\mathbf{a}_k$ is

$$\arg\max_{\mathbf{a}_k} \; \mathbf{a}_k'(\mathbf{X}'\mathbf{r}\mathbf{r}'\mathbf{X})\mathbf{a}_k \quad \text{s.t. } \mathbf{a}_k'\mathbf{a}_k = 1,\, \mathbf{a}_k'\boldsymbol{\Sigma}_{XX}\mathbf{a}_k = \mathbf{0}, \tag{7}$$

where $\boldsymbol{\Sigma}_{XX}$ is the sample covariance of $\mathbf{X}$. Clearly, if $K = p$, then PLS gives a solution equivalent to the OLS estimates. Once the weight matrix $\mathbf{A}$ is obtained, the PLS component matrix $\mathbf{Z}$ can be calculated as the product of the predictor matrix $\mathbf{X}$ and the weight matrix $\mathbf{A}$. Each column of $\mathbf{Z}$ represents a linear combination of the predictor variables that is highly correlated with the response variable (in fact, the one with the highest correlation conditional on the constraints). However, in practice, it is often unclear how many components should be used in the model, and using too many or too few components may result in sub-optimal performance. To address this issue, *cross-validation* and other model selection techniques are commonly used to determine the optimal number of components to include in the PLS model.[9] Other model selection techniques, such as information criteria and sequential hypotheses testing, can be used to determine the optimal number of components. These techniques generally involve comparing the fit of models with different numbers of components and

[9]Cross-validation involves partitioning the data into training and validation sets and evaluating the model's performance on the validation set for different numbers of PLS components. The optimal number of components is then the one that yields the best performance on the validation set.

selecting the model that achieves the best trade-off between model complexity and goodness of fit (see, e.g., Light et al. (2017), Chortareas et al. (2020), Gorman & Fabozzi (2022)). For instance, Shi (2023) has recently used PLS to extract the optimal information from five pricing factors and shows these contain stronger predictive power for asset returns than standard (unsupervised learning) PCAs.

Sparse partial least squares (SPLS) is an extension of PLS that imposes an l_1 penalty for promoting *sparsity* onto a surrogate weight vector $\mathbf{c}_k$ instead of the original weight vector $\mathbf{a}_k$, while keeping $\mathbf{a}_k$ and $\mathbf{c}_k$ "close" to each other (see Chun & Keleᴄs (2010)). The kth SPLS weight vector solves

$$\arg\min_{\mathbf{a}_k,\mathbf{c}_k} \ (\mathbf{c}_k - \mathbf{a}_k)'(\mathbf{X}'\mathbf{r}\mathbf{r}'\mathbf{X})(\mathbf{c}_k - \mathbf{a}_k + \zeta_1 \|\mathbf{c}_k\| + \zeta_2 \|\mathbf{c}_k\|^2 \quad \text{s.t. } \mathbf{a}_k'\mathbf{a}_k = 1, \tag{8}$$

where ζ_1 and ζ_2 are non-negative tuning parameters.[10] To solve SPLS, a large ζ_2 value is usually required, and setting $\zeta_2 \to +\infty$ yields a solution that has the form of the soft threshold estimator by Zou & Hastie (2005*a*). This reduces the number of tuning parameters to two, i.e., the tuning parameter ζ_1 and the number of latent factors k. For instance, Ardila et al. (2017) have presented an empirical model for the diagnosis and real-time forecasting of real estate bubbles that uses SPLS to distill a diffusion index, forecast the end of the bubbles and identify the variables that are highly relevant during the bubble regime. When building systematic trading strategies, SPLS may come to represent a popular technique for identifying a small set of highly predictive features from a large pool of potential predictors.

Artificial neural networks The most successful technique emerging within the SL methodological camp is, arguably, artificial neural networks (henceforth, ANNs). The adoption of ANNs is not new in finance (see Hawley et al. (1990) for early applications). However, the recent advancements in the field and available computational power have rekindled enthusiasm among researchers. Gu et al. (2020) tackle the problem of building a multi-factor strategy from more than 900 potential predictors of stock returns using ML. They conclude that neural networks (NNs) outperform other ML approaches in this setting. In particular, they argue that shallow networks (i.e., with only two layers; see below for an explanation) do a better job than deep ones (i.e., based on three or more layers).[11] Similarly, Rather et al. (2015) propose to use a recurrent neural network that outperforms a standard regression and is based on the idea of optimizing the merging of the predictions obtained from other, simpler models. Unlike Gu et al. (2020), they find that deep NNs yield the best results. Krauss et al. (2017) present another example of this approach: they apply deep NNs to the prediction of daily stock returns and find that they are outperformed by both random forests and gradient-boosted trees.

[10] In ML, tuning a model describes the process of searching for hyperparameters that maximize prediction performance outside the training set. A popular approach is Bergstra & Bengio (2012)'s random search method: for each hyperparameter, they define a probability distribution from which to sample its values. Next, they form various models, each defined by drawing a value for each hyperparameter. Each model will compete based on some notion of predictive performance (e.g., mean absolute error) computed on alternative validation sets. The training and validation of models are performed sequentially. The ultimate choice of the tuned model depends on the average validation score over the series of validation sets based on estimates from the underlying series of training sets.

[11] Within such classification, OLS is equivalent to extremely shallow ANNs with zero layers and based on a linear activation function.

ANNs are a family of models inspired by biological neural networks that can be trained to learn complex relationships between inputs and outputs. They have been shown to be universal approximators for any continuous function f (Cybenko (1989), Hornik et al. (1989)), in the sense that given enough components (neurons) in the network and the appropriate parameters, ANNs can approximate any continuous function to a desired degree of accuracy. There are various types of ANNs, each suited to different tasks and applications. The simplest and most widely used type of ANN is the *feedforward neural network*, which consists of input, hidden, and output layers of nodes. Other types of ANNs include *recurrent NNs*, which can process sequences of inputs, and *convolutional neural networks*, which are particularly well-suited for processing images. Recurrent neural networks extract hidden states from time-series data with nonlinear dependencies; they are basically flexible autoregressive processes. Long short-term memory (LSTM) networks are a type of recurrent NN that can capture long-term dependencies in sequences and that are often particularly useful in finance where some typical data (such as interest and exchange rates) feature near unit roots.[12]

For concreteness, in this part of our review, we briefly focus on the *multi-layer perceptron* (MLP), a type of feed-forward neural network in which information flows through the functions being estimated, from $\mathbf{X}$, through intermediate computations used to approximate f , to the output $\mathbf{r}$. These models are composed of a number of layers with multiple nodes in each layer. These consist of the input layer of the predictors, one or more hidden layers containing nodes that transform the predictors using nonlinear activation functions, and an output layer that allows a final transformation of the outcome of the hidden layers to form a prediction. Formally, feed-forward ANNs can be defined as a composition of $h^{(1)}$, $h^{(2)}$, ..., $h^{(L)}$ nonlinear activation functions for each of the L hidden layers of the network $\mathbf{Z}^{(L)} = h^{(L)}(\mathbf{X}) \circ h^{(L-1)}(\mathbf{X}) \circ ... \circ h^{(1)}(\mathbf{X})$, where

$$\mathbf{Z}^{(l)} = h\left(b_0^{(l)} + \mathbf{W}^{(l)\prime}\mathbf{Z}^{(l-1)}\right), \tag{9}$$

for $l = 1, 2, \ldots, L$, where $\mathbf{Z}^{(l)}$ is the l^{th} layer of the network with $m = 1,2,\ldots, M$ nodes, $\mathbf{W}^{(l)}$ is the matrix of weights, and $b_0^{(l)}$ is the bias term (i.e., one supposed to absorb any bias in the model, when interpreted as a regression). In the case of the first hidden layer, the input is the matrix of predictors, $\mathbf{Z}^{(0)} = \mathbf{X}$, such that $\mathbf{Z}^{(1)} = h\left(b_0^{(1)} + \mathbf{W}^{(1)\prime}\mathbf{X}\right)$. The results from each hidden layer are aggregated in the output layer, $ANN(\mathbf{X}) = b_0^{(L+1)} + \mathbf{W}^{(L+1)\prime}\mathbf{Z}^{(L)}$. The number of hidden nodes in each layer can be selected according to the so-called Master's geometric pyramid rule.[13] The activation function applied to each node can take various forms. In finance, it is typical to use the rectified linear unit (ReLU) defined as $h(x) = \max(0, x)$, which maps any negative input value to 0 and any positive input value to the same value. ReLU encourages sparsity in the number of active nodes and avoids the so-called vanishing gradient problem, which may cause instability during parameter estimation.[14] However, also

[12] Autoencoder networks are another type of ANN that can be used for unsupervised learning and feature extraction. Generative adversarial networks (GANs) are a relatively recent development because they are able to generate synthetic samples that are almost indistinguishable from real data.

[13] This method involves starting with a small number of nodes in the first layer and then increasing the number of nodes in each subsequent layer by a fixed ratio. The ratio used is typically between 1.5 and 2. The purpose is to gradually increase the complexity of the network, which can help prevent over-fitting.

[14] The vanishing gradient problem is a phenomenon that happens during the training of neural networks when back-propagation is used to calculate the gradient starting from the final layer and moving backward. It happens when the

the sigmoid and the hyperbolic tangent functions are often used. Another popular activation function is the softmax function, which is used in the output layer of classification problems to map the output of the NN to a probability distribution over multiple classes (see the discussions in da S. Gomes et al. (2011), Aldridge & Avellaneda (2019)). Needless to say, the choice of activation function depends on the specific application and the nature of the problem being addressed.

The weights and bias-correction intercepts of an ANN are estimated by minimizing the objective function:

$$\arg\min_{\mathbf{W},\mathbf{b}_0} \mathcal{L}(\mathbf{r}, ANN(\mathbf{X})) = \arg\min_{\mathbf{W},\mathbf{b}_0} \|\mathbf{r} - ANN(\mathbf{X})\|^2 . \tag{10}$$

Estimation of an ANN involves solving a non-convex optimization problem that is typically trained using stochastic gradient descent (SGD). Instead of evaluating the gradient of the entire training sample at each iteration, SGD calculates it from a random subset of data, minimizing the objective function iteratively through back-propagation, a supervised learning technique that adjusts the weights of the connections between neurons to minimize the difference between the actual output and the desired output. To address the difficulties posed by the large number of parameters and the non-convexity of the objective function, various modifications to the SGD algorithm have been proposed. For example, the adaptive moment estimation algorithm (ADAM), introduced by Kingma & Ba (2014), calculates individual adaptive learning rates for the model parameters (the weights) by estimating the first and second moments of the gradients. Another approach is to modify the loss function by adding a parameter norm penalty. These modifications aim to improve the convergence rate and prevent over-fitting in the training process. For instance, under an elastic net type of penalty, based on the l_1 and l_2 norm of the weights, the regularized objective becomes

$$\arg\min_{\mathbf{W},\mathbf{b}_0} \mathcal{L}(\mathbf{r}, ANN(\mathbf{X})) + \zeta_1 \|\mathbf{W}\| + \frac{1}{2}\zeta_2 \|\mathbf{W}\|^2 , \tag{11}$$

where higher values of the hyperparameters $\zeta_1 > 0$ and $\zeta_2 > 0$ correspond to stronger regularization. To enhance the performance of ANN models in finance, several approaches have been proposed, such as early stopping, batch normalization, and forecast averaging. In each iteration of the optimization algorithm, the parameter estimates are updated to reduce prediction errors in the training sample, and the predictive performance of the model is evaluated using data from the validation sample. *Early stopping* is implemented by halting the training process prematurely when the validation error no longer decreases. This prevents over-fitting and speeds up the training significantly. *Batch normalization* shrinks the variability of predictors by scaling the input of the activation functions. It was introduced to solve the problem of internal covariate shift when the distribution of the inputs of the hidden layer changes during training as the parameters of the previous layers change. For each node in each training step, batch normalization standardizes the output of a previous activation to restore the representation power of the node, thereby increasing the stability of the NN and its training speed. Finally, to reduce the prediction variance of the NN, due to the stochastic nature of the optimization, many researchers in finance are increasingly adopting an ANN *ensemble approach* (see, e.g., Hansen & Salamon (1990)), which entails forming the final prediction from the average of the forecasts from

gradient used to update the network becomes extremely small. The opposite phenomenon is known as the exploding gradient problem.

a large number of models initialized from different random seeds.

A few applications of ANN in the realm of finance are noteworthy. Swales Jr & Yoon (1992), Wong et al. (1992) and Kryzanowski et al. (1993) were early applications of ANNs to discriminate between stocks that are predicted to provide positive returns from those that provide negative returns, based on the recognition that an ANN may learn the relationships between a company's stock return one year forward and the most recent four years of financial data for a company and its industry, as well as data for seven macroeconomic variables. To train the model efficiently, continuous data is simplified into discrete states using machine coding. This involves representing each feature point with a three-bit binary vector, simplifying trends and performance into upward, stable, or downward states. In fact, Kryzanowski et al. (1993) reported that their ANN (a so-called Boltzmann Machine) estimated using simulated annealing (a discrete optimization method that uses a gradually decreasing amount of random noise while searching for a globally optimal solution) correctly classified 72 percent of the positive/negative returns.

Chen, Pelger & Zhu (2024) have estimated a linear stochastic discount factor (SDF) starting from new test assets that are hard to price, combining no-arbitrage conditions together with three types of ANNs.[15] Knowing the SDF would allow investors to build economically driven trading strategies that buy under-priced assets and sell the over-priced ones. Chen, Pelger & Zhu (2024) assume the SDF weights linking the pricing kernel to variables and the corresponding loadings are functions of both macroeconomic and firm-specific characteristics. To find them, they solve the method of moments classical condition ($E_t\left[M_{t+1}r^*_{i,t+1}\right] = 0,\ i = 1, ..., N$), where M_{t+1} is approximated by a flexible nonparametric function and $r^*_{i,t+1}$ is the excess return of N optimized test portfolios that depend on another flexible nonparametric function of a set of potential macro variables and stock characteristics, implied by no-arbitrage condition using a GAN approach. GAN, in a loose sense, can be summarized as a min-max problem over model-induced mispricing. Within the GAN framework, the flexible nonparametric function generating the N optimal portfolios is determined through an iterative zero-sum game. Initially, an 'adversary' selects portfolios that are most challenging to price, followed by adjustments made by the asset pricer to the SDF, denoted as M_{t+1}, to accurately price these assets until all relevant information is incorporated. In practice, the GAN seeks these flexible nonparametric functions to address the min-max problem by assigning one ANN to each.[16] The algorithm iterates through the two ANNs until convergence is achieved. The optimal model incorporates dropout and ensemble learning techniques to mitigate sensitivity to specific hyperparameter selection. Implemented with 46 firm-specific characteristics and 178 macro variables as inputs, GAN attains a cross-sectional R-square exceeding 90 percent for each of the 46 corresponding anomaly portfolios. It outperforms simpler models that lack non-linearities and an adversarial approach for both Fama-French portfolios and individual stocks. Section 10 deals with additional examples of applications of ANNs.

[15] The SDF is a key concept used to price assets. It helps determine how much future cash flows or payoffs are worth today by adjusting for both the time value of money and the risks involved.

[16] The process unfolds in two main steps. Initially, a Recurrent Neural Network (RNN) with Long Short-Term Memory (LSTM) cells is employed to extract relevant hidden state variables for pricing. Subsequently, a feed-forward ANN (FFN) minimizes pricing errors by estimating asset characteristics crucial for pricing. Similarly, another RNN identifies hidden states, serving as inputs for the FFN to determine stock characteristics essential for optimizing the objective of the problem. This iterative approach focuses on constructing assets that induce significant mispricing, thereby yielding a more robust SDF compared to one derived from assets with a strong factor structure like the Fama-French portfolios.

Support vector machine regressions A support vector machine (SVM) regression is a non-parametric technique used to learn a nonlinear function by mapping it into a high-dimensional kernel-induced feature space. It does so by finding the linear function that best separates the data points in this higher-dimensional space. This approach allows for the modeling of nonlinear relationships between the input and output variables without the need to specify a specific functional form for the relationship. The resulting model is determined by a set of data informing the support vectors that lie closest to the decision boundary (the line that separates the data into different classes or regions) rather than by the entire dataset. By focusing on the support vectors, SVM regression can create a model that is robust to outliers and noise in the data, as it is only concerned with the data points that have the largest impact on the decision boundary. This property can also make the resulting model more efficient and faster to compute, as it only needs to consider a subset of the data rather than the entire dataset. Indeed, SVM regressions are particularly effective when dealing with high-dimensional datasets.

In some cases, it may be desirable for an SVM to allow a certain amount of error or deviation from the target output. This is where the *epsilon-insensitive SVM* (e-SVM) comes into play. The objective of an e-SVM is to find a function $f(\mathbf{X})$ that approximates the target output (represented by $\mathbf{r}$) within a certain threshold of error (represented by ϵ) for each training observation of $\mathbf{X}$. Additionally, the function should be as flat as possible. To achieve this, the algorithm introduces $\mathbf{a}$ and $\mathbf{a}^*$, which are non-negative vector multipliers for each observation on $\mathbf{X}$ that maximize a dual objective function. By solving this dual problem, the optimal hyperplane that maximizes the margin between the positive and negative classes can be computed. This hyperplane separates the two classes with the largest possible margin and is called the maximum margin hyperplane,

$$\arg\min_{\mathbf{a}} \ (\mathbf{a}-\mathbf{a}^*)'\mathbf{Q}(\mathbf{a}-\mathbf{a}^*) + \mathbf{r}'(\mathbf{a}-\mathbf{a}^*) + \epsilon\mathbf{1}'(\mathbf{a}+\mathbf{a}^*) \tag{12}$$

$$\text{s.t. } \mathbf{1}'(\mathbf{a}-\mathbf{a}^*) = \mathbf{0} \qquad a_i \geq 0,\, a_i^* \leq C,\, i = 1, 2, ..., t, \tag{13}$$

where $\mathbf{1}$ is the unit vector, $\mathbf{Q}$ is a $t \times t$ positive semi-definite matrix with generic element $Q_{i,j} \equiv \phi(x_i)'\phi(x_l)$, a nonlinear kernel function based on $\phi(x_i)$, a transformation that maps $\mathbf{x}$ into a high-dimensional space. Based on the Kuhn-Tucker conditions, only a certain number of Lagrange multipliers $\mathbf{a}$ and $\mathbf{a}^*$ will assume non-zero values. The observations associated with them will be affected by approximation errors equal to or larger than $\epsilon > 0$ and are referred to as support vectors.[17] Because support vectors are usually only a small subset of the training observations, this characteristic is referred to as the sparsity of the solution.

One of the problems of e-SVM is choosing the appropriate value for the parameter ϵ. The nu-support vector machine (nu-SVM) regression is a method that automatically adjusts ϵ by considering

[17]The parameter ϵ captures the trade-off between the sparseness of the representation and closeness to the data, with large values of ϵ resulting in fewer support vectors and thus a sparser representation of the solution. The cost parameter, $C > 0$, controls the penalty imposed on observations that lie outside the epsilon margin, ϵ, and helps to prevent over-fitting as it represents the balance between the flatness of $f(X)$ and the extent to which violations to ϵ are tolerated.

it as part of the optimization problem. The dual optimization problem for nu-SVM becomes

$$\arg\min_{\mathbf{a}} \ \frac{1}{2}(\mathbf{a}-\mathbf{a}^*)'\mathbf{Q}(\mathbf{a}-\mathbf{a}^*)+\mathbf{r}'(\mathbf{a}-\mathbf{a}^*) \tag{14}$$

$$\text{s.t. } \mathbf{1}'(\mathbf{a}-\mathbf{a}^*)=\mathbf{0} \qquad \mathbf{1}'(\mathbf{a}+\mathbf{a}^*)=Cv \tag{15}$$

$$a_i \geq 0,\ a_i^* \leq C,\ i=1,2,...,t, \tag{16}$$

where the parameter $0 \leq v \leq 1$ determines the proportion of the number of support vectors kept in the solution relative to the total number of observations. The final SVM regression function is then given by

$$f(\mathbf{X}) = \sum_{i=1}^{t}(a_i^* - a_i)K(x_i, x).$$

Typical applications (see, e.g., Kim (2003)) use the radial basis function (RBF) kernel, with parameter $\theta > 0$, given by $K(x_i, x) = \exp(-\theta|x_i - x|^2)$. The set of hyperparameters to be selected are $\{C, \theta, \epsilon\}$ for e-SVM and $\{C, \theta, v\}$ for nu-SVM.

A Support Vector Regression (SVR) is a related, robust method for estimating nonlinear relationships in financial time series. Unlike traditional regression techniques, SVR aims to minimize the risk of a model over-fitting the training data. In SVR, the goal is to find a hyperplane that best fits the data while minimizing the prediction error. The hyperplane is constructed such that the maximum margin between the data points and the hyperplane is achieved. The margin is defined as the distance between the hyperplane and the closest data point. This approach not only reduces the over-fitting potential but also improves the model's ability to predict future observations. SVR is based on the Structural Risk Minimization (SRM) principle, which seeks to minimize the expected error of the model while controlling for its complexity. The SRM principle is accomplished through a regularization term that penalizes the model for having too many parameters, thus preventing it from over-fitting.

SVMs and SVRs are among the most recently developed ML tools and, possibly because of that, have not been studied as much as ANNs. Nevertheless, several studies have successfully employed SVMs in financial modeling and forecasting (e.g., Cao & Tay (2003), Kim (2003), Huang et al. (2005), Chen et al. (2006), Arrieta-Ibarra & Lobato (2015)). With specific reference to systematic trading strategies, SVMs are increasingly used to predict the direction of stock price movements based on technical indicators either by themselves or in combination with fundamental data. This stems from the power of SVMs to detect relationships between predictor variables and target variables that are nonlinear, their robustness to noise and outliers, and the fact that–because they can be fitted under a variety of kernel functions–they are flexible. For instance, Huang et al. (2005) investigated the directional predictability of stock price changes at a weekly frequency and obtained that SVM outperformed ANN, even though ensemble methods eventually performed the best. Lee (2009) developed a trading system based on SVM with a hybrid feature selection method to predict trends in the stock market. He reported that SVM can outperform back-propagated ANNs. Luo & Chen (2013) have argued that SVMs offer good power against over-fitting and local minima traps that are instead typical of the way ANNs work within systematic trading strategies. They show that a remarkable perfor-

mance may be generated by obtaining stock price turning point forecasts by modeling the change rate of the closing price between the current turning point and the next one with a weighted SVM. Chen & Hao (2018) have investigated stock trading signals prediction using algorithms that integrate principal component analysis (PCA) into weighted SVM to forecast stock turning points. Henrique et al. (2018) have used SV regressions to predict stock prices for large and small capitalization in three different markets. Their findings suggest that the SVR has predictive power, especially when using a strategy of updating the model relatively often.

SL with shrinkage and regularization Within the SL camp, shrinkage methods "regularize" the coefficient estimates while fitting a model to all p predictors contemporaneously. Differently from KS, these procedures shrink the coefficients towards zero relative to the OLS estimates and aim at significantly reducing their standard errors. Shrinkage methods can also perform variable selection depending on the type of regularization. A shrinkage regression is similar to a simple linear model in that it considers only the baseline, un-transformed predictors; however, it modifies the least squares problem by adding one additional term within the loss function, a penalty reflecting the complexity of the model:[18]

$$\arg\min_{\boldsymbol{\theta}}[\mathcal{L}_2(\boldsymbol{\theta}) + \mathcal{P}(\boldsymbol{\beta};\boldsymbol{\kappa})] \qquad \mathcal{P}(\boldsymbol{\beta};\boldsymbol{\kappa}) = \sum_{i=1}^{p} \mathcal{P}(\beta_i;\boldsymbol{\kappa}). \tag{17}$$

There are several choices for the penalty function $\mathcal{P}(\cdot)$ defining the shrinking method as ridge, LASSO, elastic net, adaptive lasso, bridge, smoothly clipped absolute deviation, minimax concave penalty and smooth integration of counting, and absolute deviation, respectively. The first set of shrinkage methods is based on convex penalties that can be written as

$$\mathcal{P}(\beta_i;\boldsymbol{\kappa}) = \frac{\kappa_1}{|\hat{\beta}_i^{OLS}|^{\kappa_3}}\left[\kappa_2|\beta_i| + (1-\kappa_2)\beta_i^2\right] \tag{18}$$

where κ_1 is a tuning parameter which controls the amount of shrinkage, κ_2 is a hyperparameter that controls the trade-off between l_1 and l_2 regularization, and $\kappa_3 \geq 0$ is a hyperparameter that controls the strength of the weight $1/|\hat{\beta}_i^{OLS}|^{\kappa_3}$ assigned to each coefficient. When $\kappa_3 = 0 = \kappa_2$, (18) becomes a *ridge regression* that shrinks the coefficients towards zero but retains them all, $\mathcal{P}^{ridge}(\beta_i;\kappa) = \kappa_1\beta_i^2$;[19] when $\kappa_3 = 0$ and $\kappa_2 = 1$, (18) yields the LASSO, introduced by Tibshirani (1996), which allows for both shrinkage and variable selection, by setting some of the coefficients equal to zero, $\mathcal{P}^{LASSO}(\beta_i;\kappa) = \kappa_1|\beta_i|$. Equivalently, LASSO delivers sparseness in the estimation problem. However, LASSO is known to struggle with strongly correlated predictors, as it is indifferent between including one, the most strongly correlated one, or even both covariates as long as they get nonzero estimated coefficients. As a remedy to this deficiency, the elastic net (EN), proposed by Zou & Hastie (2005*a*), combines both l_1 and l_2 terms in the penalty, thus simultaneously performing continuous shrinkage and automatic variable selection while it can also select groups of correlated variables. The penalty in (18) produces

[18] The intercept, μ, is not included in the penalty term, else this would make the optimization procedure dependent on the origin selected for the return equation. As usual with ML methods, the hyper-parameters that control the penalization functions are typically found through cross-validation

[19] The name "ridge regression" comes from the fact that the penalty term in the objective function has the effect of shrinking the estimated regression coefficients towards zero, effectively pushing them closer to a "ridge" that keeps those points from straying too far from the center (which is zero).

the EN solution when $\kappa_3 = 0$ and $\kappa_2 \in (0, 1)$:

$$\mathcal{P}^{EN}(\beta_i; \boldsymbol{\kappa}) = \kappa_1 \left[\kappa_2 |\beta_i| + (1 - \kappa_2)\beta_i^2\right] \tag{19}$$

where $\kappa_1 = 1$ is typical. The adaptive LASSO (ALASSO), developed by Zou (2006), solves the drawback of the original LASSO, i.e., that it does not necessarily satisfy the "oracle properties" (see Fan & Li (2001)).[20] This is achieved by modifying the LASSO to include adaptive weights that penalize the individual coefficients less severely. The ALASSO solution is derived by setting $\kappa_3 > 0$ and $\kappa_2 = 1$, to obtain:

$$\mathcal{P}^{ALASSO}(\beta_i; \kappa) = \frac{\kappa_1}{|\hat{\beta}_i^{OLS}|^{\kappa_3}} |\beta_i|. \tag{20}$$

The convex penalties mentioned earlier combine l_1 and l_2 terms, which can result in rather complex models, as Fan & Li (2001) have noted. Such models may contain too many parameters, variables, or features. In such cases, non-convex penalties that include an l_0 term, which penalizes the number of non-zero coefficients in the model, can be used. These methods, which are often data greedier, produce sparser solutions with similar or improved prediction accuracy and are better suited for variable selection. The bridge regression developed by Friedman (2012) involves a penalty term based on the l_{κ_2} norm and is given by

$$\mathcal{P}^{BRIDGE}(\beta_i; \boldsymbol{\kappa}) = \kappa_1 |\beta_i|^{\kappa_2}, \tag{21}$$

where $\kappa_1 > 0$ and $\kappa_2 \geq 0$ are the tuning parameters. For $0 \leq \kappa_2 \leq 1$, the bridge penalty term represents all the penalties between ridge regression and best-subsets selection. Best-subsets selection is a method to identify the best subset of predictor variables to include in a model, and it involves fitting all possible combinations of predictor variables and selecting the subset that provides the best fit according to a chosen criterion.[21] When using the squared error loss, (21) includes ridge ($\kappa_2 = 2$), the LASSO ($\kappa_2 = 1$), and best-subsets ($\kappa_2 = 0$). Bridge penalties tend to yield the sparsest solutions by forcing many coefficients to be equal to zero but apply no shrinkage to the non-zero coefficients. For $\kappa_2 > 1$, all coefficients are strictly non-zero, and all penalties in the power family are convex, while for $\kappa_2 < 1$, the penalties are non-convex.

The smoothly clipped absolute deviation (SCAD), proposed by Fan & Li (2001), is a non-convex penalty function given by

$$\mathcal{P}^{SCAD}(\beta_i; \boldsymbol{\kappa}) = \begin{cases} \kappa_1 |\beta_i| & \text{if } |\beta_i| \leq \kappa_1 \\ \frac{2\kappa_1 \kappa_2 |\beta_i| - |\beta_i|^2 - \kappa_1^2}{2(\kappa_2 - 1)} & \text{if } \kappa_1 < |\beta_i| \leq \kappa_1 \kappa_2 \\ \frac{\kappa_1^2 (\kappa_2 + 1)}{2} & \text{if } |\beta_i| > \kappa_1 \kappa_2 \end{cases}. \tag{22}$$

For $\kappa_2 > 2$, SCAD coincides with the LASSO if $|\beta_i| \leq \kappa_1$, then smoothly transitions to a quadratic

[20] The term oracle refers to a hypothetical model or process that has access to perfect information about the future and can therefore make perfectly accurate predictions. An oracle forecast function would be one that can predict future outcomes with 100% accuracy if fed with perfect information about the future.

[21] Best-subsets selection methods, also called stepwise variable selection technique, have the advantage of considering all possible combinations of predictor variables, which allows for a comprehensive search for the best model. However, as the number of predictors increases, the number of possible subsets grows exponentially, making it computationally intensive and potentially impractical.

function for $\kappa_1 < |\beta_i| \leq \kappa_1\kappa_2$, and next remains constant for all values of β_i as long as $|\beta_i| > \kappa_1\kappa_2$. For small coefficients, the SCAD penalty implies a similar penalization rate as LASSO but leaves large coefficients not excessively penalized. The minimax concave penalty (MCP), developed by Zhang (2010), is defined by

$$\mathcal{P}^{MCP}(\beta_i; \boldsymbol{\kappa}) = \begin{cases} \kappa_1|\beta_i| - \frac{|\beta_i|^2}{2\kappa_2} & \text{if } |\beta_i| \leq \kappa_1\kappa_2 \\ \frac{\kappa_2\kappa_1^2}{2} & \text{if } |\beta_i| > \kappa_1\kappa_2 \end{cases}. \tag{23}$$

For $\kappa_1 > 0$ and $\kappa_2 > 1$, there is a continuum of penalties and threshold operators varying from hard ($\kappa_2 \to 1^+$) to soft thresholding ($\kappa_2 \to +\infty$). MCP starts with the same rate of penalization as LASSO but smoothly relaxes the penalization rate to zero as the absolute value of the coefficient increases. Furthermore, MCP relaxes the penalization rate immediately, compared with SCAD, where, instead, the rate remains flat for a while before decreasing.

Finally, the smooth integration of counting and absolute deviation (SICA) penalty (see Lv & Fan (2009)) takes the form

$$\mathcal{P}^{SICA}(\beta_i; \boldsymbol{\kappa}) = \kappa_1 \frac{(\kappa_2 + 1)|\beta_i|}{\kappa_2 + |\beta_i|}, \tag{24}$$

with $\kappa_1 > 0$ and a generally small shape parameter, $\kappa_2 > 0$. SICA is another non-convex regularization method, which is a combination between the l_0 and l_1 penalties and therefore gives sparse solutions. For smaller values of κ_2, SICA yields results closer to the best-subsets selection, while for larger values of κ_2, it is closer to LASSO.

Knight & Fu (2000) have proven that the distribution of the LASSO estimator for the parameters related to the irrelevant variables is non-Gaussian. Wang et al. (2007), for the case of dependent errors in a regression model, showed LASSO is model selection consistent (i.e., across repeated samples with designs with fixed number of regressors, it will end up selecting the regressors that actually belong to the DGP with probability one), whereas a modified LASSO similar to the AdaLASSO, is both model selection consistent and has the oracle property (the ability of the LASSO method to recover the true underlying model parameters accurately under certain conditions). Interestingly, the oracle property distinguishes LASSO from other methods like ridge regression, which does not necessarily perform variable selection. While ridge regression can perform well in terms of prediction accuracy, it may retain all variables without effectively determining which ones are truly informative.[22]

Regularized regression methods, particularly LASSO, have proven popular in empirical finance and asset management. Chinco et al. (2019) have suggested using LASSO to enhance 1-minute-ahead return forecasts when dealing with a large pool of potential predictors, which are identified with the lagged returns of the entire NYSE universe over the preceding three minutes. Utilizing LASSO regressions within 30-minute rolling windows, they report a rather impressive out-of-sample (OOS) adjusted R-square of 2.47 percent. A trading strategy based on these model predictions generates an annualized Sharpe Ratio (SR) of 1.79 OOS and a notable alpha relative to a four-factor model comprising market, size, value, and momentum factors, without significant loadings on any of these,

[22] Chan & Chen (2011) show oracle properties and model selection consistency for lag selection in ARMA models. Yoon et al. (2013) derive model selection consistency and asymptotic distribution of the LASSO, adaLASSO, and SCAD, for penalized regressions with autoregressive error terms.

except the market. Interestingly, LASSO typically selects the lagged returns for no more than 15 minutes, indicating that the predictors are transient. It primarily retains returns of stocks recently in the news concerning their fundamentals, aligning with economic reasoning. Feng et al. (2020) and Freyberger et al. (2020) employ LASSO to create data-driven combinations of stock return predictors from large sets of signals, which may support factor-driven trading. Similarly, Rapach et al. (2019) use LASSO to identify the most significant predictors and create optimal combinations among a large set of industry and market returns. Rapach et al. (2013) use adaptive elastic nets to explore the predictive ability of lagged US market returns on global indexes. Gu et al. (2020) provide an example of the application of elastic nets to the selection and combination of both fundamental and technical signals to predict stock returns. Messmer & Audrino (2017) conclude that adaptive LASSO outperforms both ordinary least squares and LASSO when used to select from a vast number of features in the perspective of deriving trading signals. Chinco et al. (2019) use LASSO to predict one-minute-ahead returns for a large number of stocks listed on the New York Stock Exchange. Their findings indicate that using penalized regression via LASSO enhances OOS predictive accuracy by selecting predictors that effectively reflect the impacts of unexpected news announcements. Dai et al. (2022) compare different shrinkage approaches to predict stock return volatility with a large set of variables and find that shrinkage, including LASSO, adaptive lasso, elastic net, and ridge regression, exhibit superior performance compared to competing models. Finally, Guidolin & Pedio (2022) compare the OOS predictive power of best-subsets selection methods and hidden Markov models applied to commodity futures returns and find evidence that either types of model outperform the benchmarks mainly in low-volatility regimes. However, trading strategies built on best-subsets methods (especially stepwise predictive regressions) achieve higher realized Sharpe ratios compared to an investor employing AR(1) forecasts.

