IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 18, 2021, accepted February 10, 2021, date of publication February 12, 2021, date of current version February 25, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3059187

Ensembling and Dynamic Asset Selection for Risk-Controlled Statistical Arbitrage

SALVATORE M. CARTA[1], (Member, IEEE), SERGIO CONSOLI[2], ALESSANDRO SEBASTIAN PODDA[1], DIEGO REFORGIATO RECUPERO[1], AND MARIA MADALINA STANCIU[1]

[1]Department of Mathematics and Computer Science, University of Cagliari, 09124 Cagliari, Italy
[2]European Commission, Joint Research Centre (DG-JRC), Directorate A-Strategy, Work Programme and Resources, Scientific Development Unit, I-21027 Ispra, Italy

Corresponding author: Sergio Consoli (sergio.consoli@ec.europa.eu)

We would like to thank the Centre for Advanced Studies at the Joint Research Centre of the European Commission for guidance and support during the development of this research work. This work was also partially supported by the POR FESR 2014-2020 project: "AlmostAnOracle - AI and Big Data Algorithms for Financial Time Series Forecasting."

ABSTRACT In recent years, machine learning algorithms have been successfully employed to leverage the potential of identifying hidden patterns of financial market behavior and, consequently, have become a land of opportunities for financial applications such as algorithmic trading. In this paper, we propose a statistical arbitrage trading strategy with two key elements: an ensemble of regression algorithms for asset return prediction, followed by a dynamic asset selection. More specifically, we construct an extremely heterogeneous ensemble ensuring *model diversity* by using state-of-the-art machine learning algorithms, *data diversity* by using a feature selection process, and *method diversity* by using individual models for each asset, as well models that learn cross-sectional across multiple assets. Then, their predictive results are fed into a quality assurance mechanism that prunes assets with poor forecasting performance in the previous periods. We evaluate the approach on historical data of component stocks of the S&P500 index. By performing an in-depth risk-return analysis, we show that this setup outperforms highly competitive trading strategies considered as baselines. Experimentally, we show that the dynamic asset selection enhances overall trading performance both in terms of return and risk. Moreover, the proposed approach proved to yield superior results during both financial turmoil and massive market growth periods, and it showed to have general application for any risk-balanced trading strategy aiming to exploit different asset classes.

INDEX TERMS Stock market forecast, statistical arbitrage, machine learning, ensemble learning.

I. INTRODUCTION

Statistical arbitrage trading, or StatArb for short, exploits some statistical patterns in the dynamics of security prices, thus obtaining, with a high probability, a return larger than the risk-free return. StatArb roots back from pairs trading strategy [2], and was first developed at Morgan Stanley by a quantitative trading group under the lead of Nunzio Tartaglia in the mid-1980s on Wall Street [3]. Pairs trading, a simplified form of StatArb, involves forming portfolios of two related stocks with relatively close pricing. The intuition behind pairs trading is to exploit the spread of expected returns of financial assets. When using such a trading strategy, investors go long on the underpriced asset with the highest expected return and short the portfolio of assets with the lowest expected return. These strategies typically tend to make a large number of individual independent trades with a positive expected return, thereby reducing the risk of the strategy. The arbitrage opportunities exist as a consequence of the market inefficiency and the profits are realized by taking trading positions when the mispricing of the assets correct themselves in the future. Moreover, because the spread between assets' prices is considered to be uncorrelated with market returns, pairs trading and, by extension, StatArb, are market-neutral strategies.

On a high level, StatArb implies automatically trading a set of assets that construct a portfolio [4] and comprises two phases: (i) the *scoring* phase, where each asset is assigned a relevance score, with high scores indicating assets that should be held long and low scores indicating assets that are candidates for short operations; and (ii) the *risk reduction*

The associate editor coordinating the review of this manuscript and approving it for publication was Bohui Wang.

phase, where the assets are combined to eliminate, or at least significantly reduce the risk factor [5], [6].

In the context of StatArb, there are two challenges when implementing such a strategy. First, correctly identifying pairs of assets that exhibit similar behavior and determining the point in time when the prices start moving away from each other and open trading positions. For this reason, over the years, a plethora of *statistical and econometric techniques* have been developed to analyze financial data such as distance based [7], co-integration approach [8], and models based on stochastic spread [9]. Second, determine the correct moment in time when the prices converge and the equilibrium is reached, to close the trading positions.

This insufficiency, coupled with the large number of assets involved in trading, has led to the need of introducing a forecasting component and, consequently, the rise of machine learning models [10]. However, when incorporating machine learning algorithms, the famous efficient market hypothesis (EMH) gives a pessimistic view as it implies that the financial market is efficient [11], and, as such, technical or fundamental analysis (or any analysis) would not yield any consistent over-average profit to investors. Furthermore, since investors often exhibit irrational behavior, the price changes differently. One possible cause for that is the under-reaction of the investors, like *anchoring*. Conversely, when the asset price reaches the underlying value of the company, it usually continues moving in the same direction, exceeding the real value of the asset, a phenomenon called *overreaction* which can be attributed to investors' overconfidence and bias. This leads to financial data containing a large amount of noise, jump, and movement, and as a consequence, such time-series are highly non-stationary in time and notoriously unpredictable [12], thus the forecasting performances are substandard. To mitigate the noise problem, a successful approach has proven to be the use of ensembles. They have demonstrated superior predictive performance compared to individual forecasting models, hence their notable success in different domains such as credit scoring [13], sentiment analysis [14], power systems control [15] or natural calamities forecasting [16]. In the literature, we can find several implementations of StatArb that apply classification to construct the trading portfolio [17], [18]. In this regard, our work applies predicted return in building the portfolio by buying long assets with the highest expected return and short selling assets with the lowest expected return. Although regression in the context of financial predictions poses more challenges [19], [20], it allows for a more *granular ranking*, without reference to any balance point.

