Expert Systems With Applications 206 (2022) 117763

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Statistical arbitrage powered by Explainable Artificial Intelligence

Salvatore Carta [a], Sergio Consoli [b,*], Alessandro Sebastian Podda [a], Diego Reforgiato Recupero [a], Maria Madalina Stanciu [a]

[a] Department of Mathematics and Computer Science, University of Cagliari, 09124 Cagliari (CA), Italy
[b] European Commission, Joint Research Centre (JRC), Via E. Fermi 2749, I-21027 Ispra (VA), Italy

ARTICLE INFO

Keywords:
Stock market forecast
Statistical arbitrage
Machine learning
Explainable Artificial Intelligence

ABSTRACT

Machine learning techniques have recently become the norm for detecting patterns in financial markets. However, relying solely on machine learning algorithms for decision-making can have negative consequences, especially in a critical domain such as the financial one. On the other hand, it is well-known that transforming data into actionable insights can pose a challenge even for seasoned practitioners, particularly in the financial world. Given these compelling reasons, this work proposes a machine learning approach powered by eXplainable Artificial Intelligence techniques integrated into a statistical arbitrage trading pipeline. Specifically, we propose three methods to discard irrelevant features for the prediction task. We evaluate the approaches on historical data of component stocks of the S&P500 index and aim at improving not only the prediction performance at the stock level but also overall at the stock set level. Our analysis shows that our trading strategies that include such feature selection methods improve the portfolio performances by providing predictive signals whose information content suffices and is less noisy than the one embedded in the whole feature set. By performing an in-depth risk-return analysis, we show that the proposed trading strategies powered by explainable AI outperform highly competitive trading strategies considered as baselines.

1. Introduction

The quantitative trading strategy known as Pairs Trading[1] has become increasingly popular since the mid-1980s. The seminal paper by Gatev et al. (2006) is one of the first comprehensive studies on the profitability of a simple pairs trading strategy of U.S. stocks. This investing strategy identifies pairs of stocks whose prices have historically moved together (i.e., they have high correlation) over a specific period. Once having identified stocks that tended to behave in a "similar" way in the past, traders short the outperforming stock (i.e., betting on a value of a security to fall and profiting from that fall), and buy long the under-performing one, assuming an equilibrium relationship between the two securities (i.e., stocks) and a mean reversion occurring in price spread. This happens on the consideration that if the future performance resembles that of the past, then the relative deviation of price is temporary, and market prices are likely to correct and finally converge again, thereby generating profits. This concept of pairs trading can be extended to groups of stocks and this strategy is often referred to as generalized pairs trading or statistical arbitrage (StatArb) (Avellaneda & Lee, 2010; Khandani & Lo, 2011).

In Krauss (2017), the authors pointed out that although academic research about this strategy is still at its early stages, if compared with other quantitative trading strategies, StatArb is gaining momentum on different research streams: distance based (Gatev et al., 2006), co-integration based (Vidyamurthy, 2004), models based on stochastic spread (Kaufman & Lang, 2015), and a bucket of further approaches, that the authors label as "other approaches", that involve Machine Learning (ML) technologies. StatArb is a natural application field for ML as this variant of algorithmic trading involves a large number of securities and substantial computational, trading, and information technology infrastructures (Henrique et al., 2019; Lo, 2010).

In recent years, this trend of adopting ML has surged, due to more accessible computing power. Furthermore, financial time-series datasets have unique characteristics that make prediction tasks challenging, such as, e.g., their non-linear and non-stationary nature. In this context, studies as Cheng et al. (2015) have shown that the

The code (and data) in this article has been certified as Reproducible by Code Ocean: (https://codeocean.com/). More information on the Reproducibility Badge Initiative is available at https://www.elsevier.com/physical-sciences-and-engineering/computer-science/journals.

* Corresponding author.
E-mail addresses: salvatore@unica.it (S. Carta), sergio.consoli@ec.europa.eu (S. Consoli), sebastianpodda@unica.it (A.S. Podda), diego.reforgiato@unica.it (D.R. Recupero), madalina.stanciu@unica.it (M.M. Stanciu).

[1] https://www.investopedia.com/terms/p/pairstrade.asp

https://doi.org/10.1016/j.eswa.2022.117763
Received 20 October 2021; Received in revised form 24 May 2022; Accepted 2 June 2022
Available online 9 June 2022

use of classical statistical tools cannot reveal the entire spectrum of information. For these reasons, ML has come to play an integral role in many parts of the quantitative financial ecosystem.

Although employing ML techniques for decision making, and by extension for StatArb, seems to be the best viable option, practitioners require to understand how the employed ML models undertake their decisions. Often referred to as "black boxes", ML models are developed with a single goal of maximizing predictive performance (Barbaglia et al., 2021). The European GDPR regulation (Goddard, 2017) states that the existence of automated decision-making should carry meaningful information about the logic involved, as well as the significance and the envisaged consequences of such processing for the data subject. Under the GDPR, users of a ML system are entitled to receive meaningful information about the logic of the automated decision-making process. To fill this gap, researchers have developed frameworks to "X-ray" the ML models and increase their transparency (Molnar et al., 2020), and the field of eXplainable Artificial Intelligence (XAI) has surged. Under the umbrella of XAI, in the literature (Adadi & Berrada, 2018) we can find different perspectives and motivations: explanations to justify, explanations to control, explanations to discover, and finally, explanations to improve classification or regression tasks. We focus our work on the latter. Specifically, we aim at defining a feature selection framework tailored for financial forecasting that increases the predictive performance of the employed ML models.

However, we must note that there are some undesirable, natural and unavoidable limitations to XAI (Emmert-Streib et al., 2020). The authors of the cited paper show what XAI methods can be instead than presenting what they should be and have summarized their findings as reported in the following three items:

- An AI system represents an instrument for data analysis and the explainability of an even perfect AI system are limited by the random sample of the used data.
- The more exhaustive an AI system becomes the more complex its underlying mathematics is.
- The most powerful AI systems such as neural networks should not be preferred by default. A careful analysis against alternatives should be carried out and differences should be quantified.

This means that, although a XAI method provides a useful tool to analyze and choose the features of a given dataset, a certain degree of uncertainty of any AI model may still be left when trying to uncover the black-box underlying it.

As far as the feature selection process is concerned, it aims at identifying a representative subset of features from a larger cohort. One can either choose to manually select the features or to apply an automated method. The challenge in the manual selection of the features is that this process requires expert knowledge about the data at hand. Besides, due to unclear dependencies within financial data, identifying significant information from irrelevant information is a challenging task that can be better tackled in an automated fashion. This constitutes the goal of this paper.

In the context of machine learning, feature selection has high potentials (Barbaglia et al., 2021). When handling high-dimensional input data, finding the most descriptive features might reduce model complexity and accelerate computation, model training and prediction. Furthermore, it supports model interpretation and diagnosis (Zhao et al., 2019), as understanding what the representative features mean leads to gain a deeper insight of the problem at hand. Besides it tendentially helps tempering model over-fitting, which can significantly hinder the performance of the predictive models (de Prado, 2018). However, as also shown by Smolander et al. (2019), the use of feature selection should be balanced, in the sense that it needs ensuring that the overall usage and set-up is simple, the needed computational resources are little and the execution time is faster compared to other approaches alone.

Feature selection methods can be roughly divided into three categories: wrapper methods, embedded methods, and filter methods (Yamada et al., 2020). Wrapper methods evaluate subsets of features by recomputing the model for each of them. Embedded methods learn the model while simultaneously selecting the subset of relevant features. Filter methods attempt to remove irrelevant features using a per-feature relevance score. In this paper, a filter method based on permutation importance score is studied and evaluated within an automated machine learning platform. The advantage of the filter method is the quality of being applicable to different machine learning models.

Combining these three dimensions, i.e., machine learning, explainable AI, and trading/statistical arbitrage, is the main objective of this work. Although these points are discussed in the literature on an individual basis, there is a lack of empirical work dealing with the use of XAI techniques in a StatArb trading context.

The reader notices that our aim is not to provide a novel contribution to XAI research, nor to propose new techniques – or variations of existing ones – in this field of research. Rather, we integrate machine learning and XAI technologies to tackle the StatArb problem, leading to a (proven) increased performance with respect to that obtained by a baseline consisting of machine learning approaches alone. Therefore, this work is more focused in improving the financial performance of existing methods rather than to make the models (more) explainable or to propose a new general XAI method. However, we would like to point out that this goal is fully in agreement with the purpose and usefulness of XAI, as these techniques are not limited – as said – to contribute to the explanability of models, but are now widely used also for the purpose (equally important) of improving the performance of existing algorithms (Krishnan, 2020; Langer et al., 2021; Zhou & Chen, 2019).

Specifically, we focus on StatArb strategies developed by buying the top and selling the bottom stocks according to a given sorting criterion, i.e., ranked on the daily returns. To achieve this goal for a large set of stocks, we train a ML model with different types of features, i.e., lagged returns or series of technical indicators, and forecast the next-day return. To improve the prediction performance, we propose a framework to select the features that are most relevant for the return prediction task. We build a ML model either for each stock, or a model for each sector industry the stocks are grouped in. The rationale behind this choice lies in the fact that, given the heterogeneous nature of the stock data, in many cases the predictions made by a model may rely on a different subset of features for different subgroups within the data (de Prado, 2018). By performing feature selection at a more granular level, instead of the entire stocks set, allow us to select the features that are most relevant to that specific subgroup, rather than having to choose the features globally. Selecting the features globally may perform well on the average across all stocks, but may not be able to explain the predictions for each stock or sector satisfactorily.

In summary, our contributions are:

1. we integrate machine learning and XAI technologies to tackle the StatArb problem leading to increased performance with respect to that obtained by machine learning approaches alone.
2. We employ a feature selection method based on permuting the values of each feature in the out-of-bag sample to derive a feature importance score. In this setup, we propose three strategies to select the best subset of features to remove for each stock, with the goal of improving the overall predictive performance. Our proposed approach is capable of selecting features without prior knowledge of how a certain feature may contribute to the prediction outcome.
3. We develop an *automatic* feature selection method as we aim at determining a feature importance *threshold* such that we can divide the features into two subsets: important features that meaningfully contribute to the prediction task, and non important features that do not.

4. We carry out an empirical evaluation of the above on large set of stocks, i.e., S&P500 index constituents, demonstrating potential for cost reduction in terms of feature engineering and data labeling efforts.

The remainder of this paper is organized as follows. Section 2 describes relevant related works in the literature. Section 3 introduces the problem we are tackling, along with the details of the proposed feature section methods. In Section 4 we continue by describing the setup considered for the trading back-test, while Section 5 presents the computational experiments we have carried out. Section 6 concludes the paper and discusses further directions of research.

2. Related work

The prediction of the stock market is one of the most challenging tasks in the realm of time series forecasting. In recent years, clear evidence was built that ML techniques are capable of identifying (non-linear) structures in financial market data, and such applications of ML and neural networks in finance have been presented and analyzed in several works, such as Atsalakis and Valavanis (2009), Carta et al. (2019), Cavalcante et al. (2016), Henrique et al. (2019), Kumar et al. (2021), Thakkar and Chaudhari (2021). Although there are various streams of research and applications, this section presents only a limited number of articles, highly correlated to this paper, that employ XAI techniques for financial forecasting, and even more in depth, XAI applied to StatArb.

Chronologically covering prior works on XAI techniques applied to financial forecasting, we start by mentioning the work by Lee (2009), where the authors used a hybrid feature selection method (filter and wrapper) for a classification task on stock trends. To prove its performance, they applied the method on 17 different stocks obtaining an enhanced prediction performance, i.e., 87.3% accuracy on average. The proposed method was an hybrid, since it firstly used the F-Score to initially filter the features, and then it used a Supported Sequential Forward Search, that is a technique created for Support Vector Machines to sequentially perform a forward search for the best features to add to the final feature set. Moreover, in the mentioned work, the method selected 17 features, out of a total of 30 features, as the best ones that significantly improved the overall classification accuracy. In Hafezi et al. (2015), the authors aimed at forecasting the DAX index closing price and devised a pipeline with four layers: data preprocessing, feature selection, modeling, and finally a reporting and scenario planning. The authors used cross-correlation as a feature selection technique, and out of 20 features, they chose 13 as appropriate for predicting the stock price. Pimenta et al. (2018) introduced a feature selection approach based on trading rules generated from technical indicators and genetic programming. The authors defined the pipeline as an automated investment system envisaged to identify the right moments for executing buying and selling orders. The authors tested its performance by applying it to six shares on two study periods: February 2013 to February 2015 and July 2015 to July 2016. They identified that the feature selection strategy had a positive impact in 5 out of 6 shares. Gunduz (2021) proposed to use Variational Autoencoders as an unsupervised feature reduction mechanism, followed by a recursive feature elimination technique, to further decrease the number of features in the input feature set. The selected features constituted the input for three different classifiers: Gradient Boosting, Support Vector Machines, and two variants of Long Short-Term Memory (LSTM) networks, with and without attention. The proposed ensembled approach was used to forecast the hourly direction of eight different stocks of the Borsa of Istanbul on a study period from 2011 to 2015. In Man and Chan (2020) the authors proposed an "instability index" strategy for feature selection based on the features ranks variance. Their investigation focused on identifying a convergence point for feature importance stability. To this end, for the same time series, the authors trained various Random Forests models having different configurations, and computed the feature importance for each of these models to infer the "instability" index. Continuing in the realm of XAI techniques applied to financial forecasting, Carta, Consoli, Piras et al. (2021) proposed an approach to indicate any links between news and stock behavior. To this end, the authors employed an explainable ML model based on decision trees to discover keywords relevant to stock return trends.