Ensemble methods Within the domain of portfolio selection, the use of SL algorithms is often problematic due to several reasons. One of the primary issues is that past asset returns are not always representative of their subsequent behavior because of the occurrence of structural breaks and regime shifts in the dynamics of returns. Additionally, without specific safeguards and precautions, SL models tend to over-fit the historical data, which can lead to poor performance when applied to new data. Finally, the iterative steps from ML prediction through policy development, backtesting, and parameter optimization are fragile, slow, and prone to error. Such a fragility, and the corresponding occasional under-performance of SL methods in OOS experiments, has recently led many researchers to select and combine existing individual signals originated by individual SL methods, often by additionally using ML methods (see, e.g., Bartram et al. (2021)). In fact, SL methods may be supplemented by additional ML to produce forms of optimal *ensembles*. Ensembling achieves a better balance between bias and variance, which can lead to more accurate predictions.[23] Equivalently, if many different algorithms trained on different training sets all find similar patterns and reach similar

[23]Bias refers to the error introduced by approximating a real-world problem, which may be complex, with a simplified model. High bias can lead to under-fitting, where the model fails to capture the underlying patterns in the data. Variance refers to the model's sensitivity to fluctuations in the training data. High variance can lead to overfitting, where the model captures noise instead of the underlying signal, performing well on training data but poorly on unseen data. In many applications, rich models characterized by low bias are plagued by large variance; simple models that are not prone to overfitting are easily biased. See Belkin et al. (2019).

conclusions, we can be more confident that the forecast is robust and not the result of over-fitting.[24]

There are several examples of ensembling being used successfully in equity market trading signal generation. For instance, Tsai et al. (2011) and Krauss et al. (2017) have all applied this approach. Borghi & De Rossi (2020) conducted an experiment in which they combined an ensemble of random forest models, neural networks, gradient-boosted trees, and regularized regressions to predict stock returns. They found that a trading strategy based on model combinations tended to outperform strategies based on individual models. Similarly, Wolff & Echterling (2020) built ML models to predict weekly returns for the constituents of the S&P 500 index and found that an ensemble of deep NNs, random forests, and long short-term memory (LSTM) ANNs delivered the best risk-adjusted performance.

An ensemble of models is often optimized by a second supervised ML model that would decide on the most profitable model weighting scheme; this is then known as a *stacked* model. Three key types of nonlinear models from the ML literature are structural ensembles: regression trees, support vector machines, and ANNs. We now briefly describe them. A *regression tree* is a non-parametric model that is constructed using a recursive binary splitting approach (Breiman et al. (1984)). At first, starting from the top of the tree, we divide the predictor space into two distinct and non-overlapping rectangular regions (also called "leaves") and model, say, the asset returns as the simple average of $\mathbf{r}$ within that region. Then, one or both of those regions are split into two more regions, etc. This process continues until certain stopping criteria are met. The predictor variable upon which a branch is based and the value where the branch is split are chosen to minimize the mean squared forecast error or some other loss function. Specifically, the prediction of a tree, $\mathcal{T}$, with M leaves $R_1, R_2, \ldots, R_M$ and maximum depth D (the maximum number of levels that the tree can have, which in turn controls the number of splits and the complexity of the model), is defined as $\mathcal{T}(\mathbf{X}, c, M, D) = \sum_{m=1}^{M} c_m I_{\{\mathbf{X} \in R_m(D)\}}$, where $R_m(D)$ represents the mth partition of the predictor space. The score c associated with partition m, c_m, is the simple average of returns within region R_m, written as

$$c_m = \frac{1}{T_m} \sum_{\mathbf{X} \in R_m(D)} \mathbf{r}(\mathbf{X}), \tag{25}$$

where T_m denotes the number of observations in region m. Typically, at each branch, we choose the sorting variable among the set of predictors and the split value that minimizes the loss function:

$$Q_m(c, T_m) = \frac{1}{T_m} \sum_{\mathbf{X} \in R_m} (\mathbf{r}(\mathbf{X}) - c_m)^2. \tag{26}$$

Branching halts when the depth, D, of the tree reaches a pre-specified threshold that is tuned adaptively using cross-validation methods.

Since locating the globally optimal partition is often computationally infeasible, it becomes necessary to employ an algorithm to approximate the solution. In this regard, the greedy algorithm (see Hastie et al. (2009)) is a local method that focuses on a single branch at a time, disregarding the

[24] Rasekhschaffe & Jones (2019) emphasize the importance of model averaging to mitigate over-fitting and recommend various types of forecast combinations, such as across different models, training sets, and forecast horizons.

remainder. For each splitting variable $j = 1, \ldots, Q$, a splitting point s creates two half-planes of the covariates $\mathbf{X}$, that is, $\{\mathbf{X}|x_j \leq s\}$ and $\{\mathbf{X}|x_j > s\}$. We therefore seek both j and s to minimize the local measure

$$\min_{j,s} \left\{ \min_{c_1} \sum_{\mathbf{X} \in \{\mathbf{X}|x_j \leq s\}} (r(\mathbf{X}|x_j \leq s) - c_1)^2 + \min_{c_2} \sum_{\mathbf{X} \in \{\mathbf{X}|x_j > s\}} (r(\mathbf{X}|x_j > s) - c_2)^2 \right\}.$$

One can show that the constants c_1 and c_2 are the average returns in each sub-region as defined by the predictors. Thus, identifying the optimal splitting point for each variable is easy. We keep partitioning the feature space into half-planes, refining the splits until we develop M leaves. This iterative process is termed recursive binary splitting.[25]

Regression trees remain unaffected by monotonic transformations of the data and can capture up to $D - 1$ interactions with a depth D. The depth and the corresponding number of leaves serve as tuning parameters that govern the model's complexity, and their selection should involve techniques aimed at reducing overfitting. In this matter, *cost-complexity pruning* involves initially growing a large tree with M leaves and subsequently "pruning" it down to a smaller one by consolidating some leaves to minimize a Ridge- or LASSO-penalized Sum of Squared Residuals over the terminal nodes. This process generates a series of smaller trees, with the one having the lowest cost emerging as the final tree. For instance, because building a cross-section of asset returns conditional on asset-specific characteristics is equivalent to forming managed portfolios, which can be used for unconditional modeling, Bryzgalova et al. (2020) use regression trees to tackle the problem of the selection of test asset portfolios in the search for tradeable risk factors useful in devising trading strategies. On the one hand, they show that the conventional sorting-based portfolios drastically fail to span the SDF and hence lead to the wrong conclusions when used to construct asset pricing models. On the other hand, within a mean-variance framework, their solution (which they call *Asset Pricing Trees*) addresses all problems of conventional sorts by resorting to conditional tree portfolios pruned of the overall portfolio set based on the SDF spanning requirement. In fact, Bryzgalova et al. (2020) suggests that while conventional approaches in the literature have segmented the development of profitable strategies with ML into two distinct phases–first employing advanced methods to extract signals for predicting future returns and then utilizing these signals to construct profitable portfolios–they emphasize instead the importance of integrating these steps: ML techniques should extract signals most pertinent to the overall economic problem, rather than solely focusing on return prediction.[26]

Nonetheless, regression trees are among the methods that tend to over-fit, typically resulting in inferior OOS performance. To improve the prediction accuracy of regression trees, in finance applications, researchers typically resort to approaches that combine a large number of trees, $\mathcal{T}_1$, $\mathcal{T}_2$, ..., $\mathcal{T}_k$ to yield a single consensus prediction. The first set of methods is based on *boosting*, proposed by

[25] While the greedy algorithm is efficient, it does not guarantee that the resulting sequence of thresholds and splits is optimal (see the discussion in Gu et al. (2020)).

[26] In fact, when Bryzgalova et al. (2020) compare the OOS performance of pruned AP-Trees to that of long-short prediction portfolios formed by leading tools in ML, namely random forest and neural networks from Gu et al. (2020), they find that while retaining easy interpretability of the portfolios, AP-Trees are superior in their empirical performance, based on both the Sharpe ratio and SDF-based alpha estimates.

Schapire (1990) and Freund (1995) in classification problems. In a boosting scheme, one recursively combines a large number of shallow trees, known as "weak learners ", to form an ensemble of trees with greater stability than an individual, more complex tree. The first ensemble method we consider is the gradient boosting machine (GBM), proposed by Friedman (2001), which extends boosting to regression frameworks. *Gradient boosting* is initialized by fitting a shallow tree (i.e., a seed tree with a small number of levels or nodes); then, a second tree with the same depth is used to fit the residuals of the previous model and the forecasts of these two trees are added together to form a single ensemble prediction. This procedure is repeated sequentially until the total number of iterations k is reached. The goal is to minimize the objective function, typically based on a squared loss:

$$\arg\min_{f} \mathcal{L}(\mathbf{r}, f_k(\mathbf{X})) = \arg\min_{f} \|\mathbf{r} - f_k(\mathbf{X})\|^2 . \tag{27}$$

Gradient boosting is an additive model that can be expressed as

$$f_k(\mathbf{X}) = \sum_{j=1}^{k} f_j(\mathbf{X}, c_j, M_j, D, v), \tag{28}$$

where k is the total number of trees and f_j is given by $f_j(\mathbf{X}, v) = f_{j-1}(\mathbf{X}) + v\mathcal{T}_j(\mathbf{X}, c_j, M_j, D)$. The parameter v controls the learning rate of the boosting procedure that scales the contribution of each tree to the ensemble by a factor of $0 < v < 1$ and prevents the model from over-fitting.

Recently, many applications in empirical finance (see, e.g., Kynigakis & Panopoulou (2022)) have also considered a regularized gradient boosting machine (RGBM), which is an extension of GBM that includes a regularization term in the loss function to control for the complexity of the model and avoid over-fitting. In this case, the objective function is

$$\arg\min_{f} \mathcal{L}(\mathbf{r}, f_k(\mathbf{X})) + \Omega(f_k(\mathbf{X})). \tag{29}$$

Refining the definition of a tree to be $\mathcal{T}(\mathbf{X}) = \vartheta_{q(\mathbf{X})}$, where ϑ is a vector of scores for each region and q is a function assigning each observation to the corresponding leaf, the regularization term $\Omega(\cdot)$ is defined as

$$\Omega(f_k(\mathbf{X})) = \kappa_2 M_f + \frac{1}{2}\kappa_1 \|\boldsymbol{\vartheta}\|^2 + \kappa_3 \|\boldsymbol{\vartheta}\| , \tag{30}$$

where the regularization parameter κ_2 captures the complexity cost caused by the introduction of an additional leaf to the tree and the parameters κ_1 and κ_3 control the strength of the l_2 and l_1 regularization of the weights, respectively.

The second set of ensemble methods is based on *bootstrap-aggregating or bagging* (Breiman (1996)). These approaches combine many noisy but approximately unbiased models to reduce the variance of the forecasts and improve their accuracy. Bagging involves creating multiple samples of the training dataset by randomly selecting observations with replacement. This means that some observations may be selected more than once, while others may not be selected at all. For each sample, a new model is trained on the bootstrapped data, resulting in a set of diverse models. The individual models are then combined using an ensemble method, such as averaging or voting, to make a final prediction.

Because bagging is particularly effective when applied to unstable or high-variance models, such as regression trees, we shall present the methodology in that context. The baseline bagging procedure estimates $\mathcal{T}_1$, $\mathcal{T}_2$,..., $\mathcal{T}_B$ based on B different bootstrap samples of the data and then averages their forecasts to obtain a single low-variance model:

$$f_B(\mathbf{X}) = \frac{1}{B}\sum_{b=1}^{B}\mathcal{T}_b(\mathbf{X}, c_b, M_{bj}, D). \tag{31}$$

In bagging, usually the regression trees are simply i.d. (identically distributed, but not independent), with the variance of the average estimates that depend on the product of the variance of each tree and the correlation among trees as the number of bootstrapped trees increases (see Hastie et al. (2009)).[27] Typically, in empirical finance, users adopt the stationary bootstrap (see Politis & Romano (1992)) instead of a standard (i.i.d.) bootstrap, due to the underlying time series structure of the data (see Jin et al. (2014)).

It is well known that boosting tends to take longer to run than bagging algorithms do. Boosting sequentially fits estimators on the training set and gives more weight to mis-classified observations in successive boosting rounds. The resulting strong boosted learner is an accuracy-weighted average of the weak learners. By giving more weight to more successful learners, boosting can address bias. If we allow the boosting algorithm to overweight successful weak learners too aggressively, however, this benefit will be more than offset by increasing variance. Because of this trade-off, boosting algorithms tend to require more careful parameter tuning than bagging algorithms and conservative learning rates tend to be preferable in finance, at least to device trading rules for stocks.

One popular variation of bagging that has occasionally found application in empirical finance (see, e.g., Sadorsky (2022)) is *random forests* (RF), proposed by Breiman (2001*a*), which aims to reduce the variance of the average forecast by minimizing the correlation among the regression trees in the ensemble. A random forest algorithm creates multiple decision trees by randomly selecting a subset of input features at each split. This reduces the correlation between trees and improves prediction accuracy. Extremely Randomized Trees (ERT) is a variant of RF that adds more randomness to the tree construction process (see Geurts et al. (2006)). ERT selects the split value at each node randomly from a uniform distribution instead of choosing the optimal split based on a specific criterion. This further reduces the correlations among trees and results in improved accuracy and generalization. For example, ERT can be used to predict stock prices based on multiple input features such as company financial data, market trends, and macroeconomic indicators. ERT would randomly select a subset of these features at each split and then select the best split based on the highest-performing random split.[28] Jiang et al. (2020) and Saifan et al. (2020) are two successful examples of applications of ERT

[27] The dependence derives from the fact that the regression trees used in bagging are correlated with each other as they are all constructed using slightly different versions of the same training data. The variance of the bagged model is affected by two factors: the variance of each individual tree and the correlation among the predictions of the trees. If the trees are highly correlated, then their predictions tend to be similar, which reduces the variance of the bagged model. If the trees are independent, then their predictions tend to be more diverse, which increases the variance of the bagged model.

[28] The hyperparameters in ERT include the number of trees in the ensemble, the maximum depth of the trees, and the size of the random feature subset used at each split. ERT is especially useful for high-dimensional data with many

to forecasting stock prices and building trading strategies.

Decision and regression trees are indeed a popular class of ML methods employed in generating multi-factor signals. In fact, Bryzgalova et al. (2019) point out that the standard factor models à la Fama & French (1993) and Fama & French (2015) can be viewed as simple tree models with just two (and five) splitting dimensions based on the quantiles of the distributions of company fundamentals, such as the book to price ratio or operating profitability. Applying this intuition, Coqueret & Guida (2020) have produced tree-based forecasts of the returns of a large set of US equities between 2002 and 2016 using extreme gradient-boosted trees. Also Gu et al. (2020) estimate a number of tree-based models using gradient-boosted regression trees. Most papers reach the conclusion that shallow trees and models with a limited number of trees outperform in applications to equities. Leung et al. (2021) use gradient-boosted trees to predict individual monthly stock returns out- or underperformance relative to its peers in the same region and industry using 20 stock characteristics categorized as size, book-to-market, profitability, and asset growth (inspired after the Fama–French five-factor model), momentum, and short-term reversal. They conclude that, despite the statistical advantage of ML predictions, the economic gains tend to be limited and depend critically on the ability to take on risk and to implement trades efficiently, which appear to be more critical than devising the trading strategies themselves.

2.2 Unsupervised learning

Unsupervised learning (USL) is a type of ML in which an algorithm is trained to identify patterns and structures in the data without a need for labeled examples, i.e., without any supervision. A USL framework is useful when a researcher is not quite sure what to look for in the data. It is often used for an exploratory analysis of the raw data. In finance, unsupervised learning techniques are commonly used for tasks such as anomaly detection, clustering, and dimensionality reduction. In particular, the latter is a technique that involves reducing the number of features in a dataset while preserving the most important information. In our applications, while in the case of the dimensionality reduction methods that fall under SL, the directions that best represent the predictors $\mathbf{X}$ are derived in a supervised way since the vector of asset returns, $\mathbf{r}$, is used to determine the component directions, under USL, the algorithm derives the latent factors regardless of the values taken by the asset returns in the sample.

Principal component analysis (PCA) is the most widely used method to obtain estimates of latent factors in empirical finance and now scores hundreds of applications in the finance literature, also to offer grounds to trading strategies (see, e.g., Guidolin & Pedio (2022), with reference to commodities). PCA can be viewed as a regression-type problem in which the goal is to find the first $K \leq p$ principal component weight vectors collected in $\mathbf{A}$ as:

$$\arg\min_{\mathbf{A}} \ \left\|\mathbf{X} - \mathbf{X}\mathbf{A}\mathbf{A}'\right\|^2 \quad \text{s.t. } \mathbf{A}'\mathbf{A} = \mathbf{I}_k. \tag{32}$$

The solution to this problem is most often obtained via the singular value decomposition, $\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{A}'$,

noisy features.

where the columns of $\mathbf{A}$ are the principal component weights and each column is used to derive the mth principal component, $\mathbf{z}_m = \mathbf{X}\mathbf{v}_m$; thus, $\mathbf{ZV}$ is the dimensionality-reduced version of the original predictors. The derived variable $\mathbf{z}_1$ is the first principal component of $\mathbf{X}$ and has the largest sample variance amongst all normalized linear combinations of the columns of $\mathbf{X}$; $\mathbf{z}_2$ is the second principal component of $\mathbf{X}$, it is orthogonal to $\mathbf{z}_1$, and it has the second largest sample variance amongst all normalized linear combinations of the columns of $\mathbf{X}$; etc. PCAs constitute the closest subspace (e.g., surface) to the data cloud, thereby providing a k-dimensional approximation in terms of Euclidean distance. The proportion of variance explained by each of the PCs is the ratio of its variance to the total variance of the data. Hence, one shall keep estimating PCs until their cumulative proportion of variance explained reaches a user-defined target. Nonetheless, PCA is not without problems. The method is particularly beneficial in scenarios in the presence of high collinearity, where shrinkage methods may lead to sub-optimal forecasts. Although PCA has often displayed significant explanatory power across various applications, it is susceptible to overfitting. Another limitation is its sensitivity to the structure of the data. For instance, if one variable accounts for a large proportion of data variability, the first PC is likely to align closely with that variable. Furthermore, if this variable were divided into L smaller segments, the first L PCs would roughly align with these segments and, despite no deep changes in the data, the outcomes can vary significantly based on how the splits are applied, as a shift from one principal component to multiple ones occurs. Moreover, PCA extracts features with no knowledge of the predicted variable (since it is an unsupervised method) and that can be problematic because the PCA transformation is useful only if the predicted variable is highly correlated with the principal components. That is not generally the case in applications (e.g., prediction of default probabilities in a bond portfolio), in which it would have been useful to extract features that allow one to discriminate best between the most relevant cases (e.g., defaulted and non-defaulted bonds) of the target variable. Finally, it is well known that the eigenvectors associated with the smallest eigenvalues on which the PCAs are based cannot be robustly estimated. This is particularly true in the context of the covariance matrices for financial returns because of the inherently low signal-to-noise ratio. López de Prado (2022) emphasizes how crucial it is to identify what eigenvalues are associated with noise and shrink only those, especially in the estimation of return covariance matrices.

As an example of an application of PCA to trading strategies, Kozak et al. (2020) demonstrates that a low-dimensional SDF, constructed using principal components from the covariance matrix of a set of test assets, can effectively price multiple portfolios that were previously difficult to value, thereby potentially generating positive average abnormal returns. However, this does not provide insights into the rationality of the economy underlying such an SDF. Mis-pricing may still occur because arbitrageurs typically exploit near-arbitrage opportunities only to the extent that they are orthogonal to common risk sources. Moreover, they are often hesitant to trade aggressively against anomalies that expose them to factor risk.

Sparse principal component analysis (SPCA), as developed by Zou et al. (2006), is based on the regression/reconstruction property of PCA–the fact the original data can be reconstructed from the reduced set of principal components by projecting them back onto the original space–and produces modified principal components with sparse weights, such that each principal component is a linear

combination of only a few of the original predictors. Zou et al. (2006) show how PCA can be viewed as the solution to a ridge regression problem by adding the l_1 penalty to the optimization in (32); they also convert it to an elastic net regression, which allows for the estimation of sparse principal components. In general, the following regression criterion is proposed to derive the sparse principal component weights,

$$\arg\min_{\mathbf{A},\mathbf{C}} \ \left\|\mathbf{X}-\mathbf{X}\mathbf{A}\mathbf{C}'\right\|^2+\zeta_1\|\mathbf{c}\|+\zeta_2\|\mathbf{c}\|^2 \quad \text{s.t. } \mathbf{A}'\mathbf{A}=\mathbf{I}_k, \tag{33}$$

where $\mathbf{C}$ is $p\times K$. If $\zeta_1=\zeta_2=0$ and $\mathbf{C}=\mathbf{A}$, then the minimizer of the objective function is exactly the first K weight vectors of ordinary PCA. The l_1 penalty imposed on $\mathbf{c}$ induces sparseness of the weights, with larger values of ζ_1 leading to sparser solutions. The algorithm by Zou & Hastie (2005*b*) is then used to compute the sparse approximations of each principal component.

The *independent component analysis* (ICA), developed by Comon (1994), aims at finding a linear representation of any non-Gaussian data so that the components are statistically independent. ICA is particularly useful when dealing with sources of shocks that are mixed with a signal: by separating the independent components of the signal, ICA can help identify the sources of noise and eliminate them from the signal. For instance, ICA can be used to identify the independent components of a portfolio's returns, which can then be used to construct a diversified portfolio. By identifying the independent sources of returns, ICA can help investors reduce their exposure to common market factors and diversify their portfolios across uncorrelated assets. The ICA objective is a variation of (32):

$$\arg\min_{\mathbf{A}} \ \|\mathbf{X}\mathbf{A}\| \quad \text{s.t. } \mathbf{A}'\mathbf{A}=\mathbf{I}_k. \tag{34}$$

Solving the ICA problem amounts to finding an orthogonal $\mathbf{A}$ such that the components of the vector random variable $\mathbf{Z}=\mathbf{X}\mathbf{A}$ are independent and non-Gaussian, which is a reasonable assumption for many economic and financial variables. In practice, the independent components are obtained iteratively by estimation of the matrix $\mathbf{A}$, systematically increasing the degree of independence of the components.[29] Ordinary ICA has two drawbacks; it requires constrained optimization, which can be computationally challenging in high dimensional settings and is sensitive to whitening, a pre-processing step that de-correlates the input data that, however, cannot always be computed exactly when $p>>T$. Le et al. (2011) have proposed a *Reconstruction Independent Component Analysis* (RICA), which overcomes the drawbacks of ICA by replacing ICA's orthonormality constraint with a reconstruction penalty. This leads to the unconstrained problem,

$$\arg\min_{\mathbf{A}} \ \|\mathbf{X}\mathbf{A}\|+\zeta\left\|\mathbf{X}-\mathbf{X}\mathbf{A}\mathbf{A}'\right\|^2, \tag{35}$$

where $\zeta>0$ is a regularization parameter. RICA is equivalent to ICA when $K<p$, so that the data can actually be whitened first and $\zeta\to+\infty$.

[29]The most commonly used algorithm for ICA estimation is called the FastICA algorithm. It uses a fixed-point iteration scheme to estimate the independent components of the mixed signal. At each iteration, the algorithm updates the values of the parameters in $\mathbf{A}$ to maximize the non-Gaussianity of the components. Other optimization algorithms, such as maximum likelihood estimation (MLE) and Bayesian methods, can also be used for ICA estimation. These methods typically require more computational resources and may not be as efficient as FastICA.

Clustering is a technique used to group similar observations together. The main goal of clustering is to find structure or patterns in data without any prior knowledge of the data's labels or groups. There are several methods of clustering, including *hierarchical clustering*, *K-means clustering*, and *density-based clustering*. Hierarchical clustering involves creating a tree-like structure of clusters in which each node represents a group of data points. The algorithm works by iteratively merging the closest pair of clusters until all data points belong to a single cluster. Figure 2 visually illustrates the logic of the algorithm. Hierarchical clustering is useful when the number of clusters is not known beforehand. K-means clustering involves partitioning the data into K clusters, where each data point is assigned to the nearest cluster center. The algorithm works by iteratively updating the position of the cluster centers until the clusters are stable. K-means clustering is useful when the number of clusters is known beforehand and the clusters are roughly spherical, i.e., characterized by compactness and isotropy (i.e., when they are evenly distributed around a central point and have approximately the same radius in all directions).[30] Density-based clustering involves identifying areas of high density in the data, which are then used to form clusters. The most common density-based clustering algorithm is DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*).[31]

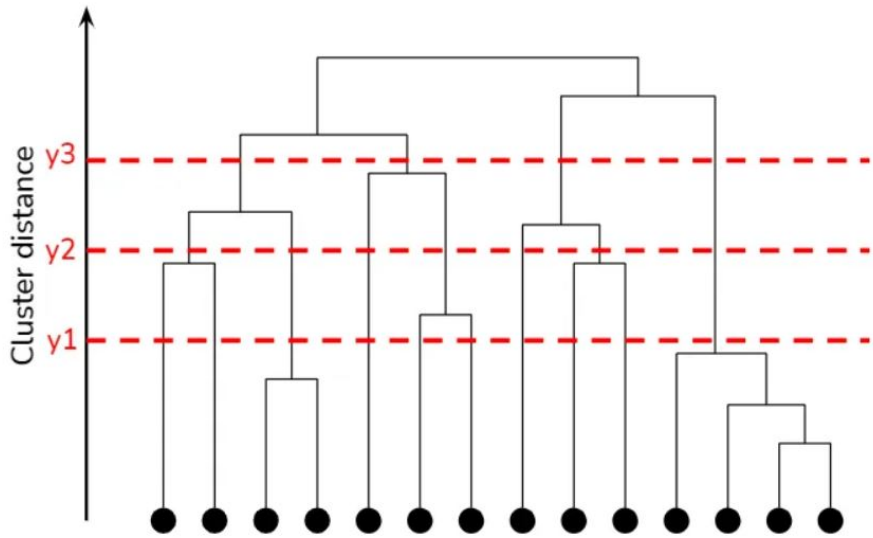

Figure 2: Schematic visualization of the working of a hierarchical clustering algorithm

In portfolio decisions and trading strategy applications, clustering can be used in a variety of ways. For example, clustering is useful to identify similar stocks or assets based on their historical returns. This can be useful in portfolio optimization, as it can help investors identify diversification opportunities and minimize exposure to common risk factors. Another example is when a trader uses clustering to identify a group of stocks that tend to move together based on certain factors, such as sector or industry. One application of clustering in systematic trading is mean reversion strategies. If a trader has identified a group of assets that tend to mean revert together, the trader

[30]Spherical clusters are a common assumption deriving from the fact that each cluster is modeled as a multivariate Gaussian distribution with a spherical covariance matrix; the latter feature means that the variance of the data points within each cluster is approximately equal in all dimensions.

[31]DBSCAN works by identifying points that are in dense regions and separating them from points that are in less dense regions. Density-based clustering is useful when the clusters have irregular shapes, departing from Gaussianity

may systematically buy assets that have fallen below their historical mean and sell assets that have risen above their historical mean, betting that they will eventually revert to their mean. If a trader has identified a group of assets that tend to exhibit momentum together, she may develop a momentum strategy. For example, the trader may buy assets that are trending upwards and sell assets that are trending downwards, betting that the trends will continue, also obtaining substantial diversification benefits. If a trader has identified two assets that tend to move together but have recently diverged, she may implement a *pair trading strategy*, i.e., buy one asset and short the other, betting that the prices will eventually converge. Recently, Gorman & Fabozzi (2022) have used a variety of supervised and unsupervised dimensionality reduction algorithms to investigate whether the smart beta (alternative) risk premia may represent a separate investment category consisting of a wide range of rule-based strategies targeting returns representing either compensation for bearing risk or behavioral biases.

3 ML in portfolio optimization

Almost 70 years after the work by Markowitz (1952) was published, the standard approach to portfolio construction in the financial industry is still deeply rooted in the mean-variance (MV) traditional approach. Key textbooks on quantitative portfolio management, such as those by Grinold & Kahn (2000), Goetzmann et al. (2014), make substantial use of MV. However, critics, such as Michaud 1989 and Kolm et al. (2014), have pointed out that, in practice, the MV paradigm suffers from the difficulty of precisely estimating its key inputs, in particular, expected returns. A disturbing consequence of such inaccuracy is that a simple equally weighted portfolio of all available assets, which makes no assumptions on the direction of future returns, may often turn out to be a hard benchmark to beat for optimized portfolios (DeMiguel et al. (2009)).[32] Hence, it is natural to view portfolio construction as an area in which the recent advances in ML techniques may inspire wide-ranging innovation.

3.1 Improving portfolio optimization through more accurate input forecasting

An obvious starting point is represented by the use of ML algorithms to obtain more accurate (either less biased, less volatile, or both) forecasts of future expected returns on each asset or asset class entering a portfolio problem. One can read Section 2 of our paper as a discussion of such methods and attempts as they have appeared so far in the quantitative finance literature. Of course, improving the quality of the MV (or related) weights through more careful prediction of expected returns implies that ML methods are used to forecast returns, which are to be seen as continuous random variables. Yet, most often ML algorithms are best employed to forecast the classification rank (e.g., trivially, the sign) of asset returns, i.e., ML is often more useful at *classifying* than at continuously predicting prices or returns directly (which is instead best seen as a regression problem), see e.g., Pirayesh & Sallan (2024).[33] Interestingly, in this approach, one first classifies assets to determine their likelihood

[32] However, a number of more recent papers have questioned such a finding, see among the others, Bianchi & Guidolin (2014), Fugazza et al. (2015).

[33] Forecasting through regressions requires the algorithm to learn complex relationships between various factors that affect the returns of an asset. These relationships can be nonlinear and involve multiple factors, making it challenging for the algorithm to predict accurately. Additionally, financial markets are inherently noisy, and there is massive uncertainty involved, making it even more difficult to forecast returns instead of their sign or cluster. Nonetheless, in

of achieving the expected return, and then, only with reference to the assets classified as likely to meet the expected return are used in the next stage, the MV model is used to determine the proportion of capital allocated to each asset (see, e.g., Paiva et al. (2019)).

Regardless of whether ML has been used or not in obtaining forecasts of expected asset returns, the next step is the potential use of ML to predict portfolio risk. There are various ML techniques that support portfolio risk forecasting, including ANNs, decision trees, and SVMs.[34] For example, SVRs can be used to estimate the conditional variance of financial returns based on past returns and other market variables. Peng et al. (2018) have combined traditional GARCH with ML approaches to volatility prediction, estimating the mean and volatility equations using SVR and comparing it to GARCH models. Using data on cryptocurrencies and exchange rates, they show that SVR-GARCH significantly outperforms GARCH, EGARCH and GJR-GARCH models with Normal, Student's t and Skewed Student's t distributions. Recently Zhang & Qiao (2021) have investigated whether the nonlinear SVR method applied to a Heterogeneous Auto-Regressive model (SVR-HAR) can compete with combination methods in terms of OOS realized volatility forecasting, reporting encouraging results.[35] Wang et al. (2020) have proposed using short-term memory networks and a mean-variance model for optimal portfolio formation in conjunction with a procedure of asset pre-selection, in which any long-term dependence of financial time-series can be captured. They report that their SL method outperforms all benchmarks in terms of the cumulative return per year, Sharpe ratio over three-year sub-samples as well as average return scaled by realized risk per month.

The ANNs applied to volatility forecasting typically use a multi-layer feedforward architecture, in which the input layer consists of past returns and other market variables, the output layer consists of predicted volatility, and one or more hidden layers capture the non-linear relationships between the input and output layers. The weights of the NN are adjusted during training using back-propagation to minimize the difference between the actual and predicted volatility. One important aspect of using ANNs in volatility forecasting is the need for careful data pre-processing and feature engineering to extract relevant information from the raw data. Feature engineering is the process of selecting, transforming (e.g., by scaling or normalization), and creating new features from raw data to improve the performance of ML models, where the features are the individual properties defining the data that are used as input variables. Additionally, hyperparameter tuning is required to optimize the network architecture and regularization parameters. For instance, already Donaldson & Kamstra (1997) had constructed a semi-nonparametric nonlinear GARCH model based on an ANN and reported OOS volatility forecasts for international stock index returns which encompassed those from plain vanilla

finance, predicting the sign is often more important than predicting the size of a return: failing to predict the size is an opportunity loss, but failing to predict the sign is an actual loss. In addition, it is common in finance to find that the sign and size of an outcome depend on different features; see the discussion in de Prado (2020).

[34] We refrain from classifying classical GARCH models as belonging to the ML domain, even though in an ML perspective, GARCH certainly class as SL regression-type architectures, see, e.g., Chen et al. (2010) and Yamaka et al. (2021).