Therefore, in this paper, we propose a general approach for risk-controlled trading based on machine learning and StatArb. The approach employs an ensemble of regressors for which we ensure three levels of heterogeneity: (i) *forecasting algorithms*, as the approach can be implemented with any state-of-the-art forecasting algorithms; (ii) *data level*, as we train our models with information pertaining to constituents of financial time series with a diversified feature set, considering not only lagged daily prices return but also a series of technical indicators; and (iii) *diversified models*, as we are training the models using either data from individual assets or aggregated assets' data pertaining to the same industry. Finally, in our approach, after the assets have been ranked in descending order, we propose the use of a *dynamic asset selection*, which looks at the past and influences the ranking by removing assets with bad past behavior. Then, the strategy buys (performing long operations) the flop k assets and sells (performing short operations) the top k assets. Furthermore, we propose one possible instance of our approach that has been configured for intra-day operations and on the well-known S&P500 Index. The regressors we have employed for such an instance are the following state-of-the-art machine learning algorithms: Random Forests (RF), Light Gradient Boosted trees (LGB), Support Vector Regressors (SVR), and the widely known statistical model, Auto-Regressive Integrated Moving Average (ARIMA). To validate the chosen configuration, we evaluate the performance of our approach from both return and risk performance perspectives. The comparisons against the baselines clearly illustrate the superiority of the proposed methodology in performing the forecast.

This paper extends a previous conference work [1], which has been completely revised and rewritten, providing new analyses and results. In summary, its contributions are the following:

1) We propose a general approach for risk-controlled trading based on machine learning and StatArb, by defining the problem as a regression of price returns, and to be easily implemented using different types of assets;
2) We propose an ensemble methodology for StatArb, tackling the ensemble construction from three different perspectives:
   - *model diversity*, by using machine learning algorithms and even statistical algorithms;
   - *data diversity*, by considering lagged price returns and technical indicators to enrich the data used by models;
   - *method diversity*, by simultaneously training single models across several assets (*i.e.*, models per industries) and, conversely, models for each asset;
3) We provide a possible instance of our approach for intra-day trading with different kinds of regressors (machine learning algorithms and statistical models) for StatArb within the S&P500 index;
4) We develop a dynamic asset selection based on models' most recent prediction performance, with brand new experimental results aimed at finding the influence of the lookback period on the proposed approach;
5) We performed an in-depth risk-return evaluation, enabling a detailed overview of the proposed approach in terms of risk exposure and control;
6) Finally, we carried out a performance evaluation of our approach, showing that it outperforms several strong

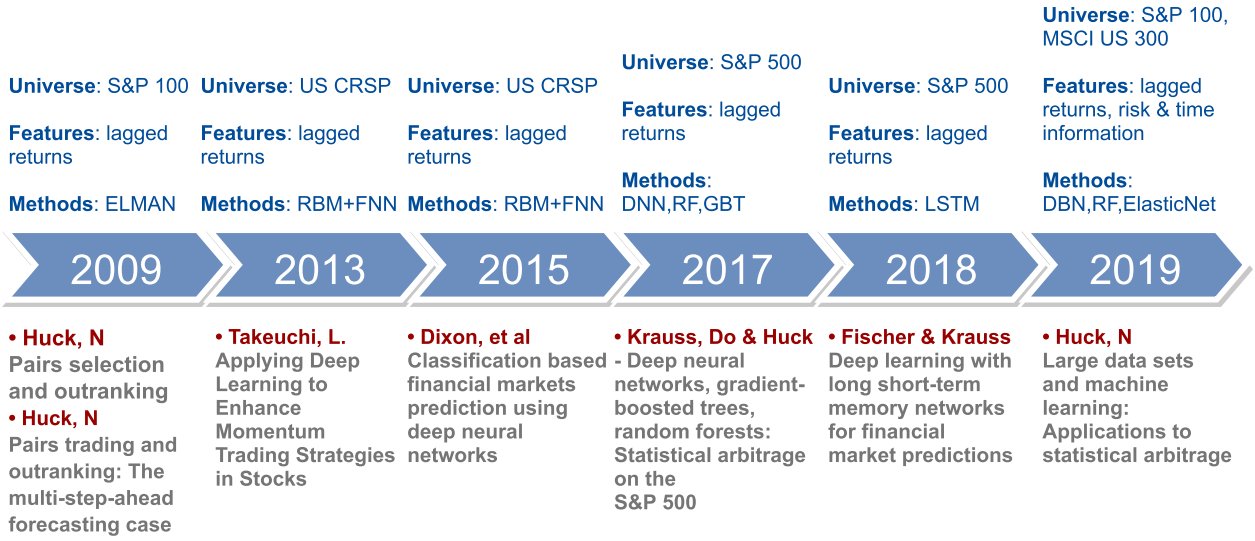

FIGURE 1. A road map of the forecasting methods and asset universes in StatArb trading along the years (ELMAN - ELMAN Recurrent Neural Network, RBM - Restricted Boltzman Machines, FNN - Feedforward Network, DNN - Deep Neural Network, RF - Random Forests, GBT - Gradient Boosting Trees, LSTM - Long Short Term Memory, DBN - Deep Belief Network).

baseline methods on the S&P500 index for intra-day trading.

The remaining of this paper is organized as follows. Section II briefly describes relevant related work in the literature. Section III introduces the problem we are aiming to tackle, the general architecture of the proposed approach, and details about the instance we have generated. We continue by describing the features that we have used (Section IV), details of the regressors (Section V), the proposed ensembling methodology, and finally the dynamic asset selection (Section VI). Section VII discusses the setup considered for the trading back-test, while Section VIII presents the experiments we have carried out. Section IX concludes the paper and discusses further directions of research where we are headed.

II. RELATED WORK

The prediction of the stock market is one of the most challenging problems in time series forecasting research and, hence, the interest in performing efficiently such a task from the academic finance community is growing. The literature dealing with applications on machine learning and neural networks in finance is presented and analyzed in several works [10], [21]–[23]. Although there are various streams of research and applications, this section highlights only a limited number of articles, highly correlated to this paper and by extension to StatArb, as it will be discussed in the next paragraphs. Figure 1 depicts the road map for the techniques behind StatArb algorithmic trading and corresponding input features and the universe (set of assets) considered.

Starting chronologically, the first work that considers coupling machine learning with StatArb is [24]. Here the author proposes a StatArb system that entails three phases: forecasting, ranking, and trading. For the forecasting phase, the author uses an Elman recurrent neural network to perform weekly predictions and anticipate return spreads between any two securities in the portfolio. Next, a multi-criteria decision-making method is considered to outrank stocks based on their weekly predictions. Lastly, trading signals are generated for top k and bottom k stocks, considering a variable k. This work is later extended in [25], where the approach is evolved by introducing a multi-step-ahead forecast. Both studies consider constituents of the S&P100 Index on a period spanning from 1992 to 2006. Although these approaches also consider regression, they lack scalability as their application is limited to 100 stocks, and in the case of broader indexes such as S&P500 or Russell 1000, would become computationally intractable.