When it comes to explainable methods applied to StatArb applications, we should necessarily mention the work in Krauss et al. (2017), where the authors used an ensemble of classifiers composed of Random Forest, Gradient Boosted Trees, and Deep Neural Networks to predict stock return daily trends, for the stocks of the S&P500 index. In few words, the approach computes the probability that one stock outperforms the cross-sectional median return of all stocks in the holding period. The authors conducted daily trading first, by ranking the stocks according to the forecasted probability, and then, by generating the daily trading signals on the top-decile of stocks that were bought long, and the flop-decile stocks that were sold short. To understand the key features (lagged stock returns) of the long–short strategy, and how they contributed to the trading strategy results, the authors presented the features importance as given by Random Forest and Gradient Boosted Trees (Molnar et al., 2020). Fischer and Krauss (2018) used an LSTM network for the same prediction task. This enhanced approach outperformed memory-free classification methods such as Random Forests. To explain the results of the StatArb strategy, the authors used the coefficients of a Carhart regression (Carhart, 1997) applied to the returns and first and second-order time-series statistics and, in doing so, they untangled the dependence between the market regime, stock behavior, and the results obtained by the long–short trading strategy. Huck (2019) proposed the use of four predefined feature sets and constructed the target similar to the previous works, i.e., by computing the probability that stocks outperform their peers in the S&P100 or S&P300 indexes. The author used various ML models such as Deep Belief Networks, Elastic Net, and Random Forest. Concerning the XAI techniques, the author also inferred the feature importance by assessing the performance of the algorithmic trading strategy given the four groups of features. Also, the author reported the feature importance per feature sub-groups according to Random Forests and permutation importance, and noted that increasing the number of features does not translate into enhanced financial performance. Flori and Regoli (2021) examined an approach of stocks clustering to complement investor practices for the identification of pairs-trading opportunities among co-integrated stocks. The authors proposed to predict the trend of price or return gaps between each stock and its peers.

With respect to the works illustrated so far, in this paper we aim to fill the gap of applying XAI techniques into the StatArb context (Carta, Consoli, Podda et al., 2021). In contrast to some of the recent works, such as Fischer and Krauss (2018), Huck (2019), Kraus and Feuerriegel (2017), our work is focused on applying explainability methods that are enhancing the predictions rather than finding the causes of stock behavior and establishing causality claims. Furthermore, we propose a solution for improving the prediction of daily returns by selecting the most appropriate features either for single stocks or groups of stocks (sectors). To this end, we employ different ML models and different feature types (i.e., lagged returns and technical indicators), nevertheless on a large set of stocks as the StatArb trading strategy requires. Having said that, it is clear that our purpose aims to define our approach such that (i) is not specifically designed for a type of ML model (such as the work in Lee (2009)); (ii) captures the particularities of the underlying trading domain (by using two levels of forecasting and feature selection, i.e., sector or stock); (iii) and, at the same time, is computationally tractable and applicable for a large set of stocks, i.e., the S&P500 index.

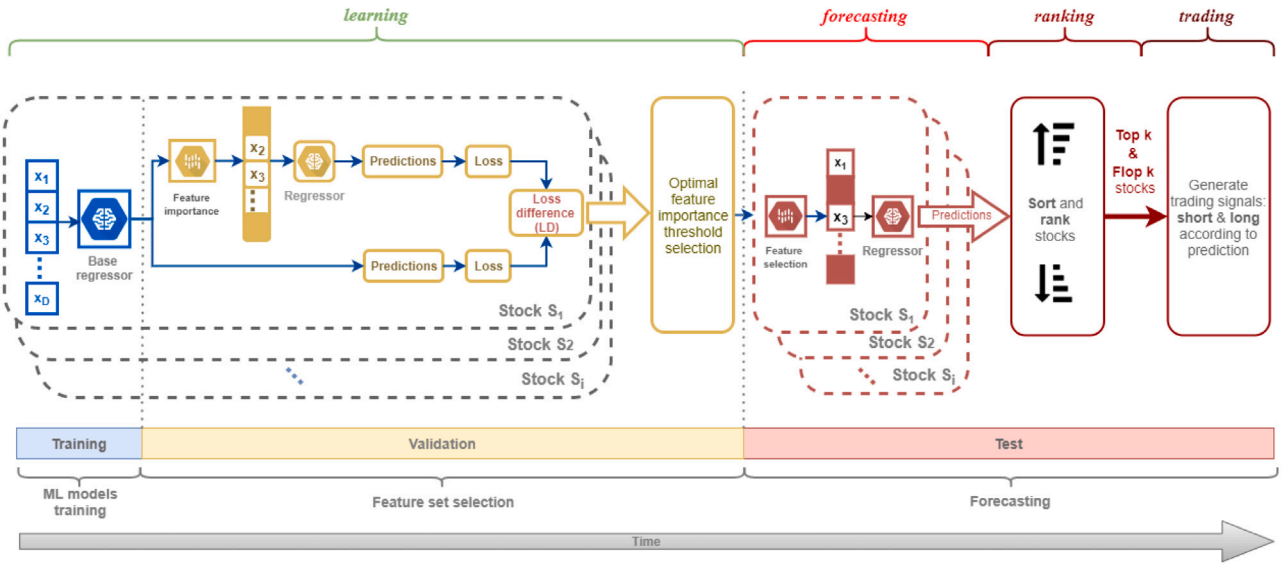


Fig. 1. Block diagram of the StatArb XAI trading strategy.

3. Methodology

3.1. Overview

StatArb is a contrarian trading strategy (Avellaneda & Lee, 2010) as its source of profitability lies in the identification of short-term anomalous stock behaviors, i.e., abnormally high returns or remarkable low ones, and trading both the *winners* and *losers* stocks (Huck, 2019). In the following, we describe the application of ML models in order to identify such deviant stocks. More specifically, we employ the ML models based on financial information to generate predictions of increasing/decreasing returns in the near future for each stock. Moreover, our goal is to complement the StatArb trading strategy with an automatic feature selection technique taken from the XAI literature and, in doing so, verify whether it is possible to increase the predictive performance of the underlying ML models, thereby generating even better financial performances. Fig. 1 presents the block-scheme for the proposed approach where typically a StatArb XAI trading strategy comprises three main phases: the forecasting phase, ranking of the stocks given a specific criterion, and, finally, issuing the trading signals and performing the actual trading.

More specifically, in the *forecasting phase*, similarly to Carta, Consoli, Podda et al. (2021), we aim at forecasting daily price returns. Additionally, we introduce a XAI mechanism related to a feature selection algorithm for each stock. The forecasting phase is naturally preceded by an "estimation", derived by learning the relationships between the stocks and the input features. At a high level, we construct a feature vector containing daily financial information that is fed into a ML model, and obtain the predictions of the base regressor of the stock returns. Next, feature importance scores are computed and, based on them, features are divided into two categories: important features and unimportant ones. The unimportant features are then discarded, and a new model is trained. The new model receives as input only the features that are considered to be important and makes consequent predictions. The loss of the newly trained model is subtracted from the loss of the base regressor. The loss difference is used to compute an optimal feature score threshold, below which features are indeed set as unimportant — our intuition is that, if removed, these features do not worsen the overall prediction performance. The described process constitutes the learning phase. In the trading phase for each stock, we use the model trained with the optimal features set.

In the *ranking phase*, based on the forecasted price returns for each stock, we rank the features in descending order. While doing so, we find at the top of the ranking the stocks whose returns are expected to be positive, therefore representing candidates for buy (long) operations, whereas at the bottom of the ranking we find the stocks whose returns are expected to be negative — for this reason they are candidates for short operations.

In the last phase, i.e., the *trading phase*, we generate the appropriate trading signals for the top k and the flop k stocks of the ranked set of stocks.

Provided that financial data represent a particular type of time series, we account for this as depicted in Fig. 1, and the data are chronologically split into three chunks: training dataset, validation dataset, and testing dataset. We use the training set for ML models training, while the validation set serves for feature importance scoring, optimal feature importance threshold computation, and the subsequent optimal feature set selection. The test set, considered out of sample data, is used for the StatArb XAI strategy performance analysis. Note that the length of the bars indicating the three datasets (i.e., train, validation, and test) does not correspond to the actual duration of these time segments. Each bar is placed under the portion of the architecture corresponding to the bar function.

In the following subsections we present the details of each block forming the StatArb XAI trading strategy.

3.2. Data and input features

Let $\mathcal{D} = \{(\mathbf{x_t}, y_t)\}_{t=1}^{T}$ be a time-series dataset where $\mathbf{x_t} \in \mathbb{R}^D$ denotes the observation at time $t \in \{1, \dots, T\}$. D represents the dimension of the feature set. Thus, for a single feature, i, an observation at time t is indicated by $x_{t,i}$. Our goal is to predict a target variable $Y = \{y_t\}_{t=1}^{T}$ for $y_t \in \mathbb{R}$, based upon the multivariate predictive variable $X = \{\mathbf{x}_t\}_{t=1}^{T}$ by using a learning algorithm that outputs a model function $f(\cdot)$.

For each stock, we construct our dataset D by collecting daily raw financial information such as *Open Price*, *High Price* in the day, *Close Price*, *Low Price* in the day, and *Volume* of stocks traded during the day (i.e., OHCLV variables). Based on this information, we create two different types of features:

Lagged returns (LR) — computed with respect to day t and expressed as

$$LR_{t-j} = \frac{close_{t-1} - open_{t-j}}{open_{t-j}},$$

where $j \in \{1, 2, 3, 4, 5, 21, 63, 126, 252\}$ and denotes the length of the time-window for which the return is computed.[2] The feature vector $\mathbf{x_t}$ is therefore composed of 9 lagged returns for each day t. Similar types of input are used in previous works such as Carta et al. (2022), Huck (2019), Kraus and Feuerriegel (2017) and the lagged returns are meant to provide the ML algorithms with information within three different time horizons: (i) the recent past, i.e., $j \in \{1, 2, 3, 4, 5\}$, (ii) medium past of the last trading month, last three trading months, respectively, for $j \in \{21, 63\}$, and finally (iii) returns corresponding to the last six months and one year (i.e., for $j \in \{126, 252\}$).

Technical Indicators (TI) — represent statistical tools that investors extensively use to make their investment decisions. They are extensively used within the academic literature to either generate trading signals (Pimenta et al., 2018) or as input features for ML models (Kara et al., 2011; Patel et al., 2015a, 2015b). To construct the feature vector $\mathbf{x_t}$, similarly to Carta, Consoli, Podda et al. (2021), Carta et al. (2020), we use the same set of nine technical indicators: Exponential Moving Average (EMA(10)), Williams %R, Stochastic Oscillator (%K), Relative Strength Index (RSI), Rate of change (ROC), Accumulation/distribution indicator (AccDO), Moving Average Convergence Divergence (MACD), and two disparity measuring indicators, respectively on the last 5 days and on the last 10 days (Disp(5), Disp(10)). To compute the feature vector of technical indicators for a day t, we use financial information over the past days to t (each indicator needs a different number of previous days to t, according to their formula).

Roughly, the indicators can be divided into four categories[3]:

- Trend followers — identify the main movements of stock prices in recent past (EMA(10)).
- Divergence identifiers — identify possible regime switches, e.g. the current trend ends and the prices that will start moving in the opposite directions (AccDO, MACD).
- Momentum indicators — determine the momentum that a stock has with respect to its upward or downward trajectory in the market (ROC, RSI, Disp (5), and Disp (10)).
- Oscillators — follow the price variation of the stocks in order to identify possible reverse points (%K and Williams %R). To note that when stochastic oscillators are increasing, the asset prices are likely to go up and vice-versa.