[35] A HAR (Heterogeneous Autoregressive) model is a popular econometric tool used to forecast volatility. This model was developed by Corsi (2009) to capture the heterogeneity in the persistence of volatility across different time horizons. The HAR model is based on the idea that volatility can be predicted by past observations at different frequencies. The HAR model has been shown to outperform other popular models such as GARCH and EGARCH; see Corsi et al. (2012) for an introduction and Guidolin & Panzeri (2024) for recent evidence

GARCH models.[36] Kristjanpoller et al. (2014), Kristjanpoller & Minutolo (2015, 2016) test hybrid Neural Networks-GARCH models for volatility of the returns of three stock indices and find that ANN improves the forecasting performance of GARCH. Ramos-Pérez et al. (2019) introduce a model based on a set of ML techniques that include ANN as well as Gradient Descent Boosting, Random Forest, and SVM, and stack those algorithms to predict S&P500 volatility. They show that this approach outperforms other common benchmarks.[37]

In MV optimization, two SL approaches are typically used to estimate the covariance matrix. The first is the graphical LASSO, which is a static estimator that derives a sparse version of the covariance matrix. The second is a dynamic estimator based on the DCC GARCH, already known from time series econometrics (see Guidolin & Pedio (2018)). The graphical LASSO algorithm, proposed by Friedman et al. (2008), estimates the sparse precision matrix (the inverse of the covariance matrix), using the l_1 penalty to enforce sparsity. The graphical LASSO problem consists of the maximization of the following penalized log likelihood:

$$\ln\det(\mathbf{\Sigma}_t^{-1}) - tr(\mathbf{S}_t\mathbf{\Sigma}_t^{-1}) - \rho\left\|\mathbf{\Sigma}_t^{-1}\right\|_1 \tag{36}$$

Here $\mathbf{S}_t$ is the sample covariance matrix, $\rho \geq 0$ is a tuning parameter controlling the amount of regularization, and $\left\|\mathbf{\Sigma}_t^{-1}\right\|_1$ is the l_1 norm of $\mathbf{\Sigma}_t^{-1}$ is the sum of the absolute value of the elements of the inverse covariance matrix. The penalty parameter is chosen by a validation approach to make the value of $\ln\det(\mathbf{\Sigma}_{1,t}^{-1}) - tr(\mathbf{S}_{2,t}\mathbf{\Sigma}_{1,t}^{-1})$ large, where $\mathbf{\Sigma}_{1,t}$ is the covariance matrix estimated using the training set and $\mathbf{\Sigma}_{2,t}$ is the covariance estimated over the validation set.[38] As an alternative to graphical LASSO, one can consider portfolios based on the linear shrinkage estimator of the covariance matrix proposed by Ledoit & Wolf (2004). Specifically, the sample covariance matrix is shrunk towards a one-parameter matrix, where all the variances are the same and all the covariances are zero. The weight given to the target matrix is typically determined by a shrinkage parameter. The linear shrinkage estimator has been shown to perform better than the sample covariance matrix estimator in a variety of settings, especially when the number of variables is large relative to the sample size or when the variables are highly correlated. The estimator can improve the accuracy of statistical inference and prediction and reduce the risk of over-fitting.

Often, classical portfolio estimates are optimal in sample (on the training set) but perform poorly out of sample (on the test set). One way to deal with this problem is to drop forecasts of expected returns altogether and, for instance, by simply weighting the assets by the inverse of their predicted volatility, what is called the *risk parity approach* (RP). The problem is that both RP and MV require the inversion of a positive-definite covariance matrix. An example is a study by de Prado (2016)

[36] When the forecasts of one model encompass those from another, it means that the first model's predictions are more inclusive and cover a wider range of possible outcomes than the second model's predictions.

[37] Mourtas & Katsikis (2022) propose a continuous-time Black-Litterman portfolio optimization problem, which is solved using a novel ANN architecture to suggest that it is a promising alternative to traditional approaches in portfolio optimization.

[38] The validation sample approach is a technique to evaluate the performance of a model on an independent dataset that was not used during the model's training phase. This approach involves splitting the available data into three sets: training, validation, and testing. The training set is used to train the model, while the validation set is used to evaluate the model's performance during training and adjust the model's hyperparameters. The testing set is used to evaluate the final performance of the trained model.

that replaces the structure of return covariances among assets with a tree structure using hierarchical cluster analysis. Hierarchical risk parity (HRP) works by grouping similar investments into clusters based on a distance metric. HRP leverages graph theory to improve portfolio diversification by re-arranging the correlation matrix into a hierarchical structure, mitigating the impact of estimation error. The rows and columns of the covariance matrix are reorganized so that the largest values lie along the diagonal. Finally, the allocations are split through recursive bisections of the reordered covariance matrix. This means that one starts by defining a distance measure between assets, ranging from zero to one, $d_{i,j}$, after which one clusters the pair of columns (i^*, j^*) together such that $(i^*, j^*) = argmin_{(i,j)} \{d_{i,j}\}$ and $i \neq j$. The next step is to update $\{d_{i,j}\}$ with the new cluster and recursively repeat until all $N-1$ clusters are formed. Then one places the correlated assets close together and the uncorrelated ones further apart and carries out a top-down allocation. Although the information processed by this method is the same as in the traditional MV paradigm, it requires fewer estimates and, therefore, delivers better stability and robustness. Lopez-Lira & Tang (2023) reports that the proposed method to build a minimum variance portfolio yields a 31% improvement in the OOS Sharpe ratio compared to the standard approach. Jaeger et al. (2021) use interpretable machine learning techniques (XGBoost, a gradient regression tree boosting algorithm known for its speed and accuracy) on a multi-asset futures portfolio bootstrapped to generate a range of data sets to show that HRP exhibits superior risk-adjusted performance (in terms of the Calmar ratio, a path-dependent performance measure emphasizing both returns and drawdown mitigation) to standard RP based on the full covariance matrix, especially in scenarios where the asset universe exhibits high drawdown heterogeneity. A Shapley value analysis reveals that univariate drawdown measures are key drivers of HRP's success and can be considerably heterogeneous.[39] Edirisinghe & Jeong (2024) have explored a new methodology for creating efficient portfolios that addresses limitations of traditional MV, particularly in high-dimensional small-sample (HDSS) scenarios, when the selection concerns many assets for which only relatively short time series are available. HDSS leads to difficulties in accurately estimating asset return parameters: small sample sizes amplify the negative impact of parameter estimation errors on portfolio performance and MV can lead to unrealistic short positions, exposing the portfolio to high risk. Edirisinghe & Jeong (2024)'s Elastic Net approach uses constraints on the absolute and Euclidean norms of asset positions to control the number of investments and short portfolio exposure; sparsity and leverage restrictions are imposed as probabilistic constraints. By calibrating these norm parameters through a cross-validation process, evaluation using SP 500 Index stocks demonstrates significant genuine OOS performance improvement over a 2021-2022 sample compared to the standard MV model, also those based on cardinality constraints using binary variables, achieving desired sparsity and leverage levels. Interestingly, the spread in performance grows when the stock pool grows and as the imputed risk aversion coefficient increases; such results come from a reduction in realized risk in the face of a minor reduction in realized mean returns.

[39] While classical RP (here optimized equal risk contribution) works well when correlations are stable (particularly negative between stocks and bonds), HRP is robust to realistic time-varying correlations.

3.2 Automatic portfolio optimization

A separate strand of literature consists of papers that have attempted to use ML directly to obtain optimal portfolio weights. While the papers listed in Section 3.1 aim at improving the quality–for instance, forecasting accuracy–that enter a standard, classical MV problem (hence, predicted means, variances, and/or covariances), in this case, the goal is to use ML to directly infer weights from the data and hence:

- skip the rigid distinction between predicted performances (e.g., mean excess returns) and risk (e.g., the variance of a portfolio) and
- generalize the MV framework to more general asset allocation schemes (e.g., to higher-order-moments or to pure rule-based portfolio policies).

The first approach to direct optimization of portfolio weights using ML involves typical SL algorithms, such as ANNs or SVMs. In this case, the first step is to define a set of input features, such as historical asset returns, market indices, and macroeconomic indicators. The output of the model is a set of predicted portfolio weights, which can be obtained by training the model on the data. During the training phase, the model is fed with the features and corresponding portfolio weights, and the model learns to map the input features to the output weights using a loss function. The loss function can be designed to optimize under alternative investment objectives, such as maximum Sharpe ratio or minimum Value-at-Risk (VaR). Once the model is trained, it can be used to predict portfolio weights based on new market data in real-time. The model's output can be adjusted dynamically as new data becomes available, allowing the portfolio to adapt to changing market conditions.

ANNs are a common approach to achieving these goals owing to their flexibility in accommodating complex rules and constraints. Chapados & Bengio (2001), for example, used ANNs to learn the optimal asset allocation subject to VaR constraints. They also showcase the effective use of *committee methods* to systematize the choice of hyperparameters during ANN training.[40] Similarly, Yu et al. (2008) went beyond the first two moments of portfolio returns by training an ANN to build mean-variance-skewness efficient portfolios.Zimmermann et al. (2001) incorporated Black and Litterman's celebrated asset allocation framework into an ANN model to augment the set of inputs to include views about future returns. Tsang & Wong (2020) have investigated a deep-learning solution to high-dimensional multi-period asset allocation problems with constraints. They propose a deep ANN architecture to describe the underlying control process. The feedback control function (the mechanism designed to adjust the system's actions based on the feedback it receives from the environment) is determined solely by the network parameters. In this way, the multi-period problem is linked to the training of the deep ANN, which can be efficiently tackled by standard optimization techniques. Bradrania & Pirayesh Neghab (2022) have proposed an ANN approach to conditional asset allocation, which directly relates state variables to portfolio weights, and find that the proposed approach generates better performance when compared to traditional methods. Du (2022) present

[40]Committee methods involve creating multiple ANNs, training each one with a different subset of the available data, and then combining the outputs of the individual ANNs to produce a final prediction. Therefore, this represents an ensemble approach that can improve the accuracy and robustness of the prediction compared to using a single ANN. Bagging and boosting are related to committee methods but require training–of identical models–on different sub-samples.

an MV model constructed using stationary portfolios composed of cointegrated stocks. The expected returns of this new model are predicted by using ML models, such as SVM, RFs, and attention-based long short-term memory (LSTM) networks. [41] They show that attention-based LSTM networks can more accurately model long-short portfolio returns using technical indicators and lagged returns. They successfully select pairs of cointegrated stocks to construct a more profitable MV portfolio.

A widespread challenge, especially in the area of AI-based portfolio decisions, is the interpretability of results. The issue is particularly relevant to ANNs and to the task of building a predictive model for active portfolio management. This occurs because ANNs consist of many interconnected layers of neurons, making it difficult to determine how specific inputs are being processed and transformed into specific outputs. This lack of transparency can make it challenging to identify the features that the model is using to make predictions and to explain how the model arrives at a particular decision. Moreover, ANNs can be easily influenced by small changes in the input data, leading to hardly explainable changes in the output. This instability leads to difficulties in understanding how the model will perform in different situations or with different inputs. In the context of a day-to-day investment process, the capacity to audit previous decisions is essential to portfolio managers. Therefore, models that enable a clear attribution of risk and return to individual components of the model are highly valued. This lack of transparency of ANNs can be problematic to money managers who require a comprehensive understanding of the decision-making process to assess the model's robustness, detect biases, and identify potential areas of improvement. This issue has spurred a series of attempts (e.g., Gu et al. (2020)) to use recent advances in the ML literature to interpret the importance of individual features. Dixon et al. (2019) explore ways to interpret ANNs statistically by using confidence intervals and by ranking the importance of input variables and interaction effects. Section 9 returns on these issues.

3.3 Reinforcement learning

One approach that has been gaining traction in academic research as well as considerable media coverage (see, e.g., Inc. (2017), The Economist (2022)) is the use of RL algorithms, which enable the optimization of a policy function for selecting portfolio weights without requiring the explicit calculation of expected returns, variances, and higher-order moments. Conceptually, RL implies a paradigm by which one stops being solely focused on the predictive power of a model for the mean and variance of asset returns (which are in any case just auxiliary tasks), but rather on the direct optimization of portfolio decisions, i.e., the primary goal. A wise combination of SL and RL algorithms may lead to indirectly re-creating well-known trading strategies without having to pre-define them. For example, the gradient step that leads the machine agent to buy more of what did the best in the past contributes to discovering a momentum investing strategy but however will be entirely

[41] An attention-based LSTM network is a type of recurrent neural network (RNN) that incorporates an attention mechanism to selectively focus on specific parts of the input sequence when making predictions. In a standard LSTM network, the hidden state of the network is computed at each time step using the input at that time step and the previous hidden state. This hidden state is then used to make a prediction at each time step. In an attention-based LSTM network, the additional mechanism allows the model to selectively focus on the most relevant parts of the input sequence when making predictions, which can improve the model's performance on tasks that involve long input sequences.

data-driven.

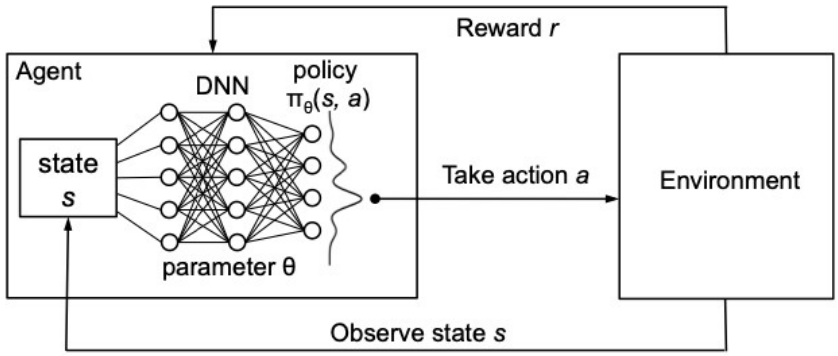

Figure 3: Flow diagram illustrating reinforcement learning

In finance, RL consists of modelling an agent that learns how to take actions in an environment to maximize some notion of cumulative reward. The Markov decision process (MDP) paradigm, which describes decision-making as a series of states, actions, and rewards, serves as the foundation for RL algorithms. An agent receives as input the current state S_t and is asked to choose an action A_t to receive a reward R_{t+1}; the information on the reward can be used to identify the next optimal action, A_{t+1}, given the new state S_{t+1}, etc. 3 shows the flow of information characterizing RL. In the context of portfolio optimization, the state space could include factors such as historical price data, market indices, and economic indicators, while the action space could represent the portfolio weights assigned to different assets. The reward function could be defined based on the portfolio's performance relative to a benchmark or based on other criteria, such as maximum Sharpe ratio, minimum risk, maximum drawdown, and minimum Value-at-Risk measures. Next, an RL algorithm is trained using historical data to learn an optimal policy for selecting portfolio weights. The algorithm could adopt a model-based approach that learns a transition model for the state space or a model-free approach that directly learns the policy through trial and error. The weights selected by the policy can be adjusted dynamically as new data becomes available, allowing the portfolio to adapt to changing market conditions.

While supervised learning (SL) aims to predict specific outcomes, such as the change in the price of securities, RL optimizes decision-making in dynamic, uncertain environments. By using RL to devise a trading strategy, portfolio managers can make better-informed decisions, streamline the process, and reduce human error. RL also conveniently accounts for market frictions, such as transaction costs and liquidity constraints, as these can all be simultaneously incorporated into the calculation of the actions' rewards. Yet, despite (or because of) its ability to compute (near-)optimal portfolio weights with less human involvement, RL takes longer to train and is computationally intensive. Under RL, an agent learns by interacting with the environment and receiving rewards or penalties. Feedback is used to update the agent's policy to maximize her long-term rewards. However, the agent's decisions affect the environment, resulting in sparse, noisy, and delayed feedback that can hinder accurate learning. Therefore, RL algorithms tend to require a large amount of data to be effective. This can be particularly true in complex environments with many possible actions and state transitions, when

the decision maker may need to explore a wide range of actions and states before finding an optimal policy.[42] For instance,Zhang et al. (2020) suggest an RL approach to directly optimize a portfolio's Sharpe ratio. Clearly, the advantage of this approach is that it delivers optimal portfolio weights by updating model parameters through gradient ascent without the need for the forecasting step required by classical MV.

Interestingly, RL's convergence to an optimal value and, hence, to a policy solving a portfolio problem is not guaranteed. The famous Bellman updating algorithm can only guarantee the achievement of the optimal value if every state is visited an infinite number of times and every action is tried an infinite amount of times within each state, so essentially, it is never guaranteed in practice. Luckily, in applied portfolio decisions, we do not need a truly optimal value: approximate optimality is often acceptable. The key issue is, instead, that the sample size needed to obtain a good level of approximate optimality increases with the size of the state and action space. Furthermore, without any additional assumptions, there is no better way to approach the problem than to explore the space randomly, so progress at first is modest and slow. Finally, the use of RL in real-world financial decisions can be challenging due to the cost and complexity of testing RL algorithms in simulated environments, particularly when accurate feedback is necessary. In such cases, it may be necessary to revert to the real environment, which can be prohibitively expensive and time-consuming as it may require deploying the algorithm in the actual system, collecting data, and waiting for the results from genuine OOS experiments.

Fischer (2018) has categorized RL algorithms as *critic (value function)-only*, *actor-only*, and *actor-critic*; this distinction appears to progressively become important in portfolio applications. Critic-only algorithms (see Kolm & Ritter (2019)) are the most extensively studied RL approaches. They aim at learning value functions that produce the expected reward from each action at each point in time without explicitly learning a policy; this allows the algorithm to choose the action with the best outcome. These algorithms are often unable to deal with multiple asset problems, they can only consider discrete action spaces (e.g., buy, sell) and are not commonly used in portfolio optimization, as the domain of the problem typically involves a combination of both RL and optimization techniques, such as MV optimization, which require an explicit modeling of the policy function. Actor-only (see Jiang et al. (2017), Chaouki et al. (2020)) algorithms learn to map states (e.g., market conditions, see Snow (2020*b*)) to actions (the actor) directly without explicitly estimating the value function (the critic). These algorithms rely on a parameterized policy, which can be proxied by an ANN, to output the best action to take in a given state. These models are more transparent and can accommodate continuous action spaces (e.g., portfolio weights). Actor-only RL algorithms are generally used in problems where it is difficult or impossible to estimate the value function accurately but where the optimal policy can still be approximated. However, they require the reward function (e.g., a portfolio's Sharpe ratio, an investor's utility, ect.) to be differentiable when the action space is continuous. In particular, the Deterministic Policy Gradient (DPG) algorithm is a type of actor-only RL algorithm that is commonly used in portfolio optimization. Chaouki et al. (2020) address the question of whether the DPG algorithm can learn optimal trading strategies for a dynamic portfolio allocation problem

[42]Furthermore, the use of function approximation techniques, such as deep neural networks, can also increase the amount of data needed.

and report encouraging results: DPG can learn optimal trading strategies for the problem, provided that it can be formulated as a Markov decision process.

Actor-critic approaches combine the benefits of both value-based and policy-based methods and comprise two steps. The first (the actor model) determines the action depending on the current state, and the second (the critic model) evaluates the performance of the chosen action. The actor is trained to output a probability distribution over possible actions, which is then used to select an action. To select an action, one typically samples from the probability distribution produced by the actor (in some implementations, the action with the highest probability can be selected directly, but this approach reduces exploration). The critic, on the other hand, evaluates the performance of the actor by estimating the value function of the current policy and allows one to iteratively adjust the actor so that it maximizes the expected reward produced by the critic model. Fischer (2018) reviews a large number of studies related to each of these approaches—critic-only, actor-only, and actor-critic—and concludes that there is no clear winner in terms of performance in a general perspective. Garc'ıa-Galicia et al. (2019) provide an RL solution to the portfolio problem based on an actor/critic architecture in which the market is characterized by a restriction called transaction cost, involving time penalization.

The application of RL to portfolio construction is not a new concept, as it dates back at least to Moody et al. (1998). However, recent research has explored larger samples of stocks and conducted more extensive analyses, resulting in a better understanding of the potential benefits of RL. Some of the recent studies include Wang & Zhou (2020), Zhang et al. (2020). They focus on the US financial markets and report yearly Sharpe ratios between 0.75 and 5.5, both higher than the Sharpe ratio of the S&P 500. Almahdi & Yang (2017) and Almahdi & Yang (2019) propose to use a combination of recurrent RL and a particle swarm algorithm in a constrained asset allocation problem.[43] Using S&P100 index stocks, they show such a system with a Calmar ratio-based objective function yields a better efficient frontier than the Sharpe ratio and than MV portfolios.[44] Bertoluzzo & Corazza (2012) discuss two different RL approaches, the Temporal Difference and the Kernel-based RL, and take into account not only the usual buy and sell signals but also a third signal, a stay-out-from-the-market one. Among the recent studies, Cong et al. (2020) stands out for two methodological innovations. First, they use encoders (discussed below) to extract features from a list of technical and fundamental variables. This approach is particularly useful for reducing the dimensionality of otherwise high-dimensional panel data while preserving complex nonlinear and interaction effects. Second, they use surrogate modeling and polynomial sensitivity analysis (a technique to assess the impact of varying parameters or hyperparameters on the performance of an algorithm) to interpret the RL algorithm's outputs and to uncover the model's drivers. Overall, the direct optimization of portfolio weights using RL provides a powerful tool for portfolio managers, enabling them to make informed decisions based

[43]Particle Swarm Optimization (PSO) is a computational method of evolutionary type inspired by social behavior observed in bird flocking or fish schooling. It is a stochastic optimization technique used for solving optimization problems based on the idea of mimicking the social behavior of organisms within a group to iteratively search for optimal solutions in a multidimensional space. In PSO, a group of particles moves around the search space, where each particle represents a potential solution to the optimization problem.

[44]The Calmar Ratio is a risk-adjusted performance measure calculated by dividing the mean return of an investment by its maximum draw-down over a certain time period.

on real-time data and adapt their strategies to changing market conditions. However, care must be taken to ensure that the chosen reward function aligns with the desired investment objectives and that the algorithm is properly calibrated to avoid over-fitting. With reference to portfolio choice modelling, it is likely that SL will still rule the pack in the foreseeable future (see Snow (2020*a*)).

A related and yet different strand of AI applied to portfolio management is based on *evolutionary algorithms (EAs)*, which are optimization techniques inspired by natural selection, see, e.g., Metaxiotis & Liagkouras (2012). EAs have been used to search for an optimal portfolio by evolving a population of candidate solutions over several generations. EAs can handle complex, nonlinear, and non-convex problems, which are common in finance. EAs can incorporate various constraints and objectives, such as transaction costs and risk aversion. However, EAs are different from ANNs, although they entertain a tight connection with them related to their power to handle complex and high-dimensional data. RL can be used to predict the fitness of candidate solutions in an EA, while EAs can be used to optimize the hyperparameters of a RL model. This approach, known as *neuroevolution*, combines the strengths of EAs and RL to create more efficient and accurate models. Branke et al. (2009) show that the EA approach can be used to incorporate complex rules, such as constraints on the number of assets in the portfolio and minimum holding thresholds. Skolpadungket et al. (2016) report significant improvements in Sharpe ratios by incorporating model risk in a portfolio construction framework through EAs.

3.4 Deep learning

Occasionally, in empirical finance, there has been some confusion and overlap between RL and the so-called "deep learning" approach. Deep learning is a set of ML algorithms that use complex, multi-layer ANNs to solve complex problems.[45] This approach has been highly successful in tasks such as image and speech recognition, natural language processing, and autonomous driving. The key difference between deep learning and RL is their approach to learning. Deep learning algorithms learn to represent and analyze data, while RL algorithms learn how to best make decisions based on rewards and punishments.[46] However, in contrast to traditional RL algorithms (which often use simpler, hand-crafted representations of the policy or value function, which may not be suitable to complex environments), deep learning and RL can be used jointly, an approach called deep RL, to combine the representational power of deep ANNs with the decision-making capabilities of RL.

Deep RL (DRL) first gained attention in the finance sector for hedging tasks. JP Morgan, through the work of Buehler et al. (2019), developed a DRL-based model known as deep hedging, which they successfully applied to actual operations. This innovation significantly influenced numerous financial institutions. Recently, they further developed and announced the second version of their deep hedging

[45] A deep neural network (DNN) is a type of neural network that has multiple layers of interconnected nodes (or neurons) between the input and output layers. Each layer of nodes learns to represent increasingly abstract and complex features of the input data. The main difference between a DNN and a shallow neural network (SNN) is the number of layers. SNNs typically have one or two hidden layers, while DNNs may have many more hidden layers. This allows DNNs to learn more complex representations of the data. Another key difference is that DNNs require significantly more data and computational resources to train compared to SNNs.

[46] This has been recently emphasized by Ngo et al. (2023) whose results suggest that RL consistently outperforms established methods and benchmarks—including deep ANNs—even when using a very similar degree of diversification in portfolio construction and the same input data.

model (see Du et al. (2020) for a comprehensive review of derivative hedging using RL). Benhamou et al. (2021) have recently stressed that DRL has reached an unprecedented level of effectiveness when applied to complex tasks like game solving and autonomous driving.[47] However, it remains an open question whether DRL can reach "super human levels". Benhamou and co-authors have showcased state-of-the-art DRL methods for selecting portfolios based on a final DNN concatenating three individual networks using layers of convolutions to reduce the network's complexity. They show that these DRLs can over-perform traditional portfolio methods. Aboussalah & Lee (2020) address the challenge of continuous action and multi-dimensional state spaces and propose a so-called Stacked Deep Dynamic Recurrent RL architecture to construct real-time optimal portfolios. Their system is trained and tested in an online manner for 20 successive rounds with data for ten selected stocks from different sectors of the S&P 500 between January 1, 2013 and July 31, 2017. Betancourt & Chen (2021) propose a novel portfolio optimization method based on DRL applied to markets under a dynamically evolving asset menu.[48] Their architecture considers all assets in the market, and automatically adapts when new ones are introduced, making their method more efficient than previous approaches. Their tests on cryptocurrency data outperform state-of-the-art asset allocation methods. Lastly, Jang & Seong (2023) investigate a novel DRL approach that combines modern portfolio theory and deep learning to show that their method outperforms state-of-the-art algorithms in terms of realized OOS Sharpe ratio, average annualized returns, and maximum drawdown. As highlighted by Sezer & Ozbayoglu (2018) and Tsantekidis et al. (2020), in finance, convolutional NNs (CNNs) can be adapted to analyze time-series data by treating sequences of price movements or financial indicators as images. For example, they can process historical price data or technical indicators as two-dimensional arrays where one dimension represents time and the other represents different features or metrics. Their papers shows that CNNs can generate trading signals for buying, selling, or holding assets based on historical data. Sezer & Ozbayoglu (2018) found that CNNs could identify profitable trading opportunities more effectively than the buy-and-hold strategy and other trading algorithms and may offer a more robust approach to trading by capturing complex nonlinear relationships in the data that traditional models often miss.

3.5 Autoencoders

An autoencoder is a type of ANN designed to learn a compressed representation or encoding of some input data. The network is composed of two parts: an encoder and a decoder. The encoder takes the input data and maps it to a lower-dimensional latent space representation, while the decoder takes the latent space representation and maps it back to the original input space. The goal of the autoencoder is to learn an encoding that is as close as possible to the original input whilst being as compact as possible. To achieve this, the autoencoder is trained using an USL approach, where the loss function

[47] For example, in 2015, DeepMind AlphaGo's algorithm defeated the world champion in the ancient Chinese board game, Go. This was a major milestone in the field of DRL because Go is an incredibly complex game; its complexity arises from the vast number of possible moves and strategies that a player can choose from. Since then, other DRL algorithms have been able to solve complex games such as chess, shogi, and even video games like Atari games. In 2018, Alphabet's Waymo self-driving cars drove 1.2 million miles in California, with human intervention necessary only every 11,000 miles.

[48] This problem is especially important in cryptocurrency markets, which support the trading of hundreds of assets, with new ones being added every month.

is typically based on the reconstruction error between the original input and the decoded output. During training, the encoder and decoder parameters are updated to minimize this reconstruction error. Autoencoders can be used for a variety of tasks, including data compression, anomaly detection, and feature extraction. Figure 4 shows a schematic representation of an autoencoder algorithm used in the estimation of a multi-factor asset pricing relationship. In asset allocation, autoencoders are used to extract relevant features from financial and macroeconomic variables that can be used to optimize portfolios. Autoencoders are often used in deep learning because they are able to learn a hierarchical representation of input data, i.e., multiple levels of abstraction in the data, with each layer capturing increasingly complex features.[49]

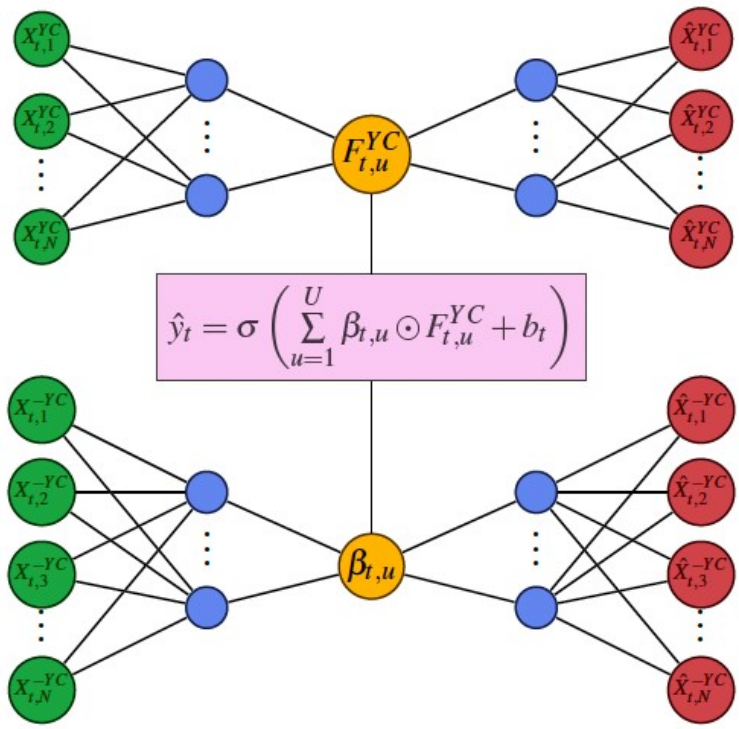

Figure 4: Schematic representation of an auto-encoder algorithm (in the diagram, $\boldsymbol{X}^{YC}$ represents any type of variables that may explain returns, $\mathbf{F}$ a lower-dimensional (factor-) representation, $\boldsymbol{X}^{-YC}$ is any type of variables that drives the time-varying factor-exposure, $\boldsymbol{\beta}$ the time-varying factor loadings, and $\sigma(\cdot)$ any type of linear or non-linear activation function)

In portfolio management, auto-encoders have been popularized by Heaton et al. (2017), who describe a procedure for data-driven portfolio selection consisting of four steps. The first step, the auto-encoding step, involves fitting historical returns. The goal of the second step (termed the decode step) is to find a portfolio map to achieve a pre-specified goal. Third, an out-of-sample validation step is used to tune the hyperparameters and, particularly, to optimize the amount of regularization

[49] Gu et al. (2021) have used an autoenconding ANN to model individual stock returns allowing the information in a set of covariates to guide dimensionality reduction. The model allows stock characteristic covariates to have nonlinear and interactive effects with factor exposures. Uddin & Yu (2020) have used an autoencoder to generate hidden features from daily returns of all stocks in an index. They use these latent factors to identify representative and non-representative stocks in an index, using them as a measure of how representative the stocks are in terms of the norm of the difference between the original and reconstructed data from the latent, encoded factors.

needed in the first two steps. This leads to the generation of an efficient frontier that can be used, in a fourth step, for model selection. **?** develop a portfolio framework based on a DRL framework called DeepBreath, that combines a restricted stacked autoencoder and a convolutional ANN. The autoencoder is employed to achieve dimensionality reduction and feature selection, thus ensuring that only the most informative features are retained; the ANN is then used to learn and implement the investment policy. Their results show that the average returns achieved outperform current expert investment strategies while minimizing risk.

4 Robo-advising in the wealth management industry

Recently, a new type of automated asset management advisory firms, often called *robo-advisors* (henceforth, RAs), has grown in importance. These firms counsel individual clients on their optimal portfolio using ML algorithms and digital platforms. In turn, these algorithms use investor characteristics such as age, net income, and assessments of individual risk aversion to recommend suitable asset allocations. Several robo-advisors are fully automated. These types of RAs are usually less costly than traditional wealth managers and thus oriented towards the retail market. Nevertheless, other RAs exist that offer a hybrid system in which clients can benefit from human interactions, albeit limited. These RAs charge higher fees, but are still cheaper than traditional human advisors.

Automated, algorithmic financial guidance has unquestionably the strength to disrupt the traditional wealth management industry's distribution model. This model has often been criticized for its high costs, limited scalability, and excessive subjectivity.[50] Two catalysts are at work in supporting the growth of this new strand of *fintech* development, at least until recently.[51] Initially, the near zero interest rates prevailing between 2009 and 2020 made it increasingly challenging to rationalize the exorbitant fees commonly associated with the financial planning sector. Furthermore, in the United States, there has been a notable transition from defined-benefit to defined-contribution plans. This shift has resulted in individual investors assuming greater accountability for their retirement investments. Additionally, investors with relatively modest wealth, unable to meet minimum account requirements, often found themselves stranded with limited access to conventional planning services. Such a drive towards heavier individual financial responsibilities has encouraged increasing experimentation with RAs.

Currently, RAs are widely recognized as one of the most important disruptive trends in the asset and wealth management industries. Assets under management (AuM) are expected to grow at an

[50] Brenner & Meyll (2020) provide evidence of a strong inverse relationship between using RAs and seeking human financial advice. The aggregate substitution effect of RAs is especially driven by investors who fear to be victimized by investment fraud. A variety of conflicts of interest plague the client-adviser relation. For instance, Hackethal et al. (2012) show that advised accounts perform worse than unadvised accounts largely because the former trade more often, producing commissions for advisers at the expense of investors. Linnainmaa et al. (2021) show that advisers make the same mistakes in their investment accounts as they do in their clients' accounts and hence transmit their own behavioral biases to clients. These results cast doubts on whether traditional financial advisers can create value for the investors whose accounts they manage.