In [18], the authors use deep neural networks for classification and as features cumulative returns. The approach computes the probability that one stock outperforms the cross-sectional median return of all stocks in the holding month. Next, all stocks are ranked according to the forecasted probability, and then, the trading signals are constructed based on the top decile of predictions, which are bought long, and flop decile, which are sold short. The stock universe used is the U.S. CRSP and the study period spans from 1965 until 2009. The works in [17], [26] adopt a similar strategy. The former in a high-frequency setting with five-minutes binned return data. In the latter, the authors construct similarly a classification problem using cumulative returns as input features and employ models like deep neural networks, random forests, gradient boosted trees, and three of their ensembles. They validate their study using *S&P*500 Index constituents on a period ranging from 1992 to 2015, with the trading frequency of one day. The ensemble proposed in this work uses the set of input features for all the models whereas in our work we consider diversified features set. Later, the authors extend

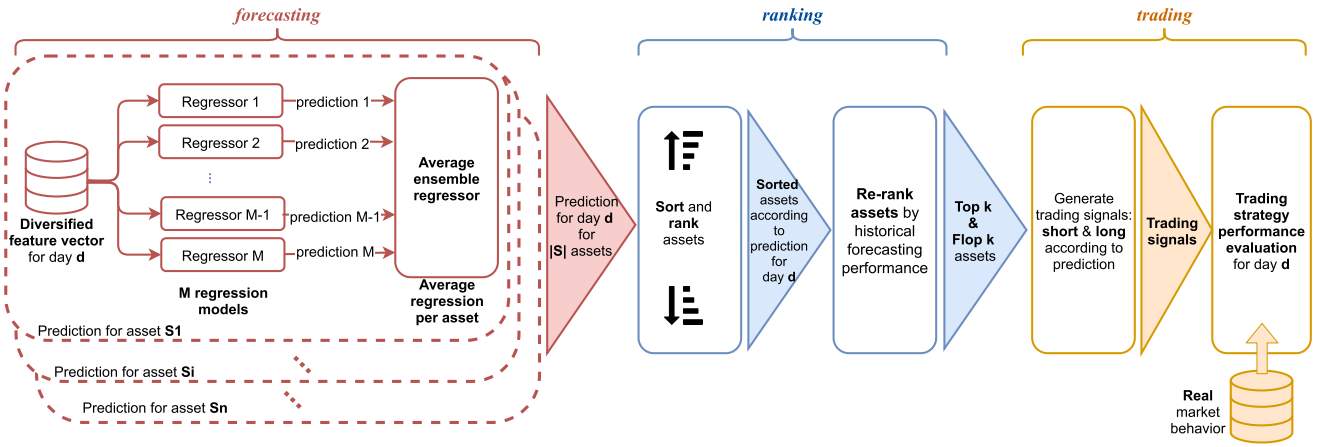

FIGURE 2. Architecture of the proposed general approach for risk controlled trading.

their work in [27] by using a Long Short-Term Memory network for the same prediction task. This enhanced approach outperforms memory-free classification methods.

A different approach in terms of features, in the context of StatArb, is presented in [28]. The author uses 4 types of one-hot encoded features on a period from January of 1993 to June 2015, with two forecasting/holding periods (*e.g.* one and five days). The used machine learning algorithms are Random Forests, Elastic Net, and Deep belief networks in a classification setup. The most salient finding in this work is that increasing the number of features does not translate into increased performance. One difference concerning this work is that we employ an ensemble strategy to mitigate the results of all the used models.

In [29], the authors take a different approach for predicting returns of S&P500, where the used features are stock tweets information. The aim is to unveil how the textual data reflect in stocks' future returns. For this goal, they use factorization machines and support vector machines. The proposed system performs prediction in a 20 minutes frequency over a two years period: from January 2014 to December 2015. The selection of flop and top stocks is made at the formation period based on the algorithm's performance evaluation (*i.e.* lowest root relative squared error) and trading signals are generated based on Bollinger bands.

Thus far we can state that the actual research on machine learning with application to StatArb has the following limitations we aim to tackle in this paper: (i) ranking based on intra-day price forecasting of single stocks; (ii) use of diversified features and data, with further distribution of them to different models of an ensemble; and (iii) definition of more efficient ranking technique to avoid trading bad stocks. Moreover, the proposed approach can be regarded as general and it can be instantiated with a different configuration: number and types of regressors, transaction frequency (*e.g.* intra-day), selected features (*e.g.* lagged returns, technical indicators), number of assets to buy or sell (choice for k).

III. METHODOLOGY

A. PROBLEM STATEMENT

Using StatArb as a trading strategy, our general approach aims at solving a risk-controlled trading problem. In this regard, it leverages machine learning to identify possible sources of profit and balance risk at the same time. At a very high level, our approach takes various assets' time series as an input and outputs the subset of assets to invest in, together with the appropriate trading signals, *i.e.* long or short. Our approach follows these three steps:

i *forecasting* - we tackle StatArb as a regression problem, investigating the potential of forecasting price returns for each of the assets in a selected assets collection, on a target trading day.

ii *ranking* - based on the anticipated price returns for the assets, we rank them in descending order. Next, we *balance the risk* due to inaccurate predictions by pruning the "bad" assets based on their past behavior, thus the *dynamical asset selection* yields a reorganized ranking of the assets.

iii *trading* - in the last step we issue trading signals for the top k and flop k assets from the ranked set of assets obtained in the previous step.

B. OVERVIEW

In Figure 2 we present the architecture for such a risk-controlled trading approach. Let s_i be an individual asset in the asset collection S. We start by collecting historical raw financial information for each asset s_i in the selected asset collection S. To note that we are using information available before the trading day d. Using the historical data, we generate the diversified feature set denoted by $\mathcal{F}_d^{s_i}$, that is used as input to each regressor m in our regressors pool $\mathcal{M}$.