Target variable is a continuous variable given by the intra-day price return defined as:

$$y_t = \frac{close_t - open_t}{open_t}.$$

3.3. Feature selection

With respect to Fig. 1, the second block of the forecasting phase aims to assign feature importance scores with the goal of identifying uninformative features for the prediction task, and proposes them for removal. In the following paragraphs, we detail the methodology for assigning the feature importance scores and the subsequent feature removal strategies. Note that the feature selection importance score computation, the optimal threshold computation, and the feature selection process, are computed using the validation dataset only, thus ensuring the absence of data-leakage.

[2] For example, $j = 1$ denotes the return in the previous day for each observation. At the opposite pole $j = 252$ denotes the return in the past year, provided that there are 252 trading days in a year.

[3] Information about technical indicators use and their interpretation has been collected from https://www.investopedia.com/.

Feature importance score with permutation importance (PI)

The feature importance score indicates the amount each feature contribute to the model prediction (Strumbelj & Kononenko, 2010). The PI method (Breiman, 2001) provides such a score and quantifies the impact on the predictive power of the estimator by replacing each feature with noise. The motivation in choosing the PI approach is threefold: (i) it is model agnostic, i.e., it can be applied to any learning model; (ii) it computes the feature importance on out-of-bag samples , which makes it possible to highlight which features contribute the most to the generalization power of the inspected model (Strobl et al., 2008); (iii) it has no tuning parameters and relies only on average values, making it statistically very stable.

Generically, given the input variables X and the corresponding target variable Y, let $X^{\pi,i}$ be the results of randomly permuting the ith feature of X. Let $L(Y, f(X))$ be the loss for predicting the target variable Y from the data X using the model $f(\cdot)$. Then, the feature importance can be mathematically expressed as follows:

$$\begin{aligned} VI_i^{\pi} &= L(Y, f(X^{\pi,i})) - L(Y, f(X)), \\ VI_i &= \frac{1}{N} \sum_{\pi} VI_i^{\pi}, \end{aligned} \quad , \tag{1}$$

where N represents the number of permutations applied to each feature. Specifically, the importance of the feature i is given by the increase in loss due to replacing $X_{:,i}$ with values randomly chosen from the distribution of the same feature i. Regarding the computation of the feature importance and of Eq. (1), there are variations of the equation that scale better the changes in loss due to permuting feature i values with the loss of the base regressor, such as the following:

$$VI_i = \frac{1}{N} \sum_{\pi} \frac{L(Y, f(X^{\pi,i})) - L(Y, f(X))}{\mathbf{L}(\mathbf{Y}, \mathbf{f}(\mathbf{X}))}. \tag{2}$$

The feature importance expressed as above can be interpreted as the average change in prediction loss relative to the loss of the base regressor.

In any of the variants (Eq. (1) or Eq. (2)), the feature importance score will have either (i) positive, (ii) close to zero, or (iii) negative values. As a consequence, we can classify the features as follows. *Important features* will have *positive* scores, as replacing the corresponding features values with other random values, increases the loss with respect to the original regressor. On the other hand, *unimportant features* will have *negative* feature importance scores, as their replacement with noise-like information improves the prediction performance, i.e., the loss decreases with respect to the original predictions. In the middle, we have the *unimportant features* whose feature importance score is close to 0, thereby they do not significantly contribute to the prediction outcome. As a consequence, replacing them with noise does not introduce a substantial loss change. Out of these three categories of features, we consider as candidates for removal the last two categories, as long as their feature importance is lower than 0.

Feature removal strategies

Given the possible values of the feature importance score and the implication on feature categorization, we propose three strategies to remove the features:

1. *PI best* — identifies the features which have feature importance lower than 0 and, at the same time, have the *highest* feature importance among them. We remove those features if their importance is lower than a certain threshold.
2. *PI worst* — identifies the features whose importance is lower than 0, and, simultaneously have the *lowest* feature importance among them. Similarly to *PI best*, if those features have their importance score below a certain threshold, we remove them.
3. *PI running* — identifies the features with the importance lower than a variable threshold, and proposes for removal the one(s) with the *highest* feature importance score.

Stock	Feature	Feature importance
GOOGL	LR_2	-0.1077
GOOGL	LR_{63}	-0.3906
GOOGL	LR_5	-0.5298
GOOGL	LR_3	-1.1571
IBM	LR_1	0.1060
INTC	LR_5	-0.2911
INTC	LR_{252}	-0.3211
INTC	LR_3	-0.3420
INTC	LR_{63}	-0.7658
INTC	LR_{126}	-1.1547

(a) Feature importance

	PI best		PI worst		PI running	
Stock	Feature	Feature importance	Feature	Feature importance	Feature	Feature importance
GOOGL	N/A		LR_3	-1.1571	LR_{63}	-0.3906
IBM	N/A		N/A		N/A	
INTC	LR_5	-0.2911	LR_{126}	-1.1547	LR_5	-0.2911

(b) Selected features

Fig. 2. Illustration of the proposed feature selection strategies for three stocks: *GOOGL*, *IBM*, and *INTC*. Panel (a) shows, for each stock, the features and their importance. To note that, for this example, we present only a limited number of features. In our working scenario, the feature importance panel would include all the features associated to a stock. Moreover, the example illustrates the removal of only one feature, while the strategies allow for multiple features dropping. Panel (b) shows the selected features for each method for an assumed threshold of −0.15.

The main difference between *PI running* and the other strategies is that *PI running* uses the threshold to identify the best features to remove. *PI best* and *PI worst* use, instead, the threshold to identify whether the best features (i.e., the features with the highest importance below 0) or the worst feature (i.e., the features with the lowest importance below 0) will be removed.

To better grasp the differences between the three proposed strategies, we illustrate in Fig. 2 a simplified example. The figure presents information for three stocks: Alphabet, Inc. (GOOGL), International Business Machines (IBM), and Intel Corporation (INTC). The example uses as features lagged returns (LR) whose indices correspond to the window lag when computing the returns (see Section 3.2 for further details on features). The left-hand side panel, i.e. (a) Feature importance, shows the stocks, the features, and their corresponding feature importance score computed by Eq. (2). For each stock, the features are sorted in descending order by their feature importance. For sake of simplicity, we show for stocks *GOOGL* and *INTC* only the features with negative feature importance. For completeness of the example, for the *IBM* stock, we show the least important feature. Such a scenario can happen if the stock has no features whose importance is lower than 0, therefore all features contribute to the prediction outcome. The right-hand side panel, i.e. (b) Selected features, shows the effect of the three feature selection strategies that use a threshold equal to −0.15.

Starting with *PI best*, for the *GOOGL* stock, it does not select for removal any feature, as the feature with the highest negative feature importance is LR_2. However, its feature importance is not below the threshold −0.15. For the *IBM* stock, for obvious reasons no features are removed. In the case of the *INTC* stock, the feature with the highest negative feature importance, and that also falls below the threshold −0.15, is LR_5; thus, *PI best* will propose it for removal.

Switching to *PI worst*, in the case of *GOOGL*, the strategy will remove LR_3, since it has the lowest value and is also below the threshold. For the *INTC* stock, the strategy will remove LR_{126} for the same reasons.

Finally, for the *PI running* strategy, with respect to the *GOOGL* stock, the strategy will remove LR_{63}, since it is the first feature whose importance value is below the threshold. For *INTC*, the strategy proposes to remove LR_5, the same feature chosen using the *PI best* strategy.

Optimal feature importance threshold selection

For each of the proposed feature selection strategies, i.e., *PI best*, *PI worst*, and *PI running*, we compute an optimal feature importance threshold. For each feature selection method, model and stock, we remove the indicated features according to the identified feature importance threshold and retrain a new model. Then, for each threshold value, we compute the mean loss difference between the base regressor (without any feature removed) and the corresponding newly trained model.

Under this premise, for each feature selection method, we compute the optimal threshold as the one that maximizes the mean loss difference above. The introduction of the optimal threshold enables the proposed approach to *learn* in cross-section the features that can be removed in order to increase the overall prediction performance, i.e., also, to decrease the mean loss (across all the stocks). In other words, in the case of *PI best* and *PI worst*, the optimal threshold controls the sparsity of stocks and models that have features removed, whereas in the case of *PI running*, it controls the highest score below which the features can be considered unimportant, thus suitable candidates for removal. Note that, as highlighted in Fig. 1, for the computation of the optimal feature threshold we use the validation dataset only, in order to avoid data-leakage.

3.4. Forecasting algorithms

In our forecasting process, we are considering three well-established ML models, i.e., *Random Forests*, *Light Gradient Boosting*, and *Support Vector Machines*. The first two are *decision-tree*-based models, and therefore they are intrinsically explainable (Breiman, 2001; Molnar et al., 2020). Moreover, all types of models have been successfully used in financial prediction tasks which are particularly challenging for reasons that we are going to discuss in the paragraphs below.

3.4.1. Light gradient boosting

Light Gradient Boosting (LGB), first proposed by Ke et al. (2017), has been widely applied in machine learning tasks, and it supports efficient parallel training. LGB applies iteratively weak learners (decision trees) to re-weighted versions of the training data (Hastie et al., 2009). After each boosting iteration, the results of the prediction are evaluated according to a decision function, and data samples are re-weighted in order to focus only on examples with higher loss in previous steps. By comparison to the traditional gradient boosting techniques, LGB grows the tree vertically (i.e., leaf-wise tree-growth, until the maximum depth is reached), whereas other alternative algorithms extend their structures horizontally (i.e., level-wise tree-growth). This fact makes

LGB quite effective in processing large-scale and high-dimensional data, with the downside of being more prone to over-fitting. In their work, authors of Carta, Consoli, Podda et al. (2021), Carta et al. (2020) use LGB to predict daily price returns.

3.4.2. Random forests

Random Forests (RF) belongs to a category of ensemble learning algorithms introduced in Breiman (2001). This learning method is the extension of traditional decision trees techniques where random forests are composed of many deep de-correlated decision trees. Such a de-correlation is achieved by bagging and by random feature selection. These two techniques make the RF algorithm quite robust to noise and outliers. When using RF, the larger the size of the forest (the number of trees), the better the convergence of the generalization error. However, a higher number of trees, or a higher depth of each tree, induces computations costs; therefore a trade-off must be made between the number of trees in the forest and the improvement in learning after each tree is added to the forest. In the academic literature, we can find that RF has been extensively used in financial forecasting (Carta et al., 2022; Patel et al., 2015b) and also in StatArb applications (Krauss et al., 2017). Moreover, RF is an explainable model and is therefore used to identify the most important features in trading strategies (see, e.g. Huck (2019)).

3.4.3. Support vector regressors

Support vectors were proposed initially as supervised learning models in classification, and later revised for regression, with the name of Support Vector Regressors (SVR), by Vapnik (1999). Given the dataset D described in Section 3.2, the goal is to find a function that deviates from actual data, y_t, by a value no greater than ϵ for each training point, and, at the same time, being as flat as possible. SVR extends least-square regression by considering an ϵ-insensitive loss function. Furthermore, to avoid over-fitting of the training data, the process of regularization is often applied. SVR is trained by solving the following optimization problem:

$$\min_{w} f(w; C, \epsilon), \text{ where}$$
$$f(w; C, \epsilon) = \frac{1}{2}\|w\|^2 + \frac{C}{T}\sum_{t=1}^{T}\underbrace{\max(0, |y_t - f((x_t); w)| - \epsilon)}_{\epsilon\text{-insensitive loss function}}, \tag{3}$$

and where $C > 0$ is the regularization parameter, and $\epsilon > 0$ is an error sensitivity parameter. The training finds the weights w so that the function f minimizes empirical risk. The above concepts can also be extended to the non-separable case (linear generalized SVR). In our study, we select the radial basis kernel function, which can be formalized as $k(\mathbf{x_{t1}}, \mathbf{x_{t2}}) = exp(-\gamma\|\mathbf{x_{t1}} - \mathbf{x_{t2}}\|^2)$. Here $\mathbf{x_{t1}}$ and $\mathbf{x_{t2}}$ represent two feature vectors of the input space, γ is a free parameter, considered as a design parameter of SVR, and $\|\cdot\|^2$ is the squared Euclidean norm. Lee et al. (2019) use SVR not only for price prediction purposes, but also in conjunction with a feature selection algorithm.

4. Experimental setup

In this section, we describe the technical setup of the experiments that we carried out. We follow along with the main components of our trading strategy: data, model training, feature selection, ranking process, and trading execution.