[51] Fintech refers to the application of innovative technological solutions to improve and automate the delivery of financial services. This interdisciplinary field investigates a wide range of activities, including but not limited to online banking, mobile payment solutions, cryptocurrencies, peer-to-peer lending, robo-advisers, and algorithmic trading, see, e.g., Goldstein et al. (2019).

annual rate of 7% between 2024 and 2028, resulting in a projected total amount of USD 2.33tn by 2028 (see https://www.statista.com/ outlook/dmo/fintech/digital-investment/robo-advisors/worldwide). In recent years, the wealth management sector has witnessed the emergence of a new breed of clientele. These clients are tech-savvy, highly educated, inclined towards maintaining active and continuous oversight of their investments. They also tend to base their decisions on information from various sources, primarily online, rather than relying solely on financial advisors.[52]

The main unifying value proposition of RAs is to provide cheap access to diversified beta, especially through a focus on passive investing via exchange-traded funds (ETF). Uhl & Rohner (2018) demonstrate that investors utilizing RAs enjoy annual savings of more than 4% when considering both direct and indirect costs in comparison to conventional alternatives like advisory fund services offered by banks. Moreover, RAs typically require minimal initial deposits, often as low as USD 10, and levy cost-effective fees, typically hovering around 0.25% annually. In contrast, traditional financial advisory services in the United States entail annual fees amounting to 0.7% of AuM, while in the United Kingdom, these fees are approximately 1.25%. Nonetheless, the blanket term "robo-advising" hides a variety of different models that differ in several dimensions, four of which are:

1. Personalization of the advice, at least in principle, as discussed below.
2. Involvement of the investor in financial planning (most RAs usually ask investors to approve every single trading decision before it is executed); nonetheless, there are also RAs tailored to long-term investment horizons. These platforms not only generate financial plans and strategies automatically but also execute trades on behalf of investors without manual intervention. Initially, these RAs seek approval for an initial plan. However, once the plan gains approval, they take charge of conducting trades autonomously, requiring no further input from the investor.[53]
3. Investors' discretion to deviate from the automated advice. RAs offering flexibility enable investors to adjust the allocations suggested by the algorithm. In some instances, investors can also decide whether to execute the trades recommended by the algorithm. Additionally, investors have the option to incorporate stocks and other assets into their financial plan that may not align with the RAs' recommendations. In such cases, the robo-advisor will optimize the portfolio by considering a blend of recommended assets and those picked by the investor.
4. The presence of any form of human interaction. Numerous RAs, exemplified by platforms like Wealthfront and Betterment, operate as fully automated systems devoid of human advisory access. In contrast, certain RAs targeting a wealthier and more mature client base adopt a hybrid approach. While the algorithm predominantly handles the intricate task of the portfolio allocation, human advisors engage with investors during pivotal junctures, such as the initial sign-up process and when investors have inquiries.[54]

[52] According to a study by the Spectrem Group, the average age of investors who use the services of RAs is 48, compared with an average age of 62 for investors who do not use robo (see https://www.wealthadviser.co/2020/05/18/285696/ us-robo-advisory-industry-hit-usd1tn-value-year.) Nonetheless, D'Acunto et al. (2019), using data from an Indian RA, find that users and non-users are indistinguishable along several demographic characteristics, including their gender, age, and trading experience. At the same time, users have a larger amount of wealth invested in the brokerage house and are more sophisticated in the sense that they are more involved with the management of their portfolios.

[53] A better taxonomy would define such form of automated advice *robo-managers* rather than robo-advisors, because robo-managers manage wealth directly rather than providing advice about each step in the strategy.

[54] A specific form of hybdrid robo-advising that has had success is the so-called "super adviser". Super advisers are

Phoon & Koh (2018) document that typically, RAs raise the necessary inputs required for their investment algorithms via web-based questionnaires that users of their services need to answer. A rather typical set of questions will include (see Beketov et al. (2018)):

1. Do you invest for retirement or to generate/preserve general savings?
2. What is your age?
3. What is your net income after taxes?
4. What is your saving rate?
5. What is the value of your current (liquid) investments?
6. What is your previous experience with risky investments?
7. What is your investment horizon?
8. When deciding how to invest your money, do you worry more about maximizing gains, minimizing losses, or both equally?
9. In the event that your investment portfolio experienced a decline of $x\%$ within a particular period (month), would you choose to liquidate the entire portfolio, sell a portion of your investments, maintain your current holdings, or would you consider increasing your investments?

Two alternative methods exist to channel this information in the algorithms used by RAs (see Faloon & Scherer (2017)). The first method is *ad hoc* and, at least to some extent, driven more by the chances of legal action against the RA in case of large losses than by optimization of an investor's preferences subject to constraints. Under this approach, the questions are used in a scoring model that maps scores to points on a given efficient frontier (see Scherer & Lehner (2023)).[55] The second approach aligns with decision-theoretic models of portfolio selection. These models commonly encompass household assets and liabilities and also need additional data on lifetime career trajectories, retirement age, and an assumption about the characteristics of the investor's human capital. The nature of human capital influences both its correlation with financial returns and the discount rate employed to assess its present value. Optimized portfolios typically undergo rebalancing using trigger-based mechanisms, when rebalancing is triggered only when the portfolio deviates substantially in terms of risk from the recommended portfolio due to realized market performance.

There is little doubt that RAs encourage substantial diversification:[56] D'Acunto et al. (2019) report that investors holding less than ten stocks before using the RA optimizer increased the number of stocks they held and experienced sharp declines in portfolio volatility; for investors with ten or more stocks, the number of stocks held decreased after the RAs portfolio optimizer usage, suggesting

human financial advisers who make use of robo-advising to produce financial plans and strategies, but represent the only interface between users and their investment strategies and performance. Super advisers resemble traditional human advisers on the client side but are closer to RAs in charging lower fees as they do not need to spend time producing financial plans and strategies and implementing such strategies. Super advisers may offer a solution to the problem of the potential transmission of human advisers' own biases and misguided beliefs to their clients' portfolios (see Linnainmaa et al. (2021)).

[55] Tertilt & Scholz (2018) have investigated how different questionnaire answers relate to recommended equity allocations. They document that many questions have no impact on portfolio advice and, hence, must serve legal protection purposes.

[56] It is well known that participants in the stock market tend to be under-diversified. This results in investors bearing idiosyncratic risk, see Badarinza et al. (2016). Financial advising can potentially help mitigate under-diversification and help investors realize better outcomes (see Gennaioli et al. (2015) but, for many retail investors, traditional financial advisers are too costly.

the optimizer recommended closing positions in stocks that would be shorted had the short-sales constraint not been binding. Following the adoption of these strategies, investors held fewer stocks in their portfolios. Surprisingly, this shift did not result in increased portfolio volatility. The investment performance, when adjusted for market conditions, exhibited improvements, particularly among less diversified investors. Notably, these investors paid more attention to their portfolios, as evidenced by an increase in trading volume, which is reflected in the overall amount of trading fees incurred. Moreover, several classical behavioral biases, including the disposition effect (where investors tend to realize gains rather than losses), trend chasing, and the rank effect (investors' tendency to trade the best- and worst-performing stocks in their portfolios), became less pronounced after the implementation of the RAs' strategies, although they were not entirely eliminated.

With reference to data from the largest robo-advisor, owned by Vanguard, Rossi & Utkus (2020) reported that RAs reduced idiosyncratic risk by decreasing holdings in individual stocks and active mutual funds while increasing exposure to low-cost index mutual funds. Furthermore, they found that RAs eliminated investors' home bias and enhanced overall risk-adjusted performance, primarily by reducing portfolio risk.[57] The improved diversification comes, however, with no performance sacrificed. For instance, Rossi & Utkus (2020) report that at the 6-month horizon, the annualized Sharpe ratio average is 0.115 and exceeds that of the US market portfolio. This difference turns out statistically significant at the 1% level. In contrast, prior to resorting to robo-advisors, the differential in Sharpe ratios averages -0.014, also significantly different from zero at the 1% level. Importantly, these two figures are statistically significantly different from each other. Furthermore, their analysis reveals that the enhancement in performance is primarily driven by a reduction in portfolio risk rather than an increase in portfolio average returns. Using boosted regression tree methods, they also unveil a positive relationship between the cash share and the trading volume at the time of sign-up and the post-advice performance improvement. This suggests that investors who were actively trading or holding a substantial portion of their wealth in cash benefited more from the RA services.

Of course, generic RAs often provide 'canned advice', in the sense that personalized information about the individual investor is often not collected or used. Investors differ materially in their ability to take risks, although they share the same age, net income, and savings rate. For instance, RAs make little attempt to better understand the sources of retirement income (whether payoffs are to come from defined-benefit or defined-contribution plans, life insurance, etc.). The current practice of offering pre-calculated portfolios through web portals, embraced to prevent delays in presenting real-time solutions, may become impractical if RAs were to truly personalize their advise. RAs have also been reported to adopt rather holistic approaches when determining a client's risk aversion. This is somewhat surprising, given that risk aversion significantly influences optimal portfolio allocations. The challenge arises from the fact that risk aversion is a latent, unobservable variable. To address this, RAs can either seek to measure observable client characteristics highly correlated with risk aversion, such as responses to specific questions or psychometric questionnaires, or derive risk preferences from

[57]The percentage of wealth in index mutual funds almost doubles, from 47% to 83%. Investors' international diversification increases threefold: the percentage of wealth in international mutual funds increases from 10% to 32%. Of course, moving investors into index mutual funds translates into lower fees: average expense ratios are more than halved from 19 to 9 basis points. RAs also change investors' share in risky assets. It increases investors' bond-holdings from 24% to 40% and decreases investors' cash and money market mutual fund holdings from 22% to 1%.

observable experimental or investment decisions.[58]

Interestingly, customization should come at little marginal costs for a web-based platform. For instance, Scherer & Lehner (2023) conducted a study utilizing scraped data on portfolio recommendations from 16 German RAs, collectively representing approximately 78% of the assets in the German robo-advisory market. Their findings suggest that RAs offer only limited individualization in their recommendations. Crucial factors in contemporary portfolio selection, such as the extent and nature (beta exposure) to human capital or shadow assets shocks, receive minimal consideration. Instead, portfolio recommendations are said to often align with investor pre-conceptions or adhere to the regulator's perspectives on safe portfolio choices. This raises questions about the genuine economic benefits provided by RAs.[59] Scherer and Lehner conjecture that these choices are not made because of ignorance of the existing academic literature but for commercial reasons: complicated models that may deliver counter-intuitive solutions to the financially untrained client will not maximize revenues in a highly competitive market.

One recently developed ML solution that applies to robo-advising is to use *experience sampling*. The concept revolves around investors using simulations, whether historical or forward-looking, to gain insights into how a selected portfolio might perform over time and how end-of-period wealth could vary under various market conditions. Essentially, this approach can be likened to a fast-forwarding of capital markets and may serve as a viable substitute for accumulating market and investment experience. A collection of studies, exemplified by Kaufmann et al. (2013), provides extensive cross-validated evidence demonstrating that investors exposed to experience sampling tend to, on average, opt for riskier portfolios with lower instances of regret. These portfolios are less likely to undergo revisions following adverse return shocks when compared to those shaped by alternative risk profiling methods. This is particularly advantageous for the digital asset management advisory, where there is no direct interaction with client relationship officers. Modern ML techniques may play a pivotal role in simulating the experience sampling process, as highlighted by recent work such as Schultz & Maedche (2023).

Beketov et al. (2018) have analyzed 219 RAs worldwide and showed that Modern Portfolio Theory remains the main framework used in robo-advising, followed by Sample Portfolios and Constant Portfolio Weights.[60] They conducted an analysis of the web pages of these RA systems to gather information about their asset allocation methodologies. Beketov, Lehmann and Wittke's dataset encompassed RAs from 28 different countries, with 30% of these companies based in the USA, 20% in Germany, 14% in the UK, 9% in Switzerland, and the remaining 27% situated in other countries.

[58] Some robo-advising firms utilize sophisticated questionnaires designed by psychologists to uncover individual risk aversion. This results in investors being assigned a risk index score that resembles a normalized measure, somewhat akin to an IQ score, when compared across the population of participants.

[59] It must be acknowledged that also traditional financial advisors have a poor track record in terms of taking client characteristics into account. Foerster et al. (2017) find that only 12% of the cross-sectional variation in advice (across clients) arises from differences in client characteristics such as risk aversion, wealth, experience, occupation or time horizon. Mullainathan et al. (2012) show that advisors are systematically biased against passive investments and even ignore stated client preferences.

[60] Yet, information about the asset allocation methods was only available for 73 RA systems. Moreover, the true methodologies behind the various Sample Portfolios methods is unknown, and they may actually be rather sophisticated. However, it is a fact that Sample Portfolios-based RAs had limited capacity for individualization (i.e., they could not offer portfolios that were unique to each client).

They report that assets under management (AuM) tend to be higher for systems employing newer and more sophisticated approaches, such as full-scale optimization. This poses a challenge because clients who invest through RAs are typically tech-savvy and expect both advanced and personalized digital services and methodologies, while Modern Portfolio Theory, which has been around for more than 65 years, may not fully resonate with the preferences of the new generation of investors. These investors consider state-of-the-art methods and technology as the standard for various digitized services (see also D'Acunto et al. (2019)). Nevertheless, advanced methods like Full-Scale Optimization and the Black-Litterman's model still attract substantial AuM volumes despite appearing relatively less frequently in Beketov et al.'s data. In contrast, simpler and more broadly defined methods, such as Sample Portfolios and Constant Portfolio Weights, are utilized more frequently but tend to attract relatively lower AuMs.

As discussed by Phoon & Koh (2018), leveraging data analytics and AI in the realm of robo-advising could yield significant benefits. It may enhance the alignment of the investment opportunities with clients' needs, ultimately leading to improved investment outcomes, including the optimization of behavioral-type preferences. Advanced analytical tools, like those powered by ML, have the potential to help clients steer clear of human judgmental errors and behavioral biases. The use of such algorithms can empower individual investors with enhanced capabilities, enabling them to compete more effectively with institutional investors. Moreover, AI-driven data chatbots can offer intuitive responses to common customer queries. This functionality may serve to liberate relationship managers, allowing them to concentrate on addressing more complex requests and promoting specialized products. By integrating chatbots into the RA ecosystem, it will be possible to expand the client base, catering to an additional market segment that embraces technology and social media.

5 ML in the ETF and smart beta spaces

ML has also gained significant interest in the ETF and passive strategies spaces. Passive investment strategies, such as ETFs, seek to track the performance of a particular index or asset class, rather than to outperform it. ML techniques can be used to optimize portfolio construction, enhance risk management, and improve asset allocation decisions, even when they concern strategies and portfolios characterized by a passive, index-tracking nature. In fact, according to Phoon & Koh (2018), RAs may even eventually replace traditional ETF fund managers altogether and deeply integrate themselves within the ETF landscape.

One area in which ML is applied ETF and passive strategies is in the selection of securities to include in a portfolio. ML algorithms can analyze large amounts of data, identify patterns and correlations, and make informed decisions about which securities to include. For example, ML can be used to screen securities based on various characteristics, such as price-to-earnings ratios, dividend yields, and market capitalization, and select those that are likely to provide the best risk-adjusted returns. To manage the short-term index tracking risks of typical ETF strategies, Fons et al. (2021) have proposed a novel smart allocation system based on a Feature Saliency HMM (Hidden Markov Model) algorithm that performs feature selection simultaneously with the training of the HMM to

improve regime identification.[61] They report results (up to a 60% increase in risk-adjusted returns annually) that emphasize model performance improvement with respect to portfolios built using full feature HMMs. Lee et al. (2022) use financial statement data of individual US stocks to train ANNs to predict the future performance of the individual stocks and use these predictions to construct (a portfolio of) ETFs. They report that their approach outperforms all classical benchmarks.

Index tracking is, therefore, an area of finance that could significantly benefit from an AI approach. An early example is Lowe (1994), in which the author applies feedforward ANNs to portfolio optimization under various constraints. Lowe argues that the proposed methodology is able to replicate the FTSE 100 index using a rather small subset of the constituents, thereby reducing transaction costs and allowing for higher efficiency. Among others, Ouyang et al. (2019) have proposed to apply deep ANNs and autoencoders to track index performance, and they introduce a dynamic weight calculation method to measure the direct effects of the stocks on the index. They show that their tracking method based on deep ANN combined with the capital asset pricing model (CAPM) leads to a smaller error compared to all non-ML benchmarks. Kim & Kim (2020) associate a deep latent representation of asset returns, obtained through a stacked autoencoder, with a benchmark index's return to identify the assets for inclusion in the tracking portfolio. Their results indicate that to improve the performance of previously proposed deep learning-based tracking, the deep latent representation needs to be learned in a strictly hierarchical manner. Bradrania & Pirayesh Neghab (2022) examine the stock selection step of the index tracking problem and propose an application of deep ANNs to select the best selection method based on the state of the market. Their empirical evidence shows that the approach outperforms standard cointegration techniques that ignore market regimes.[62]

The financial crisis of 2008 sparked a renewed interest in asset allocation strategies that are transparent and rules-based, leading to the emergence of smart beta as a trend among institutional investors (see, e.g., Gerakos et al. (2016). Smart beta strategies seek to deliver superior risk-adjusted returns by selecting and weighting stocks based on specific factors, such as value, momentum, and profitability, rather than market capitalization alone (see, e.g., Jacobs & Levy (2014)). While smart beta strategies have demonstrated strong performance in the long run, they can also suffer from severe short-term drawdowns and fluctuations in performance across market cycles and regimes (see, e.g., Chincoli & Guidolin (2017)). This is because they often imply sharp, considerable factor exposures compared to traditional market-cap weighted strategies, making them more sensitive to shifts in market sentiment and changes in economic conditions. As such, smart beta requires careful monitoring and management to avoid unwanted exposures to risks and to optimize factor weights. Therefore ML has increasingly been applied in the smart beta space in recent years. ML techniques can be used to identify and exploit patterns in the data that may not be captured by traditional quantitative models, which typically rely only on the linear relationships between variables captured by standard factor models. ML can then sharpen and robustify smart beta schemes by analyzing large amounts

[61] Index tracking is a popular strategy that aims to replicate the performance of a benchmark index, such as the S&P 500 or the Nasdaq Composite. The goal is to achieve a similar return to the index, with minimal deviation, called tracking error.

[62] This strand of literature remains distinct from papers that have attempted to build algorithmic ETF trading models that can produce abnormal returns by using ML, see e.g., Baek et al. (2020) and the references therein. These papers test trading models based on ETFs, which are not to be used by ETF fund managers.

of data, identifying nonlinear relationships between variables, and making more informed investment decisions based on these insights (see, e.g., Gu et al. (2021)). For example, ML can be used to identify stocks that have historically exhibited strong momentum or value characteristics in a regime-specific but decoupled fashion; this information can be used to construct a smart beta portfolio that is more likely to outperform the market.

Recently, the advantages of smart beta factor construction using ML have been shown by Feng et al. (2020), who have combined LASSO with classical Fama-MacBeth regressions to systematically evaluate the contribution of an additional factor to a set of pre-existing factors, finding that many factors proposed in the literature may fail to add much incremental value. Kelly et al. (2019) propose Instrumental PCA (IPCA), where factors are latent, and the time-varying loadings depend on stock characteristics to show that a small number of factors can explain the cross section of average returns more accurately than other leading, less parsimonious factor models.[63] Feng et al. (2018) build factors that maximize the fit to the cross-section of returns using a deep feed-forward ANN, while He et al. (2022) find that using reduced-rank regression outperforms popular factor models, such Fama and French's. Conlon et al. (2021) have examined the use of latent factors generated from SL and USL dimensionality reduction techniques for asset allocation within the framework of factor-based covariance matrices used to construct minimum-variance portfolios. They report evidence that factor-based covariance matrices tend to significantly reduce the risk of a portfolio consisting of individual stocks, and that using ML can lead to significant economic gains. In their work, the best-performing methods to generate the covariance matrix turn out to be ANN autoencoders and SPCA. Surprisingly, shallow learning outperforms deep learning, a fact that can be attributed to the small size of the data set and the low signal-to-noise ratio. Maguire et al. (2018) detail the construction of a monthly re-weighted portfolio involving two independent smart beta strategies; the first component is a long-short beta-neutral strategy derived from running an adaptive boosting classifier (AdaBoost) on a suite of momentum indicators; the second component is a minimized volatility portfolio which exploits the empirical fact that low-volatility stocks tend to yield higher risk-adjusted returns than high-volatility stocks. They find that the the combined leveraged strategy achieves a stunning Sharpe ratio of 0.96. Avramov et al. (2021) highlight two advantages of ML-driven strategies over the isolated exploitation of classical anomalies through regression methods: ML-driven strategies profit both from their long and their short positions and, unlike a number of well-established anomaly portfolios, do not suffer from a steep decline in their performance over the last 20 years. Moreover, they show that USL appears to perform better than SL methods. Zhang et al. (2022) introduce smart beta factors built on textual analysis, which include readability, tone and sentiment factors, similarity, semantic, uncertainty, accuracy, and popularity. For instance, they report that both a firm management's opinion about ongoing concerns and textual features of management discussion and analysis disclosure can predict whether the firm will face distress as early as three years prior to when a real bankruptcy happens.

Hsu et al. (2022) have recently presented results on building portfolios of smart beta strategies.

[63] Similarly, with reference to corporate bond returns, Kelly et al. (2023) employ IPCA to extract latent factors. They use 30 characteristics with predictive power already reported in the literature and find that the risk factors are dominated by option-adjusted bond spread, duration, bond volatility, equity momentum, and spread-to-distance-to-default ratio.

Smart beta products using common factors like value, low volatility, quality, and small cap experienced an underwhelming performance between 2005 and 2022. On average, long-only factor portfolios built from a wider set of global factors identified in the finance literature generated significantly positive excess returns across countries, suggesting that diversifying across many factors is more prudent than selecting a handful that have performed the best. Moreover, long-only portfolios built from expected returns extracted from their 87 factors using linear ridge and nonlinear ML (like GBM) generated larger and more statistically significant excess returns in nearly all countries.

Using data from ETF Global, Bartram et al. (2021) report that a number of active exchange-traded funds (ETFs) claim to use AI or ML in their investment strategies. Although these funds do not represent a significant share of the market, recently, their assets under management have grown rapidly. Their active risk seemed to be largely driven by factor exposures, which indicates that they tended to focus on smart beta bets. At 0.75%, their average management fees were slightly above the level typically charged by active equity ETFs. An equally weighted portfolio of all of these funds generated a Sharpe ratio of 0.88 and outperformed a broad market benchmark since inception. However, over their relatively short sample period, these funds failed to exhibit a significant alpha after accounting for the portfolio's exposure to a set of standard style factors, which confirms their role of smart beta packaging vehicles.

6 Natural language processing in portfolio management

Natural Language Processing (NLP) is a sub-field of artificial intelligence focusing on techniques enabling computers to analyze, model, and comprehend human language. NLP encompasses a variety of tasks, ranging from data collection and pre-processing to text representation, modeling, and performance evaluation. NLP has become increasingly popular in portfolio management due to its ability to extract valuable insights from both structured and unstructured data sources, often called alternative data (alt-data).[64] The body of literature concerning NLP is vast to the point that several surveys have been published on this subject (see, e.g., Das (2014), Fisher et al. (2016), Loughran & McDonald (2016)).

The methodological workflow of a typical NLP problem starts with the acquisition of relevant documents. These may be collected from numerous sources, both traditional and non-traditional ones. Traditional sources are exchanges, regulatory disclosures, economic releases and indicators. Non-traditional (alternative) sources are those created by individual users, business processes and sensors (Sun et al. (2022)) and include data from social media, consumer transaction electronic receipts, internet searches, and satellite images, to list a few. In general, the text gathered from these sources may appear in the form of electronic documents or be scraped directly from web pages. The collection of documents or population of texts collected represents the *corpus*, while a plurality of subsets of the corpus is referred to as the *corpora* (see Grimmer & Stewart (2013)).

During the pre-processing phase, the corpus is transformed into a manageable format suitable for

[64] Alt-data may be defined as any dataset that does not originate from exchanges, regulatory disclosures, or economic releases of official indicators.

the analysis to be conducted. This step involves representing the text as data with specific techniques, which may vary depending on the purpose of the analysis. Generally, the text is first segmented into sentences using full stops and question marks as delimiters. Sentences may then be further split into words. These two steps, known as sentence tokenization and word tokenization, respectively, allow the text to be viewed as a collection of single words, known as tokens. Other common practices include lowercasing, removing stop words, removing common words, stemming and lemmatization. Lowercasing refers to the practice of converting the text to lowercase, removing punctuation and converting plural words into singular words. Since common words often serve only grammatical functions and do not convey any additional meaning, they are frequently removed through the practice of removing stop words. [65] Also common words are discarded because they are deemed unlikely to be discriminating (e.g., routine and auxiliary verbs, like to do, to have or to be). Stemming is instead a process where the final parts of words are removed to reduce the total number of unique tokens in the data set. For instance, the three words "send", "sent" and "sending" will all become "send". In simpler terms, stemming involves mapping words related to the same concept to a single root. This step reduces the dimensionality of the corpus. Since the stemming process may return a word that is not necessarily meaningful, one may opt for lemmatization. Lemmatization is the process of mapping various forms of a word to its canonical or base form, known as the lemma. The lemmatizer will understand, for example, that "good" is the base form of "better". Notably, not all pre-processing steps are compulsory or appropriate in every application. The selection and order of these steps depend on the objectives of the analysis. Most of the steps are carried out through the pre-processing phase using regular expressions ("RegEx"), a programming language that identifies patterns through the documents and manipulates them. The formatting structure of the document is, therefore, paramount for this step to be performed correctly. However, when handling financial documents, the formatting structure can be intricate and inconsistent over time, leading to systematic biases (Loughran & McDonald (2016)).[66]

Commonly, the subsequent phase involves disregarding the order or arrangement of words in the documents. Documents are then treated as a collection of words, where the order and context does not affect the analyses. The most common way to represent a text is the Bag of Words (BoW) method of feature extraction. The goal of BoW is to transform the text into a numerical format, create vectors, and then assemble these vectors into a document term matrix (DTM). In this matrix, each document is represented as a vector that counts the frequency of each of the previously identified unique words, known as *unigrams*. However, it may be desirable to preserve the general meaning of sentences and not completely discard the order of words. To achieve this, more sophisticated approaches can be employed. Through the Bag of N-grams (BoN) approach, the word order can be retained through word pairs (bigrams), word triples (trigrams) or N-contiguous words. Indeed, the BoW and BoN approaches show some noticeable drawbacks. For instance, they treat individual words

[65] Stop words include, for example "a", "an", "the", "of", etc.

[66] Systematic biases may also arise from lowercasing. The aim of this step is to avoid discriminating words appearing at the start of the sentence from words appearing mid-sentence. To counter these biases, corpus annotation techniques like part-of-speech (POS) tagging, parsing, named entity recognition (NER), and co-referencing can be employed. Co-referencing refers to the task of determining when two or more expressions in a text refer to the same entity. Co-reference resolution may be challenging because it requires understanding the context of a text and resolving ambiguous references.

or groups of words as isolated units without considering the relationships among them. Additionally, these methods often result in sparse vector representations, where many entries in the vectors are zeros, leading to inefficient and less effective text representations. Furthermore, these representations create high-dimensional vectors, which can lead to computational inefficiencies and require significant resources, especially when dealing with large datasets. Lastly, two additional techniques are worth mentioning: term frequency-inverse document frequency (TF-IDF) and word embedding techniques. As opposed to the BoW and BoN approaches that treat all words as equally important, TF-IDF evaluates the relative importance of a word in a document relative to the entire corpus. Embedding techniques, instead, are low-dimensional representations of texts obtained using neural networks.

Also in the case of NLP approaches, the selection of a statistical method is influenced by the intended purpose of the analysis. When it comes to text classification and sentiment analysis, two common pathways can be pursued: dictionary-based approaches or machine-learning approaches. *Dictionary-based approaches* are widely used in accounting and finance (El-Haj et al. (2019)). They utilize the rate at which key words appear to categorize a document. However, applying dictionaries outside their intended contexts can lead to notable inaccuracies; see, e.g., the discussion in Loughran & McDonald (2011). To address this challenge, a number of scholars have curated specialized dictionaries. At the forefront of this endeavor, Loughran & McDonald (2011) undertook an analysis of 10-K filings from 1994 to 2008, resulting in the creation of six distinct word lists: negative, positive, uncertain, litigious, strong modal, and weak modal. These lists, encompassing 354 positive and 2,329 negative words, were carefully tailored to capture the nuanced interpretations of words within the realm of financial applications. Despite their simplicity, dictionary-based techniques are susceptible to subjectivity, homograph-related errors, and significant challenges with negation. Homographs are words spelled identically but carrying diverse meanings (e.g., "Tear" can mean to rip, a verb, or refer to a drop of liquid from the eye, a noun), while negation issues arise when a sentence is miss-classified as positive due to a positive target word being preceded by terms of negation such as "no" and "not". Furthermore, these methods' contextual specificity may not be suitable to certain sources, such as social media, which exhibit distinct linguistic traits.[67]

ML approaches to NLP present a valuable alternative approach to heuristic-based dictionary methods. However, for instance according to El-Haj et al. (2019), fewer than 20% of textual analysis papers in leading accounting and finance journals currently utilize machine learning methods. When ML techniques are deployed, they mainly revolve around supervised ML, particularly for tasks related to classification. This is why our focus will be directed towards such methods. The commonly applied supervised methods include naive Bayes and SVM (El-Haj et al. (2019)). Generally, all supervised methods consist of a standard sequence of steps (Grimmer & Stewart (2013)). The first step is that of creating a training set of $i = 1, ..., N_{train}$ documents. Each document has to be coded in one of the $k = 1, ..., K$ categories. One then considers a smaller DTM_{train}, comprising only the documents belonging to the training set, and through SL methods, attempts to understand the

[67]Recently, Garcia et al. (2023) have proposed to rely on stock price reactions to color words in order to provide new dictionaries of positive and negative words in a finance context. In a horse race comparison, their dictionaries outperform the standard bag-of-words approach (e.g., in Loughran & McDonald (2011)) when predicting stock price movements out-of-sample. By comparing their composition, word-by-word, their method refines and expands the sentiment dictionaries in the literature.

connection between terms and categories in the training set

$$Y_{train} = f(DTM_{train}) \tag{37}$$

Each algorithm (naive Bayes, SVM, random forest, neural networks, etc.) attempts to estimate f with $\hat{f}$. $\hat{f}$ can then be used to train the SL algorithm to understand the connection between features and categories in the training set. One can then employ this understanding to deduce categories in the test set, which will have a corresponding DTM_{test}. Finally, each document category will be estimated as

$$\hat{Y}_{test} = \hat{f}(DTM_{test}). \tag{38}$$

The performance of the model is evaluated using different metrics which depend on the modeling choices. In the case of classification tasks, one typically uses a *confusion matrix*. A confusion matrix is a KxK matrix where the columns represent the categories assigned by the algorithm, whilst the rows are the true categories of the text. Each element in the matrix shows how many documents were correctly classified into a specific category (column) that actually belong to that category (row). The main diagonal will, therefore, contain the counts of correct predictions. The accuracy for each category will be measured as

$$A = \frac{O(k,k)}{\sum_{k=1}^{K} M(k)} \tag{39}$$

where $O(k,k)$ are the actual number of documents in class k classified as class k and $M(k)$ is the total across row k of the confusion matrix. The accuracy A is expected to be more than $1/K$, achieved by guessing the categories at random. The empirical literature suggests that ML algorithms tend to achieve higher classification accuracy compared to dictionary-based methods (see, e.g., Li (2010)). This occurs because ML algorithms have the capability to identify patterns, relationships, and nuances in the data that may not be evident to human designers of pre-defined dictionaries. This adaptability allows ML algorithms to improve their performance over time as they are exposed to more data. In particular, ML algorithms can scale effectively to handle large datasets with high-dimensional feature spaces. They can learn from massive amounts of data and identify complex patterns efficiently, whereas dictionary-based methods may struggle to scale or may become computationally expensive as the size of the dictionary or the complexity of the problem increases, see Van Atteveldt et al. (2021).

6.1 Portfolio management applications

NLP has numerous applications in portfolio management, such as sentiment analysis, text mining, and topic modeling. We shall now briefly touch upon them. *Sentiment analysis* (SA) is an NLP technique that uses ML to assess the tone and emotion of texts, which may be useful to predict stock returns (Lo et al. (2023)). Often, a given text is classified into positive, negative, or neutral sentiment categories. SA algorithms operate by analyzing the syntactic, semantic, and contextual features of the text to extract its underlying sentiment. Some of the main features that are considered include the use of emotional words, negations, intensifiers, and other linguistic cues that indicate the overall tone

of a text. SVMs and naive Bayes are two of the most commonly used ML algorithms for sentiment modelling in portfolio management. SVMs in the context of sentiment analysis aims at finding the hyperplane that best separates positive and negative instances, while naive Bayes is a probabilistic model that calculates the conditional probabilities of each word given the sentiment category.[68] These algorithms can be trained on a set of labeled SA examples, which can improve their accuracy and performance over time. By using SA, portfolio managers can gain valuable insights into the overall market sentiment towards a particular company or the entire market.[69]

Another important application of NLP in portfolio management is *text mining*. Text mining helps investors extract useful information from unstructured data. The process of text mining involves collecting, analyzing, and transforming unstructured textual data into structured and usable information. By using techniques such as Named Entity Recognition (NER) and clustering, portfolio managers can acquire information on key themes or topics present in large amounts of textual data. NER involves identifying and classifying named entities (e.g., persons, organizations, locations or even dates) in text into predefined categories. Clustering involves grouping similar textual data together based on content similarities. This can help portfolio managers identify trends or topics that may emerge or change over time.