In the *forecasting* step, each trained model m makes its prediction, $o_d^{s_i,m}$ for day d and asset s_i. Then, their results are averaged by a given ensemble method, to obtain a final prediction output denoted by $o_d^{s_i,ENS}$.

In the *ranking* step, we *sort assets* in descending order based on the forecast output $o_d^{s_i,ENS}$. This means that we will find at the top assets whose prices are expected to increase, and at the bottom assets whose prices are expected to drop. Assets at the top and at the bottom of our sorted list represent the most suitable candidates for trading. After the ranking is performed, we introduce the *dynamic asset selection* step: from this pool of assets, we discard those that do not satisfy a prediction accuracy higher than a given threshold ε in a past trading period, rearranging the ranking accordingly.

The *trading* step consists of selecting the top k (winners) and flop k (losers) assets and issue the corresponding trading signals: k long signals for the top k stocks and k short signals for the bottom k stocks. These selections are repeated for every day d in the trading period.

Finally, we evaluate the performance of our architecture by using a back-testing strategy [5].

As stated throughout this paper, we tackle StatArb as a regression problem, investigating the potential of forecasting intra-day stock price returns. We choose such a feature as it allows refined trading, as opposed to classification methods used in the previously mentioned work. Moreover, by clearly separating the forecasting phase from the ranking and trading signal generation, we aim to make the system-agnostic of any alterations that can happen over time to the set of stocks and, hence, tackle the survivorship[1] and data snooping biases.[2] Additionally, in this study, we consider an ensemble of forecasters with a high degree of diversity provided by the input features and dissimilar prediction models, each of them capturing different structures in the financial market data. The set of features and the regressors will be described in Section IV and Section V, respectively.

IV. FEATURE ENGINEERING

As mentioned in the introduction, we have instantiated one example out of our general framework by using the S&P500 index as a reference dataset. By reference to Figure 1, only two studies [17], [27] focus on such a large pool of companies. As stated by the authors in [17], the S&P500 constituents represent the leading 500 companies in the U.S. stock market, and at the same time, the large-capitalization segment. These characteristics make these companies highly attractive to investors, and as a consequence, they are very challenging for any trading strategy.

For each stock, we collect 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 (OHCLV variables). Based on this information, we have created two different types of features:

Lagged intra-day price returns (LR) - for a stock s_i and a trading day d, in a given time-frame Δ, we compute the intra-day $LR_{d-\Delta}^{s_i}$ expressed as:

$$LR_{d-\Delta}^{s_i} = \frac{closePrice_{d-\Delta}^{s_i} - openPrice_{d-\Delta}^{s_i}}{openPrice_{d-\Delta}^{s_i}}. \quad (1)$$

We considered $\Delta \in \{1, \ldots, 10\}$, thus yielding for each trading day d, a series of 10 historical price returns as it follows:

$$\mathbf{LR}_d^{s_i} = [LR_{d-10}^{s_i}, LR_{d-9}^{s_i}, LR_{d-8}^{s_i}, LR_{d-7}^{s_i}, LR_{d-6}^{s_i}, LR_{d-5}^{s_i}, \times LR_{d-4}^{s_i}, LR_{d-3}^{s_i}, LR_{d-2}^{s_i}, LR_{d-1}^{s_i}] \quad (2)$$

By using the lagged returns in the nearest past, we input each of the prediction models with the information about asset behavior in the last days. Glancing at Figure 1, historical price returns are the preferred choice in financial studies. With respect to previous works, we chose to use the intra-day price returns. This decision was motivated by recent studies [32] where the authors decompose the returns in overnight returns and intra-day price returns. They point out that after-hours trading happens much more seldom than trading while markets are open. Moreover, the pre-open auctions on the U.S. stock exchanges only average one to four percent of median daily volume, as pointed out by the same authors. Also, trading in the first half-hour of the day (the interval in which we measure the open price) is significantly less than the volume one observes intra-day, particularly near or at the close.

Technical Indicators (TI): We use a set of technical indicators summarized in Table 1. For this second type of feature we have built the following vector:

$$\mathbf{TI}_d^{s_i} = [EMA(10), \%K, ROC, RSI, AccDO, MACD, \%R, \times Disp(5), Disp(10)] \quad (3)$$

The first technical indicator, **EMA(10)**, is a type of moving average that places a greater weight and significance on the most recent data samples. We use this type of indicator as we are predicting the short-term future. **%K** and **Williams %R** are stochastic oscillators, both being trend indicators for any asset. When stochastic oscillators are increasing, the asset prices are likely to go up and vice-a-versa. **ROC**, **Disp (5)**, and **Disp (10)** are momentum-based technical indicators. **ROC** measures the percentage change in price between the current price and the price a certain number of periods ago. ROC is used to spot divergences, overbought and oversold conditions. **Disp (5)** and **Disp (10)** have a similar interpretation, i.e., a value greater than zero suggests that the asset is gaining upward momentum, whereas, a negative value is a sign that selling pressure is increasing, forcing the price to drop. **RSI**, ranges between 0 and 100 and is generally used for identifying the overbought and oversold assets. That is, if **RSI** exceeds the level of 70, it is an indication that the asset is overbought, so, the asset's price may go down in near future, whereas values below 30 level indicate that the asset is oversold, so, the asset's price may go up in near future. **AccDO** provides insight into how strong a trend is by using both price and volume information. If the price is rising

[1]Survivorship bias refers to the tendency to exclude from performance studies the failed companies because they no longer exist [30].

[2]Data snooping refers to the use of data mining to uncover misleading relationships in data [31].

TABLE 1. Selected technical indicators and their acronyms throughout this paper.