4.1. Data

We consider stocks composing the S&P500 index as a reference dataset. The S&P500 constituents represent the leading 500 companies in the U.S. stock market and, at the same time, a large-capitalization segment. These characteristics make these companies highly attractive to investors and, as a consequence, they are very challenging for any trading strategy, as various works in the literature have pointed out (Fischer & Krauss, 2018; Krauss et al., 2017).

We back-tested our trading strategy using the *walk-forward* validation approach, which represents a common practice in the related literature (Carta, Consoli, Podda et al., 2021; Flori & Regoli, 2021). The walk-forward validation consists of splitting the time interval into overlapping training and validation periods, and non-overlapping test (trading) periods, as shown in Fig. 3. The example depicts values for the closing price of the AT&T stock over the whole study period, ranging from January 2003 to January 2021. Each triplet *(training, validation, test)* forms a walk. By sliding forward the triplet with the length of the test period, we construct the out-of-sample contiguous period from January 2007 to January 2021, i.e. a total of 14 years. Each walk entails three years for ML models training, one year of validation used for optimal threshold computation and feature selection, and one year considered as out-of-sample.

The input features for the ML models represent a multivariate time-series composed, as indicated in Section 3.2, either of the lagged returns, the technical indicators, or both. For each stock and each walk, we start by collecting raw financial data[4] and compute the two types of input features mentioned in Section 3.2, as well as the target variable for the training and validation periods. Then, the features are rescaled within the range $[-1, 1]$ by applying a MinMaxScaler (Pedregosa et al., 2011) fit on the training set.[5] The MinMaxScaler scales and translates each feature individually such that it is in the given range on the training set, i.e., $[-1, 1]$. The rescaling is a necessary step as it increases the convergence likelihood of the SVR model, as well as un-biased learning in the case of decision trees based models such as RF and LGB.

Similarly to Carta, Consoli, Podda et al. (2021), we build machine learning models containing data either for a single or multiple stocks belonging to the same sector in the S&P500 index. Herein we refer to this particular setup as the data level, also referred to as stock level, or sector level. In Fig. 4 we show the distribution of the S&P500 stocks into their corresponding sectors.

4.2. Model training

As mentioned throughout the paper and reflected in Figs. 1 and 3, the ML models are trained using a 3 years time-span for each rolling window. The hyperparameters of the models are listed in Table 1.

LGB, as stated in Section 3.4, is prone to over-fitting due to its manner of constructing the decision trees. To control this behavior, we defined the maximum depth levels of the tree, *max_depth*. Also, considering that LGB constructs the decisions trees sequentially, a large number implies high computation costs and also the possibility of over-fitting, therefore we chose for *n_estimators* conservative values lower than 250. Depending on the type of features and on the level of the model, we also considered different *num_leaves* values, aiming at achieving a balance between a conservative model and a good generalization capability. The percentage of features considered when doing a split is restricted by *colsample_by_tree*, which can be thought of as a regularization parameter. The work in Hastie et al. (2009) suggests in general to set a learning rate lower than 0.1; we therefore considered values lower than 0.01 in order to account for a better generalization over the dataset.

RF models are normally not subject to over-fitting when a large number of estimators (decision trees) is used. Therefore, we set *n_estimators* to high values up to 500. We limit the maximum number of features in each split by imposing *max_features* equal to 1, and limit at the same time the depth of the tree by setting either the *min_samples_leaf* or *max_depth* parameters.

[4] We use the publicly available data from Yahoo Finance, available at: https://finance.yahoo.com/.

[5] The transformation is learnt on the training set only in order to ensure that no data-leakage happens.

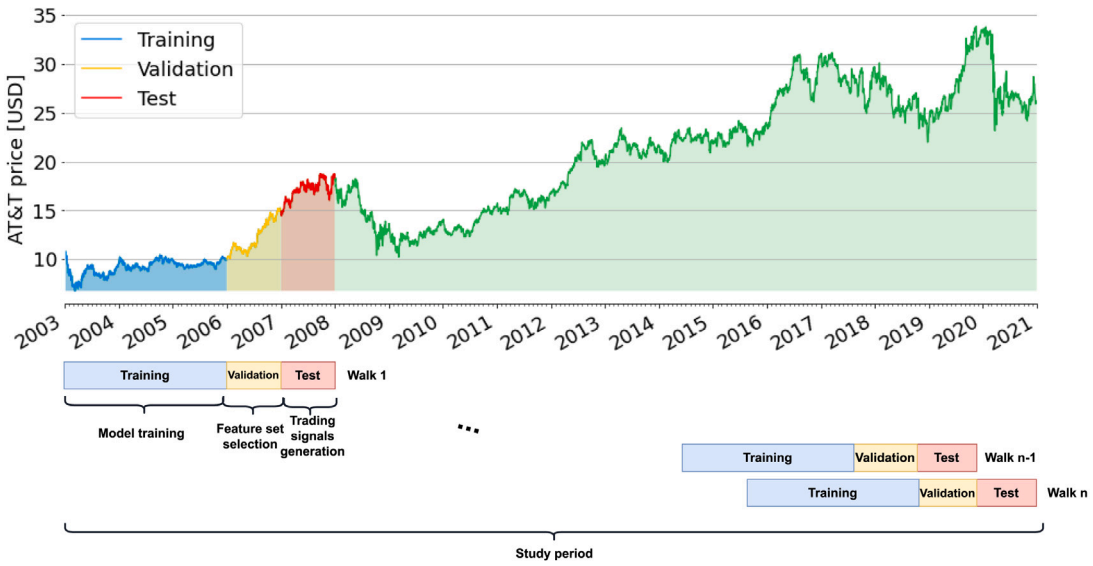


Fig. 3. Illustration of walk-forward procedure, considering a study period starting from January 2003 to January 2021. On the y-axis it is presented the development of the closing price for the AT&T stock across time..

Table 1
Hyperparameters of the ML models.

Level	Feature Type	ML Model Type	Hyperparameters
Stock	LR	LGB	n_estimators=220, num_leaves=252, max_depth=7, learning_rate=4e−2, reg_alpha=9e−4, bagging_fraction=0.65
		RF	n_estimators=500, min_samples_leaf=5, max_features=1
		SVR	C=10, tol=1e−5, epsilon=1e−5, gamma=9e−3
	TI	LGB	n_estimators=100, num_leaves=21, max_depth=7, learning_rate=5e−2, colsample_bytree=9e−4, bagging_fraction=0.65
		RF	n_estimators=500, min_samples_leaf=5, max_features=1
		SVR	C=10, tol=1e−5, epsilon=1e−5, gamma=9e−3
Sector	LR	LGB	n_estimators=250, num_leaves=230, max_depth=12, colsample_bytree=.9,learning_rate=5e−3,
		RF	n_estimators=300, max_samples=0.5, max_features=1
		SVR	C=2, tol=1e−5, epsilon=1e−5, gamma='scale'
	TI	LGB	n_estimators=100, num_leaves=100, colsample_bytree=0.8, max_depth=15, learning_rate=0.03, subsample=0.8
		RF	n_estimators=300, max_depth=40, max_features=1
		SVR	C=2, tol=1e−5, epsilon=0.25e−2, gamma='scale'

SVR solves an optimization problem that involves two parameters: the regularization parameter, C, and the error sensitivity parameter, ϵ. The C parameter controls the trade-off between model complexity and number of non-separable samples. A lower C generally encourages a larger margin, whereas higher C values lead to a harder margin (Vapnik, 1999). Instead, the ϵ parameter controls the width of the ϵ-insensitive zone and is used to fit the training data. An excessively high value leads to flat estimations, whereas a too small value is not appropriate for large or noisy datasets. The work in Chalimourda et al. (2004) suggests that the γ value of the kernel function should vary together with C, and high values of C normally require high values of γ too. As this is highly dependent on the data variance, we set it such that to be scaled accordingly with the data.

4.3. Feature selection

Next to model training, we compute, for each model and each feature, the feature importance using *PI* over $N = 100$ repeated permutations. We have chosen this value experimentally in order to achieve a good trade-off between computational running time and reliable estimations of the feature importance. Then, for each of the proposed feature selection strategies, i.e., *PI best*, *PI worst*, and *PI running*, we perform a preliminary feature removal process by selecting the threshold values in the range $[0, -0.025]$ with a step of 0.0001. For each model we remove the indicated features $K \in \{1, 3, 5, 7\}$ at a time and retrain a new model. In our experiments we use a prespecified number of features to remove, where $K < no\ features$ (9 for TI, 9 for LR and 18 for TI+LR), and leave the dynamic feature selection as a future improvement of the proposed XAI methods. Finally, for each feature selection method, we compute the optimal threshold as the one that maximizes the mean loss difference between the base regressor (without any feature removed) and the corresponding newly trained model. In our experiments we consider two metrics to quantify the loss: the canonical MSE and the `mean return`.[6] When considering the MSE,

[6] To distinguish between the MSE and mean return as an evaluation metric of the XAI trading strategies and the loss metric for the feature selection, we use the type-writer notation, i.e., `MSE` and the `mean return`.

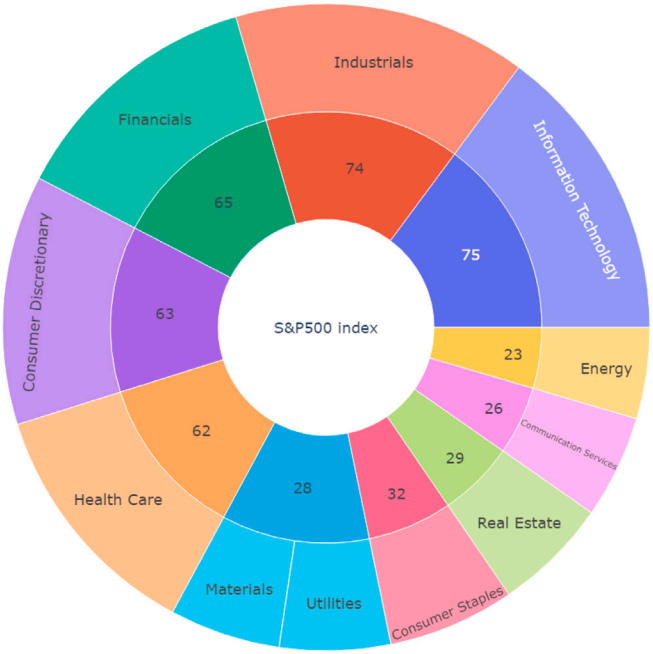


Fig. 4. Number of constituents of the S&P500 index into sectors.

the computation is straightforward and aligned with common practice. The `MSE` metric consists of computing the squared residual between the true observations (on the validation set) and the corresponding estimated values provided by the machine learning model. Instead, in the case of the `mean return` loss, we adopt the following approach. Considering a stock and its return forecast at time t, we "trade" the stock, i.e.: long, if its return forecast is positive, or short, if its forecast is negative. Then, considering the actual stocks return, we can determine the daily stocks return when investing according to the forecast. The mean return is computed as the mean of the daily returns on the entire validation period. In this manner, we want to validate whether a financial metric, such as the mean return, can serve as a good proxy of the ML model performance. Or, differently, we try to answer the question whether the incorporation of a financial metric into the feature selection process can improve the financial performance of our overall XAI strategy.

4.4. Trading execution and portfolio construction

The back-testing experiments consist in running the signals (i.e., long and short) according to the ML models forecast and evaluating the performance against historical data. As stated already throughout the paper, StatArb consists of trading long and short the predicted k winners and k losers, respectively. We fix the number of pairs to be traded to $k = 5$, similarly to Carta, Consoli, Podda et al. (2021). The trading session is set as intra-day, meaning that we are opening the positions at the beginning of the trading day and close them at the end of the day. In other words, we are rebalancing our portfolio daily.

4.5. Baselines

To assess the value added by the feature selection strategies, they are benchmarked against two statistical arbitrage trading baselines: the portfolio constructed using only the base regressors without having any features removed (**BaseRegressors**), and the S&P500 buy-and-hold strategy (**Buy-and-hold** (Li & Hoi, 2014)). The comparison against the **BaseRegressors** is quite natural and straightforward, as we aim to assess the performance improvements of the proposed XAI StatArb methods over the **BaseRegressors**. **Buy-and-hold** is instead a well-established quantitative strategy, and largely used as baselines to evaluate the profitability of other investment approaches (Flori & Regoli, 2021; Huck, 2019). The strategy buys in our application in January 2007, and holds the S&P500 exchange-traded fund (SPY security) during the whole back-testing period, i.e. until January 2021. This passive strategy runs without any trading signals.