Topic modeling (which has recently been connected to the extraction of narratives from texts, see, e.g., Blanqu'e et al. (2022)) is an unsupervised NLP technique used to identify underlying themes present in a corpus of textual data. It is particularly useful for portfolio managers who need to stay informed about changes in market trends and sentiment to adapt their strategies accordingly. The technique works by analyzing the co-occurrence of words and phrases within a document and across a corpus of documents to identify groups of words that likely belong to the same theme. One of the most common algorithms used for topic modeling is Latent Dirichlet Allocation, which assigns a probability distribution over topics to each document in a corpus. By analyzing these probabilities and identifying the most important themes, decision-makers can gain insights into the relationships among different themes and how these may impact their performance.

It is also useful to survey the literature on applications of NLP to asset management on the basis of the main sources of textual data that scholars and users commonly employ, namely news and social media. Analyzing news articles has been one of the earliest and most widely practiced uses of NLP in finance. Tetlock (2007), for example, used a dictionary-based approach to assess the tone of a daily column in the Wall Street Journal. He constructed a pessimism factor and established that changes in this factor could predict substantial moves in daily stock prices. This analysis was further expanded in Tetlock et al. (2008), where the impact of negative words in all the Wall Street Journal and in the Dow Jones News Service (DJNS) stories about S&P 500 firms from 1980 to 2004 was tested with reference to stock returns. Negative terms found within news articles specifically related to companies consistently predicted lower returns on the subsequent trading day. A heuristic-based method was also

[68] Chen et al. (2006) consider three textual document representations: BoW, noun phrases, and named entities. They use SVMs to model the impact of news articles on equity prices 20 minutes after publication. They conclude that noun phrase outperforms the traditional methods and that SVM is successful in capturing the impact of news on stock prices.

[69] News analytics is a notable area of interest, which can be traced back to the early work of Tetlock et al. (2008). Papers in this strand of literature typically focus on short-term predictions of returns based on changes in sentiment that are detected in real-time from the news coverage attracted by an issuer.

favored by Heston & Sinha (2017) who, by analyzing a large sample of Thomson-Reuters news articles, uncovered a significant link between positive (negative) tone and subsequent positive (negative) stock returns. An SVM methodology and an n-gram representation of the text were instead selected by Manela & Moreira (2017), who constructed a measure of news-based implied volatility. He reported that news-based implied volatility explained the time variation in stock risk premia by investigating the front page content of the Wall Street Journal between 1889 and 2009. Heston & Sinha (2017) used instead a sophisticated neural network to pursue a similar goal and found that, similarly to previous studies, stocks with positive (negative) news on one day scored predictably high (low) returns on the subsequent one-to-two days. Moreover they showed how aggregating news over longer time periods produced a dramatic increase in the predictability of stock returns. Recently, Füss et al. (2020) have used NLP to investigate stock pricing at the international level by introducing a sentiment-augmented asset pricing model that recognizes to sentiment a risk factor role in global equity markets. They show that indicators derived from news and social media sentiment as distilled through Refinitiv's MarketPsych Indicator significantly correlate with the excess returns of international stock indices. By integrating sentiment factors into both classical and contemporary pricing models, they observe improved model performance, revealing positive risk premiums associated with positive sentiment and negative premiums for negative sentiment. Finally, Chen, Yu & Lin (2024) investigate whether news sentiment words from Loughran & McDonald (2011)'s dictionary can be used to develop a short-term profitable investment strategy. They propose a strategy that automatically classifies news articles from the Wall Street Journal into profitable categories based on the presence of positive and negative words. The study uses a large dataset of financial news articles and stock returns to train and test the strategy using decision trees and RF, finding that it has a positive short-term return but that the return is relatively low and essentially negated even by modest transaction costs. The authors also investigate whether the strategy's performance varies depending on the sentiment level and the size of the companies being reported, concluding that the strategy performs better in negative sentiment categories than in positive categories (as in Tetlock et al. (2008)) but that the strategy does not perform significantly differently for large and small companies. Negative returns associated with negative news sentiment tend to reverse within a few (three) days, indicating a short-term overreaction by investors. This finding, already highlighted by Tetlock (2007), suggests potential short-term trading opportunities.

NLP methods have been routinely applied to sentiment detection in social media forums and internet search data. Early examples of this approach include the work by Antweiler & Frank (2004) and Das & Chen (2007), who analyzed large sets of posts in internet message forums. Sprenger et al. (2014) found that sentiment in tweets from stock market micro-blogs was associated with subsequent stock returns. Azar & Lo (2016) focused on analyzing tweets related to the scheduled Federal Open Market Committee (FOMC) meetings and assessed their power to forecast future stock returns. They assigned polarity scores to each tweet on a scale from -1 to +1 and the resulting sentiment index based on tweets was employed to predict returns. After controlling for standard asset pricing factors, they discovered that their Twitter-derived sentiment index exhibited predictive power, particularly on days coinciding with FOMC meetings. Nonetheless, further research such as Checkley et al. (2017) and Oliveira et al. (2017) reports that sentiment measures based on Twitter may be relatively weak

predictors of stock returns. The main concern in this strand of literature is with the amount of noise that characterizes data extracted from platforms such as Twitter. One solution proposed by Sohangir et al. (2018) is to focus on platforms that attract finance experts, such as StockTwits. Liew & Budavari (2017), show how security characteristics obtained from social media have significant power in explaining the time series of daily returns across 15 stocks. Their estimated Social Media Factor (SMF) retains its significance even in the presence of established traditional factors, such as the Fama–French five factors. Furthermore, these authors propose an extension of the Fama–French 5-factor model into an encompassing 6-factor model that incorporates SMF, similarly to Füss et al. (2020). Agrawal et al. (2018) find how extremely bullish or bearish sentiment derived from Twitter and StockTwits is significantly correlated with higher values of various liquidity measures. In particular, they report a positive correlation between returns and sentiment scores and that liquidity increases after highly positive and negative sentiment messages. More recently, Groß-Klußmann et al. (2019) have proposed a method to identify only expert users who primarily focus on financial topics. They find a strong correlation between their proposed directional sentiment metrics and aggregate stock returns at a global level. This method can be clearly used to improve trend-following strategies. Ke et al. (2019) attempt to predict returns directly from textual data using RL, as opposed to building and training a sentiment indicator that can then be used as a predictor in a SL framework, reporting some preliminary success. Finally, Alvarez et al. (2024) propose a new model for predicting stock prices by combining NLP techniques (particularly the FinALBERT, a specialized model based on the ALBERT architecture, designed to handle financial text classification tasks and developed from Stocktwits data) with CNN and LSTM networks. The model uses both structured data (stock prices) and unstructured data (social media opinions) to predict a stock's future value. The model uses a fine-tuned FinALBERT model to analyse the sentiment of tweets related to specific stocks. The output is then fed into a CNN to reduce dimensionality and extract the most relevant features for predicting polarity (positive or negative sentiment). An LSTM network processes the historical stock prices and the sentiment polarity output from the CNN. The authors report that their model can significantly improve forecast accuracy over weekly horizons. The model, tested on various stocks from the NYSE, demonstrates significant improvements over a baseline LSTM model for most tested companies, achieving a mean absolute error difference ranging from 0.3% to 26%. Yet, the model showed less accurate results when predicting the price of highly volatile stocks such as Tesla.

6.2 Large scale language models and the use of GPT

A large-scale language model (LSLM) is a type of AI model designed to process and generate human language text. The term "large-scale" refers to the extensive computational resources, including powerful hardware and massive datasets, used in training these models. These models are typically based on deep learning techniques, specifically neural networks or transformer architectures, and are trained on vast amounts of text data from the internet or other sources.[70] LLMs entail millions, or

[70] Unlike recurrent and convolutional neural networks, *transformers* networks do not rely on sequential processing. Instead, they use self-attention to weigh the importance of each word or token in relation to all others, facilitating more effective modeling of contextual relationships within the input data. Self-attention is a mechanism to weigh the importance of different parts of the input sequence by computing for each word/token in the sequence, weights that determine how much focus should be placed on other words/tokens in the sequence by comparing the current word/token

even billions, of parameters. These parameters are the trainable elements of the model, empowering it to comprehend and ultimately produce human language. The immense magnitude of these models enables them to encapsulate intricate language nuances and context, yielding text generation capabilities that closely resemble human language.

ChatGPT is an example of a LSLM developed by OpenAI based on the GPT (Generative Pre-trained Transformer) architecture introduced in Vaswani et al. (2017) and based on self-attention mechanisms to focus on the most relevant parts of an input. Launched in November 2022, it is one of the most advanced NLP models developed to date and trained on a massive corpus of text data to understand the structure and patterns of natural language.[71] The GPT architecture is a deep learning algorithm used for NLP tasks. The GPT architecture employs a multi-layer neural network to capture the intricacies and patterns of natural language. Through unsupervised learning techniques, it undergoes pre-training on an extensive corpus of text data, encompassing sources like Wikipedia articles or stand-alone web pages. This training data undergoes pre-processing and filtration to eliminate low-quality and repetitive text while guaranteeing that the model remains impartial to any specific group or viewpoint.

It remains an object of debate whether LSLMs may be useful in portfolio management and in devising trading strategies. On the one hand, as these models are not explicitly trained for this purpose, but instead, are trained on historical, backward-looking text corpora, one may expect that they would give little value. On the other hand, to the extent that these models are more capable of understanding natural language, one could argue that they could be a valuable tool for processing textual information to predict asset returns. Lopez-Lira & Tang (2023) have examined the potential of ChatGPT in predicting stock market returns using sentiment analysis of news headlines (sourced from RavenPack Dow Jones). The headline data comprise news headlines from a variety of sources, such as major news agencies, financial news websites, and social media platforms.[72] In practice, they use ChatGPT to filter whether a given headline is good, bad, or irrelevant news for stock market investors.[73] They proceed to calculate a numerical score and establish a positive correlation between these "ChatGPT scores" and subsequent daily US stock market returns using data from the Center for Research in Security Prices (CRSP) for a sample period spanning from October 2021 to December 2022. Their key finding is that ChatGPT outperforms many other traditional sentiment analysis

to every other word/token in the sequence. This innovation has made transformers exceptionally well-suited for a wide range of NLP tasks, such as machine translation, text summarizing, sentiment analysis, and text generation.

[71] Other LLMs, such as Bard and BERT/RoBERTa from Google or Claude 2 from Anthropic, may be considered to offer alternatives to GPT.

[72] These headlines undergo pre-processing and filtering to detect company-specific news, allowing for the assessment of the impact of sentiment scores on individual stock returns. They make use of the "relevance score" provided by RavenPack, ranging from 0 to 100, to gauge how closely a piece of news relates to a specific company. A score of 0 (100) suggests a passive (active) mention. Lopez-Lira & Tang (2023)'s sample focuses on news stories with a relevance score of 100, and the authors concentrate their tests on full articles and press releases. To avoid redundant news coverage, they also require the "event similarity days" to exceed 90, as supplied by RavenPack, ensuring that only fresh information about a company is considered.

[73] Prompts are essential for enabling ChatGPT to perform its required tasks. In fact, they allow the model to generate responses tailored to the user's needs. Lopez-Lira & Tang (2023) use the following prompt in their study: "Forget all your previous instructions. Pretend you are a financial expert. You are a financial expert with stock recommendation experience. Answer "YES" if good news, "NO" if bad news, or "UNKNOWN" if uncertain in the first line. Then elaborate with one short and concise sentence on the next line. Is this headline good or bad for the stock price of *company name* in the *term-term*? [Headline follows]."

methods. More basic models like GPT-1, GPT-2, and BERT fail to provide accurate return forecasts, indicating that the ability to predict returns is an emerging capability of complex models. After adjusting for ChatGPT sentiment scores, the impact of other sentiment scores on daily stock market returns diminishes to zero. This leads to the conclusion that enhancing and integrating advanced language models into the investment decision-making process can result in improved performance of quantitative trading strategies.

Yet, Romanko et al. (2023) have stressed that pushing the power of ChatGPT much further text classification and, for instance, into the domain of portfolio selection may be challenging. Trusting investment advice from GPT models is problematic due to the phenomenon of the LSLM "hallucinating" (when the AI, although generating text based on its training, does so without a solid conceptual framework behind it), which makes careful verification and validation of the output difficult. Moreover, it is nonetheless important to recognize that biases exist not only in the datasets used for the LSLM training but also due to the probabilistic nature of generative AI. The frequency and positive slant of mentions pertaining to a particular stock in its training data–be it from various analysts, company reports, blog posts, news articles, research publications, or other text documents–may influence its inclusion in the selection. However, it remains uncertain whether risks are considered at the same level as performance and potential rewards.

Hence, Romanko et al. (2023) contemplate an alternative approach using ChatGPT to identify a selection of stocks from the S&P 500 market index that exhibit potential attractiveness for investment decisions. This consideration is made in the context of GPT-4's training data, which predates September 2021. The underlying rationale is that, owing to its extensive training on substantial text datasets, GPT-4 can indirectly infer the overall aggregate sentiment regarding future performance and even the risks associated with various stocks and potentially other financial assets such as bonds. Romanko et al. (2023) usee historical data up to prior to September 2021 as their in-sample data for portfolio selection and post-September 2021 data for out-of-sample testing. They ask ChatGPT to generate three distinct trading populations of stocks from the S&P500 market index, with the goal of outperforming the index.[74] These trading universes varied in size, consisting of 15, 30, and 45 stocks, respectively. They also asked ChatGPT to assign asset weights to each of the stocks, leading to the creation of three distinct ChatGPT-weighted portfolios (one for each trading universe).[75] A fundamental challenge associated with this method lies in the fact that ChatGPT may yield distinct sets of assets with each request made to the GPT-4 API. In order to bolster the reliability of their approach, they also lodged the API request to ChatGPT on 30 separate occasions and document the frequency with which each stock appeared in the output. Based on the outcomes of these repeated

[74]They used the following prompt: "Using a range of investing principles taken from leading funds, create a theoretical fund comprising of at least N stocks (mention their tickers) from the S&P500 with the goal to outperform the S&P500 index", (where N represents 15, 30, or 45 assets). When the above prompt is used without the phrase "comprising of at least X stocks", the outputs are portfolios consisting of about 15 stocks on average.

[75]They solicited weight assignments for these assets from ChatGPT using the following GPT-4 prompt: "Assume you're designing a theoretical model portfolio from these S&P500 stocks: {*input*}. Provide a hypothetical example of how you might distribute the weightage of these stocks (normalized, i.e., weights should add up to 1.00) in the portfolio to potentially outperform the S&P500 index. Also mention the underlying strategy or logic which you used to assign these weights." Here, {*input*} comprised the list of N most frequently occurring stock tickers as per the earlier prompts (where N = 15,30, 45). ChatGPT rationalized its weight assignments by citing the consideration of various factors such as sector diversification, market capitalization, growth potential, and stability, among others.

requests, they chose the 15, 30, and 45 stocks that were most frequently recommended for their respective portfolios.[76] They report that ChatGPT is effective in performing stock selection but may not perform as well in assigning optimal weights to stocks within the portfolio. Yet, when stocks selection by ChatGPT was combined with established portfolio optimization models, they achieved superior results.[77] Blending the strengths of AI-generated stock selection with advanced quantitative optimization techniques lies the ground for a hybrid approach to effective and reliable investment decision-making.

A significant challenge arises from the opaque nature of generative AI models, characterized by intricate non-linearities that obscure the manner in which they analyze their training data to formulate investment strategies. For example, the specifics of the training data employed for the GPT-4 model remain undisclosed. Furthermore, machine learning and NLP models reliant on historical data, may encounter difficulties when their anticipation of the future diverges from historical patterns. Similarly, NLP models may not be the optimal solution for scenarios demanding logical reasoning. In contrast, well-trained humans may possess a deeper comprehension of the future and excel in logical reasoning tasks (see Liu et al. (2023)). In such scenarios, the presence of humans in the decision-making loop will integrate human expertise and understanding into the data-based machine models, what has been occasionally defined "quantamental investing" (see Ma (2020)).

Existing LLMs excel in general-purpose tasks but often fall short in specialised domains like finance. The intricate language, unique terminology, and nuanced context of financial data necessitate a domain-specific approach. In fact, Bloomberg has recently identified a significant opportunity to create an LLM tailored for financial tasks, which has led to the creation of BloombergGPT, a 50.6 billion parameter language model tailored for the financial industry. Unlike previous domain-specific LLMs trained exclusively on in-domain data, BloombergGPT adopts a mixed approach. The researchers constructed a massive dataset combining a domain-specific dataset, arguably the largest of its kind (FinPile), to include diverse financial documents like news articles, filings, press releases, and web content, all sourced from Bloomberg's Archives and a range of public datasets, to ensure general language proficiency.[78] Wu et al. (2023) have backtested the performance of BloombergGPT, reporting that it outperformed existing LLMs on tasks spanning finance-specific and general-purpose categories, to test the hypothesis that training on high-quality, finance-specific data would yield superior results on financial tasks, while maintaining robust performance on general tasks. The financial tasks involved publicly available datasets covering a range of NLP tasks, as well as Bloomberg-internal high-quality evaluation sets for sentiment analysis and named entity recognition (NER). General-purpose tasks were assessed using benchmarks such as BIG-bench Hard, Knowledge Assessments,

[76] In some rare cases, ChatGPT also returned stocks that were not in the S&P500 index. To address this issue, Romanko et al. (2023) verified whether each of the assets returned by ChatGPT belonged to the S&P500 market index.

[77] In practice, Romanko et al. (2023) computed the Markowitz mean-variance efficient frontier for the universe of assets suggested by ChatGPT. To ensure a fair comparison, they impose constraints on asset weights within their optimized portfolios. More precisely, they mandate that each asset's weight should be approximately between half and double the equal weight of $1/N$, where N is the number of assets in the portfolio.

[78] BloombergGPT's architecture is a decoder-only causal language model inspired by BLOOM. It features 70 layers of transformer decoder blocks, where each block comprises multi-head self-attention (SA), layer-normalization (LN), and a feed-forward network (FFN) with a GELU non-linear function. The model incorporates ALiBi positional encoding in the self-attention component and ties the input token embeddings to the linear mapping before the final softmax.

Reading Comprehension, and Linguistic Tasks. BloombergGPT also demonstrated competitive or superior performance compared to similarly-sized and even larger general-purpose LLMs. The case of BloombergGPT provides compelling evidence of the benefits of a mixed training approach, combining domain-specific and general-purpose data for superior performance in both areas.

Glasserman & Lin (2024) have stressed how also using LLMs (in their case, Chat-GPT 3.5) to support trading strategies should be approached with caution and being mindful of the potential biases that the very nature of LLMs imply. In particular, their paper examines the potential for using sentiment analysis based on LLMs to predict stock returns and emphasize two major limitations. First, the look-ahead bias, where the LLM has knowledge of future events that influences its predictions; this occurs when an LLM, trained on historical data, has prior knowledge of stock returns following specific news articles during backtesting. For example, if an LLM knows that a company released poor Q3 earnings after a headline announcing the earnings release, it might incorrectly classify the headline as "bad news" due to its knowledge of future events. Second, the distraction effect, when the LLM's general knowledge about a company interferes with its analysis of news sentiment. This general knowledge could include both past and future information about the company. Glasserman & Lin (2024) propose a simple method to mitigate these biases: anonymizing company information in news headlines when these are fed to the LLM for sentiment estimation. They find that this anonymization technique significantly improves the accuracy of LLM sentiment analysis, also reporting that the distraction effect outweighs the look-ahead bias in LLMs trained on vast amounts of data. This finding suggests that GPT-3.5's general knowledge of companies negatively impacts its sentiment analysis, overshadowing any potential benefits from look-ahead bias. While trading strategies based on GPT-3.5 sentiment analysis show significant profitability out-of-sample (post-training period), aligning with previous research, in-sample (within the training period), strategies using original headlines (with company identifiers) underperform those using anonymized headlines, especially for larger companies. Because look-ahead bias must be positive, they conclude that the distraction effect must be negative: the general knowledge GPT-3.5 has about companies does more harm than good. Moreover, trading strategies based on anonymized headlines demonstrate lower correlation with overall market performance, indicating more idiosyncratic returns driven by company-specific news. However, the study found only weak evidence, not statistically significant, that anonymizing headlines could improve GPT-3.5's out-of-sample sentiment analysis. This is because the distraction effect can still persist outside the training window, influencing the LLM's assessment of news sentiment.

7 ML and NLP in sustainable and impact investing

Impact (sustainable) investing is a form of investment that aims to generate a positive social and/or environmental impact alongside financial returns. This is in opposition to philanthropy, which primarily aims to create social or environmental benefits without reference to performance so that private profitability and returns are made *sustainable*. ML, especially NLP models, play a crucial role in the so-called practice of *impact investing* by offering various applications. It assists in tasks such as document classification, sustainability information analysis, sentiment analysis, and the detection of "green washing". ML and NLP models automate the categorization of sustainability reports and

disclosures into specific impact themes like climate change, human rights, and biodiversity. The ML algorithms have the power to analyze and categorize sustainability-related information within a company's annual report and sustainability disclosures. Additionally, they can perform sentiment analysis on news articles and social media posts to monitor and quantify a company's social and environmental impact. The applications of NLP models vary in complexity. They range from basic dictionary-based methods used to identify thematic exposure to advanced deep-learning models like climateBERT. ClimateBERT specializes in detecting climate-related context and fact-checking claims related to climate risk. Moreover, deep learning-based language models are employed to identify instances of corporate green washing, the deceptive practice of making sustainability claims for marketing purposes without genuine sustainability efforts, possibly a prevalent issue in sustainable investing.[79]

A first area in which ML may come to play a considerable role is that of Environmental, Social, and Governance (ESG) scores, which are quantitative assessments of companies' commitments to sustainability that have become popular tools in the financial industry.[80] Recent evidence suggests that approximately two-thirds of the investors frequently use ESG ratings (see Bancel et al. (2023)). However, achieving transparency in the ESG assessment process remains a significant challenge. This is primarily because there is no complete disclosure regarding how these ratings are calculated. Rating agencies derive ESG ratings, which are based on distinct Environmental (E), Social (S), and Governance (G) scores, using proprietary models. The extent of public knowledge is limited to what data providers choose to reveal. Furthermore, ESG ratings exhibit notable heterogeneity and heavily change with the agency assigning them. For instance, Berg et al. (2022) and Billio et al. (2021) have documented substantial discrepancies in ESG ratings provided by five major data providers. They have identified three specific sources of these divergences: scope, measurement methods, and weighting mechanisms.[81] These papers find that measurement divergence is responsible for more than half of the overall discrepancy and that scope and weight are moderately less relevant, yet non negligible.

In fact, recent literature has demonstrated that it is possible to gaining insights into the internal workings of proprietary black-box rating models through the application of ML *global interpretability techniques*. These techniques aim to bring to the surface what can be inferred from replications.

[79] Oddly enough, as discussed in Lo et al. (2023), one concern regarding the use of deep learning-based LSLM in impact investing is the carbon footprint of these very models, as they are highly computationally intensive. For instance, the training of climateBERT caused 115.15 kg of CO2 emissions. Ligozat et al. (2021) have proposed to study the possible perverse, negative impact of AI systems often presented as a solution to climate change, presenting different methodologies. Strubell et al. (2019) have estimated the consumption of large NLP models, comparing it in CO2 equivalents with illustrative general life examples. They conclude that training a transformer LSLM with neural architecture search can emit up to six times what a car produces (including fuel) in its lifetime. These results advise to prioritize computationally efficient hardware and algorithms.

[80] ESG, Corporate Social Responsibility (CSR), SRI and sustainable investment are concepts related to the integration of sustainable goals into business and investment decision-making. In our treatment, these terms will be used interchangeably, even though these may carry in practice slightly distinct meanings. ESG is a set of criteria used to objectively evaluate a company's performance and risk profile, while CSR refers to a company's voluntary actions to improve its impact on society and the environment. SRI and sustainable investment are investment approaches that consider ESG and related criteria to identify companies that are making positive contributions to society and the environment and are able to generate financial returns while doing so.

[81] Divergence in scope arises from variations in what each rater considers as sustainable themes to include in the screening process. Measurement divergence pertains to the selection of specific indicators provided by data providers to evaluate a firm's quality within each pillar. Weight divergence involves differences in how importance is attributed among the various raters, meaning that rating agencies hold varying perspectives on the significance of the same attributes.

For instance, D'Amato et al. (2022) deploy random forests algorithms to predict companies' ESG scores based on accounting data, including balance sheet items.Del Vitto et al. (2023) leverage ML to shed light on the ESG ratings issuance process. Their analysis focuses on Refinitiv ESG data for the year 2021, spanning three distinct industries (finance and insurance; manufacturing; information and professional, scientific, and technical services) across three geographical regions (USA, Europe, and China). Their Refinitiv ESG scores encompass a substantial portion of listed firms worldwide, accounting for over 80% of the global market capitalization, and offer insights into over 12,000 public and private companies. Del Vitto et al. (2023) assess both white-box (explainable) models, such as Ridge and Lasso regularized regression, and black-box models, like random forests and Artificial Neural Networks (ANNs), to reconstruct assessments of E, S, and G ratings. Across all models, the learning process entails the minimization of the Root Mean Squared Error on a training dataset. Furthermore, they assess and compare the models' performances on an independent test dataset.

Del Vitto et al. (2023) report that it is possible to replicate the rating assessment with a satisfactory level of accuracy to achieve an understanding of the proprietary models employed by their data providers. However, they also uncover evidence of persisting and non-learnable noise that even more complex models cannot remove. The best results are achieved when predicting the environmental rating scores for companies within the information technology sector using regularized linear regression models like Ridge and Lasso. Conversely, the lowest accuracy is observed when assessing the social pillar for manufacturing companies, for which the best-performing models are of the ANN type. In a broader perspective, Ridge and Lasso regressions yield uniformly superior results, characterized by consistently low average prediction errors. This observation suggests that the Refinitiv ESG scores can be effectively replicated with sparse regression models without a need for overly complex algorithms. Examining individual features, it is evident that eco-friendly products exert a substantial influence on the E pillar: eco-labeled products are more likely to be developed by companies with environmental target strategies, which may be linked to management remuneration schemes. Furthermore, the presence of a "resource reduction policy" and the variable representing total CO2 equivalent emissions emerge as critical factors. The latter variable has become a well-established metric in carbon risk studies within the sustainable finance literature (as seen in Bolton & Kacperczyk (2021)). Regarding the social pillar, the analysis by Del Vitto et al. (2023) emphasizes the significance of policies supporting the freedom of association (for workers) in determining a company's S score. In contrast, in the case of the governance score, the results are less robust across sectors. The only factor consistently present in their interpretability analysis pertains to board diversity, a relationship that has been extensively studied in finance literature (see, e.g., Harjoto et al. (2015)). Sokolov et al. (2021) propose an approach to automatically convert unstructured text data into ESG scores and show how a state-of-the-art NLP technique, BERT, can be incorporated to improve the accuracy of assessing relevance and content of documents in an ESG context. Using social media data as an example, they discuss the relevance of this approach to automating ESG scoring. Modern NLP methods offer a valuable opportunity to continually enhance algorithmic capabilities for processing ESG-related documents.

One last field of application related to sustainability in which ML may provide a valuable contribu-

tion concerns the reporting and auditing of greenhouse gases (GHG) emissions.[82] Many environmental impact mitigation frameworks, like carbon pricing policies, the measurement of the alignment to climate scenario like the Paris Agreement Capital Transition Assessment (PACTA), or moving toward net zero GHG emissions via Net Zero Banking Alliances, all need a correct GHG emission baseline estimation. However, currently, such a reporting framework is not mandatory for all companies, and there is a lack of standardized measurement and estimation methodologies. For instance, while a standardized framework exists for large firms in developed countries to calculate scope 1 and 2 greenhouse gas (GHG) emissions, with results being either published, validated, or both by independent bodies like external auditors and the Carbon Disclosure Project, this practice was followed by just over 4,000 companies worldwide in 2021. Consequently, considering a typical investment universe of 15,000 companies, this implies that approximately 11,000 companies (73%) did not report their scope 1 and 2 GHG emissions.[83]

To remedy to this state of affairs, Nguyen et al. (2021) propose to use an ensemble approach that relies on an optimal set of predictors combining OLS, Ridge regression, Lasso regression, ElasticNet, multilayer perceptron, K-nearest neighbors, random forest, and extreme gradient boosting as base learners to forecast GHG emissions. Their approach generates more accurate predictions than previous models do, even in out-of-sample exercises. Additionally, Nguyen et al. (2021) find limited predictive capabilities from ML models when it comes to forecasting indirect carbon emissions (i.e., scope 3) by firms. This limitation is primarily due to a high prevalence of missing and incomplete data. In contrast, Assael et al. (2023) have proposed a ML model designed to estimate scope 1 and 2 GHG emissions for companies that have not yet reported such data. This encompasses an investment universe of approximately 50,000 companies. Their ML approach leads to robust OOS performance, both on a global scale and with a granular focus, i.e., across various sectors, countries, and revenue categories. Moreover, the model incorporates explainability tools based on Shapley values, which ensures that users can gain insights into the factors contributing to GHG emissions for each individual company.[84] In terms of interpretable Shapley values, the energy consumption feature turns out to be the most important one used by the model for both scope 1 and scope 2 forecasts. As expected from the definition of scope 2, the Employees, Country of Incorporation, and Country Energy Mix Carbon Intensity features are more important for the estimation of scope 2 than the estimation of scope 1.

[82] As it is well known, greenhouse gases affect the radiative forcing of the planet. When this radiative forcing is positive, the Earth system captures more energy than it radiates to space: this is a common measure for global warming. The calculation of the carbon footprint tends to account for all GHG emissions caused by an individual, event, organization, service, place, or product, and is expressed in units of carbon dioxide equivalent (CO2-eq).

[83] In fact, it is common to analyze GHG model estimates from the data provider Trucost, see, e.g., Bolton & Kacperczyk (2021). In general, most data providers, such as Bloomberg, MSCI ESG, Refinitiv ESG, and S&P Global Trucost and CDP, use models to estimate the GHG emissions of companies that fail to publish emissions data. Such models rely mainly on rules of proportionality between emissions and the size of the company operations and on industry-specific historical data as a foundation for their calculations. Their primary objective is to predict the logarithm of GHG emissions. When assessing the fit of these models to logarithmic emissions data, the reported in-sample performances typically hover around 60% in terms of R-square for most datasets.

[84] In ML applications, Shapley values serve as a tool to interpret the contribution of individual features or variables to a model's predictions, see, e.g., Shalit (2021). The process of computing Shapley values for a particular feature within an ML model involves considering various permutations of a feature's presence or absence from the model. For each permutation, the model's prediction is assessed, and the difference between predictions with and without the feature is recorded. Next, Shapley values are derived by calculating a weighted average of these differences across all conceivable permutations. These weights correspond to the likelihood of each feature being incorporated into the model.

More generally, the financial sector has the potential to become an important ally in alleviating the adverse consequences of climate change. This was recognized by the Paris Agreement in 2015, which formally stated that the financial system will be crucial in mobilizing capital towards new (green) assets for climate mitigation and adaptation purposes (see Wang et al. (2024)). The recognition of the role of finance in the fight against climate change has led to the emergence of a new field in the literature called *climate finance*, that focuses on "(...) the tools of financial economics designed for valuing and managing risk which can help society assess and respond to climate change" (Giglio et al. (2021). This emerging field is confronted with distinct challenges appearing at the horizon. Climate finance grapples with classical issues, notably the presence of endogeneity and non-linearities, which are due to the dual nature of climate's impact on the economy, driven by intricate feedback loops. Additionally, the conventional, substantial scale of datasets in climate finance often necessitates robust statistical tools, a characteristic of ML. To illustrate this complexity, consider the massive disparities in temperature predictions among the 20 global climate models from diverse laboratories worldwide that inform the Intergovernmental Panel on Climate Change (IPCC) across a dataset spanning over a century. Furthermore, some of the most intriguing datasets in climate finance are not only highly dimensional but also unstructured. They encompass information sourced from news articles, voice recordings, or satellite images. Coupled with the intricate nature of the phenomena they capture, many of these datasets extend beyond the realm of traditional econometrics. These challenges increasingly justify the adoption of ML.[85]

8 ML Applications in Trading Execution

To regularly re-align holdings with investment objectives, investment portfolios are rebalanced by executing orders based on planned re-optimization activity. Quantitative methods can help portfolio managers make informed decisions about when and how to execute trades to minimize transaction costs and reduce risks. Early studies by Bertsimas & Lo (1998) and Almgren & Chriss (2001) had formalized order execution as a dynamic programming problem, in which the goal is to split trades optimally over an interval of time. Two main components are needed to characterize this problem: a market impact framework–used to assess the costs and risks of executing trades–and a model of the intraday dynamics of security prices of a discrete- or continuous-time nature.[86] The optimal solution needs to strike a balance between minimizing price risk (which can be achieved by rapidly executing trades) and minimizing market impact (which typically requires a passive execution style that applies dilution over time).

One notable case in which ML seems to be required to increase the effectiveness of order execution concerns the construction of smart beta factor strategies. A common result is that ML-driven multi-factor models tend to load predominantly on factors that generate high turnover, such as price

[85] This has been recently recognized in the Conference of the Parts (COP26), where it has been stated that AI and ML can play a key role in important climate-related topics like prediction, mitigation, and adaptation, in ways we cannot ignore.

[86] Market impact models consider a variety of factors that influence the price effect of a trade, such as liquidity, the volume and timing of the trade, and the characteristics of the security being traded. These models can be developed using different approaches, including empirical regressions, market microstructure, and agent-based models. In particular, regressions estimate the relationship between trade size and price impact on the basis of historical trading data.

momentum and short-term reversals. As a consequence, it is crucial to limit transaction costs when implementing such strategies. Wolff & Echterling (2020) warn that transaction costs have to be marginal for investors to be able to capitalize on smart beta factor strategies. Avramov et al. (2021) reach a similar conclusion after analyzing several ML approaches to building multi-factor signals and argue that, particularly in the case of deep ANN techniques, their profitability tends to be concentrated in small, illiquid stocks or securities issued by distressed firms. This makes order execution of the greatest importance.