Name of technical indicator	Formula
Exponential Moving Average ($EMA(10)$)	$(C_t \times a) + (EMA_{t-1} \times (1-a))$ where $a = 2/(n+1)$
Stochastic %K (%K)	$\%K = \frac{(C_t - LLt-n)}{(HH_{t-n} - LL_{t-n})} \times 100$
Price rate of change (ROC)	$\frac{C_t - C_{t-n}}{C_{t-n}} \times 100$
Relative Strength Index (RSI)	$100 - \frac{100}{1+(U/T_n)}$
Accumulation Distribution Oscillator (AccDO)	$\frac{(C_t - LL_{t-n}) - (HH_{t-n} - C_t)}{HH_{t-n} - LLt-n} \times V$
Moving Average Convergence - Divergence (MACD)	$EMA_{12}(t) - EMA_{26}(t)$
Williams %R	$\frac{HH_{t-n} - C_t}{HH_{t-n} - LL_{t-n}} \times 100$
Disparity 5 (Disp (5))	$\frac{C_t}{MA_5} \times 100$
Disparity 10 (Disp (10))	$\frac{C_t}{MA_{10}} \times 100$

C_t is the closing price at time t, L_t the low price at time t, H_t high price at time t, LL_{t-n} lowest low in the last $t-n$ days, HH_{t-n} highest high in the last $t-n$ days, MA_t the simple moving average of t days, U represents the total gain in the last n days and T_n represents the total loss in last n days. In this paper n has been set to 10.

but the indicator is falling this indicates that accumulation volume may not be enough to support the price rise, thus, the price might drop in the near future. **MACD** follows the trend of the asset, i.e., if its value goes up then asset price also goes up and vice-versa.[3]

Technical analysis and by extension technical indicators are valuable statistical tools through which investors extensively use to make their investment decisions. This fact partially motivated the reasoning behind our choice for this second type of features. Moreover, we are interested in predicting the price movement range and also its direction and each of the technical indicators has its own inherent opinion about the asset price movement. When we give these data as inputs to the model, we are already inputting trend information as perceived by each of the individual technical indicators. Several other financial studies use technical indicators in conjunction with machine learning algorithms to predict individual stock direction movement [33], [34] or closing price [35].

For regressors' training, we also generate the associated *target value* (label), y_d^{si} which is the intra-day price return for the current day.

$$y_d^{si} = \frac{closePrice_d - openPrice_d}{openPrice_d}. \tag{4}$$

V. FORECASTING ALGORITHMS

In the proposed instance of our general approach, we considered the following three different state-of-the-art machine learning models, and the widely known statistical model, ARIMA. We based our choice to employ such models on the following criteria:

1) robustness to noisy data and over-fitting;
2) diversity amongst models in the final ensemble;
3) computational efficiency;
4) adoption of such models in the scientific community for similar tasks.

A. LIGHT GRADIENT BOOSTING

Light Gradient Boosting (LGB) first proposed by Ke, *et al.* [36], is another novel gradient boosting framework, which has been widely applied in machine learning tasks and supports efficient parallel training. LGB applies iteratively weak learners (decision trees) to re-weighted versions of the training data [37]. 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 on examples with higher loss in previous steps. The LGB algorithm integrates two cutting-edge techniques, respectively referring to the gradient-based one-side and the exclusive feature bundling methods. These two techniques reduce the data size by rows and by columns, respectively, which makes the algorithm less computationally expensive and, at the same time, maintains the accuracy. Comparing to the traditional gradient boosting techniques, LGB would grow 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), which makes an effective method for LGB in processing large-scale and high-dimensional data, on the one hand, but more prone to over-fitting, on the other hand. To control this behavior we defined the maximum depth levels of the tree, *max_depth*, to 8. We chose to vary the *num_leaves* parameter in the set [70, 80, 100], achieving a balance between a conservative model and a good generalization. The feature selection is

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

restricted by a parameter *colsample_by_tree* set at 0.8 of the total number of features, which can be thought of as a regularization parameter. The work in [37] suggests a learning rate lower than 0.1, so we set it to 0.01 to account for a better generalization over the data set. Its adoption in the scientific community for financial forecasting is scarce [38], but by contrast, we can find this algorithm in the leading positions on machine learning competitions platforms.[4]

B. RANDOM FORESTS

Random Forests (RF) belong to a category of ensemble learning algorithms introduced in [39]. 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 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. But, 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. We opt to vary the number of trees by ranging *n_estimators* from 50 to 500 with a 25 increment, similarly to [28]. Random feature selection operations substantially reduce trees' bias, thus we set *min_samples_leaf* to 3 of the total number of features in a leaf. The learning rate is set to 0.01. In the academic literature corpus, we can find that RF has been extensively used in financial forecasting [35] and moreover in StatArb applications [17], [24].

C. SUPPORT VECTOR REGRESSORS

Support Vector Regressors (SVR) were proposed initially as supervised learning model in classification, and later revised for regression in [40]. Given a finite d-dimensional set of training data $x_i \in \mathbf{R}^d$ for $i \in \{1 \dots m\}$ of length m, the goal is to find a function that deviates from actual data, $y_i \in \mathbf{R}$ for $i \in \{1 \dots m\}$, by a value no greater than ϵ for each training point, and at the same time is as flat as possible. SVR extends least-square regression by considering an ϵ-insensitive loss function. Further, to avoid over-fitting of the training data, the concept of regularization is usually applied. SVR is trained by solving the following optimization problem:

$$\min \frac{1}{2}\|w\|^2 + \frac{C}{m}\sum_{i=1}^{m} \underbrace{\max(0, |y_i - f(x_i; w)| - \epsilon)}_{\epsilon \text{ - insensitive loss function}}, \quad (5)$$

where C is the regularization constant. The training finds the weights w so that the function f minimizes empirical risk. Also, SVR can map the input vectors x_i into a high dimensional feature space using kernel functions. Furthermore, it can be applied to nonlinear regression problems by employing such kernel functions. In this study, we selected the radial basis kernel function which can be formalized as $k(x_i, x_j) = exp(-\gamma\|x_i - x_j\|^2)$. Here x_i and x_j represent two feature vectors of the input space and γ is a free parameter, considered as a design parameter of SVR.

SVR, thus, solves an optimization problem that involves two parameters: the regularization parameter, C, and the error sensitivity parameter ϵ. C controls the trade-off between model complexity and the number of non-separable samples. A lower C will encourage a larger margin, whereas higher C values lead to a hard margin [40]. Thus, we set our search space in $\{8, 10, 12\}$. Parameter ϵ controls the width of the ϵ-insensitive zone and is used to fit the training data. A too high value leads to flat estimates, whereas a too-small value is not appropriate for large or noisy data-sets. Therefore, we set it to 0.1. The work in [41] suggests that the γ value of the kernel function should vary together with C, and higher values of C require higher values for γ too. Therefore, we set a smaller search space in $\{0.01, 0.5\}$. Authors in [34], [35] used successfully SVR in financial forecasting tasks.