4.6. Implementation details

The approach proposed in this paper has been developed in *Python*, by using the `scikit-learn` library (Pedregosa et al., 2011) and the *LightGBM* python API (Microsoft/LightGBM, 2022). The experiments have been executed on a desktop system with the following specifications: an Intel(R) Xeon(R) Gold 6136 CPU @ 3.00 GHz, 32 GBytes of RAM, and 64-bit Operating System (Linux Ubuntu). The full code of the solution, for reproducibility purposes, has been made publicly available at: https://github.com/Artificial-Intelligence-Big-Data-Lab/xai_statarb.

5. Results

In the following section, we present the results obtained by our proposed XAI StatArb strategy. We show the results from three perspectives: (i) goodness of fit of the ML models; (ii) returns generated over the entire trading period; and, finally, (iii) risk evaluation.

In this sense, please observe that quantifying the *explainability* characteristic of a statistical model is a completely unsettled territory to date, and there is still a huge need for proper evaluation methodologies in order to quantify the quality of the explanatory methods as well as to devise suitable approaches able to assess and choose the most appropriate XAI practices. As thoroughly motivated by Zhou et al. (2021), the existing research has been attempting only recently to formulate some approaches for explainability assessment. However, there is no common agreement on the right metrics to use in order to properly assess the quality of the explanation methods (Carvalho et al., 2019). The main reason behind this unsolved problem is that explainability is inherently a very subjective concept, hard to formalize, and the perceived quality of an explanation is contextual and dependent on users, the explanation itself, as well as the type of information that users are interested in Carvalho et al. (2019), Rüping (2006). On the other hand, explanation is also a domain-specific note, hence there cannot be an all-purpose type of explanation (Rudin, 2019).

In light of this evidence, at present there is no clear, precise metric available to estimate explainability (e.g. accuracy, MSE, etc.). Moreover, in the financial domain this results to be even more difficult as a good accuracy/error value is not necessarily correlated to a good financial performance, especially for long–short portfolios, such as ours. Consider, for example, the case in which a large number of transactions with a modest impact (which tend to be easier to predict) is correctly matched, while a single transaction with a very high impact (which is usually more difficult to predict) is missed: in such a case, which is quite frequent in the financial sector – especially during long sideways markets – the outcome would be a good statistical performance, but a potentially very bad financial performance. This would inevitably affect also the type of explainability metric used, if any. For all these reasons, we believe that, for the purposes of our work as well as for the specific domain considered, the use of financial metrics is the most scientifically rigorous way, to date, to provide analysis of the performance of XAI techniques used in the financial context.

Hence, hereby, the following evaluation results are quantified using the following metrics:

(i) Mean squared error (MSE) — one of the most popular goodness of fit measure for ML models for a continuous dependent variable, such as in our case. The MSE is defined as:

$$MSE(f, X, Y) = \frac{1}{T}\sum_{t}^{T}(\hat{y}_t - y_t)^2 = \frac{1}{T}\sum_{t}^{T} r_t^2,$$

where $\hat{y}_t$ is the forecast of the model f at time t (see also Section 3.2 for further details on notations), and r_t is the residual for the observation at time t. Thus, MSE can be seen as a sum of squared residuals. As the measure weighs all differences equally, large residuals have a large impact on MSE. Thus, the measure is prone to outliers. For a "perfect" model, which predicts (fits) all y_t exactly, we would have $MSE = 0$.

(ii) Annual return — represents the return on an investment generated over a year and calculated as a percentage of the initial amount of investment. Formally, it is expressed as:

$$\text{annual return} = 100 \times \left(\left(\prod_{i=1}^{n} (1 + R_n) \right)^{1/n} - 1 \right),$$

where n represents the number of years, and R_n the corresponding return at year n. High values are to be sought.

(iii) Maximum drawdown (MDD) — the maximum amount of wealth reduction that a cumulative return has produced from its maximum value over time. Formally, the drawdown is defined in two steps: first, we find the maximum up to time t, i.e., $M_t = \max_{u \in [0,t]} R_u$, where R_t are the cumulative returns up to time t. The drawdown up to time t (as a percentage) is, then: $DD_t = \frac{R_t}{M_t} - 1$. The maximum drawdown is thus the largest drop in cumulative returns from its maximum rolling over a given time frame:

$$MDD_t = \max_{u \in [0,t]} DD_u.$$

Smaller absolute values are an indication of lower incurred risk.

5.1. Overview

Briefly, we train three types of machine learning models, i.e., LGB, RF, and SVR. Each of them are fed with different types of input features, i.e., LR, TI, or jointly LR and TI. Also, we build models at two levels of data — stock or sector. Thus, we have 3 types of models × 2 levels × 3 types of features, for an overall of 18 base regressors. For each of these combinations, we apply the three strategies of feature selection, namely *PI best*, *PI running*, and *PI worst*. To identify unimportant features (feature selection), we employ two metrics to evaluate the loss, that is `MSE` or `mean return` (details provided in Section 4.3). Moreover, we discard $K = 1, 3, 5, 7$ unimportant features as given by each of the methods.

Firstly, in Fig. 5 we show an overview of the performance obtained by the proposed feature selection strategy compared to the **BaseRegressors**. Specifically, we show the percentage of models that obtain improved performance compared to the **BaseRegressors**, given the three evaluation criteria, i.e., MSE, annual returns, and MDD. Additionally, we show, side-by-side, the performances of the models when employing the two loss metrics for feature selection, i.e., `MSE` and `mean return`. Note that the `MSE` (as loss function for feature removal) is marked with diagonal lines, whereas the `mean return` has no filling pattern. For each performance evaluation criteria, we have on the x-axis the three feature removal methods strategies and the number of features that we discard. The bars represent the percentage of models that have an improved performance over the corresponding **BaseRegressors**.

Fig. 5(a) (MSE evaluation) shows that more than half of the models (more than 9 models out of 18 for each x-value) have an improvement over the **BaseRegressors**, reaching high values such as 88.89%. The lowest performance is registered by *PI best*, which is an expected outcome given that we remove unimportant features but with higher importance than others. In terms of annual returns (Fig. 5(b)), the performance improvements are not as prominent as in the case of the MSE criterion, nevertheless having the highest value of 72.2% and the lowest of 38.89%. Values below 50% indicate that the **BaseRegressors** have a better performance than the feature selection method in more than half of the used setups, i.e. ML model type, data level, input feature type. For the MDD, Fig. 5(c) shows an interesting trend where most of the models have an enhancement. The only exception to the rule is *PI best* with 1 and 3 features removed. Also, we can see that the improvement is almost opposite to the percentage of models that have an annual return increase, which is an indication that a higher return comes with the cost of higher risk. Comparing the feature selection loss metrics, i.e., `MSE` versus `mean return`, we cannot deduct a clear indication whether one metric is emerging over the other.

We complement these results by those depicted in Fig. 6, where we qualitatively assess the performance improvements between the different approaches under the same criteria used for the experiments reported in Fig. 5. Thus, for each feature selection method, loss metric, and number of removed features, Fig. 6 illustrates the average differences of the MSE, annual returns, and MDD among the feature selection methods and their corresponding **BaseRegressors**.

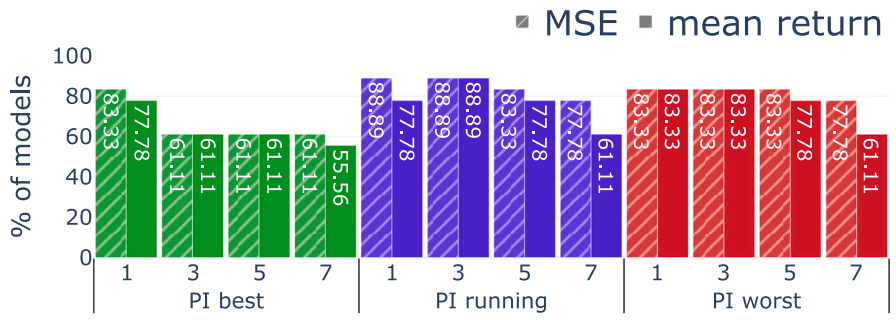
Starting with the forecasting performance evaluation, in Fig. 6(a) negative values are sought, since lower MSE values relative to the ones obtained by the base regressors imply a better forecasting performance. Conversely, for annual returns (Fig. 6(b)), positive values are preferable, since an increase of the annual return relative to the **BaseRegressors** assumes an improvement of the trading strategy when removing features. Furthermore, given that a higher MDD (Fig. 6(c)) means a higher risk, negative values for this metric are desired as this denotes that the feature removal process decreases the trading strategy risk. Under these considerations, we can state that all the feature selection methods bring a performance improvement in the predictive performance (Fig. 6(a)) since all the values are below 0.

In terms of both annual returns (Fig. 6(b)) and MDD (Fig. 6(c)), every feature removal method brings improvements with regards to the financial performance, i.e., by increasing the returns or by lowering the risk. However, there are exceptions to the rule, such as *PI best*, when removing one feature. Also, it is evident a similar pattern as in Fig. 5, namely, higher returns imply increased risk. Furthermore, crossing the two figures (Fig. 5(b) and Fig. 6(b)), another surprising finding can be deducted: *PI best* improvements are significantly higher than *PI worst* or *PI running*.

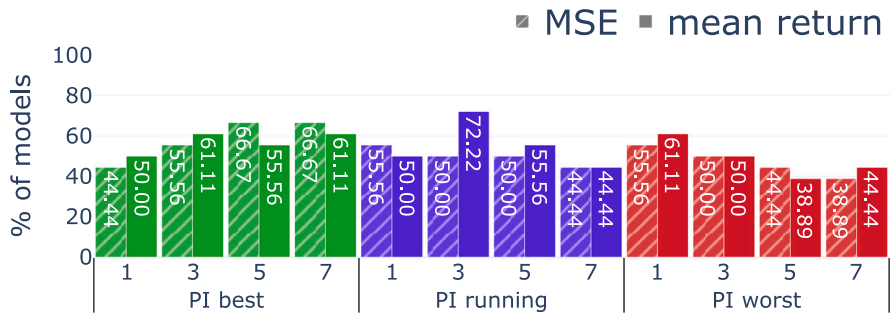
5.2. Financial performance of the XAI StatArb trading strategies

The results from the experiments reported in Figs. 5 and 6 motivate us to take a closer look at the financial performance of the XAI StatArb strategies. Therefore, we present in Fig. 7 statistics of the returns and risk characteristics. Provided that in this paper we have built a large number of models and configurations (3 types of models × 2 data levels × 3 types of features × 3 feature selection methods × 2 feature selection metrics × 4 number of features removed = 432 in total), we choose to present the performances of the best model per configuration, given by the obtained annual return. Note that a configuration is provided by the ML model type, the data level, and the input feature type. Therefore, we present the XAI StatArb performances for 18 such configurations, in contrast to their corresponding **BaseRegressors** and **Buy-and-hold** strategies, i.e., 6 XAI StatArb strategies per each ML model. In Fig. 7 we show a representative set of financial performance indicators, and, in the right-most columns, we present the differences between the annual returns/MDD of the XAI StatArb strategies and the **BaseRegressors**. The differences are color-coded: red colors represent values where the **BaseRegressors** have a better performance. Conversely, blue or green represent better performances of the strategies compared to the **BaseRegressors**. Also, the dimension of the bar is proportional to the difference.

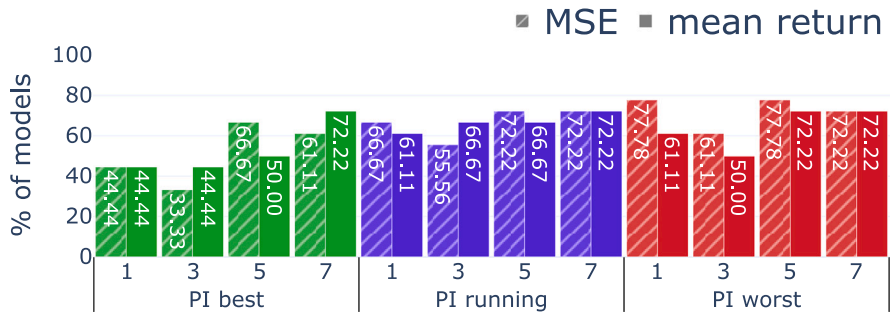
As the reader can notice, all the models have a positive annual return, between 4% and 21%, and most of the models have annual returns above the **Buy-and-hold** strategy (7%). The best performing model is LGB with technical indicators (TI) as features. Moreover, LGB



(a) MSE



(b) Annual returns



(c) MDD

Fig. 5. Percentage of ML models that exhibit improvements over the **BaseRegressors** in terms of (a) MSE, (b) annual returns, and (c) MDD, developed from 2007 to 2021 for the three feature selection methods when removing $K = 1, 3, 5, 7$ features. The MSE as loss function for feature removal is marked with diagonal lines, whereas the `mean return` has no filling pattern.