Estimating market impact, particularly in equity markets, naturally relies on ML techniques because of the vast number of available inputs, the nonlinearity of the relationships, and the complex dynamics typical of the data. In the SL camp, Zheng et al. (2012) use logistic regressions to investigate the relation between market impact and a large set of potential predictors. LASSO is used for variable selection. They find that the sign of the trade, market order size, and liquidity estimated on the basis of limit order prices, tend to be selected as the most relevant features. Booth et al. (2015) propose using performance-weighted RFs to build data-driven non-parametric models that predict market impact as a function of the characteristics of an order. They find that this approach significantly outperforms linear regressions, ANNs, and SVMs in OOS tests. Park et al. (2016) perform a similar analysis and conclude that several ML models (e.g., ANNs, Bayesian ANNs) outperform a standard parametric benchmark, whereas SVRs yields less unambiguous conclusions. Briere et al. (2020) use Bayesian networks to forecast implementation shortfall as a measure of transaction cost.[87] This approach is particularly useful in the case of missing data because it can impute the most likely value given the available information. The authors incorporate net order flow imbalance as a predictor and show that the resulting model improves on standard approaches, particularly when the order size is large and market volatility is low. Li, Cao & Pan (2019) build a system that exploits a deep learning architecture to improve feature representations, and adopt an Extreme Learning Machine (ELM) to predict market impact.[88] Li, Cao & Pan (2019) evaluate the performance of the system by comparing different configurations of representation learning and classification algorithms, and conduct experiments on intraday tick-by-tick price data and corresponding commercial news archives for stocks to find that to make the system achieve good performance, both the representation learning and the classification algorithm play important roles.

Alongside SL methods such as the ones discussed above, USL, e.g., cluster analysis, can be used for market impact modeling. Bloomberg's liquidity assessment tool and the stock clustering framework developed by Goldman Sachs (see Smith (2021)) are examples of this approach. Bloomberg's liquidity assessment tool is a set of metrics used to evaluate the liquidity of a security or portfolio. It provides portfolio managers with a quantitative assessment of the liquidity profile of securities, which can

[87] Implementation shortfall is the difference between the market price of the security at the time the order is placed and the final execution price. It is a way for traders to evaluate how well they execute their trades and to compare the cost of execution across different strategies.

[88] ELM is a feedforward neural network that is trained by randomly selecting the input weights and then solving for the output weights in closed-form using a Moore-Penrose pseudo-inverse. This approach is computationally efficient and can be applied to large datasets. ELM is different from traditional ANNs in that the hidden layer is not trained iteratively through back-propagation. Instead, the input weights and intercepts of the hidden layer are randomly generated and held fixed. This allows for much faster training and enables ELM to be applied to large datasets.

help them make informed decisions about trading strategies.[89] Cluster analysis can be used to group securities that have similar characteristics in a trade execution perspective into a small number of categories using a large set of historical microstructure features.

Recently, Philip (2020) has advocated the use of RL to estimate price impact. The reward function needed by RL is suggested to be based on the execution cost or the deviation of the actual execution from the target execution schedule. The paper shows that vector autoregressive models–which are common tools for estimating permanent price impact–may lead to incorrect inference in the presence of nonlinear impact dynamics. RL can be used to learn a model that predicts the impact of a trade given its characteristics, such as trade size, order type, and time of day. The RL agent can then use this model to optimize its execution strategy, taking into account the expected impact and other relevant predictors. The advantage of RL is that it can flexibly learn the nonlinear relations between trades and quotes. Another argument for using RL in impact estimation is that it can incorporate multiple objectives, such as minimizing execution cost, reducing impact, and minimizing risk. In fact, RL can optimize a trade-off between these objectives by exploring different execution strategies and balancing the trade-off between short-term execution cost and long-term performance.

Another field of potential application of ML is optimal execution, where optimality is measured in terms of decision timing. Rather than relying on dynamic programming, a growing literature adopts a data-driven approach to let an algorithm learn the optimal execution strategy as a function of the evolution of the inputs. RL has proven popular in this area. Dab'erius et al. (2019) examined two cutting-edge approaches provided by two popular algorithms in RL.[90] They show that both methods are able to outperform a simple, classical time-weighted-average-price (TWAP) benchmark, one of the benchmarks used by traders in the industry.[91] This result is mostly driven by environments in which TWAP is not optimal, such as when prices contain a drift, are mean reverting, or both. Tashiro et al. (2019) adopt convolutional ANNs to extract predictive signals from order-based features. A common theme among these studies is the emphasis on feature engineering, which plays a crucial role in intensely data-driven forecasting (see Rasekhschaffe & Jones (2019)). Given the low signal-to-noise ratio, it is important to incorporate domain knowledge (the understanding of the specific problem or environment in which the RL agent operates) to avoid over-fitting. Baldacci & Manziuk (2020) develop a framework that optimizes partial execution of limit orders at multiple venues by capturing dependencies between market conditions, impact, hidden liquidity, and the imbalance and spread across venues. Meanwhile, Leal et al. (2022) use a deep ANN to generate trading rules for

[89] The tool uses various inputs, including historical trade data, bid-ask spreads, market depth, and volatility, to produce metrics that quantify the liquidity of a security.

[90] These are Proximal Policy Optimization (PPO) and Deep Double Q-Network (DDQN). PPO is a method that updates the policy iteratively by maximizing a surrogate objective function that approximates the true objective of maximizing the expected discounted return. The key idea of PPO is to limit the size of the update in each iteration, so that the updated policy remains close to the previous policy. DDQN is a variant of the Q-learning algorithm, which is used to learn the optimal action-value function in RL by addressing the problem of the overestimation of the action-value function in traditional Q-learning by using two separate networks to estimate the action-value function: one network is used to select the best action, while the other is used to evaluate the value of that action.

[91] In order execution, traders use algorithms to break up large orders into smaller batches and execute them over time. This is done to minimize market impact and avoid moving the price in an unfavorable direction. TWAP is a commonly used benchmark to evaluate the performance of these procedures. For example, if a trader wants to execute a large order over a 4-hour period, she may use a TWAP algorithm to execute the order at a given average price over that time period.

high-frequency data. Their approach learns the mapping between a trader's risk aversion and optimal trading speed, which can be projected onto the functional space spanned by the closed-form solution of the optimal control problem. Mounjid & Lehalle (2019) propose a methodological improvement of RL algorithms that introduces a dynamic optimal policy for the choice of the learning rate used in stochastic approximations. Their empirical study shows improved results for the optimal execution of a large number of shares.

Recently, Hultin et al. (2024) proposed a novel iterative deterministic policy gradient (DPG) algorithm for optimizing order execution in the presence of both temporary and permanent, linear and non-linear market impact. Traditional RL methods assume that the agent's actions have no lasting impact on the market, which is unrealistic. Hultin and co-authors depart from this assumption by considering a mixed state space comprising both deterministic (e.g., inventory, time) and stochastic components (e.g., asset price). They use deep ANN to represent and optimize trading policies (through Stochastic Gradient Descent, which involves adjusting the ANN's parameters based on the calculated gradient of the objective function). By accurately accounting for non-instantaneous market impact, the proposed algorithm offers a more realistic framework for optimising trading strategies. The authors successfully apply their method to order execution, where they seek to minimize the expected cost of trading, but their results extend to exponential utility, which accounts for an agent's risk aversion. Through numerical experiments, the method is shown to be more robust and efficient than previous model-free DRL approaches (PPO and Q-learning) and time-weighted strategies, as it requires fewer design choices and performs more consistently across objectives and market dynamics. The incorporation of market impact in a more realistic manner can contribute to developing more effective and efficient algorithms for order execution and portfolio management.

9 What are the problems with ML in portfolio decisions?

ML has quickly become a popular tool for generating trading signals and optimizing portfolios in finance. However, practitioners and researchers face several challenges when applying ML techniques to these tasks (see Israel et al. (2020)). First, ML tools are highly dependent on data quality. Poor-quality, noisy data can easily result in unreliable models. By ML standards, financial time series are often very short relative to now commonly available cross sections of data. This significantly inhibits the models' full potential to learn from the data. Second, even in cases in which the time-series dimension is large (e.g., high-frequency market data), the signal-to-noise ratio is typically found to be lower than in successful ML applications outside of the field of finance (say, in marketing analytics). Apart from prices and key accounting quantities, financial data is often incomplete, inconsistent, or poorly labeled, making it challenging to create high-quality datasets for ML applications. Third, financial markets are complex systems characterized by a multitude of interacting factors, making it difficult to identify the relevant variables to consider. As a result, models can be over-fitted, meaning they capture the noise and random fluctuations in the data rather than the underlying, actual patterns (see Arnott et al. (2019)). Moreover, most popular methods, such as ANN and RL, are often difficult to interpret and do not provide insights into how they generate results. This lack of *interpretability* makes it difficult to understand whether the model is capturing meaningful structure or noise. The

immediate consequences of this could be poor model performance and risk measurement. The latter could be a concern because asset managers often rely on risk management and oversight procedures to gain investors' trust, especially during turbulent times. To address this issue, practitioners and researchers must strike a balance between model complexity and interpretability; see the discussion in Hoepner et al. (2021).

Besides interpretability, recently also the *explainability* of ML in finance has been debated as a distinct concept. Explainability goes beyond interpretability by requiring that a model's decisions or predictions can be explained to a human audience in a clear way. For instance, in the case of portfolio weight optimization, in order to gain the trust of investors and regulatory bodies, it is important to be able to explain how the model arrived at its recommendations in a transparent manner. Of course, this requires the model to be interpretable so that the portfolio manager can understand the factors that drove the model's decision-making process. In our example, the manager may want to know which asset pricing factors had the greatest influence on the model's output and how the model balanced different risk drivers (see, e.g., Bagnara (2024)). Explainability may involve creating visualizations or other tools to help clients understand the rationale behind the model's recommendations or providing detailed reports that explain how the model has arrived at its output. In practice, the distinction between interpretability and explainability can be blurred, and the two terms are often used interchangeably. Yet, the key difference is that interpretability focuses on understanding a model's internal workings, while explainability focuses on communicating the model's decision-making process to a human audience.

Recently, progress has been made in so-called *explainable artificial intelligence* (XAI) to provide solutions for interpreting opaque ML. For instance, it is becoming common (see, e.g., Babaei et al. (2022)) to use Shapley values–a concept from cooperative game theory we have introduced earlier–to interpret the importance of individual features in ML. Another approach is to use Shapley values to identify specific stocks or sectors that are driving portfolio performance, providing investors with a granular view of their exposures (see, e.g., Avramov et al. (2023)). In fact, a careful approach to the selection of relevant features in applications of SL may deliver important payoffs. For instance, Carta et al. (2022) propose three methods to discard irrelevant features in prediction tasks and assess these approaches on historical data on the component stocks of the S&P500 index. They show that trading strategies that include feature selection methods improve performances by providing predictive signals whose information content is less noisy than the one embedded in the whole feature set. However, these solutions are still far more limited than the statistical inference tools available for selection in econometric modelling. López de Prado (2022) argues that the the black box view of ML is a misconception fueled by popular industry applications, in which the search for better predictions outweighs the need to achieve a theoretical understanding. In fact, ML models can be interpreted through a number of procedures, such as partial dependence plots, individual conditional expectation analysis, accumulated local effects, Friedman's H-statistics, mean decrease accuracy, local interpretable model-agnostic explanations, and the already explained Shapley values, among others (see Molnar (2020) for details).[92]

[92]For example, the method of mean decrease accuracy (MDA) follows these steps: (1) Derive the OOS cross-validated

Many users have traditionally commented on the peril that ML may over-fit the data. *Over-fitting* occurs when a model picks up noise instead of signals. Over-fitted models have good in-sample performance but little predictability when applied to unseen data.[93] For instance, in empirical asset pricing, the relationships between factors and returns are frequently noisy, and many potential factors exist, which increases the dimensionality of the problem.[94] Practical examples of over-fitting in portfolio management include the use of complex ML models to generate trading signals that result in highly concentrated portfolios, which can lead to significant losses if the model fails to perform OOS. Another example is the use of data-mining techniques to identify spurious patterns in historical data, which can result in overestimation of future returns and lead to poor investment decisions. Yet, Israel et al. (2020) have pointed out that the very fact that the systematic patterns detected by ML may be traded away by profit-seeking arbitrageurs may lower the signal-to-noise ratio of financial data and give the impression that ML simply over-fits the data, which is of course an outcome and not a permanent attribute of applications of ML in finance per se.

To mitigate the risk of over-fitting, practitioners and researchers are pressed to use techniques such as regularization, cross-validation, and out-of-sample testing. As we have seen in Section 2, regularization methods add a penalty term to the objective function of the learning algorithm to prevent over-fitting, while *cross-validation* is used to estimate the generalized performance of the model by assessing how well a predictive model will perform on new, unseen data. It involves dividing the sample into two or more sets, one of which is used to train the model, while the other(s) is (are) used to validate its performance. By using multiple rounds of the process, in which different subsets of data are used for training and validation, it is possible to obtain a more reliable assessment of the model's ability to generalize to new data (see the discussion in Arnott et al. (2019)).[95]

Two further approaches help mitigate the perils of over-fitting: forecast combinations and feature engineering. Technically, *forecast combinations* are related to but different from ensembling. Forecast combinations refer to the process of combining the predictions of multiple models to generate a more accurate prediction. In this approach, individual models are developed and trained on the *same* data set, and their predictions are then combined using a weighted average or other techniques. The aim of forecast combinations is to leverage the strengths of each individual model to produce more accurate predictions. Ensemble methods, on the other hand, involve the creation of a group or ensemble of models, each of which is trained on a different subset of the data. The individual

accuracy of an ML algorithm on a particular dataset; (2) repeat step 1 after re-shuffling the observations of individual features or combinations of features; (3) compute the decay in accuracy between steps 1 and 2. Shuffling the observations of an important feature will cause a significant decay in accuracy. Thus, although MDA does not uncover the underlying mechanism, it discovers the variables that should be part of a theory.

[93]The in sample data can be divided into a train set and a test set, resulting in two forms of overfitting. Train-set overfitting occurs when a model so closely fits the training set that it misidentifies noise for signal. Test-set overfitting occurs when a researcher assesses the performance of a model on the same test set multiple times and picks the best result, hence concealing the existence of worse outcomes.

[94]In contrast, many ML applications outside finance, such as image recognition, have high signal-to-noise ratios. For example, some image classification tasks (such as classifying dogs versus cats) have error rates below 1%.

[95]Out-of-sample testing is a technique used to assess the performance of a predictive model on data that was not used during the model's training phase. This method involves splitting the available data in (at least) two separate sets: one for training and another for testing a model's performance. Clearly, cross-validation implies a structural use of OOS testing but not the opposite. Needless to say, OOS testing is also popular in standard time series econometrics; see Guidolin & Pedio (2018).

models in the ensemble are then combined to make a final prediction. Therefore, ensemble methods focus on creating diverse models that can capture different aspects of the same data, whereas forecast combinations focus on leveraging the strengths of multiple models *each trained in the most appropriate way on the data*, to make more accurate predictions.[96] For example, forecasts generated from different estimation windows can detect market regimes and often exhibit low correlations. Additionally, different algorithms capture different data features, ranging from those captured by simple linear regressions to highly nonlinear relationships. While more complex algorithms can potentially reveal complex relationships, they usually require a higher signal-to-noise ratio and/or more training data to work effectively. Combining forecasts from various training windows also reduces forecast variance, which may eventually increase risk-adjusted returns because of higher precision.

As discussed in Section 2, ensemble methods have shown promise when applied to financial data. Their goal is to combine weak learners, either by equal-weighting forecasts (bagging) or by accuracy-weighting forecasts (boosting), to produce a strong learner. The strong learner tends to do better than any of its constituent weak learners. Both boosting and bagging can address the bias versus variance trade-off encountered, especially with SL methods. Bias results when the estimation method does not effectively capture fundamental relationships in the data (under-fitting). Variance is systematic error that arises from small changes in the training set, which means the estimator does not learn relationships that generalize to OOS periods, which reflects over-fitting. Finally, *dropout*, a tool related to ensemble algorithms, also harnesses concepts of model averaging. Dropout is regularization technique used in ANNs that randomly drops out or deactivates nodes in a layer during training. This prevents the network from learning redundant representations of the data and prevents over-fitting. Dropout effectively produces an ensemble of ANNs, and the final prediction is made by averaging the outputs of these networks. Like bagging, this technique helps to reduce the variance of the model, leading to better generalization performance on unseen data, see, e.g., Kwak et al. (2021) for a recent application to portfolio decisions.

Feature engineering uses domain knowledge to structure a problem so that it is amenable to ML solutions. Feature engineering determines which tasks we ultimately ask the algorithms to perform and which algorithms we can use to tackle them. It is one of the most effective ways to overcome over-fitting because it allows an increase in the signal-to-noise ratio before training the algorithms. Feature engineering is where domain knowledge flows into the process. For instance, in the context of stock selection, it can provide input on such questions as: What are we trying to forecast exactly? Which algorithms are likely to be most effective? Which training windows are likely to be most informative? How should we standardize factors and returns, if relevant? And which factors are likely to provide valuable information? To limit the risk of over-fitting, the best approach is often to forecast discrete variables with ML algorithms. Rather than predicting returns, as we would with linear regression, ML algorithms most often predict categories–for instance, outperformers versus underperformers–which tend to be less noisy than returns.[97] After the selection of categories, a second decision involves

[96] Timmermann (2006) proposed a framework that helps determine the effectiveness of forecast combinations. According to the framework, forecast combinations are likely to be effective when different forecasters use distinct features of the same data and techniques, resulting in relatively uncorrelated forecast biases.

[97] Users may want to select an additional category, such as market neutral performance, or even more categories to reflect various levels of performance, but each new category increases the risk of over-fitting and may provide little

how to define these categories. If we were interested in the cross-section of returns, we would define categories by dividing stocks into outperformers and underperformers for each date in the training set. Most investors want to outperform net of risk, so a natural approach is to define performance categories for risk-adjusted excess returns.

The use of AI and ML in (especially active) portfolio management presents several challenges, including the potential of unreliable and biased models, which could result in significant financial losses. To mitigate these risks, in 2021, the European Commission has published a regulatory framework aimed at ensuring the trustworthiness of AI.[98] This regulatory framework represents the first major international effort to regulate the use of AI and provides seven key requirements for a trustworthy AI approach:

- Human agency and oversight, which emphasizes the importance of ensuring that humans have the final decision-making power in the use of AI.
- Technical robustness and safety, which emphasizes the need for AI models to be reliable, accurate, and safe.
- Privacy and data governance, which highlights the need for appropriate data protection measures to be in place.
- Transparency, which emphasizes the need for AI models to be explainable and interpretable.
- Diversity, nondiscrimination, and fairness, which emphasizes the importance of ensuring that AI models do not perpetuate biases and discrimination.
- Societal and environmental well-being, which highlights the need for AI to benefit society as a whole.
- Accountability, which emphasizes the importance of ensuring that individuals and organizations are responsible for the use of AI.

Obviously, the first, second, and fourth requirements directly relate to the technical challenges discussed above. The successful implementation of these regulations will likely lead to other international regulatory bodies to follow suit, thus reshaping the future landscape of ML research and its applications to portfolio management. However, it is not yet clear how different sectors will comply with these requirements.

9.1 Machine learning vs. econometrics?

A Reader may wonder in what ways, if any, ML differs from classical econometrics. We include an analysis of the literature that has addressed this important distinction as the existence of any differences implying that econometrics carries superior importance or practical valence over ML could *per se* indicate that ML shows some relevant limitations. Let's start by clarifying what the difference between ML and econometrics–especially interpreted as time series analysis–is not about. Because econometrics is in no way limited to linear regression models and methods, it is faulty to argue that

additional accuracy for data that are as noisy as stock returns.

[98] In 2023, the European Parliament, the European Commission, and the Council reached a political agreement on the Artifcial Intelligence Act (henceforth AIA), marking it as a historic step in AI regulation. This agreement sets out to create a comprehensive legal framework that addresses the risks associated with AI systems based on their potential to cause harm, see https://commission.europa.eu/news/ai-act-enters-force-2024-08-01$_en$).

ML puts an emphasis on highly nonlinear data that include complex patterns while econometrics would or could not. It is well known that a distinct, nonlinear branch of time series econometrics exists that has had remarkable success in economics and finance (see, e.g., Franses & Van Dijk (2000), Potter (1999), and Tsay & Chen (2018)). Moreover, in no way econometrics can be identified with simple regressions, linear or not. In fact, while nonlinear econometrics has developed and empirically validated–not least for their potential in forecasting applications–a large variety of multivariate time series models, ML still appears to go through a stage in which most of the applied empirical work is performed at the simplest, univariate level. Finally, as stressed by Masini et al. (2023), usually both ML and econometric methods need to assume stationarity for their most elementary theoretical properties to hold.

Nonetheless, a number of differences between ML and econometrics do exist. For instance, Simonian (2024) has emphasized that econometrics, with its focus on economic theory and understanding the underlying causal processes driving data generation, essentially through inferential methods and procedures (such as point estimation, confidence interval construction, and hypothesis testing), excels in explaining past events; ML, on the other hand, prioritizes predictive accuracy and generalization ability, making it more suitable for forecasting. Of course, this does not imply that, on the one hand, ML has no power to understand empirical phenomena or to be useful in inference or, especially, that econometrics has not developed the methods or a keen interest in accurate forecasting (see, e.g., Armstrong (1978)). The *algorithmic modeling culture* popularized by Breiman (2001*b*) provides the foundations of modern ML and assumes that the relationships between input and response variables are ultimately too complex to uncover with any genuine insight. Consequently, the primary concern of algorithmic methods is the development of tools that can be used to successfully predict responses from inputs without any a priori assumptions regarding the particular distribution of the data. The *data modeling culture*, which has been the dominant one in traditional econometric practice, is primarily concerned with providing information about the relationships that exist between input and response variables. It assumes that the data are generated by a specific stochastic process. In contrast, the algorithmic modeling culture assumes that the relationships between input and response variables are ultimately too complex to uncover with any genuine insight. Consequently, the primary concern of algorithmic methods is the development of tools that can be used to successfully predict responses from inputs without any a priori assumptions regarding the particular distribution of the data. As a result, ML runs into important identification problems when multiple theoretical models can explain the same observed data, making it impossible to determine the true underlying structure.

A key trade-off between interpretability and predictive accuracy exists that may tilt the balance in favor of econometrics. Econometric models, while offering valuable insights into economic relationships, often fall short of predicting complex market dynamics. ML is characterized by strong flexibility in capturing complex relationships, a structural ability to handle high-dimensional data, automated model building and testing, and a capacity to incorporate diverse data types (including unstructured data).[99] Machine learning, particularly black-box models like deep neural networks,

[99]The automated model-building power of ML can be best appreciated with reference to a multivariate linear regression and the task of including all possible interaction effects among variables taken as pairs. For instance, if one were to use econometrics and specify a model including all possible interaction effects defined by pairs of variables, then for any

can achieve high predictive performance but often lack the transparency to understand the rationale behind their predictions.[100]

These pros and cons of econometrics vs. ML have led a few authors to argue against viewing econometrics and machine learning as mutually exclusive and advocate for a hybrid approach. Here, we shall not pursue the view that a number of concepts and methods recently advocated by ML proponents have been often known and used in econometrics for decades. For instance, even though the application of LASSO to regression analyses has heralded the advent of ML for most users since approximately 2015, in econometrics, the regularized estimators covered in Section 2.1 gained attention after the seminal papers by James and Stein, who popularized the bias-variance trade-off in statistical estimation (see Efron & Morris (1973)) and, although these are now best construed as examples of SL, they are covered by all econometrics textbooks, e.g., Hayashi (2011).) Yet, a hybrid approach involves leveraging the strengths of each methodology within a single investment process, potentially combining them in a modular or integrated fashion that considers the specific requirements of each investment problem and allows for the selection of the most appropriate tool, whether it be traditional econometrics or machine learning.[101] Iskhakov et al. (2020) have, in fact, emphasized how ML can enhance what they call structural econometrics (SE), with the latter pursuing causal inference and testing economic theories; similarly, ML can aid the application of dynamic programming principles to model decision making. ML can provide approximate solutions to large-scale programming problems, making them more tractable. Techniques like neural networks offer greater flexibility in modelling complex relationships and relaxing restrictive assumptions like unbounded rationality. For instance, ML can overcome the curse of dimensionality in dynamic programming, the exponential growth in computational complexity as the number of variables in a problem increases. ML techniques, such as neural networks, can mitigate this curse by approximating high-dimensional functions with fewer parameters. While SE typically relies on economic theory and imposes assumptions about agents' rationality and equilibrium behavior, ML (e.g., RL) takes a more data-driven approach, attempting to learn a reward function that makes the observed behavior appear optimal. Moreover, counterfactual analysis involves estimating the effects of hypothetical scenarios or policy changes. While ML can contribute to counterfactual analysis by predicting outcomes under different scenarios based on learned patterns, SE provides a more robust framework for this task.

Equivalently, as stressed by Mullainathan & Spiess (2017), while ML (their focus is particularly on SL) offers a powerful toolkit for prediction (estimating $\hat{y}$), it should not be misconstrued as a

realistic multivariate model, such a number may easily exceed the sample size, making the regression impossible. On the contrary, even basic ML techniques, such as a shallow ANN, may be seen as performing the task of trying all possible combinations of interaction effects and selecting the best possible ones–especially from a predictive perspective–in an automatic way.

[100] Simonian (2024) pushes this point to claim that econometric models, by imposing specific functional forms and assumptions about the distribution of the data, which might not hold true in the future, are prone to higher bias. Instead, ML prioritizes variance reduction, meaning it aims to create models that generalize well to unseen data.

[101] Mullainathan & Spiess (2017) stress that ML may be valuable for tasks generally considered traditional estimation problems: many econometric techniques implicitly involve prediction steps where machine learning can improve performance. For instance, SL can be applied to the first stage regression to improve instrument selection and prediction, potentially leading to less biased estimates. ML algorithms can be adapted to estimate treatment effect heterogeneity by predicting individual-level effects. In fact, Mullainathan & Spiess (2017) reckon that while the language of ML may at first appear alien to econometricians, they have their roots in nonparametric statistics, see, e.g., Györfi et al. (2006) or Zhu & Nowak (2022) for a recent example of work at the intersection.

replacement for traditional econometric methods aimed at parameter estimation (estimating $\hat{\theta}$). For example, even when these algorithms produce regression coefficients, the estimates are rarely consistent. For instance, there are well-known difficulties associated with traditional inference techniques (e.g., hypothesis testing) when using penalized SL estimators like LASSO. The distribution of these estimators can be non-Gaussian and highly dependent on model selection procedures. The danger in using these tools is taking an algorithm built for $\hat{y}$, and presuming their $\hat{\theta}$ have the properties we typically associate with estimation output. They, in fact, stress that while ML excels at prediction tasks, it is not well-suited for parameter estimation and discovering causality patterns (see Hoepner et al. (2021)). They propose a framework for understanding machine learning as a tool for discovering generalizable patterns in data, which can be valuable in dealing with complex or unconventional data sources. ML's strength lies in its ability to uncover structure within data, but it cannot reliably provide stable estimates of underlying parameters. Due to the inherent flexibility of the ML algorithms, similar prediction accuracy can often be achieved with very different combinations of variables. This makes it challenging to pinpoint the causal impact of specific variables without strong, often unrealistic, assumptions about the underlying data-generating process. Yet, as already discussed in this section, ML's reliance on flexible functional forms makes it prone to overfitting, which can be mitigated through regularization (i.e., limiting the complexity of the prediction function) and what they call empirical tuning (i.e., employing techniques like cross-validation to select the optimal level of regularization by evaluating performance on held-out, cross-validation samples). Hence Mullainathan & Spiess (2017) advocate viewing ML as a powerful addition to the economist's toolkit, especially when targeted at prediction tasks, not a replacement for traditional econometric methods, which provide instead a robust framework for causal inference, allowing researchers to move beyond correlation and establish causality.

10 The empirical evidence on the performance of ML in portfolio decisions, so far

At the time this paper is being completed, the empirical finance literature is booming with surveys and empirical horse races comparing the performance of portfolio and trading decisions determined by a range of ML methods; often, these are also compared to both classical benchmarks (such as MV) and statistical methods. For instance, Pirayesh & Sallan (2024) systematically survey the applications of ML to stock trading and report that LSTM deep ANNs and random forests are the top performers due to their ability to capture long-term dependencies and nonlinear relationships within financial time series data. Therefore, in what follows, we purposely limit ourselves to summarize the key findings of only 14 papers with no goal at being comprehensive or exhaustive.[102]

10.1 Forecasting asset returns

Gu et al. (2020) have compared a large number of ML methods to predict asset returns, including elastic nets, PCA, PLS, generalized linear models with penalization, gradient-boosted regression trees

[102] Jiang (2021) is another comprehensive survey that examines over 100 articles that explored the utilization of deep learning techniques in stock market prediction between 2017 and 2019.

(GBRT), RFs, and ANNs. For US equity data, ANNs perform the best, and they are closely followed by RF and GBRT. The methods generally agree regarding the most influential stock-level predictors, which belong to four categories: price trends, liquidity variables, risk measures, and valuation ratios. At the macro-level, the aggregate book-to-market ratio stands out. The prediction-based trading strategy generating the highest SR is an ensemble of all methods, but it is difficult to find a precise economic intuition supporting it.

Lo & Singh (2023) examine the use of deep neural networks for predicting financial risk premia using skip connections to improve performance and enable the training of deeper models. They address the challenge of non-stationarity in asset returns by separating risk premia prediction into two independent training tasks: a time series model for average monthly returns using macroeconomic variables and ridge regressions; a cross-sectional model for deviations from the mean. Using monthly equity return data from CRSP for firms listed on the NYSE, NASDAQ, and AMEX from March 1957 to December 2016, 94 firm-specific features from CRSP and Compustat, and macroeconomic data from Welch & Goyal (2008), their approach leads to a threefold increase in out-of-sample R-square compared to a baseline model trained on total returns, and a 2.5x increase compared to a similar model from previous literature (e.g., Gu et al. (2020)). Techniques of explainability (Model-agnostic Explanation (LIME)) reveal that features like short-term reversion, size, and return volatility play a key role in predicting risk premia. The authors also apply their models to portfolio creation, showing that portfolios constructed based on their model predictions outperform standard benchmarks.

Similarly, Bianchi et al. (2021) employ various machine learning (ML) techniques to forecast Treasury bond returns across different maturities. Unlike stock data, the majority of these methods perform poorly when using only yields as inputs, a result rather common in the empirical finance literature. This fact leads to negative out-of-sample R-squares. However, ANNs and RFs manage to explain approximately one-fourth of the variation in 10-year Treasuries. Notably, when forward rates and macro variables are integrated as predictors, the most effective approach is a "group-ensembled" ANN, consisting of a collection of networks, each dedicated to a specific group of macroeconomic aggregates, with interactions across groups de-activated. Based on these results, Bianchi et al. (2021) contend that it is the non-linearities and interactions within a variable group that are crucial, rather than interactions across groups. Predictors associated with inflation, money, and credit exhibit significance across all maturities, while the influence of other covariates depends on the term structure; some exert effects only on level or slope, showing that averaging bond returns across maturities may be unwise. Bali et al. (2020) confirm the efficacy of ML models also in predicting corporate bond returns. They study RFs, FFANNs, and LTSM neural networks and document significant improvements in the OOS performance of stock and bond features (characteristics) when predicting future bond returns. Furthermore, imposing economic structure based on Merton's intertemporal CAPM, the ML models continue to significantly add value despite this restriction.

It remains to be seen to what extent these results can be generalized to all ML techniques, to non-US or historical data, and in particular to data that are not subject to deep and insisted scrutiny that has instead concerned stocks and bonds. Nonetheless, for instance, Baltussen et al. (2021) find that ML methods were effective in predicting stock returns in the "pre-CRSP" era, going back all

the way to 1866. Moreover, as discussed in the following subsections (see also Leung et al. (2021), even though ML may turn out to be superior to traditional linear models in predicting cross-sectional one-month-ahead returns using a set of well-documented stock characteristics, the degree to which this statistical advantage can translate into economic gains in portfolio back-tests hinges crucially on the capacity to manage risk and execute trades efficiently. For instance, under a long-only one percent tracking error strategy, the performance difference between the best-performing linear and nonlinear signals is considerably less compelling than in the absence of transaction costs. They also discuss the impact of implementation lags on trade execution, particularly for signals with short-term horizons and the high turnover induced by reversal characteristics, making them costly to implement due to transaction costs and lending fees, especially for short positions. Furthermore, none of their four GBM-based signals yields an information ratio significantly different from that of simpler models.

Li, Turkington & Yazdani (2019) introduce a comprehensive set of interpretability metrics termed *model fingerprints*, which utilize partial dependence functions to dissect the behavior of predictions made by ML models. This approach breaks down predictions into linear, nonlinear, and interaction effects among predictors, offering insights into the underlying mechanisms. Additionally, they demonstrate a method to decompose the predictive accuracy of a model into these components. Importantly, these fingerprint metrics are expressed directly in units of predicted returns, enabling comparison across different models. In their study on foreign currency investing, Li et al. apply this framework to predict one-month-forward currency returns for major currencies. They focus on total returns of forward contracts to represent investable exposures, a domain where traditional quantitative strategies have struggled after the 2008 financial crisis. The data used includes monthly returns for exchange rate pairs from G10 currencies spanning 351 months, with an OOS test sample January 2016 - March 2019. They employ a specific set of established currency factors as predictors, encompassing short-term interest rate differentials, spot return adjustments, trailing equity returns, and currency market turbulence. When comparing ML models such as random forests, gradient boosting machines, and artificial neural networks (ANNs), they find that gradient boosting exhibits superior in-sample performance over linear regression and traditional strategies. However, when evaluating performance in the testing sample, a convergence is observed across linear and ML models. Although gradient boosting performs well, there is an indication of potential mild overfitting.