D. AUTO-REGRESSIVE INTEGRATED MOVING AVERAGE

The Auto-Regressive Integrated Moving Average (ARIMA) was first introduced in [42] and, since then, it has proven to be robust and efficient for short-term prediction when employed to model economical and financial time series [43], [44]. The algorithm captures a suite of different time-dependent structures in time series. As its acronym indicates $ARIMA(p, d, q)$ comprises three parts: *Auto-Regression model* that uses the dependencies between an observation and a number of lagged observations (p); *Integration differencing* of observations with a different degree, to make the time series stationary; and *Moving Average model* that accounts the dependency between observations and the residual error terms when a moving average model is used to the lagged observations (q). Mathematically, ARIMA can be expressed as:

$$(1 - \sum_{i=1}^{p} \phi_i L^i)(1 - L^i)^d r_t = \delta + (1 - \sum_{i=1}^{q} \theta_i L^i)\varepsilon_t \quad (6)$$

where ϕ_i are coefficients for the auto-regressive part of the model, L is the lag operator, θ_i are coefficients of the moving average part of the model, and ε_t is the error term at time t, and finally, δ denotes the intercept.

We chose the lag order $p \in \{1, 5\}$, the degree of differencing $d \in \{1, 5\}$, the size of the moving average window $q \in \{0, 5\}$.

VI. MODEL TRAINING, ENSEMBLING, AND DYNAMIC ASSET SELECTION

A. MODEL TRAINING

In standard statistics, fitting a simple model on data can be done without the sensitive choice of a hyper-parameter and the model's parameters are estimated from the data, *e.g.* using the maximum-likelihood criterion. However, in high-dimensional settings, some form of regularization is

[4] https://www.kaggle.com/

needed. Choosing the amount of regularization is a typical bias-variance problem. But, in general, the best trade-off is a data-specific choice, governed by the statistical power of the prediction task, or differently put, it is governed by the amount of data and the signal-to-noise ratio. In this context, given that each of the assets exhibits different behavior in time and incurs a different amount of noise, therefore the same set of hyper-parameters do not apply to all the stocks. As such, we have embedded a hyper-parameter tuning in the model training phase.

In addition to that, for each of our machine learning models we have two variants of *data input*:

1) Data specific to each stock s_i in our stocks pool S, resulting in a model for each stock.
2) Data resulting after aggregating information from stocks pertaining to the same industry sector as given by the Global Industry Classification Standard (GICS), thus resulting in a model for each industry. To be noted that industries contain a different number of stocks, but their content does not overlap. That is, one stock cannot be part of multiple industries at the same time.

This last variant of the model was encouraged by previous work [7], where some portfolios were restricted to only include stocks from the same industry. In the spirit of the aphorism ''a rising tide lifts all boats'', companies in the same industry tend to have similar behavior, and the returns of stocks tend to follow each other [5].

As such, we are faced with the problem of selecting for each of the stocks and each type of algorithm (*i.e.* LGB, RF, and SVM) the best combination of hyperparameters as well as input data (*i.e.* per stock or per industry) or feature types (*i.e.* LR or TI) in order to obtain the model with the best predictive power for that specific stock. Standard machine learning practice advocates to measure predictive power using cross-validation [37]. This means that the available data is split into a development set, used to train the model, and a validation set, unseen by the model during training and used to compute a prediction error. By using such a scheme, one expects to obtain a better estimate of the model's out-of-sample predictive performance. To properly account for the time dependence of observations, empirical research based on traditional time series models typically reverts to validation schemes that keep the temporal order of observations between training and validation sets. However, our goal to select the best hyperparameters and best features in the same step is slightly different from standard practice. As such, we modified the machine learning algorithms selection as presented in Algorithm 1.

Our algorithm receives as input: (i) the set of stocks, S_I, with their associated financial historical data, (ii) the chosen machine learning algorithm a to be trained, (LGB, RF, or SVM) and its corresponding set of hyperparameters. The output is represented by a set of trained regressors, $\mathcal{R}$, each corresponding to each stock in S_I. The algorithm starts by acquiring historical raw financial data (*ohclv*) for each of the stocks, for the entire training period

Algorithm 1 Model Selection

Input: S_I, set of stocks in industry I
Input: $\mathcal{D}^{train}$, training dates (calendar days)
Input: a,$\mathcal{P}_{set}$, set of hyperparameters for the machine learning algorithm a
Output: Pool of regressors $\mathcal{R} = \{r_1 \dots r_{|S_I|}\}$

1: **for each** $s_i in S_I$ **do**
2: $\quad \mathbf{LR}^{s_i} \leftarrow$ computeLR(*ohclv*) using $ohclv \in \mathcal{D}^{train}$
3: $\quad \mathbf{TI}^{s_i} \leftarrow$ computeTI(*ohclv*) using $ohclv \in \mathcal{D}^{train}$
4: $\quad \mathbf{LR}^{I} \leftarrow$ append($\mathbf{LR}^{s_i}$)
5: $\quad \mathbf{TI}^{I} \leftarrow$ append($\mathbf{TI}^{s_i}$)
6: Split $\mathcal{D}^{train}$ into $\mathcal{D}^{dev}$,$\mathcal{D}^{val}$
7: **for each** $s_i in S_I$ **do**
8: $\quad$ **for each** $f in \{\mathbf{LR}^{s_i}, \mathbf{TI}^{s_i}\}$ **do** using data in $\mathcal{D}^{dev}$
9: $\quad\quad$ Select optimal hyperparameter set p^* from $\mathcal{P}_{set}$ using inner cross-validation
10: $\quad\quad m_f^{s_i} \leftarrow a.train(p^*, f)$
11: $\quad\quad$ Save $m_{I,f}$
12: **for each** $f in \{\mathbf{LR}^{I}, \mathbf{TI}^{I}\}$ **do** using data in $\mathcal{D}^{dev}$
13: $\quad$ Select optimal hyperparameter set p^* from $\mathcal{P}_{set}$ using inner cross-validation
14: $\quad m_f^{I} \leftarrow a.train(p^*, f)$
15: $\quad$ Save m_f^I
16: $\mathcal{R} \leftarrow [\,]$
17: **for each** $s_i in S_I$ **do**
18: $\quad$ **for each** $m \in \{m_{LR}^{s_i}, m_{TI}^{s_i}, m_{LR}^{I}, m_{TI}^{I}\}$ **do**
19: $\quad\quad$ Compute error $\mathcal{L}_{s_i,m}^{val}$ for model m using $\mathcal{D}^{val}$
20: $\quad \mathcal{R} \leftarrow \mathcal{R} \cup \{$model with lowest $\mathcal{L}_{s_i,m}^{val}\}$
21: **return** $\mathcal{R}$