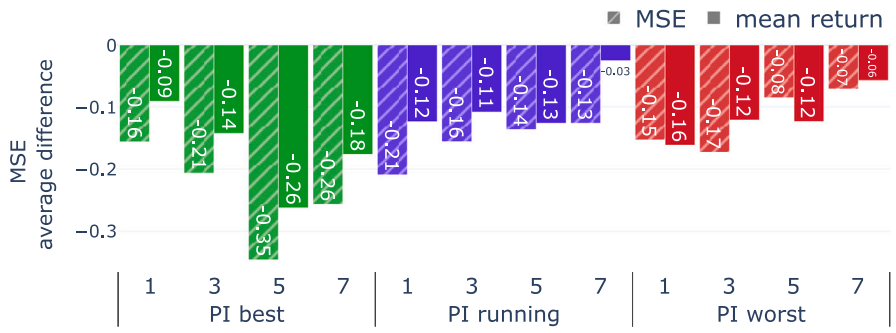
appears to be the best performing forecasting technique on average, with the highest annual returns in any given configuration. It reaches an average annual return of 15%, followed by RF with 11%, and, finally, SVR with 9%. This ranking is highly influenced by the data level as models per sector learn from a large number of observations. Consequently, SVR, that is notoriously unable to handle large datasets having a high number of observations, obtains the worst performance when trained at sector level versus stock level.

In terms of risk, all the models have a lower MDD than the **Buy-and-hold**, with the lowest value (best performing) of 17% (RF, stock level), and the highest value (worst performing) of 47% (SVR, sector level).

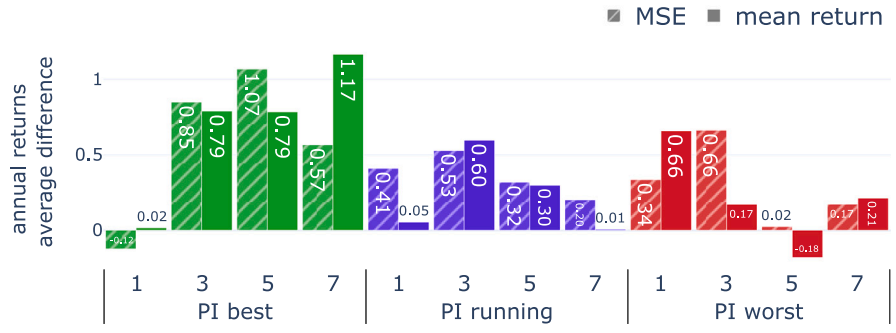
Concerning the best-performing feature selection method, we can see that out of the (best) 18 configurations presented, *PI best* yields the best returns in 10 such cases, *PI worst* in 3 cases, and *PI running* in 4. Furthermore, concerning the annual return difference, the *PI best* attains the leading positions with values such as 12.49 (sector RF with LR) and 5.22 (stock SVR with TI). The results confirm the findings in Fig. 6, where the average difference between the annual returns of the feature selection methods and **BaseRegressors** is higher for the *PI best* strategy.

Concerning the feature selection metric, we can note in Fig. 7 that the `mean return` has 11 occurrences whereas the MSE has 7, meaning that changing the canonical MSE with the `mean return` introduces a significant benefit in the financial performance.

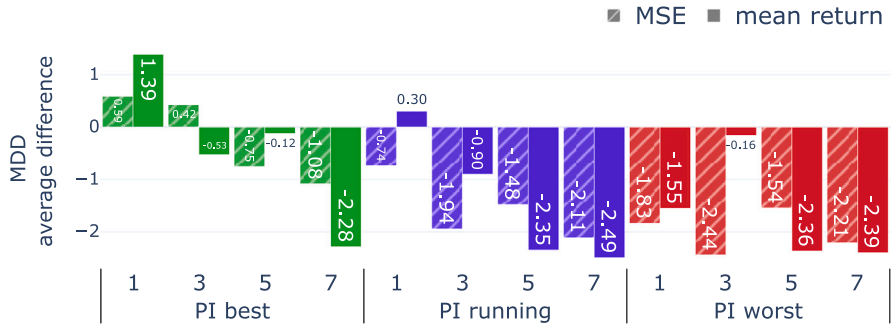
Although all the feature selection methods introduce an improvement in terms of returns, when it comes to the risk, 4 of them show a higher MDD compared to the **BaseRegressors** (i.e., positive values in the MDD difference column), signifying that these trading strategies are more exposed to financial risks. Moreover, 2 of the XAI StatArb strategies are based on SVR models. We could only suppose that XAI enabled SVR are more sensitive to outliers and produce forecasts less calibrated



(a) MSE $\times 1e4$



(b) Annual returns



(c) MDD

Fig. 6. Average difference between the ML models and the corresponding **BaseRegressors** in terms of (a) MSE, (b) annual returns, and (c) MDD, developed from 2007 to 2021 for the three feature selection methods when removing $K = 1, 3, 5, 7$ features. The MSE as loss function for feature removal is marked with diagonal lines, whereas the `mean return` has no filling pattern.

and accurate on the prediction interval extremes. This translates into more volatile down movements.

5.3. Sub-period analysis

In the following, we make a sub-period breakdown to provide more details about the performance of the XAI StatArb strategies in terms of returns and risk profiles. We present such details in Table 2. Additionally, we show in Fig. 8 an illustration of the cumulative profits of the strategies.

For this study, we selected three periods.

Low Volatility Bull Market — represents a short period of only seven months, ranging from January 2007 to August 2007. In this particular period, the XAI StartArb methods show a large diversity of returns which is hard to explain. The mean return is either strongly positive and higher than the **Buy-and-hold** (0.03%), e.g., *PI worst* LGB stock level LR+TI (0.10%), or economically non-significant, e.g., *PI best* LGB sector TI (−0.07). Works such as Huck (2019), Krauss et al. (2017) attribute similar results to the increase of computational power that made algorithmic trading readily available, and the consequent imprint in the market behavior and decrease of the returns. By comparing the average returns of the XAI StatArb methods to the average returns of the **BaseRegressors** (second-last row in Table 2), we can assess that the feature selection process, on average, introduces an improvement, that is, the average return increases from 0.02% to 0.03%.

GFC Crash — corresponds to the period of the Global Financial Crisis (GFC) starting, specifically, in August 2007, and finishing in April 2009. In this turbulent period, with the market with a

Table 2
Sub-period analysis of the XAI StatArb trading strategies performance. The best two financial returns are highlighted in bold.

Method	Feature Selection Metric	Feature Type	Model Type	Level	K	Low Volatility Bull Market						GFC Crash						New Normal					
									Base regressor						Base regressor						Base regressor		
						Mean Return	SD	MDD	Mean Return	SD	MDD	Mean Return	SD	MDD	Mean Return	SD	MDD	Mean Return	SD	MDD	Mean Return	SD	MDD
PI worst	MSE	LR	LGB	sector	3	0.01	0.54	9.34	0.01	0.54	9.34	0.15	1.64	19.39	**0.12**	1.74	24.60	0.06	0.76	15.41	0.04	0.76	17.20
PI running	MSE	LR+TI	LGB	sector	1	−0.02	0.44	8.34	−0.03	0.42	8.29	0.09	1.32	21.91	0.05	1.24	29.25	0.08	0.70	14.44	0.04	0.70	21.83
PI best	MSE	TI	LGB	sector	3	−0.07	0.49	11.07	−0.07	0.48	10.58	0.06	1.61	31.04	0.02	1.61	32.71	0.07	0.70	12.83	0.06	0.71	17.84
PI best	mean return	LR	LGB	stock	7	**0.10**	0.54	4.19	0.10	0.51	3.73	0.16	1.14	16.77	0.11	1.15	18.76	0.07	0.67	18.71	0.04	0.68	16.88
PI worst	mean return	LR+TI	LGB	stock	5	**0.11**	0.45	4.16	0.11	0.45	4.16	0.15	1.37	12.58	0.15	1.38	13.42	0.05	0.67	19.42	0.05	0.67	25.66
PI best	MSE	TI	LGB	stock	7	0.06	0.61	5.20	0.03	0.58	6.77	**0.23**	1.20	11.30	**0.22**	1.14	12.20	**0.09**	0.70	20.69	**0.06**	0.70	25.09
PI best	MSE	LR	RF	sector	7	−0.03	0.44	7.97	0.02	0.52	7.03	0.14	1.23	13.48	0.04	1.45	30.27	**0.08**	0.74	20.25	0.03	0.80	26.65
PI running	mean return	LR+TI	RF	sector	7	−0.03	0.46	11.68	−0.04	0.48	11.39	0.06	1.57	26.12	0.04	1.69	40.07	0.07	0.77	14.46	0.05	0.76	14.49
PI running	mean return	TI	RF	sector	3	0.01	0.51	5.21	−0.03	0.50	6.06	0.06	1.74	32.90	0.07	1.73	33.13	0.07	0.73	23.65	0.05	0.72	23.89
PI best	mean return	LR	RF	stock	5	0.06	0.51	4.46	0.08	0.49	3.38	0.08	1.20	14.13	0.04	1.26	22.69	0.06	0.68	16.72	0.03	0.68	20.05
PI best	mean return	LR+TI	RF	stock	7	0.03	0.49	4.82	0.02	0.49	4.79	0.13	1.28	13.53	0.12	1.29	13.58	0.03	0.72	25.88	0.02	0.72	29.91
PI best	mean return	TI	RF	stock	5	0.01	0.55	9.35	0.02	0.56	8.48	0.11	1.30	17.51	0.05	1.40	22.64	0.05	0.72	40.42	0.03	0.74	36.10
PI best	mean return	LR	SVR	sector	5	0.02	0.44	3.71	−0.03	0.49	6.32	0.06	1.81	29.50	0.08	1.70	25.89	0.06	0.85	29.83	0.03	0.84	34.72
PI best	MSE	LR+TI	SVR	sector	7	0.01	0.43	3.92	0.01	0.44	4.39	−0.03	1.39	31.55	−0.07	1.47	42.22	0.05	0.63	20.62	0.03	0.64	36.21
PI running	MSE	TI	SVR	sector	3	0.05	0.50	4.59	0.04	0.47	6.47	−0.09	1.98	46.69	−0.09	1.93	47.93	0.05	0.60	18.42	0.03	0.60	17.63
PI worst	mean return	LR	SVR	stock	1	0.05	0.48	3.64	0.05	0.46	4.08	0.15	1.62	12.48	0.13	1.59	12.39	0.06	0.65	18.79	**0.06**	0.65	18.74
PI running	mean return	LR+TI	SVR	stock	5	0.01	0.44	5.09	0.00	0.44	5.19	0.10	1.01	7.61	0.09	1.00	8.13	0.04	0.63	20.01	0.03	0.63	17.74
PI best	mean return	TI	SVR	stock	5	0.01	0.54	7.30	−0.01	0.51	6.59	**0.24**	1.42	11.46	0.09	1.37	19.25	0.03	0.59	19.12	0.02	0.58	27.98
Average Returns						0.03			0.02			0.10			0.07			0.06			0.04		
Buy-and-hold						0.03	0.80	6.42				−0.11	2.32	56.47				0.06	1.10	34.1			