Li et al. (2022) have pointed out that whilst investment strategies based on ML have a lot to offer, they are not particularly useful in practice unless they are both investable and interpretable. To build strategies that are investable, they focus on a subset of liquid SP 500 securities with large market capitalization and ensure that trading does not exceed a reasonable amount (as opposed to models that mostly invest in small or illiquid stocks or those that exploit short-term pricing effects that may be prohibitively costly to trade). To build strategies that are interpretable, they use a concise set of predictor variables (both company characteristics, such as size, short-term mean reversion, volatility, beta, leverage, profitability, investments, dividend yield, and aggregate variables, such as financial turbulence and recession likelihood indicators), and decompose the outputs of complex models into sub-components using the fingerprinting method. The prediction target is each stock's total return in the following month and forecasts are obtained through OLS regressions, LASSO regressions,

random forests, boosted trees, and one ANN, with calibration parameters chosen through cross-validation. They report compelling results in favor of supervised ML applied to stock selection within the SP 500 index. Linear methods, like equally weighting and standard regressions, exhibit reasonable performance but are outperformed by more sophisticated models like Random Forests, Boosted Trees, and ANNs. Random Forests demonstrate lower turnover but higher risk, while Boosted Trees and ANNs exhibit higher mean returns but entail more aggressive trading, hence transaction costs. ANNs marginally outperforms Boosted Trees across metrics. The fingerprints reveal differences in emphasis, with ANNs and Boosted Trees favoring momentum and value factors, while Random Forest focuses on volatility and beta. However, the benefits Li and co-authors report are modest compared to other papers, probably because of their emphasis on investability and interpretability. They suggest that investors should approach ML with reasonable expectations and should not anticipate past opportunities for superior risk-adjusted returns to last forever.

10.2 Long-short strategies tasks

Hao et al. (2023) compare gradient boosting decision trees, random forests, and deep ANN models to forecast stock prices and construct statistical arbitrage strategies involving five stock markets (the US Standard & Poor's 500 Index, the Chinese CSI 300 Index, the UK. Financial Times Stock Exchange 100 Index, the Japanese Nikkei 225 Index, and the Toronto S&P/TSX Composite Index). For each of these markets, they use data from 2005 to 2020 to construct a long-short portfolio with 20 selected stocks using their models. They find that traditional ML algorithms yield significant average excess returns in all markets except for the UK, while deep ANNs achieve strong risk-adjusted results in all markets, including the UK. However, the statistical arbitrage performance of ML varies across markets, while the deep ANN application achieves the best performance in the Chinese market, outperforming RF and GBM trees. This establishes a–possibly minor and yet unverified–advantage of deep ANN over other SL techniques.

Avramov et al. (2023) assess the effectiveness of trading strategies constructed using various ML models, including a three-layer ANN, Generative Adversarial Networks, and Independent Principal Component Analysis, while also considering transaction costs and sample restrictions. A first interesting result is that, when they exclude micro-caps, financially distressed firms, and those lacking credit ratings from the training dataset, the performance of network-based models deteriorates significantly, with more than half of the risk-adjusted returns lost, whereas IPCA leads to more resilient strategies. Concerns arise regarding the net-of-fee performance due to the high trading turnover generated by ML strategies. On a positive note, profitability under ANNs and GANs tends to increase during periods of heightened market sentiment and volatility and decreased market liquidity, aligning with common expectations. Moreover, their performance does not fade in recent years, contrasting with traditional anomalies. Analysis of stocks assigned to prediction-sorted portfolios reveals a consensus among ML techniques in taking long positions in traditional characteristics associated with positive risk premia, such as size, illiquidity, beta, and momentum.

10.3 Stock selection tasks

Rasekhschaffe & Jones (2019) have applied ML to perform stock selection on an investment universe that includes small-, medium-, and large-capitalization stocks, with an average of 5,907 stocks per month, in 22 developed markets for a 1994-2016 sample. Their factor library consists of 194 factors (i.e., company characteristics) that were assembled by IHS Markit.[103] They define their training sets in three alternative ways: (1) the recent training set of all data from the prior 12 months; (2) the seasonal training set including all data from the same calendar month over the prior ten years; (3) the hedge training set including data from the bottom half of performance over the prior ten years based on the first two training sets. Hence, every single month may appear in more than one training set. They compute risk-adjusted excess returns by dividing each stock's excess return by its past 100-day volatility. They train the four algorithms on each of the regional training sets—a bagging estimator that employs AdaBoost, a gradient boosted classification and regression tree algorithm, one shallow ANN, and a bagging estimator that is built using an SVM.

Results on predictive accuracy are strong for all algorithms and training sets, but the findings for the ensemble forecasts are even stronger. The benefits of diversifying across training windows are particularly evident. The performance of a simple OLS benchmark strategy is positive, but average returns and alphas are considerably larger for the ML approaches, also on a risk-adjusted basis. Interestingly, the ML-driven portfolios generally load negatively on the value (HML) and small size (SMB) factors, but only the US zone equal-weighted loading on SMB was significant. This indicates that the positive results are not driven by common risk factors and reflect instead the genuine contribution of ensembling.

10.4 Strategic asset allocation tasks

With reference to the returns of the component stocks of the China Securities 100 Index, Ma et al. (2021) have investigated the genuine OOS performance of SVR. RF and of three deep learning models (i.e., DMLP, LSTM neural network and CNN under a Relu activation function) in the stock pre-selection process before portfolio formation. They report that these models all lead to overwhelming performance relative to traditional time series approaches. Moreover, they combine the predictive results to advance classic MV and omega portfolio optimization models.[104].

Kynigakis & Panopoulou (2022) have explored the potential benefits of using ML to generate return forecasts from multivariate prediction models. They compare the OOS performance of portfolios (of stocks, bonds, and commodities, with reference to an OOS period 2000-2019) utilizing ML predic-

[103]They include 21 deep value factors, 18 relative value factors, ten factors focused on earnings quality, 26 factors capturing earnings momentum, 26 factors focused on historical growth, 35 liquidity factors, 29 management quality and profitability factors, and 29 technical price-based factors. They train their algorithms independently for four separate regions: the United States, Japan, Europe, and Asia ex-Japan. All factors are percentile-ranked within region and industry buckets at each date.

[104]Omega optimization maximizes the relative probability of portfolio return or loss exceeding a critical value based on the empirical distribution of returns by maximizing the omega ratio, defined as the probability weighted ratio of gains versus losses for some threshold return target. Omega is well known to avoid the limitations of the Sharpe ratio, for instance, manipulation by trading off lower realized variance for negative skewness or higher kurtosis, see Goetzmann et al. (2007)

tions to that of the equal-weighed portfolio and an MV portfolio based on the historical average (HA). The analysis is conducted for one conservative and one aggressive investor (in terms of risk aversion) and for different levels of permissible leverage. Additionally, the authors conduct robustness checks to test how alternative estimates of the covariance matrix may affect performances. They find that using ML can be beneficial to OOS portfolio performance. The majority of the portfolios outperforms the equal-weighted and HA portfolio benchmarks. MV portfolios exhibit similar OOS performance across different specifications of the covariance matrix.[105] For the equal-weighted and HA and for most of the simple forecast combination portfolios, Kynigakis & Panopoulou (2022) report higher Sharpe ratios during the expansionary periods. The pattern is similar for most of the dimensionality reduction and nonlinear SL methods, while for shrinkage methods and ML ensembling, the pattern is reversed. Finally, when they introduce transaction costs, the performance deteriorates due to the high degree of turnover of all the strategies and methods.

In Kynigakis & Panopoulou (2022), the measure of the contribution of each variable to reported performance is calculated as the change in the OOS R^2, as defined in Campbell & Thompson (2008), from setting the values of a predictor to zero within each iteration of the OOS period and then averaging these values to obtain a single variable importance measure for each predictor of a particular model. For stocks and under the KS model, the most important features are industrial production, market return, the dividend price ratio, capacity utilization, the earnings-price ratio, and the term spread. The important variables for shrinkage methods are similar to those of the KS model. In the case of dimensionality reduction methods, influential variables for PCA- and PLS-type algorithms include financial uncertainty, the payout ratio, stock variance, and the real economic activity index, while for ICA and RICA, the dividend-price ratio and excess market returns are the most important features. Nonlinear ML methods tend to use a broader set of predictors, the exception being shallow ANNs with a single hidden layer that features only the dividend-price ratio and market returns.[106] Finally, ML models favor higher rebalancing frequencies. Reducing the frequency with which a portfolio is rebalanced leads to considerably lower performance for all models. Specifically, the results based on average realized returns show that the best performing portfolio in the case of quarterly rebalancing is based on a shallow ANN. However, when the rebalancing frequency is reduced to annual, the best-performing strategies become those based on sparse PCA and an ANN with two hidden layers for the aggressive and conservative investors, respectively. On the contrary, the results based on the Sharpe ratio show that all alternative models fail to outperform the equally weighted allocation. So ML would be profitable only for investors who can rebalance and trade often, which are most likely represented by professional portfolio managers.

Because, random forests excel at capturing nonlinearities and interactions within macroeconomic and market data compared to simpler linear models, Mueller-Glissmann & Ferrario (2024) use a RF

[105] When comparing portfolios across different combinations of weight constraints, their findings indicate that allocations that allow leverage further improve the performance of portfolios based on ML and that ML methods benefit more the portfolios of an aggressive investor. Furthermore, portfolios based on ML consist primarily of stocks and commodities, with bonds representing a small share at best.

[106] The results on the frequency of inclusion of variables for the bond market indicate that most models use the information from the full set of predictors, except for RICA and ANN, which are skewed towards the dividend-price ratio. In the case of commodity returns, the influential predictors for most shrinkage methods include the market, dividend-price ratio, industrial production, capacity utilization, CFNAI and GSCI.

algorithm to identify macro regimes (as defined by growth, inflation, and policy) that drive specific tail risks (drawdowns) for the portfolio and apply it to dynamic asset allocation. The authors develop dynamic asset allocation overlays that adjust portfolio weights based on real-time probabilities of the identified regimes. These overlays involve shifting between a 60/40 portfolio and cash, rotating between equities and bonds, and strategically allocating to commodities/gold. Cross-asset performance is shown to be closely tied to business cycle swings. However, RFs struggle to capture structural shifts and market reversals during periods like the Tech Bubble and the Global Financial Crisis.

10.5 Is reinforcement learning superior to supervised learning?

Ngo et al. (2023) compare the performance of ten different portfolio construction methods, including MV-style (tangency portfolio and minimum variance) approaches, statistical ML, deep SL, and RL models. They use two experimental designs—one related to a frontier market (Vietnam) and the other based on traditional US data—with different time frames and asset types to evaluate the performance of the various construction methods. To evaluate the performance of different portfolios according to different optimization techniques, the study uses six metrics: realized mean return, standard deviation of returns, Sharpe ratio, Sortino ratio, maximum drawdown, and 1% VaR.

Their key result is that RL models outperform all other portfolio construction methods, including deep learning. In particular, an RL approach based on a proximal policy optimization (PPO) algorithm produces the highest Sharpe ratio and cumulative returns in both applications. The results of the first experiment, conducted on Vietnamese data, show that RL outperforms other deep learning models and conventional models in terms of Sharpe and Sortino ratios. The deep portfolio and PCA methods yield lower Sharpe and Sortino ratios. RL and deep ANNs lead to a slightly higher mean daily return than other models; yet, these also yield a slightly higher realized volatility than other models. Yet, RL outperforms all other models in terms of minimum Max drawdown and Value-at-Risk. In the second experiment, Ngo et al. (2023) evaluate the performance of different portfolios based on US-listed exchange-traded fund (ETF) portfolios. The results show that the RL and PCA methods outperform all other methods in terms of Sharpe and Sortino ratios. The deep learning-based portfolio produces negative Sharpe and Sortino ratios. The benchmarks lead to lower empirical Sharpe and Sortino ratios as compared to RL and PCA and traditional MV approaches perform poorly compared to other methods. This is consistent with previous research, which has shown that MV optimization may not be effective in practice due to their sensitivity to input parameters and assumptions (see Michaud (1989)). Interestingly, the statistical SL methods, such as hierarchical risk parity and PCA, perform better than MV approaches but are outperformed by deep ANNs and RL models. The study highlights the potential of RL for portfolio construction, particularly in volatile and non-stationary markets where long data sets may be less useful.

11 Conclusions and final thoughts

Machine Learning possesses boundless potential, allowing for endless customization to achieve a desired level of performance. Various adjustments can be made, encompassing different validation

techniques, hyperparameter selection, outlier detection and removal, data augmentation, and more. Additionally, features can undergo diverse transformations; their dimensions can be altered, variables can be derived through unsupervised methods, and combinations of variables can be explored. Models can be cascaded to aid optimization or enable ensemble learning. Moreover, the versatility of ML extends to the possibility of encompassing all machine learning frameworks, i.e., supervised, reinforcement, and unsupervised learning, within a single predictive task. The only means to determine whether any of these adjustments are advantageous is through empirical testing on validation data. On the one hand, this presents a remarkable opportunity and a feeling of invincibility——namely, that Machine Learning (ML) will ultimately succeed in achieving the goal of generating economic value by predicting asset returns and optimizing portfolio weights–has recently permeated both the academic literature and the practice of asset management. However, the same challenges that have confronted classical financial econometrics since the 1990s are pivotal in the current advancements of ML applications to financial decision-making. How and under what circumstances can ML efficiently and effectively support asset management tasks? We will require numerous additional empirical backtesting endeavors akin to those delineated in Section 9 before these matters are conclusively resolved.

Another emerging issue that we anticipate will attract increasing attention is the potential connection between Machine Learning (ML) methods and the behavioral patterns of investors that ML might elucidate. Such a connection could greatly enhance interpretability and explainability, thereby bolstering ML's prospects of eventually dominating the field of asset management. In this context, a recent paper by Routledge (2019) represents a significant initial step in this direction. Routledge employs a framework typical of the analysis of financial decisions under ambiguity (see Gilboa & Schmeidler (2010) and Guidolin & Rinaldi (2013), for general surveys) to characterize simplicity and parsimony in data models.[107] They propose that ML shrinkage and regularization methods may offer a natural approach to practical implementation. In Routledge's model, a potentially multivariate, complex data signal ($\mathbf{x}_n$) is known by the decision maker and may be useful in predicting some unobserved state (S_n) that is relevant for a decision problem. The investor is directly interested in the prediction of returns given the predictors, the set of possible theories is the same as the set of possible data generating processes, and the true data process belongs to the set of theories our decision maker considers. The decision maker collects data and ranks the possible theories; for example, the familiar maximum likelihood criterion is one way to construct a preference ordering. Gilboa & Schmeidler (2010) introduced appropriate axioms on preferences that allow for consideration of both "fit" (likelihood) and "simplicity":

$$\max_{\omega} E_{\theta^*}\left\{ E\left[\frac{(\omega'\mathbf{R}_n + (1-\omega'\mathbf{1})R^f)^{1-\gamma}}{1-\gamma} \right] \middle| \mathbf{x}_n \right\} \qquad \text{s.t. } \theta^* \equiv \arg\max_{\theta} \ \psi(\theta) + \sum_{(\mathbf{x}_n, S_n)\in h_n} \ell(\mathbf{r}_n, \mathbf{x}_n, \theta)$$

where γ is the coefficient of constant relative risk aversion, ω is a vector of portfolio weights associated

[107] Eghtesad & Mohammadi (2024) have recently examined a novel approach to portfolio optimization that integrates ambiguity aversion with deep learning, machine learning, and time series forecasting models. The authors examine the performance of three forecasting methods – deep ANN LSTM models, random forests, and ARIMA models – for predicting stock returns in the Iranian stock market and incorporate these predictions into a portfolio selection model that accounts for ambiguity aversion. They report that the LSTM-based portfolio optimisation model, considering ambiguity, yielded the highest long-term performance compared to Random Forest, and ARIMA.

to the risky assets whose gross returns are collected in $\mathbf{R}$, R^f is the gross risk-free rate, $\mathbf{x}$ is a vector of predictors, and θ is a vector of parameters that uniquely characterizes each theory that affects the prediction of returns conditional on the predictors. θ^* is selected on the basis on the data seen in history h_n. The functions $\psi(\theta)$ and $\ell(\mathbf{r}_n, \mathbf{x}_n, \theta)$ (the likelihood) characterize the decision maker's preferences to choose θ^* which determines how $\mathbf{x}_n$ is used to forecast $\mathbf{R}_n$. Preferences over theories, sensibly, embody a preference for fitting the data. However, the function $\psi(\theta)$ does not depend on data. This captures a preference over the theories themselves in terms of, say, simplicity. This can be interpreted as a "regularization" of a regression that imposes a preference that fewer parameters in a regression is better. Routledge provides empirically relevant examples based on US aggregate stock returns data and typical predictors (the price-dividend ratio, a big collection of macroeconomic data series, and text from the Federal Reserve Board's Federal Open Market Committee meetings Beige Book). He finds that a preference for simplicity significantly alters the optimal portfolio weights and may also improve the out-of-sample performance of otherwise rather standard strategies. Our belief is that if other persuasive models were to be developed to create this bridge between methods and behavioral investment assumption and, in a way, deep economic principles, this would offer a concrete way for ML to eventually prevail as *the* approach favored among portfolio decision-makers.

References

Aboussalah, A. M. & Lee, C. G. (2020), 'Continuous control with stacked deep dynamic recurrent reinforcement learning for portfolio optimization', *Expert Systems with Applications* **140**, 112891.

Agrawal, S., Azar, P. D., Lo, A. W. & Singh, T. (2018), 'Momentum, mean-reversion, and social media: Evidence from stocktwits and twitter', *The Journal of Portfolio Management* **44**(7), 85–95.

Aldridge, I. & Avellaneda, M. (2019), 'Neural networks in finance: design and performance', *Journal of Financial Data Science* **1**(4), 39–62.

Almahdi, S. & Yang, S. Y. (2017), 'An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown', *Expert Systems with Applications* **87**, 267–279.

Almahdi, S. & Yang, S.-Y. (2019), 'A constrained portfolio trading system using particle swarm algorithm and recurrent reinforcement learning', *Expert Systems with Applications* **130**, 145–156.

Almgren, R. & Chriss, N. (2001), 'Optimal execution of portfolio transactions', *Journal of Risk* **3**, 5–40.

Alvarez, P., Morales, A., Seguel, R. & Atkinson, J. (2024), 'Predicting stock prices using neural models based on financial textual information.', *Journal of Financial Data Science* **6**(2).

Antweiler, W. & Frank, M. Z. (2004), 'Is all that talk just noise? the information content of internet stock message boards', *Journal of finance* **59**(3), 1259–1294.

Ardila, D., Sanadgol, D., Cauwels, P. & Sornette, D. (2017), 'Identification and critical time forecasting of real estate bubbles in the usa', *Quantitative Finance* **17**(4), 613–631.

Armstrong, J. S. (1978), 'Forecasting with econometric methods: Folklore versus fact', *Journal of Business* pp. 549–564.

Arnott, R., Harvey, C. R. & Markowitz, H. (2019), 'A backtesting protocol in the era of machine learning', *The Journal of Financial Data Science* **1**(1), 64–74.

Arrieta-Ibarra, I. & Lobato, I. N. (2015), 'Testing for predictability in financial returns using statistical learning procedures', *Journal of Time Series Analysis* **36**(5), 672–686.

Assael, J., Heurtebize, T., Carlier, L. & Soupé, F. (2023), 'Greenhouse gases emissions: estimating corporate non-reported emissions using interpretable machine learning', *Sustainability* **15**(4), 3391.

Avramov, D., Cheng, S. & Metzker, L. (2021), 'Machine learning versus economic restrictions: Evidence from stock return predictability', *Management Science* . Accepted.

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

Azar, P. D. & Lo, A. W. (2016), 'The wisdom of twitter crowds: Predicting stock market reactions to fomc meetings via twitter feeds', *Journal of Portfolio Management* **42**(5), 123.

Babaei, G., Giudici, P. & Raffinetti, E. (2022), 'Explainable artificial intelligence for crypto asset allocation', *Finance Research Letters* **47**, 102941.

Badarinza, C., Campbell, J. Y. & Ramadorai, T. (2016), 'International comparative household finance', *Annual Review of Economics* **8**, 111–144.

Baek, S., Lee, K. Y., Uctum, M. & Oh, S. H. (2020), 'Robo-advisors: Machine learning in trend-following etf investments', *Sustainability* **12**(16), 6399.

Bagnara, M. (2024), 'Asset pricing and machine learning: A critical review', *Journal of Economic Surveys* **38**(1), 27–56.

Baldacci, B. & Manziuk, I. (2020), 'Adaptive trading strategies across liquidity pools', *arXiv preprint arXiv:2008.07807* .

Bali, T. G., Goyal, A., Huang, D., Jiang, F. & Wen, Q. (2020), 'Predicting corporate bond returns: Merton meets machine learning', *Georgetown McDonough School of Business Research Paper* .

Baltussen, G., van Vliet, B. & Van Vliet, P. (2021), The cross-section of stock returns before 1926 (and beyond), Technical report, Erasmus University, Rotterdam.

Bancel, F., Glavas, D. & Karolyi, G. A. (2023), 'Do esg factors influence firm valuation? evidence from the field', *Evidence from the Field (February 20, 2023)* .

Bartram, S. M., Branke, J., De Rossi, G. & Motahari, M. (2021), 'Machine learning for active portfolio management', *Journal of Financial Data Science* **3**(3), 9–30.

Beketov, M., Lehmann, K. & Wittke, M. (2018), 'Robo advisors: quantitative methods inside the robots', *Journal of Asset Management* **19**(6), 363–370.

Belkin, M., Hsu, D., Ma, S. & Mandal, S. (2019), 'Reconciling modern machine-learning practice and the classical bias–variance trade-off', *Proceedings of the National Academy of Sciences* **116**(32), 15849–15854.

Benhamou, E., Saltiel, D., Ohana, J.-J., Atif, J. & Laraki, R. (2021), Deep reinforcement learning (drl) for portfolio allocation, *in* 'Machine Learning and Knowledge Discovery in Databases. Applied Data Science and Demo Track: European Conference, ECML PKDD 2020, Ghent, Belgium, September 14–18, 2020, Proceedings, Part V', Springer International Publishing, pp. 527–531.

Berg, F., Koelbel, J. F. & Rigobon, R. (2022), 'Aggregate confusion: The divergence of esg ratings', *Review of Finance* **26**(6), 1315–1344.

Bergstra, J. & Bengio, Y. (2012), 'Random search for hyper-parameter optimization.', *Journal of machine learning research* **13**(2).

Bertoluzzo, F. & Corazza, M. (2012), 'Testing different reinforcement learning configurations for financial trading: Introduction and applications', *Procedia Economics and Finance* **3**, 68–77.

Bertsimas, D. & Lo, A. W. (1998), 'Optimal control of execution costs', *Journal of Financial Markets* **1**(1), 1–50.

Betancourt, C. & Chen, W.-H. (2021), 'Deep reinforcement learning for portfolio management of markets with a dynamic number of assets', *Expert Systems with Applications* **164**, 114002.

Bianchi, D., Büchner, M. & Tamoni, A. (2021), 'Bond risk premiums with machine learning', *Review of Financial Studies* **34**(2), 1046–1089.

Bianchi, D. & Guidolin, M. (2014), 'Can long-run dynamic optimal strategies outperform fixed-mix portfolios? evidence from multiple data sets', *European Journal of Operational Research* **236**(1), 160–176.

Billio, M., Costola, M., Hristova, I., Latino, C. & Pelizzon, L. (2021), 'Inside the esg ratings:(dis) agreement and performance', *Corporate Social Responsibility and Environmental Management* **28**(5), 1426–1445.

Blanqu'e, P., Slimane, M. B., Cherief, A., Le Guenedal, T., Sekine, T. & Stagnol, L. (2022), 'The benefit of narratives for prediction of the sp 500 index', *Journal of Financial Data Science* **4**(4), 72–94.

Bolton, P. & Kacperczyk, M. (2021), 'Do investors care about carbon risk?', *Journal of financial economics* **142**(2), 517–549.

Booth, A., Gerding, E. & McGroarty, F. (2015), 'Performance-weighted ensembles of random forests for predicting price impact', *Quantitative Finance* **15**(11), 1823–1835.

Borghi, R. & De Rossi, G. (2020), The artificial intelligence approach to picking stocks, *in* 'Machine Learning for Asset Management: New Developments and Financial Applications', pp. 115–166.

Bradrania, R. & Pirayesh Neghab, D. (2022), 'State-dependent asset allocation using neural networks', *The European Journal of Finance* **28**(11), 1130–1156.

Branke, J., Scheckenbach, B., Stein, M., Deb, K. & Schmeck, H. (2009), 'Portfolio optimization with an envelope-based multi-objective evolutionary algorithm', *European Journal of Operational Research* **199**(3), 684–693.

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

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

Breiman, L. (2001*b*), 'Statistical modeling: The two cultures (with comments and a rejoinder by the author)', *Statistical science* **16**(3), 199–231.

Breiman, L., Friedman, J., Stone, C. J. & Olshen, R. A. (1984), 'Classification algorithms and regression trees', *Classification and regression trees* **15**(2), 246.

Brenner, L. & Meyll, T. (2020), 'Robo-advisors: a substitute for human financial advice?', *Journal of Behavioral and Experimental Finance* **25**, 100275.

Briere, M., Lehalle, C., Nefedova, T. & Raboun, A. (2020), *Modeling transaction costs when trades may be crowded: A Bayesian network using partially observable orders imbalance*, pp. 387–430.

Bryzgalova, S., Pelger, M. & Zhu, J. (2019), 'Forest through the trees'. Working paper, London Business School.

Bryzgalova, S., Pelger, M. & Zhu, J. (2020), 'Forest through the trees: Building cross-sections of stock returns', *Available at SSRN 3493458* .

Buehler, H., Gonon, L., Teichmann, J. & Wood, B. (2019), 'Deep hedging', *Quantitative Finance* **19**(8), 1271–1291.

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

Cao, L. J. & Tay, F. E. H. (2003), 'Support vector machine with adaptive parameters in financial time series forecasting', *IEEE Transactions on Neural Networks* **14**(6), 1506–1518.

Carbonell, J. G., Michalski, R. S. & Mitchell, T. M. (1983), 'An overview of machine learning', *Machine learning* pp. 3–23.

Carta, S., Consoli, S., Podda, A. S., Recupero, D. R. & Stanciu, M. M. (2022), 'Statistical arbitrage powered by explainable artificial intelligence', *Expert Systems With Applications* **206**, 117763.

Chan, K.-S. & Chen, K. (2011), 'Subset arma selection via the adaptive lasso', *Statistics and its Interface* **4**(2), 197–205.

Chaouki, A., Hardiman, S., Schmidt, C., Sérié, E. & De Lataillade, J. (2020), 'Deep deterministic portfolio optimization', *Journal of Finance and Data Science* **6**, 16–30.

Chapados, N. & Bengio, Y. (2001), 'Cost functions and model combination for var-based asset allocation using neural networks', *IEEE Transactions on Neural Networks* **12**(4), 890–906.

Checkley, M. S., Hig'on, D. A. & Alles, H. (2017), 'The hasty wisdom of the mob: How market sentiment predicts stock market behavior', *Expert Systems with Applications* **77**, 256–263.

Chen, L., Pelger, M. & Zhu, J. (2024), 'Deep learning in asset pricing', *Management Science* **70**(2), 714–750.

Chen, S., Härdle, W. K. & Jeong, K. (2010), 'Forecasting volatility with support vector machine-based garch model', *Journal of Forecasting* **29**(4), 406–433.

Chen, W.-H., Shih, J.-Y. & Wu, S. (2006), 'Comparison of support-vector machines and back propagation neural networks in forecasting the six major asian stock markets', *International Journal of Electronic Finance* **1**(1), 49–67.

Chen, Y. & Hao, Y. (2018), 'Integrating principle component analysis and weighted support vector machine for stock trading signals prediction', *Neurocomputing* **321**, 381–402.

Chen, Y.-T., Yu, C.-Y. & Lin, S.-Y. (2024), 'An investment strategy based on news sentiment words and its empirical performance', *Journal of Investing* **Forthcoming**.

Chinco, A., Clark-Joseph, A. D. & Ye, M. (2019), 'Sparse signals in the cross-section of returns', *Journal of Finance* **74**(1), 449–492.

Chincoli, F. & Guidolin, M. (2017), 'Linear and nonlinear predictability in investment style factors: Multivariate evidence', *Journal of Asset Management* **18**, 476–509.

Chortareas, G., Kapetanios, G. & Liu, R. (2020), Currency portfolio selection with factors: additive gradients and model sparsity in a data-rich environment. Working paper, King's College London.

Chun, H. & Kele*c*s, S. (2010), 'Sparse partial least squares regression for simultaneous dimension reduction and variable selection', *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **72**(1), 3–25.

Comon, P. (1994), 'Independent component analysis, a new concept?', *Signal Processing* **36**(3), 287–314.

Cong, L. W., Tang, K., Wang, J. & Zhang, Y. (2020), 'Alphaportfolio: Direct construction through reinforcement learning and interpretable ai', *Working Paper* .

Conlon, T., Cotter, J. & Kynigakis, I. (2021), 'Machine learning and factor-based portfolio optimization'.

Coqueret, G. & Guida, T. (2020), 'Training trees on tails with applications to portfolio choice', *Annals of Operations Research* **288**(1), 181–221.

Corsi, F. (2009), 'A simple approximate long-memory model of realized volatility', *Journal of Financial Econometrics* **7**(2), 174–196.

Corsi, F. et al. (2012), Har modeling for realized volatility forecasting, *in* 'Handbook of Volatility Models and Their Applications', John Wiley & Sons, Inc., pp. 363–382.

Covel, M. (2007), *The complete turtle trader: The legend, the lessons, the results*, Collins.

Cybenko, G. (1989), 'Approximation by superpositions of a sigmoidal function', *Mathematics of Control, Signals and Systems* **2**(4), 303–314.

da S. Gomes, G. S., Ludermir, T. B. & Lima, L. M. (2011), 'Comparison of new activation functions in neural network for forecasting financial time series', *Neural Computing and Applications* **20**, 417–439.

Dab'erius, K., Granat, E. & Karlsson, P. (2019), 'Deep execution-value and policy based reinforcement learning for trading and beating market benchmarks', *Working paper, Linkoping University* .

Dai, Z., Li, T. & Yang, M. (2022), 'Forecasting stock return volatility: The role of shrinkage approaches in a data-rich environment', *Journal of Forecasting* **41**(5), 980–996.

Das, S. R. (2014), 'Text and context: Language analytics in finance', *Foundations and Trends® in Finance* **8**(3), 145–261.

Das, S. R. & Chen, M. Y. (2007), 'Yahoo! for amazon: Sentiment extraction from small talk on the web', *Management Science* **53**(9), 1375–1388.

de Prado, M. L. (2016), 'Building diversified portfolios that outperform out of sample', *Journal of Portfolio Management* **42**(4), 59.

de Prado, M. M. L. (2020), *Machine learning for asset managers*, Cambridge University Press.

Del Vitto, A., Marazzina, D. & Stocco, D. (2023), 'Esg ratings explainability through machine learning techniques', *Annals of Operations Research* pp. 1–30.

DeMiguel, V., Garlappi, L. & Uppal, R. (2009), 'Optimal versus naive diversification: How inefficient is the 1/n portfolio strategy?', *Review of Financial Studies* **22**(5), 1915–1953.

Dixon, M. F., Polson, N. G. & Sokolov, V. O. (2019), 'Deep learning for spatio-temporal modeling: dynamic traffic flows and high frequency trading', *Applied Stochastic Models in Business and Industry* **35**(3), 788–807.

Donaldson, R. G. & Kamstra, M. (1997), 'An artificial neural network-garch model for international stock return volatility', *Journal of Empirical Finance* **4**(1), 17–46.

Du, J. (2022), 'Mean–variance portfolio optimization with deep learning based-forecasts for cointegrated stocks', *Expert Systems with Applications* **201**, 117005.

Du, J., Jin, M., Kolm, P. N., Ritter, G., Wang, Y. & Zhang, B. (2020), 'Deep reinforcement learning for option replication and hedging', *The Journal of Financial Data Science* **2**(4), 44–57.

D'Acunto, F., Prabhala, N. & Rossi, A. G. (2019), 'The promises and pitfalls of robo-advising', *Review of Financial Studies* **32**(5), 1983–2020.

D'Amato, V., D'Ecclesia, R. & Levantesi, S. (2022), 'Esg score prediction through random forest algorithm', *Computational Management Science* **19**(2), 347–373.

Edirisinghe, C. & Jeong, J. (2024), 'Data-driven mean–variance sparse portfolio selection under leverage control', *Journal of Portfolio Management* **Forthcoming**.

Efron, B. & Morris, C. (1973), 'A bayesian derivation of the james-stein estimator', *Journal of the American Statistical Association* **68**(341), 117.

Egger, D. J., Gambella, C., Marecek, J., McFaddin, S., Mevissen, M., Raymond, R. & Yndurain, E. (2020), 'Quantum computing for finance: State-of-the-art and future prospects', *IEEE Transactions on Quantum Engineering* **1**, 1–24.

Eghtesad, A. & Mohammadi, E. (2024), 'Portfolio optimization under ambiguity aversion using deep learning, machine learning, and time series.', *Journal of Financial Data Science* **6**(2).

El-Haj, M., Rayson, P., Walker, M., Young, S. & Simaki, V. (2019), 'In search of meaning: Lessons, resources and next steps for computational analysis of financial discourse', *Journal of Business Finance & Accounting* **46**(3-4), 265–306.

Faloon, M. & Scherer, B. (2017), 'Individualization of robo-advice', *The Journal of Wealth Management* **20**(1), 30.

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

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

Fan, J. & Li, R. (2001), 'Variable selection via nonconcave penalized likelihood and its oracle properties', *Journal of the American Statistical Association* **96**(456), 1348–1360.

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

Feng, G., He, J., Polson, N. G. & Xu, J. (2018), 'Deep learning in characteristics-sorted factor models', *arXiv preprint arXiv:1805.01104* .

Fischer, T. G. (2018), Reinforcement learning in financial markets-a survey, Technical Report 12/2018, FAU Discussion Papers in Economics.