(lines 1-5). Financial information will be timestamped and we keep the temporal order of observations and any operation on data is performed using the timestamps. We split the historical dataset into two portions: development and validation sets (line 6), where the former is used for model training and the latter for model selection. Then, it proceeds with the stock level model training line 7. For each type of feature, we select the optimal hyperparameter constellation by employing an inner-cross-validation. The same process is applied at the industry level (line 12). Therefore, to forecast the return of each stock, we created 4 models: 2 models (per asset) using TI and LR, that use data of a single stock; and 2 models (per industry) using TI or LR as features, that use data of all assets associated to that industry, and, in turn, forecast one asset at a time. The second phase consists of choosing among the 4 models trained in the previous step, *i.e.* the one with the highest predictive power. Hence, using the validation set, we compute for each stock and each model the loss $\mathcal{L}_{s_i,m}^{val}$, which, in our case, is the mean squared error (MSE) between the forecast and the ground truth. Then, we choose the best model out of the four according to the lowest MSE.

ARIMA is a class of statistical models that captures temporal structures in time series data. Its methodology of

training differs from machine learning algorithms in the sense that ARIMA is not designed for multivariate input and as a consequence for ARIMA we use the series of lagged returns. Furthermore, ARIMA is designed for forecasting one-step out-of-sample forecast. Hence the methodology that we developed performs a one-step out-of-sample forecast with re-estimation, *i.e.*, each time the model is re-fitted to build the best estimation model.

In conclusion, for each stock, we train 3 types of machine learning algorithms (LGB, RF, and SVM) each of them for 2 levels of data (stock level and industry level), and 2 types of features (*i.e.* LR and TI). This yields: 3 types of algorithms $\times$ 2 data level $\times$ 2 types of features $=$ 12 models for each stock. Algorithm 1 selects the best model per machine learning algorithm type, resulting in 3 models. To this pool of regressors, for each stock, we add the ARIMA model, which is most suitably used with univariate time series (equivalent to stock level and lagged returns features).

B. ENSEMBLING

The final ensemble entails for each stock s_i three machine learning models (*i.e.* LGB, RF, SVR as given by our model selection/cross-validation phase) and a statistical model ARIMA. Their forecasted output is averaged. The benefit of this approach is that if the errors of each model are sufficiently independent, they average out: the average model performs better and displays much less variance [37], [45]. Several works in literature validated the power of averaging decisions. In [46], [47] it is revealed, by empirical evaluation, that averaging ensembles performs well in practice, especially when compared to ensembles that use more complex weighting strategies. Moreover, according to [48], simple ensembles of forecasters such as averaging decisions outperform sophisticated combination methods in empirical applications. Finally, in [17], the authors show that, for the same purpose, a simple weighted ensemble performs better when compared to performance-based or rank-based ensembles. The previous works justify our choice of model output averaging. As such, the outputs of each of the selected models, *i.e.*, LGB, RF, SVR, and ARIMA are averaged to the final output, $o_d^{s_i,ENS}$.

C. RANKING AND DYNAMIC ASSET SELECTION

The final step in our trading framework comprises ranking the assets by their predicted return to be able to issue the appropriate trading signals. Additionally, to mitigate the risk of bad forecasting performance, we propose a stock pruning mechanism by performing a *dynamic asset selection* strategy. For a stock $s_i \in S$, given its past forecastings $o_t^{s_i,ENS}$, and also its past real values $y_d^{s_i}$ in a predefined look-back period T, we compute a modified version of the mean directional accuracy [49], [50] as follows:

$$MDA_{s_i,T,d} = \frac{1}{T}\sum_{t=d-1}^{d-T-1} \mathbf{1}_{sgn(o_t^{s_i,ENS})==sgn(y_t^{s_i})}, \tag{7}$$

where d is the current trading day, T is the look-back length, and $\mathbf{1}_P$ is the indicator function that converts any logical proposition P into a number that is 1 if the proposition is satisfied, and 0 otherwise, $sgn(\cdot)$ is the sign function. In other words, $MDA_{s_i,T,d}$ compares the forecasted direction (upward or downward) with the realized direction of the return and yields the probability that the forecasting model can detect the correct direction of returns for a stock s_i on a given timespan T prior to the day d. We use such a component on a limited look-back period as a consequence of proven studies [51] that stocks exhibit behavior with short periods of significant returns predictability ('pockets'). These periods are interspersed with long periods with little or no evidence of return predictability.

Such a component introduces a new step in the StatArb pipeline: after performing the forecast, we rank the companies by their forecasted daily price returns. From this pool of stocks, we discard those that do not satisfy a prediction accuracy higher than a given threshold ε in a past trading period, rearranging the ranking accordingly.

VII. EXPERIMENTAL SETUP

In this section, we highlight the parameters of the proposed approach for reproducibility purposes. We follow along the main steps of our trading framework and present the backtesting methodology, model training and forecasting technical aspects, ranking and dynamic asset selection, trading execution, and finally we discuss the used baselines.

A. BACKTESTING

As mentioned in the introduction we have instantiated one example out of our general approach by using as pool of assets the stocks within the S&P500 Index [17], [27]. We back-tested the framework by choosing as study period a timeline starting from March 2003 to January 2016. The back-testing experiments consist in running the signals through historical data together with the estimation of forecasting hyper-parameters, signal evaluations and portfolio rebalancing [5]. We use a common approach for validating time-series data in finance, namely the *walk-forward* validation, which consists of splitting the study period into overlapping training periods and non-overlapping test (trading) periods, as shown in Figure 3. The example depicts values for the closing price of SPY (asset) over the whole study period. In this example we considered four years of training and a year of test. The period of training is further split into three years of development for individual regressors training and one year of validation for model selection, as presented in Section VI. Running the experiments under the same setup yields 9 walks.