Method	Feature Selection Metric	Feature Type	Model Type	Level	K	Annual return	Mean return	Q1	Q2	Q3	MDD	Annual return BaseRegressor	MDD BaseRegressor	Annual return difference	MDD difference
PI worst	MSE	LR	LGB	sector	3	13.59	0.05	-0.36	0.05	0.45	26.95	11.41	26.44	2.18	0.51
PI best	MSE	TI	LGB	sector	3	13.00	0.05	-0.35	0.05	0.44	31.04	11.61	32.71	1.39	-1.67
PI running	MSE	LR+TI	LGB	sector	1	14.18	0.06	-0.38	0.05	0.45	26.71	10.38	38.89	3.8	-12.18
PI best	MSE	TI	LGB	stock	7	**20.93**	**0.08**	-0.35	0.06	0.47	20.69	**19.75**	25.09	1.18	-4.4
PI best	mean return	LR	LGB	stock	7	15.88	0.06	-0.35	0.05	0.45	18.71	12.73	18.76	3.15	-0.05
PI worst	mean return	LR+TI	LGB	stock	5	12.85	0.05	-0.36	0.02	0.44	19.42	11.13	25.66	1.72	-6.24
Average return						15.07									
PI best	MSE	LR	RF	sector	7	**17.44**	**0.07**	-0.37	0.05	0.5	20.25	4.95	30.27	12.49	-10.02
PI running	mean return	LR+TI	RF	sector	7	12.02	0.05	-0.4	0.03	0.45	26.12	10.26	40.76	1.76	-14.64
PI running	mean return	TI	RF	sector	3	12.4	0.05	-0.38	0.04	0.43	32.90	11.89	33.13	0.51	-0.23
PI best	mean return	LR	RF	stock	5	11.28	0.05	-0.37	0.03	0.45	16.72	8.84	22.69	2.44	-5.97
PI best	mean return	LR+TI	RF	stock	7	7.56	0.03	-0.39	0.03	0.44	25.88	5.59	29.91	1.97	-4.03
PI best	mean return	TI	RF	stock	5	10.08	0.04	-0.39	0.01	0.43	40.42	8.09	36.1	1.99	4.32
Average return						11.80									
PI best	mean return	LR	SVR	sector	5	8.41	0.04	-0.41	0.04	0.51	29.83	6.09	34.72	2.32	-4.89
PI best	MSE	LR+TI	SVR	sector	7	4.22	0.02	-0.33	0.02	0.38	41.60	-0.58	51.44	4.8	-9.84
PI running	MSE	TI	SVR	sector	3	6.63	0.03	-0.29	0.02	0.35	46.69	3.25	49.18	3.38	-2.49
PI worst	mean return	LR	SVR	stock	1	16.44	0.06	-0.33	0.04	0.44	18.79	**14.78**	18.74	1.66	0.05
PI running	mean return	LR+TI	SVR	stock	5	10.21	0.04	-0.31	0.03	0.38	20.01	8.92	17.74	1.29	2.27
PI best	mean return	TI	SVR	stock	5	10.99	0.04	-0.33	0.03	0.41	19.12	5.77	27.98	5.22	-8.86
Average return						9.48									
Buy-and-Hold						7.17	0.03	-0.48	0.06	0.60	56.47				

Fig. 7. Financial performance of the XAI StatArb trading strategies gross of trading costs and fees, over the trading period 2007–2021. The best two return performances are highlighted in bold.

steep decline and a very high level of VIX index (Krauss et al., 2017), the XAI StatArb strategies are most proliferating. For example, *PI best* LGB TI stock obtains an all-time high of 0.23% daily mean return. By comparison, in the previous period, it reaches only a 0.06% of daily mean return, or, in the next period, only a 0.09%. The performance of the portfolios alternates rapidly between significant rises and falls, a fact confirmed by the higher standard deviations (SD) of the returns. The same can be reaffirmed by looking at the cumulative returns presented in Fig. 8. This figure also reveals that this behavior is more noticeable in the case of sector-level models. Comparable findings have been confirmed in the literature as well. For example, Huck (2019) investigated the role that input features play in the StatArb financial performance, in the period of GFC, reporting for the best performing combination of ML models and input features a daily mean return of 0.3%. However, the author have reported as the average of daily returns of the models as being 0.01%, that is significantly lower than the one of our XAI StatArb strategies (0.10%) or our **BaseRegressors** (0.07%). Even though most XAI StatArb methods exhibit positive daily returns, not all models perform notably well in this period; e.g., *PI best* SVR sector TI is the laggard of the whole set of methods and has a performance similar to the **Buy-and-hold** both in terms of returns and risk. *PI best* SVR sector TI mean return is −0.09% whereas **Buy-and-hold** has a mean return of −0.11%. The MDD for *PI best* SVR sector TI is 46% and 56% for **Buy-and-hold**. However, in Table 2, only 2 out of 18 methods register negative returns during this period. Huck (2019) reported 17 out of 44 models with negative mean daily returns, the worst performing model having −0.28%.

New Normal — the period ranges from 2009-04-01 to 2020-12-31 and corresponds to the recovery after the GFC, including short-lived market events such as the European Debt Crisis of 2011, the period between April and October 2014, the market downturn in August–September 2015, or the global COVID pandemic market crisis. Nevertheless, this time-span accounts for a bull market period. In the literature, post-Global Financial Crisis periods (Carta, Consoli, Piras et al., 2021; Krauss et al., 2017) reported modest returns by comparison with the other periods, even reaching a plateau. In this interval, investing strategies such as **Buy-and-hold** are particularly challenging as baselines, as the literature brings empirical evidence that StatArb trading strategies are not as profitable as they used to be. Even under such setback, Table 2 shows that the XAI StatArb manages to obtain mean daily returns above the **Buy-and-hold**. Indeed, the **Buy-and-hold** strategy attains a return of on par with the average returns of the XAI StatArb methods. Inspecting Fig. 8, the reader can notice that most of the models show a positive trend. But indeed, some of them reach a plateau after 2012, e.g., *SVR stock level LR+TI* or *LGB stock level LR*. Contrarily, *PI best LGB stock level TI* has a steep and positive curve in the most recent period between 2014–2015, moderate positive/negative fluctuations after 2016, and a sharp increase in the period of 2020 global COVID pandemic. We attribute such results to the trait that tree-based methods such as LGB have, that is, their robustness to noise and outliers that make them particularly suitable to a financial forecasting setting.

6. Conclusions and future work

This work proposes a machine learning approach powered by explainability techniques as an integrative part of an algorithmic trading pipeline. It aims at bridging the gap between typical financial practice (employing machine learning approaches) and robust explainable artificial intelligence applied on a large stock set. At the same time, this work aims to extend the existing literature on statistical arbitrage trading based on machine learning. We follow the well-established steps of a statistical arbitrage trading system: firstly forecasting the stock price returns, then ranking the stocks based on the predicted returns, and lastly, trading several pairs of stocks. The presented applications and forecasting models, focusing on the U.S. market, are based on various input features and report results on a trading period ranging from 2007 to 2021. Different than standard machine learning approaches, we do not simply use a framework to produce buy and sell signals. Instead, we introduce a feature selection step so that the machine learning algorithms predict the next day returns on a representative/informative feature set. To achieve this goal, we propose three methods for feature selection: *PI best*, *PI running*, and *PI worst*. The methods have the key traits of being model agnostic, and of performing the feature selection process in a post-hoc manner without interfering with the prediction and, most importantly, learning the relevant features both at stock/sector level and in cross-sections (i.e., globally). We test our proposed methods by considering several setups: various

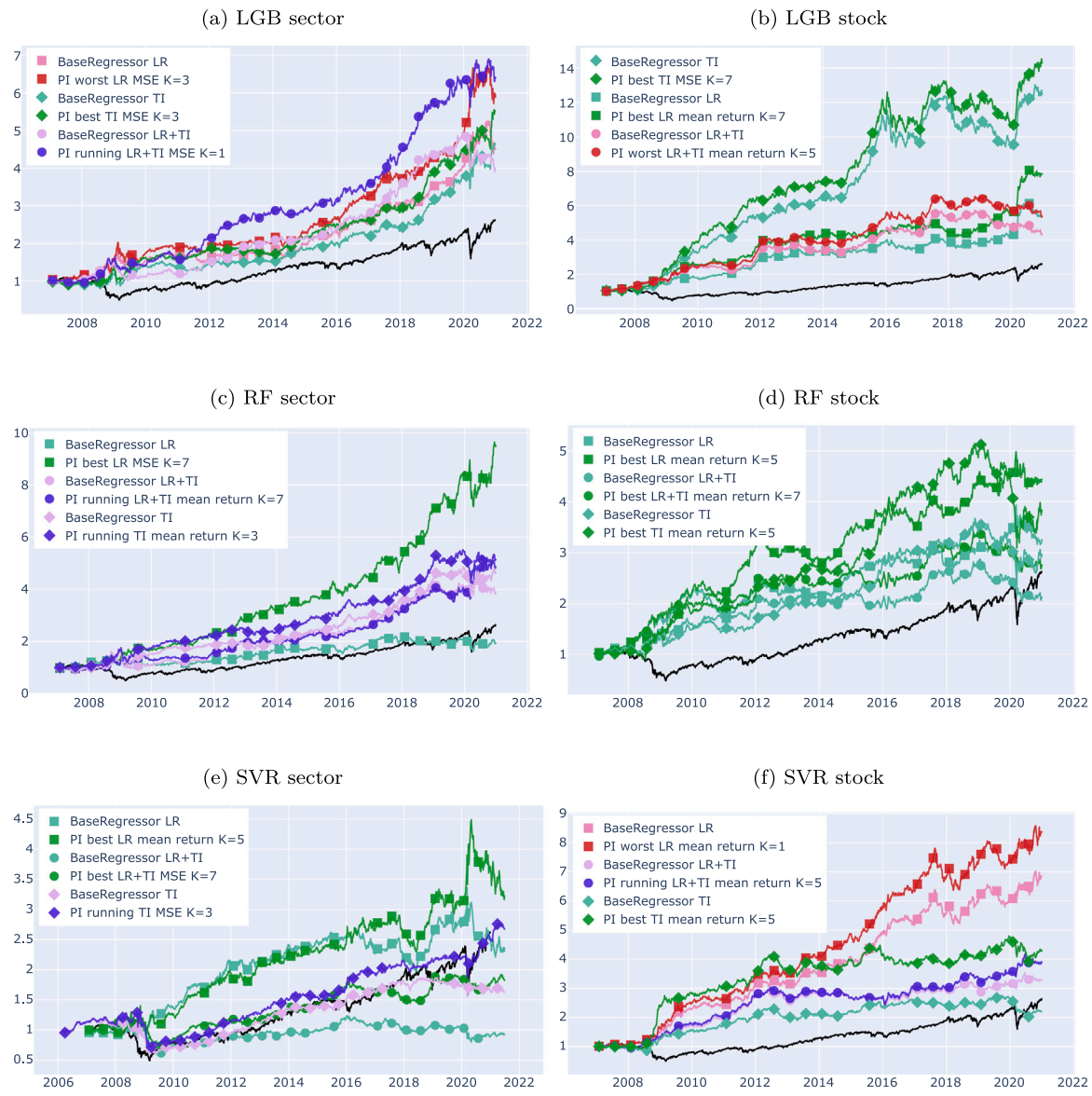


Fig. 8. Daily compounded returns gross of trading costs and fees for the XAI StatArb trading strategies compared to their corresponding **BaseRegressors** and **Buy-and-hold**, respectively. The trading period ranges from 2007 to 2021. On the left-hand side, Figs. 8(a), 8(c), and 8(e) represent *sector* level models, whereas, on the right-hand-side, Figs. 8(b), 8(d), and 8(f) represent *stock* level models. The **Buy-and-hold** is represented by a continuous black line. Input features are represented as: LR - squares, TI - diamonds, and LR+TI - circles.

machine learning algorithms (i.e., Gradient Boosting, Random Forests, and Support Vector Regressors), different input feature sets (i.e., lagged returns, technical indicators, and their combination), and distinct data levels (i.e., sector or stock levels), which yielded to an overall of 432 such models. By conducting such extensive testing from predictive as well as financial performance angles, several discussion points emerge. The feature selection methods introduce clearly improvements both in terms of predictive and financial performance. All raw returns are positive. Given that more predictors translate into potentially more noise, there is a clear improvement over the compared base regressor (i.e., the machine learning model without any features removed) when dropping several features. Besides, replacing the widely-used MSE with the mean return as a loss function within the feature selection process constitutes an essential change of the XAI StatArb strategies. In terms of annual returns, our approach brings the most significant enhancements over the base regressors.

The feature selection approach we have pursued in this article still has many possible refinements and details to be further analyzed, which we aim in future research. Although the capacity of our proposed methodology to provide valuable signals is investigated under a different number of features to remove, the dynamic adaptation of the number of features to remove would represent a valuable addition to the framework. Also, given that the main focus of this article consists in exploring the impact of feature selection on building long–short portfolios, we left over the study about the impact of the transaction cost. This could be an interesting point to investigate further in the future. Furthermore, from a more general perspective, a careful and systematic study on the influence of the various employed hyperparameters (i.e., tailored hyperparameters for each stock or sector) on the overall algorithmic performance, along with an analysis on the temporal stability of the models (i.e., sensitivity to train and test window lengths) would also be extremely useful and stimulating points to investigate more in detail in future works.