Fisher, I. E., Garnsey, M. R. & Hughes, M. E. (2016), 'Natural language processing in accounting, auditing and finance: A synthesis of the literature with a roadmap for future research', *Intelligent Systems in Accounting, Finance and Management* **23**(3), 157–214.

Foerster, S., Linnainmaa, J. T., Melzer, B. T. & Previtero, A. (2017), 'Retail financial advice: does one size fit all?', *Journal of Finance* **72**(4), 1441–1482.

Fons, E., Dawson, P., Yau, J., Zeng, X. & Keane, J. (2021), 'A novel dynamic asset allocation system using feature saliency hidden markov models for smart beta investing', *Expert Systems with Applications* **163**, 113720.

Franses, P. H. & Van Dijk, D. (2000), *Non-linear time series models in empirical finance*, Cambridge university press.

Freund, Y. (1995), 'Boosting a weak learning algorithm by majority', *Information and Computation* **121**(2), 256–285.

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

Friedman, J. H. (2001), 'Greedy function approximation: A gradient boosting machine', *The Annals of Statistics* **29**(5), 1189–1232.

Friedman, J. H. (2012), 'Fast sparse regression and classification', *International Journal of Forecasting* **28**(3), 722–738.

Friedman, J., Hastie, T. & Tibshirani, R. (2008), 'Sparse inverse covariance estimation with the graphical lasso', *Biostatistics* **9**(3), 432–441.

Fugazza, C., Guidolin, M. & Nicodano, G. (2015), 'Equally weighted vs. long-run optimal portfolios', *European Financial Management* **21**(4), 742–789.

Füss, R., Guidolin, M. & Koeppel, C. (2020), 'Sentiment risk premia in the cross-section of global equity', *University of St. Gallen, School of Finance Research Paper* (2019/13).

Garcia, D., Hu, X. & Rohrer, M. (2023), 'The colour of finance words', *Journal of Financial Economics* **147**(3), 525–549.

Garc'ıa-Galicia, M., Carsteanu, A. A. & Clempner, J. B. (2019), 'Continuous-time reinforcement learning approach for portfolio management with time penalization', *Expert Systems with Applications* **129**, 27–36.

Gennaioli, N., Shleifer, A. & Vishny, R. (2015), 'Money doctors', *The Journal of Finance* **70**(1), 91–114.

Gerakos, J., Linnainmaa, J. T. & Morse, A. (2016), Asset managers: Institutional performance and smart betas, Technical report, National Bureau of Economic Research.

Geurts, P., Ernst, D. & Wehenkel, L. (2006), 'Extremely randomized trees', *Machine Learning* **63**(1), 3–42.

Giglio, S., Kelly, B. & Stroebel, J. (2021), 'Climate finance', *Annual Review of Financial Economics* **13**, 15–36.

Gilboa, I. & Schmeidler, D. (2010), 'Simplicity and likelihood: An axiomatic approach', *Journal of Economic Theory* **145**(5), 1757–1775.

Glasserman, P. & Lin, C. (2024), 'Assessing look-ahead bias in stock return predictions generated by gpt sentiment analysis', *The Journal of Financial Data Science* **6**(1), 25–42.

Goetzmann, W., Ingersoll, J., Spiegel, M. & Welch, I. (2007), 'Portfolio performance manipulation and manipulation-proof performance measures', *The Review of Financial Studies* **20**(5), 1503–1546.

Goetzmann, W. N., Brown, S. J., Gruber, M. J. & Elton, E. J. (2014), *Modern Portfolio Theory and Investment Analysis*, John Wiley Sons.

Goldstein, I., Jiang, W. & Karolyi, G. A. (2019), 'To fintech and beyond', *Review of Financial Studies* **32**(5), 1647–1661.

Gorman, S. A. & Fabozzi, F. J. (2022), 'The data dilemma in alternative risk premium: why is a benchmark so elusive?', *Journal of Portfolio Management* **48**(5), 219–265.

Grimmer, J. & Stewart, B. M. (2013), 'Text as data: The promise and pitfalls of automatic content analysis methods for political texts', *Political analysis* **21**(3), 267–297.

Grinold, R. C. & Kahn, R. N. (2000), 'Active portfolio management'.

Groß-Klußmann, A., König, S. & Ebner, M. (2019), 'Buzzwords build momentum: Global financial twitter sentiment and the aggregate stock market', *Expert Systems with Applications* **136**, 171–186.

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

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

Guidolin, M. (2013), Preference models in portfolio construction and evaluation, *in* 'Portfolio Theory and Management', Oxford University Press, pp. 231–267.

Guidolin, M. & Panzeri, G. F. (2024), 'Forecasting the cboe vix and skew indices using heterogeneous autoregressive models', *Forecasting* **6**(3), 782–814.

Guidolin, M. & Pedio, M. (2018), *Essentials of Time Series for Financial Applications*, Academic Press.

Guidolin, M. & Pedio, M. (2022), 'Switching coefficients or automatic variable selection: An application in forecasting commodity returns', *Forecasting* **4**(1), 275–306.

Guidolin, M. & Rinaldi, F. (2013), 'Ambiguity in asset pricing and portfolio choice: A review of the literature', *Theory and Decision* **74**, 183–217.

Gupta, D., Hazarika, B. B. & Berlin, M. (2020), 'Robust regularized extreme learning machine with asymmetric huber loss function', *Neural Computing and Applications* **32**(16), 12971–12998.

Györfi, L., Kohler, M., Krzyzak, A. & Walk, H. (2006), *A distribution-free theory of nonparametric regression*, Springer Science & Business Media.

Hackethal, A., Haliassos, M. & Jappelli, T. (2012), 'Financial advisors: A case of babysitters?', *Journal of Banking & Finance* **36**(2), 509–524.

Hansen, L. K. & Salamon, P. (1990), 'Neural network ensembles', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **12**(10), 993–1001.

Hao, J., He, F., Ma, F., Zhang, S. & Zhang, X. (2023), 'Machine learning vs deep learning in stock market investment: An international evidence', *Annals of Operations Research* pp. 1–23. Forthcoming.

Harjoto, M., Laksmana, I. & Lee, R. (2015), 'Board diversity and corporate social responsibility', *Journal of Business Ethics* **132**, 641–660.

Harvey, C. R., Liu, Y. & Zhu, H. (2016), '... and the cross-section of expected returns', *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, New York.

Hawley, D. D., Johnson, J. D. & Raina, D. (1990), 'Artificial neural systems: A new tool for financial decision-making', *Financial Analysts Journal* **46**(6), 63–72.

Hayashi, F. (2011), *Econometrics*, Princeton University Press.

He, A., Huang, D., Li, J. & Zhou, G. (2022), 'Shrinking factor dimension: A reduced-rank approach', *Management Science* p. Forthcoming.

Heaton, J. B., Polson, N. G. & Witte, J. H. (2016), 'Deep portfolio theory', *arXiv preprint arXiv:1605.07230* .

Heaton, J. B., Polson, N. G. & Witte, J. H. (2017), 'Deep learning for finance: deep portfolios', *Applied Stochastic Models in Business and Industry* **33**(1), 3–12.

Henrique, B. M., Sobreiro, V. A. & Kimura, H. (2018), 'Stock price prediction using support vector regression on daily and up to the minute prices', *Journal of Finance and Data Science* **4**(3), 183–201.

Heston, S. L. & Sinha, N. R. (2017), 'News vs. sentiment: Predicting stock returns from news stories', *Financial Analysts Journal* **73**(3), 67–83.

Hoepner, A. G., McMillan, D., Vivian, A. & Wese Simen, C. (2021), 'Significance, relevance and explainability in the machine learning age: an econometrics and financial data science perspective', *European Journal of Finance* **27**(1-2), 1–7.

Hornik, K., Stinchcombe, M. & White, H. (1989), 'Multilayer feedforward networks are universal approximators', *Neural networks* **2**(5), 359–366.

Hsu, J., Liu, X., Viswanathan, V. & Xia, Y. (2022), 'When smart beta meets machine learning and portfolio optimization', *The Journal of Beta Investment Strategies* **13**(4), 123–146.

Huang, W.-Y., Nakamori, Y. & Wang, S.-Y. (2005), 'Forecasting stock market movement direction with support vector machine', *Computers Operations Research* **32**(10), 2513–2522.

Hultin, H., Hult, H., Proutiere, A., Samama, S. & Tarighati, A. (2024), 'A deterministic policy gradient method for order execution and option hedging in the presence of market impact', *Journal of Financial Data Science* **6**(3).

Inc., B. (2017), 'Could deep learning dethrone hft?', https://www.bloomberg.com/professional/blog/deep-learning-dethrone-hft/.

Iskhakov, F., Rust, J. & Schjerning, B. (2020), 'Machine learning and structural econometrics: contrasts and synergies', *Econometrics Journal* **23**(3), S81–S124.

Israel, R., Kelly, B. T. & Moskowitz, T. J. (2020), 'Can machines' learn'finance?', *Journal of Investment Management* **18**(2), 23–36.

Jacobs, B. I. & Levy, K. N. (2014), 'Investing in a multidimensional market'.

Jaeger, M., Krügel, S., Marinelli, D., Papenbrock, J. & Schwendner, P. (2021), 'Interpretable machine learning for diversified portfolio construction', *The Journal of Financial Data Science* **3**(3), 31–51.

Jang, J. & Seong, N. (2023), 'Deep reinforcement learning for stock portfolio optimization by connecting with modern portfolio theory', *Expert Systems with Applications* **119556**.

Jiang, M., Liu, J., Zhang, L. & Liu, C. (2020), 'An improved stacking framework for stock index prediction by leveraging tree-based ensemble models and deep learning algorithms', *Physica A: Statistical Mechanics and its Applications* **541**, 122272.

Jiang, W. (2021), 'Applications of deep learning in stock market prediction: recent progress', *Expert Systems with Applications* **184**, 115537.

Jiang, Z., Xu, D. & Liang, J. (2017), 'A deep reinforcement learning framework for the financial portfolio management problem', *arXiv preprint arXiv:1706.10059* .

Jin, S., Su, L. & Ullah, A. (2014), 'Robustify financial time series forecasting with bagging', *Econometric Reviews* **33**(5-6), 575–605.

Kaufmann, C., Weber, M. & Haisley, E. (2013), 'The role of experience sampling and graphical displays on one's investment risk appetite', *Management science* **59**(2), 323–340.

Ke, Z. T., Kelly, B. T. & Xiu, D. (2019), Predicting returns with text data, Technical report, National Bureau of Economic Research.

Kelly, B., Palhares, D. & Pruitt, S. (2023), 'Modeling corporate bond returns', *The Journal of Finance* **78**(4), 1967–2008.

Kelly, B. & Pruitt, S. (2015), 'The three-pass regression filter: A new approach to forecasting using many predictors', *Journal of Econometrics* **186**(2), 294–316.

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.

Kim, K.-J. (2003), 'Financial time series forecasting using support vector machines', *Neurocomputing* **55**(1-2), 307–319.

Kim, S. & Kim, S. (2020), 'Index tracking through deep latent representation learning', *Quantitative Finance* **20**(4), 639–652.

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

Knight, K. & Fu, W. (2000), 'Asymptotics for lasso-type estimators', *Annals of Statistics* pp. 1356–1378.

Kolm, P. N. & Ritter, G. (2019), 'Dynamic replication and hedging: A reinforcement learning approach', *Journal of Financial Data Science* **1**(1), 159–171.

Kolm, P. N., Tütüncü, R. & Fabozzi, F. J. (2014), '60 years of portfolio optimization: Practical challenges and current trends', *European Journal of Operational Research* **234**(2), 356–371.

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

Krauss, C., Do, X. A. & Huck, N. (2017), 'Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the sp 500', *European Journal of Operational Research* **259**(2), 689–702.

Kristjanpoller, W., Fadic, A. & Minutolo, M. C. (2014), 'Volatility forecast using hybrid neural network models', *Expert Systems with Applications* **41**(5), 2437–2442.

Kristjanpoller, W. & Minutolo, M. C. (2015), 'Gold price volatility: A forecasting approach using the artificial neural network–garch model', *Expert systems with applications* **42**(20), 7245–7251.

Kristjanpoller, W. & Minutolo, M. C. (2016), 'Forecasting volatility of oil price using an artificial neural network-garch model', *Expert Systems with Applications* **65**, 233–241.

Kryzanowski, L., Galler, M. & Wright, D. W. (1993), 'Using artificial neural networks to pick stocks', *Financial Analysts Journal* **49**(4), 21–27.

Kwak, Y., Song, J. & Lee, H. (2021), 'Neural network with fixed noise for index-tracking portfolio optimization', *Expert Systems with Applications* **183**, 115298.

Kynigakis, I. & Panopoulou, E. (2022), 'Does model complexity add value to asset allocation? evidence from machine learning forecasting models', *Journal of Applied Econometrics* **37**(3), 603–639.

Le, Q., Karpenko, A., Ngiam, J. & Ng, A. (2011), 'Ica with reconstruction cost for efficient overcomplete feature learning', *Advances in Neural Information Processing Systems* **24**.

Leal, L., Laurière, M. & Lehalle, C.-A. (2022), 'Learning a functional control for high-frequency finance', *Quantitative Finance* **22**(11), 1973–1987.

Ledoit, O. & Wolf, M. (2004), 'A well-conditioned estimator for large-dimensional covariance matrices', *Journal of multivariate analysis* **88**(2), 365–411.

Lee, J., Park, S., Ahn, J. & Kwak, J. (2022), 'Etf portfolio construction via neural network trained on financial statement data', *arXiv preprint arXiv:2207.01187* .

Lee, M.-C. (2009), 'Using support vector machine with a hybrid feature selection method to the stock trend prediction', *Expert Systems with Applications* **36**(8), 10896–10904.

Lee, Y., Thompson, J. R., Kim, J. H., Kim, W. C. & Fabozzi, F. A. (2023), 'An overview of machine learning for asset management.', *Journal of Portfolio Management* **49**(9).

Leung, E., Lohre, H., Mischlich, D., Shea, Y. & Stroh, M. (2021), 'The promises and pitfalls of machine learning for predicting stock returns', *The Journal of Financial Data Science* **3**(2), 21–50.

Li, F. (2010), 'Textual analysis of corporate disclosures: A survey of the literature', *Journal of the Accounting Literature* **29**(1), 143–165.

Li, X., Cao, J. & Pan, Z. (2019), 'Market impact analysis via deep learned architectures', *Neural Computing and Applications* **31**, 5989–6000.

Li, Y., Simon, Z. & Turkington, D. (2022), 'Investable and interpretable machine learning for equities', *The Journal of Financial Data Science* **4**(1), 54–74.

Li, Y., Turkington, D. & Yazdani, A. (2019), 'Beyond the black box: an intuitive approach to investment prediction with machine learning', *The Journal of Financial Data Science* .

Liew, J. & Budavari, T. (2017), 'The "sixth" factor—a social media factor derived directly from tweet sentiments', *The Journal of Portfolio Management* **43**(3), 102–111.

Light, N., Maslov, D. & Rytchkov, O. (2017), 'Aggregation of information about the cross section of stock returns: A latent variable approach', *Review of Financial Studies* **30**(4), 1339–1381.

Ligozat, A.-L., Lefèvre, J., Bugeau, A. & Combaz, J. (2021), 'Unraveling the hidden environmental impacts of ai solutions for environment', *arXiv preprint arXiv:2110.11822* .

Linnainmaa, J. T., Melzer, B. T. & Previtero, A. (2021), 'The misguided beliefs of financial advisors', *The Journal of Finance* **76**(2), 587–621.

Liu, H., Ning, R., Teng, Z., Liu, J., Zhou, Q. & Zhang, Y. (2023), 'Evaluating the logical reasoning ability of chatgpt and gpt-4', *arXiv preprint arXiv:2304.03439* .

Lo, A. W. & Singh, M. (2023), 'Deep-learning models for forecasting financial risk premia and their interpretations', *Quantitative Finance* **23**(6), 917–929.

Lo, A. W., Singh, M., Musumeci, J., Nagy, Z., Giese, G., Wang, X., Mirabelli, A., Keywork, N., Turetsky, A., Griffiths, B. et al. (2023), 'From eliza to chatgpt: The evolution of natural language processing and financial applications', *The Journal of Portfolio Management* .

López de Prado, M. (2022), 'Machine learning for econometricians: The readme manual.', *Journal of Financial Data Science* **4**(3).

Lopez-Lira, A. & Tang, Y. (2023), 'Can chatgpt forecast stock price movements? return predictability and large language models', *arXiv preprint arXiv:2304.07619* .

Loughran, T. & McDonald, B. (2011), 'When is a liability not a liability? textual analysis, dictionaries, and 10-ks', *The Journal of finance* **66**(1), 35–65.

Loughran, T. & McDonald, B. (2016), 'Textual analysis in accounting and finance: A survey', *Journal of Accounting Research* **54**(4), 1187–1230.

Lowe, D. (1994), Novel exploitation of neural network methods in financial markets, *in* 'Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)', Vol. 6, IEEE, pp. 3623–3628.

Luo, L. & Chen, X. (2013), 'Integrating piecewise linear representation and weighted support vector machine for stock trading signal prediction', *Applied Soft Computing* **13**(2), 806–816.

Lv, J. & Fan, Y. (2009), 'A unified approach to model selection and sparse recovery using regularized least squares', *The Annals of Statistics* **37**(6A), 3498–3528.

Ma, L. (2020), *Quantitative Investing: From Theory to Industry*, Springer Nature.

Ma, Y., Han, R. & Wang, W. (2021), 'Portfolio optimization with return prediction using deep learning and machine learning', *Expert Systems with Applications* **165**, 113973.

Maguire, P., Moffett, K. & Maguire, R. (2018), 'Combining independent smart beta strategies for portfolio optimization', *arXiv preprint arXiv:1808.02505* .

Manela, A. & Moreira, A. (2017), 'News implied volatility and disaster concerns', *Journal of Financial Economics* **123**(1), 137–162.

Markowitz, H. (1952), 'Portfolio selection', *Journal of Finance* **7**(1), 77–91.

Masini, R. P., Medeiros, M. C. & Mendes, E. F. (2023), 'Machine learning advances for time series forecasting', *Journal of economic surveys* **37**(1), 76–111.

McGrath, T., Kapishnikov, A., Tomašev, N., Pearce, A., Wattenberg, M., Hassabis, D., Kim, B., Paquet, U. & Kramnik, V. (2022), 'Acquisition of chess knowledge in alphazero', *Proceedings of the National Academy of Sciences* **119**(47), e2206625119.

Messmer, M. & Audrino, F. (2017), 'The (adaptive) lasso in the zoo-firm characteristic selection in the cross-section of expected returns', *Working paper, University of St. Gallen* .

Metaxiotis, K. & Liagkouras, K. (2012), 'Multiobjective evolutionary algorithms for portfolio management: A comprehensive literature review', *Expert Systems with Applications* **39**(14), 11685–11698.

Michaud, R. O. (1989), 'The markowitz optimization enigma: Is 'optimized'optimal?', *Financial analysts journal* **45**(1), 31–42.

Molnar, C. (2020), *Interpretable machine learning*, Lulu. com.

Moody, J., Wu, L., Liao, Y. & Saffell, M. (1998), 'Performance functions and reinforcement learning for trading systems and portfolios', *Journal of Forecasting* **17**(5-6), 441–470.

Mounjid, O. & Lehalle, C.-A. (2019), 'Improving reinforcement learning algorithms: towards optimal learning rate policies', *Mathematical Finance* .

Mourtas, S. D. & Katsikis, V. N. (2022), 'Exploiting the black-litterman framework through error-correction neural networks', *Neurocomputing* **498**, 43–58.

Mueller-Glissmann, C. & Ferrario, A. (2024), 'Dynamic asset allocation using machine learning: Seeing the forest for the trees.', *Journal of Portfolio Management* **50**(5).

Mullainathan, S., Noeth, M. & Schoar, A. (2012), The market for financial advice: An audit study, Technical report, National Bureau of Economic Research.

Mullainathan, S. & Spiess, J. (2017), 'Machine learning: an applied econometric approach', *Journal of Economic Perspectives* **31**(2), 87–106.

Neal, R. M. (2012), *Bayesian learning for neural networks*, Vol. 118, Springer Science & Business Media.

Ngo, V. M., Nguyen, H. H. & Van Nguyen, P. (2023), 'Does reinforcement learning outperform deep learning and traditional portfolio optimization models in frontier and developed financial markets?', *Research in International Business and Finance* **65**, 101936.

Nguyen, Q., Diaz-Rainey, I. & Kuruppuarachchi, D. (2021), 'Predicting corporate carbon footprints for climate finance risk analyses: a machine learning approach', *Energy Economics* **95**, 105129.

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

Oliveira, N., Cortez, P. & Areal, N. (2017), 'The impact of microblogging data for stock market prediction: Using twitter to predict returns, volatility, trading volume and survey sentiment indices', *Expert Systems with applications* **73**, 125–144.

Ouyang, H., Zhang, X. & Yan, H. (2019), 'Index tracking based on deep neural network', *Cognitive Systems Research* **57**, 107–114.

Paiva, F. D., Cardoso, R. T. N., Hanaoka, G. P. & Duarte, W. M. (2019), 'Decision-making for financial trading: A fusion approach of machine learning and portfolio selection', *Expert Systems with Applications* **115**, 635–655.

Park, S., Lee, J. & Son, Y. (2016), 'Predicting market impact costs using nonparametric machine learning models', *PLoS One* **11**(2), e0150243.

Pedio, M. (2024), Natural language processing and stock returns, *in* 'Artificial Intelligence and Beyond for Finance', World Scientific, pp. 73–102.

Peng, Y., Albuquerque, P. H. M., de Sá, J. M. C., Padula, A. J. A. & Montenegro, M. R. (2018), 'The best of two worlds: Forecasting high frequency volatility for cryptocurrencies and traditional currencies with support vector regression', *Expert Systems with Applications* **97**, 177–192.

Philip, R. (2020), 'Estimating permanent price impact via machine learning', *Journal of Econometrics* **215**(2), 414–449.

Phoon, K. F. & Koh, C. C. F. (2018), 'Robo-advisors and wealth management', *Journal of Alternative Investments* **20**(3), 79.

Pirayesh, H. & Sallan, J. (2024), 'Advancements in ai for predicting price movements: an in-depth literature survey', *Journal of Financial Data Science* **Forthcoming**.

Politis, D. N. & Romano, J. P. (1992), 'A general resampling scheme for triangular arrays of α-mixing random variables with application to the problem of spectral density estimation', *The Annals of Statistics* **20**(4), 1985–2007.

Potter, S. (1999), 'Nonlinear time series modelling: An introduction', *Journal of Economic Surveys* **13**(5), 505–528.

Ramos-Pérez, E., Alonso-González, P. J. & Núñez-Velázquez, J. J. (2019), 'Forecasting volatility with a stacked model based on a hybridized artificial neural network', *Expert Systems with Applications* **129**, 1–9.

Rapach, D. E., Strauss, J. K. & Zhou, G. (2013), 'International stock return predictability: What is the role of the united states?', *Journal of Finance* **68**(4), 1633–1662.

Rapach, D., Strauss, J. K., Tu, J. & Zhou, G. (2019), 'Industry return predictability: A machine learning approach', *Working paper, Washington University in St. Louis* .

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

Rather, A. M., Agarwal, A. & Sastry, V. (2015), 'Recurrent neural network and a hybrid model for prediction of stock returns', *Expert Systems with Applications* **42**(6), 3234–3241.

Romanko, O., Narayan, A. & Kwon, R. H. (2023), 'Chatgpt-based investment portfolio selection', *arXiv preprint arXiv:2308.06260* .

Rossi, A. G. & Utkus, S. P. (2020), 'Who benefits from robo-advising? evidence from machine learning'. Working paper, Georgetown University.

Routledge, B. R. (2019), 'Machine learning and asset allocation', *Financial Management* **48**(4), 1069–1094.

Sadorsky, P. (2022), 'Forecasting solar stock prices using tree-based machine learning classification: How important are silver prices?', *The North American Journal of Economics and Finance* **61**, 101705.

Saifan, R., Sharif, K., Abu-Ghazaleh, N. & Abdel-Majeed, M. (2020), 'Investigating algorithmic stock market trading using ensemble machine learning methods', *Informatica* **44**(3), 311–325.

Schapire, R. E. (1990), 'The strength of weak learnability', *Machine learning* **5**, 197–227.

Scherer, B. & Lehner, S. (2023), 'Trust me, i am a robo-advisor', *Journal of Asset Management* **24**(2), 85–96.

Schmidhuber, J. (2015), 'Deep learning in neural networks: An overview', *Neural networks* **61**, 85–117.

Schultz, T. & Maedche, A. (2023), 'Biosignals meet adaptive systems', *SN Applied Sciences* **5**(9), 234.

Sezer, O. B. & Ozbayoglu, A. M. (2018), 'Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach', *Applied Soft Computing* **70**, 525–538.

Shalit, H. (2021), 'The shapley value decomposition of optimal portfolios', *Annals of Finance* **17**(1), 1–25.

Shi, Q. (2023), 'The rp-pca factors and stock return predictability: An aligned approach', *North American Journal of Economics and Finance* **64**, 101862.

Simonian, J. (2024), 'Using econometrics vs. machine learning: what, when, and how', *Journal of Portfolio Management* **Forthcoming**.

Skolpadungket, P., Dahal, K. & Harnpornchai, N. (2016), 'Handling model risk in portfolio selection using multi-objective genetic algorithm', *Artificial Intelligence in Financial Markets: Cutting Edge Applications for Risk Management, Portfolio Optimization and Economics* pp. 285–310.

Smith, A. (2021), 'Goldman sachs launches pre-trade analytics on newly merged ts imagine platform', *The Trade* .
URL: *https://www.thetradenews.com/goldman-sachs-launches-pre-trade-analytics-on-newly-merged-ts-imagine-platform/*

Snow, D. (2020*a*), 'Machine learning in asset management—part 1: Portfolio construction—trading strategies', *The Journal of Financial Data Science* **2**(1), 10–23.

Snow, D. (2020*b*), 'Machine learning in asset management—part 2: Portfolio construction—weight optimization', *The Journal of Financial Data Science* **2**(2), 17–24.

Sohangir, S., Wang, D., Pomeranets, A. & Khoshgoftaar, T. M. (2018), 'Big data: Deep learning for financial sentiment analysis', *Journal of Big Data* **5**(1), 1–25.

Sokolov, A., Mostovoy, J., Ding, J. & Seco, L. (2021), 'Building machine learning systems for automated esg scoring', *The Journal of Impact and ESG Investing* **1**(3), 39–50.

Sprenger, T. O., Tumasjan, A., Sandner, P. G. & Welpe, I. M. (2014), 'Tweets and trades: The information content of stock microblogs', *European Financial Management* **20**(5), 926–957.

Strubell, E., Ganesh, A. & McCallum, A. (2019), 'Energy and policy considerations for deep learning in nlp', *arXiv preprint arXiv:1906.02243* .

Sun, Y., Liu, L., Xu, Y., Zeng, X. & Shi, Y. (2022), 'A survey on alternative data in finance and business: Emerging applications and theory analysis', *Available at SSRN 4148628* .

Swales Jr, G. S. & Yoon, Y. (1992), 'Applying artificial neural networks to investment analysis', *Financial Analysts Journal* **48**(5), 78–80.

Tang, Y., Song, Z., Zhu, Y., Yuan, H., Hou, M., Ji, J., Tang, C. & Li, J. (2022), 'A survey on machine learning models for financial time series forecasting', *Neurocomputing* **512**, 363–380.

Tashiro, D., Matsushima, H., Izumi, K. & Sakaji, H. (2019), 'Encoding of high-frequency order information and prediction of short-term stock price by deep learning', *Quantitative Finance* **19**(9), 1499–1506.

Tertilt, M. & Scholz, P. (2018), 'To advise, or not to advise—how robo-advisors evaluate the risk preferences of private investors', *The Journal of Wealth Management* **21**(2), 70–84.

Tetlock, P. C. (2007), 'Giving content to investor sentiment: The role of media in the stock market', *The Journal of finance* **62**(3), 1139–1168.

Tetlock, P. C., Saar-Tsechansky, M. & Macskassy, S. (2008), 'More than words: Quantifying language to measure firms' fundamentals', *Journal of Finance* **63**(3), 1437–1467.

The Economist (2022), 'Artificial intelligence's new frontier'.
URL: *https://www.economist.com/leaders/2022/06/09/artificial-intelligences-new-frontier*

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

Timmermann, A. (2006), 'Forecast combinations', *Handbook of Economic Forecasting* **1**, 135–196.

Tsai, C. F., Lin, Y. C., Yen, D. C. & Chen, Y. M. (2011), 'Predicting stock returns by classifier ensembles', *Applied Soft Computing* **11**(2), 2452–2459.

Tsang, K. H. & Wong, H. Y. (2020), 'Deep-learning solution to portfolio selection with serially dependent returns', *SIAM Journal on Financial Mathematics* **11**(2), 593–619.

Tsantekidis, A., Passalis, N., Tefas, A., Kanniainen, J., Gabbouj, M. & Iosifidis, A. (2020), 'Using deep learning for price prediction by exploiting stationary limit order book features', *Applied Soft Computing* **93**, 106401.

Tsay, R. S. & Chen, R. (2018), *Nonlinear time series analysis*, Vol. 891, John Wiley & Sons.

Uddin, A. & Yu, D. (2020), 'Latent factor model for asset pricing', *Journal of Behavioral and Experimental Finance* **27**, 100353.

Uhl, M. W. & Rohner, P. (2018), 'Robo-advisors versus traditional investment advisors: An unequal game', *The Journal of Wealth Management* **21**(1), 44–50.

Uysal, A. S. & Mulvey, J. M. (2021), 'A machine learning approach in regime-switching risk parity portfolios', *Journal of Financial Data Science* **3**(2), 87–108.

Van Atteveldt, W., Van der Velden, M. A. & Boukes, M. (2021), 'The validity of sentiment analysis: Comparing manual annotation, crowd-coding, dictionary approaches, and machine learning algorithms', *Communication Methods and Measures* **15**(2), 121–140.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. & Polosukhin, I. (2017), 'Attention is all you need', *Advances in neural information processing systems* **30**.

Wang, H., Li, G. & Tsai, C.-L. (2007), 'Regression coefficient and autoregressive order shrinkage and selection via the lasso', *Journal of the Royal Statistical Society Series B: Statistical Methodology* **69**(1), 63–78.

Wang, H. & Zhou, X. Y. (2020), 'Continuous-time mean-variance portfolio selection: A reinforcement learning framework', *Mathematical Finance* **30**(4), 1273–1308.

Wang, W., Li, W., Zhang, N. & Liu, K. (2020), 'Portfolio formation with preselection using deep learning from long-term financial data', *Expert Systems with Applications* **143**, 113042.

Wang, Y., Han, L., Sun, J. & Sun, Z. (2024), 'Financing the green transition: Mobilizing resources for efficient natural resource management', *Resources Policy* **89**, 104522.

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

Wold, H. (1980), *Model construction and evaluation when theoretical knowledge is scarce: Theory and application of partial least squares*, Academic Press, pp. 47–74.

Wolff, D. & Echterling, F. (2020), 'Stock picking with machine learning', *Working paper, Frankfurt University* .

Wong, F., Wang, P., Goh, T. & Quek, B. (1992), 'Fuzzy neural systems for stock selection', *Financial Analysts Journal* **48**(1), 47–52.

Wu, S., Irsoy, O., Lu, S., Dabravolski, V., Dredze, M., Gehrmann, S., Kambadur, P., Rosenberg, D. & Mann, G. (2023), 'Bloomberggpt: A large language model for finance', *arXiv preprint arXiv:2303.17564* .

Yamaka, W., Srichaikul, W. & Maneejuk, P. (2021), 'Support vector machine-based garch-type models: Evidence from asean-5 stock markets', *Data Science for Financial Econometrics* pp. 369–381.

Yoon, Y. J., Park, C. & Lee, T. (2013), 'Penalized regression models with autoregressive error terms', *Journal of Statistical Computation and Simulation* **83**(9), 1756–1772.

Yu, L., Wang, S. & Lai, K. K. (2008), 'Neural network-based mean–variance–skewness model for portfolio selection', *Computers & Operations Research* **35**(1), 34–46.

Zhang, C. H. (2010), 'Nearly unbiased variable selection under minimax concave penalty', *The Annals of Statistics* **38**(2), 894–942.

Zhang, G. & Qiao, G. (2021), 'Out-of-sample realized volatility forecasting: does the support vector regression compete combination methods', *Applied Economics* **53**(19), 2192–2205.

Zhang, Q. T., Li, B. & Xie, D. (2022), Smart beta and risk factors based on textural data and machine learning, *in* 'Alternative Data and Artificial Intelligence Techniques: Applications in Investment and Risk Management', Springer, pp. 111–128.

Zhang, Z., Zohren, S. & Roberts, S. (2020), 'Deep learning for portfolio optimization', *arXiv preprint arXiv:2005.13665* .

Zheng, B., Moulines, E. & Abergel, F. (2012), 'Price jump prediction in limit order book', *arXiv preprint arXiv:1204.1381* .

Zhu, Y. & Nowak, R. (2022), 'Active learning with neural networks: Insights from nonparametric statistics', *Advances in Neural Information Processing Systems* **35**, 142–155.

Zimmermann, H.-G., Neuneier, R. & Grothmann, R. (2001), 'Active portfolio-management based on error correction neural networks', *Advances in Neural Information Processing Systems* **14**.

Zou, H. (2006), 'The adaptive lasso and its oracle properties', *Journal of the American Statistical Association* **101**(476), 1418–1429.

Zou, H. & Hastie, T. (2005*a*), 'Regularization and variable selection via the elastic net', *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **67**(2), 301–320.

Zou, H. & Hastie, T. (2005*b*), 'Regularization and variable selection via the elastic net', *Journal of the Royal Statistical Society: series B (Statistical Methodology)* **67**(2), 301–320.

Zou, H., Hastie, T. & Tibshirani, R. (2006), 'Sparse principal component analysis', *Journal of computational and graphical statistics* **15**(2), 265–286.