CRediT authorship contribution statement

Salvatore Carta: Visualization, Investigation, Formal analysis, Writing – original draft, Validation, Supervision, Project administration,

Writing – review & editing. **Sergio Consoli:** Conceptualization, Methodology, Software, Formal analysis, Data curation, Writing – original draft, Investigation, Writing – review & editing. **Alessandro Sebastian Podda:** Software, Data curation, Formal analysis, Writing – original draft, Visualization, Investigation, Writing – review & editing. **Diego Reforgiato Recupero:** Visualization, Investigation, Formal analysis, Writing – original draft, Validation, Supervision, Project administration, Writing – review & editing. **Maria Madalina Stanciu:** Conceptualization, Methodology, Software, Formal analysis, Data curation, Writing – original draft, Investigation, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The primary funding body of this research work was the European Commission, Joint Research Centre. The views expressed are purely those of the authors and do not, in any circumstance, be regarded as stating an official position of the European Commission.

References

Adadi, A., & Berrada, M. (2018). Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). *IEEE Access*, *6*, 52138–52160. http://dx.doi.org/10.1109/ACCESS.2018.2870052.

Atsalakis, G. S., & Valavanis, K. P. (2009). Surveying stock market forecasting techniques – Part II: Soft computing methods. *Expert Systems with Applications*, *36*(3), 5932–5941.

Avellaneda, M., & Lee, J.-H. (2010). Statistical arbitrage in the US equities market. *Quantitative Finance*, *10*(7), 761–782. http://dx.doi.org/10.1080/14697680903124632.

Barbaglia, L., Consoli, S., Manzan, S., Reforgiato Recupero, D., Saisana, M., & Tiozzo Pezzoli, L. (2021). Data science technologies in economics and finance: A gentle walk-in. In *Data science for economics and finance: Methodologies and applications* (pp. 1–17). Cham: Springer International Publishing, http://dx.doi.org/10.1007/978-3-030-66891-4_1.

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

Carhart, M. M. (1997). On persistence in mutual fund performance. *The Journal of Finance*, *52*(1), 57–82. http://dx.doi.org/10.1111/j.1540-6261.1997.tb03808.x.

Carta, S. M., Consoli, S., Piras, L., Podda, A. S., & Reforgiato Recupero, D. (2021). Explainable machine learning exploiting news and domain-specific lexicon for stock market forecasting. *IEEE Access*, *9*, 30193–30205. http://dx.doi.org/10.1109/ACCESS.2021.3059960.

Carta, S. M., Consoli, S., Podda, A. S., Reforgiato Recupero, D., & Stanciu, M. M. (2021). Ensembling and dynamic asset selection for risk-controlled statistical arbitrage. *IEEE Access*, *9*, 29942–29959. http://dx.doi.org/10.1109/ACCESS.2021.3059187.

Carta, S., Corriga, A., Ferreira, A., Reforgiato Recupero, D., & Saia, R. (2019). A holistic auto-configurable ensemble machine learning strategy for financial trading. *Computation*, *7*(4), 67.

Carta, S., Podda, S., Reforgiato Recupero, D., & Stanciu, M. M. (2022). Explainable AI for financial forecasting. In *Lecture notes in computer science*: *Vol. 13164*, *The 7th international conference on machine learning, optimization, and data science* (pp. 51–69). http://dx.doi.org/10.1007/978-3-030-95470-3_5.

Carta, S., Reforgiato Recupero, D., Saia, R., & Stanciu, M. M. (2020). A general approach for risk controlled trading based on machine learning and statistical arbitrage. In *Lecture notes in computer science*: *Vol. 12565*, *The sixth international conference on machine learning, optimization, and data science* (pp. 489–503).

Carvalho, D. V., Pereira, E. M., & Cardoso, J. S. (2019). Machine learning interpretability: A survey on methods and metrics. *Electronics*, *8*(8), 832.

Cavalcante, R. C., Brasileiro, R. C., Souza, V. L., Nobrega, J. P., & Oliveira, A. L. (2016). Computational intelligence and financial markets: A survey and future directions. *Expert Systems with Applications*, *55*, 194–211.

Chalimourda, A., Schölkopf, B., & Smola, A. J. (2004). Experimentally optimal ν in support vector regression for different noise models and parameter settings. *Neural Networks*, *17*(1), 127–141, https://doi.org/10.1016/S0893-6080(03)00209-0.

Cheng, C., Sa-Ngasoongsong, A., Beyca, O., Le, T., Yang, H., Kong, Z. J., & Bukkapatnam, S. T. (2015). Time series forecasting for nonlinear and non-stationary processes: a review and comparative study. *IIE Transactions*, *47*(10), 1053–1071. http://dx.doi.org/10.1080/0740817X.2014.999180.

de Prado, M. L. (2018). *Advances in financial machine learning* (1st ed.). New York, United States: Wiley Publishing.

Emmert-Streib, F., Yli-Harja, O., & Dehmer, M. (2020). Explainable artificial intelligence and machine learning: A reality rooted perspective. *WIREs Data Mining and Knowledge Discovery*, *10*(6), Article e1368, https://doi.org/10.1002/widm.1368.

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

Flori, A., & Regoli, D. (2021). Revealing pairs-trading opportunities with long short-term memory networks. *European Journal of Operational Research*, *295*(2), 772–791. http://dx.doi.org/10.1016/j.ejor.2021.03.009.

Gatev, E., Goetzmann, W. N., & Rouwenhorst, K. G. (2006). Pairs trading: Performance of a relative-value arbitrage rule. *Review of Financial Studies*, *19*(3), 797–827. http://dx.doi.org/10.1093/rfs/hhj020.

Goddard, M. (2017). The EU General Data Protection Regulation (GDPR): European regulation that has a global impact. *International Journal of Market Research*, *59*(6), 703–705.

Gunduz, H. (2021). An efficient stock market prediction model using hybrid feature reduction method based on variational autoencoders and recursive feature elimination. *Financial Innovation*, *7*(1), 28. http://dx.doi.org/10.1186/s40854-021-00243-3.

Hafezi, R., Shahrabi, J., & Hadavandi, E. (2015). A bat-neural network multi-agent system (BNNMAS) for stock price prediction: Case study of DAX stock price. *Applied Soft Computing*, *29*, 196–210.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: data mining, inference, and prediction* (2nd ed.). New York: Springer Series in Statistics.

Henrique, B. M., Sobreiro, V. A., & Kimura, H. (2019). Literature review: Machine learning techniques applied to financial market prediction. *Expert Systems with Applications*, *124*, 226–251.

Huck, N. (2019). Large data sets and machine learning: Applications to statistical arbitrage. *European Journal of Operational Research*, *278*(1), 330–342. http://dx.doi.org/10.1016/J.EJOR.2019.04.013.

Kara, Y., Acar Boyacioglu, M., & Baykan, O. K. (2011). Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange. *Expert Systems with Applications*, *38*(5), 5311–5319.

Kaufman, C., & Lang, D. T. (2015). Pairs trading. *Data Science in R: A Case Studies Approach To Computational Reasoning and Problem Solving*, *1*(April 2015), 241–308.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in neural information processing systems (Vol. 30)* (pp. 3146–3154).

Khandani, A. E., & Lo, A. W. (2011). What happened to the quants in August 2007? Evidence from factors and transactions data. *Journal of Financial Markets*, *14*(1), 1–46.

Kraus, M., & Feuerriegel, S. (2017). Decision support from financial disclosures with deep neural networks and transfer learning. *Decision Support Systems*, *104*, 38–48. http://dx.doi.org/10.1016/j.dss.2017.10.001.

Krauss, C. (2017). Statistical arbitrage pairs trading strategies: review and outlook. *Journal of Economic Surveys*, *31*(2), 513–545. http://dx.doi.org/10.1111/joes.12153.

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

Krishnan, M. (2020). Against interpretability: a critical examination of the interpretability problem in machine learning. *Philosophy & Technology*, *33*(3), 487–502.

Kumar, G., Jain, S., & Singh, U. P. (2021). Stock market forecasting using computational intelligence: A survey. *Archives of Computational Methods in Engineering*, *28*(3), 1069–1101. http://dx.doi.org/10.1007/s11831-020-09413-5.

Langer, M., Oster, D., Speith, T., Hermanns, H., Kästner, L., Schmidt, E., Sesing, A., & Baum, K. (2021). What do we want from Explainable Artificial Intelligence (XAI)?–A stakeholder perspective on XAI and a conceptual model guiding interdisciplinary XAI research. *Artificial Intelligence*, *296*, Article 103473.

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. http://dx.doi.org/10.1016/j.eswa.2009.02.038.

Lee, T. K., Cho, J. H., Kwon, D. S., & Sohn, S. Y. (2019). Global stock market investment strategies based on financial network indicators using machine learning techniques. *Expert Systems with Applications*, *117*, 228–242. http://dx.doi.org/10.1016/J.ESWA.2018.09.005.

Li, B., & Hoi, S. C. H. (2014). Online portfolio selection: A survey. *ACM Computing Surveys*, *46*(3), http://dx.doi.org/10.1145/2512962.

Lo, A. W. (2010). *Hedge funds: An analytic perspective*. Princeton University Press.

Man, X., & Chan, E. P. (2020). The best way to select features? Comparing MDA, LIME, and SHAP. *The Journal of Financial Data Science*, http://dx.doi.org/10.3905/jfds.2020.1.047.

Microsoft/LightGBM (2022). Lightgbm documentation. https://lightgbm.readthedocs.io/en/latest/Python-API.html. (Last accessed: 12 May 2022).

Molnar, C., Casalicchio, G., & Bischl, B. (2020). Interpretable machine learning – a brief history, state-of-the-art and challenges. In I. Koprinska, & et al. (Eds.), *ECML PKDD 2020 workshops* (pp. 417–431). Cham: Springer International Publishing.

Patel, J., Shah, S., Thakkar, P., & Kotecha, K. (2015). Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques. *Expert Systems with Applications*, *42*(1), 259–268.

Patel, J., Shah, S., Thakkar, P., & Kotecha, K. (2015). Predicting stock market index using fusion of machine learning techniques. *Expert Systems with Applications*, *42*(4), 2162–2172. http://dx.doi.org/10.1016/J.ESWA.2014.10.031.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Pimenta, A., Nametala, C. A., Guimarães, F. G., & Carrano, E. G. (2018). An automated investing method for stock market based on multiobjective genetic programming. *Computational Economics*, *52*(1), 125–144.

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, *1*(5), 206–215.

Rüping, S. (2006). *Learning interpretable models* (Ph.D. Dissertation), TU Dortmund.

Smolander, J., Dehmer, M., & Emmert-Streib, F. (2019). Comparing deep belief networks with support vector machines for classifying gene expression data from complex disorders. *FEBS Open Bio*, *9*(7), 1232–1248. http://dx.doi.org/10.1002/2211-5463.12652.

Strobl, C., Boulesteix, A.-L., Kneib, T., Augustin, T., & Zeileis, A. (2008). Conditional variable importance for random forests. *BMC Bioinformatics*, *9*(1), 307. http://dx.doi.org/10.1186/1471-2105-9-307.

Strumbelj, E., & Kononenko, I. (2010). An efficient explanation of individual classifications using game theory. *Journal of Machine Learning Research*, *11*, 1–18.

Thakkar, A., & Chaudhari, K. (2021). A comprehensive survey on deep neural networks for stock market: The need, challenges, and future directions. *Expert Systems with Applications*, *177*, Article 114800. http://dx.doi.org/10.1016/j.eswa.2021.114800.

Vapnik, V. N. (1999). An overview of statistical learning theory. *IEEE Transactions on Neural Networks*, *10*(5), 988–999.

Vidyamurthy, G. (2004). *Pairs trading: Quantitative methods and analysis*. John Wiley & Sons.

Yamada, Y., Lindenbaum, O., Negahban, S., & Kluger, Y. (2020). Feature selection using stochastic gates. In *Proceedings of machine learning research*: *Vol. 119*, *Proceedings of the 37th international conference on machine learning* (pp. 10648–10659). PMLR, URL: http://proceedings.mlr.press/v119/yamada20a.html.

Zhao, Z., Anand, R., & Wang, M. (2019). Maximum relevance and minimum redundancy feature selection methods for a marketing machine learning platform. In *2019 IEEE international conference on data science and advanced analytics* (pp. 442–452). http://dx.doi.org/10.1109/DSAA.2019.00059.

Zhou, J., & Chen, F. (2019). Towards trustworthy human-AI teaming under uncertainty. In *IJCAI 2019 workshop on explainable AI (XAI)*.

Zhou, J., Gandomi, A. H., Chen, F., & Holzinger, A. (2021). Evaluating the quality of machine learning explanations: A survey on methods and metrics. *Electronics*, *10*(5), 593.