B. MODEL TRAINING

For model training and parameter tuning, we used a 10-fold *TimeSeriesSplit* combined with a *GridSearch*, both using scikit-learn implementations [52]. In this experimental scenario, the temporal linking between the observation at day

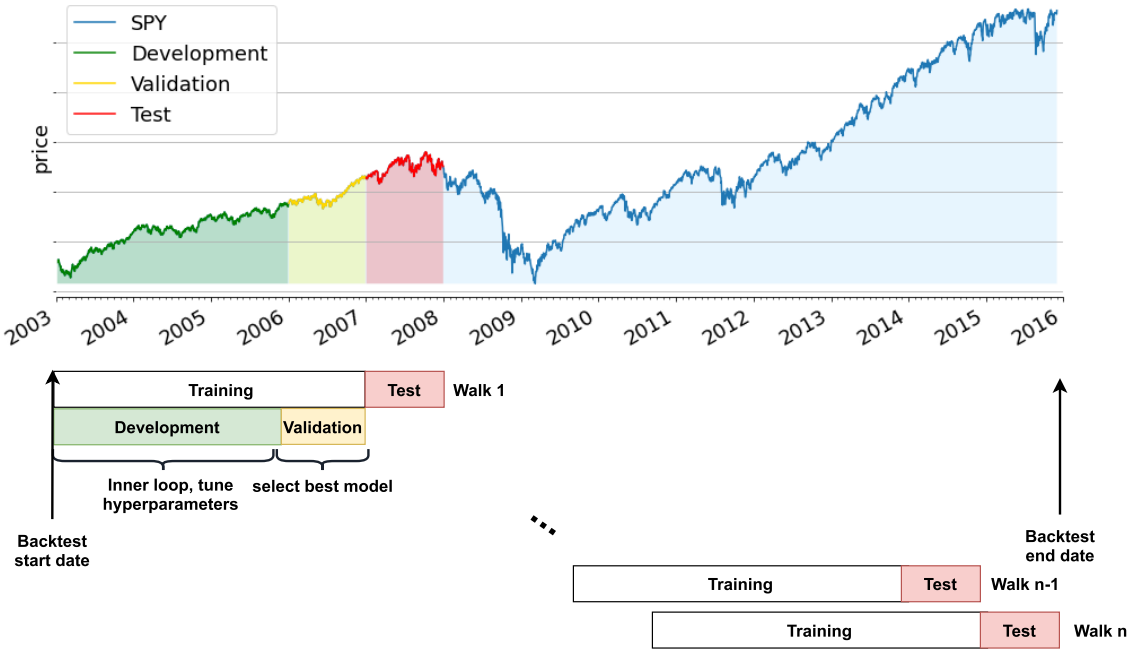

FIGURE 3. Illustration of walk-forward procedure, considering a study period starting from January 2003 to January 2016. On the y-axis it is presented the development of the closing price for SPY (asset) across time and how this overlaps, in time, to each walk and each development,validation and test period, respectively.

$d-1$ and d is taken into account to compose the same bunch of training and validation sets. This is different with respect to common machine-learning cross-validation approaches like the Leave-One-Out cross validation or the k-fold cross validation, where data are randomly sampled in different folds, no matter when they were acquired. Such an approach is quite biased when applied to time series forecasting, as features from late past and early future are mixed in the same fold of data.

C. RANKING AND DYNAMIC ASSET SELECTION

As pointed out in Section VI-C, the dynamic asset selection requires a series of parameters: the accuracy threshold ε, and rolling window length, T. The threshold value is set to $\varepsilon = 0.5$ and we considered different lengths of the rolling window $T = \{30, 40, 60\}$. We made these choices based on findings in [53] where the authors noticed that *MDA* can efficiently capture the inter-dependence between asset returns and their volatility (hence forecast-ability) when using intermediate return horizons, e.g. two months. The threshold value has been set to $\varepsilon = 0.5$ as advised in [24] for a similar scenario.

D. TRADING EXECUTION AND PORTFOLIO CONSTRUCTION

As stated throughout the paper each day we perform $2 \times k$ operations, k long and k short operations. We fixed the number of pairs to be traded to $k = 5$, based on the findings in similar works [17], [27] where higher k values lead to a decrease in portfolio performance both in terms of returns and risks. The trading session is set as intra-day, meaning that we are opening the positions at the beginning of the training day and close them at the end of the day. In other words, we are rebalancing our portfolio daily. As the authors in [29], we assume transaction costs of 0.4% daily.

E. BASELINES

To assess the value added by both our model selection strategy (**ENS**) and dynamic asset selection strategy (**ENS-DS**), they are benchmarked against two statistical arbitrage trading baselines: one based on cumulative five day return (**5-DAY**), and S&P500 buy-and-hold strategy (**Buy-and-hold**). These last two methods are well-established quantitative strategies, and largely used as baselines to evaluate the profitability of other investment approaches. They are described in the following:

1) **5-DAY** - Daily, we sort the set of stocks according to the 5-day cumulative return prior to the trading day, in ascending order. At the top we would find the stocks with the most negative cumulative returns and at the bottom stocks with the most positive 5-day cumulative return. We go long for top k and short for flop k stocks in the sorted list, building an equal weight portfolio. This approach has been considered in prior works such as [27]. Data, trading execution, and portfolio construction are the same as the other strategies (*i.e.* we open the trading position for $k = 5$ at the beginning of the day and close them at the end of the day and the portfolio is rebalanced daily).
2) **Buy&Hold** This strategy buys in 2007 and holds the S&P500 exchange-traded fund (SPY security) during

the whole backtesting period, *i.e.* until January 2016. This passive strategy runs without any trading signals. Such a baseline is widely recognized in literature as a valuable benchmark (e.g. [29], [54]).

F. IMPLEMENTATION DETAILS

The approach proposed in this paper has been developed in *Python*, by using the `scikit-learn` library [52] and the *LightGBM* python API [55]. The experiments have been executed on a desktop system with the following specifications: an Intel(R) Xeon(R) Gold 6136 CPU @ 3.00GHz, 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/stat-arb.

VIII. RESULTS

The results are presented from three perspectives which include: (i) predictive results of the individual regressors and their resulting ensemble under different training setups; (ii) return evaluation before and after transaction costs, (iii) exposure to common risk factors by analyzing risk metrics, and (iv) comparisons against state-of-the-art trading strategies.

A. MODEL SELECTION PERFORMANCE

First we compare the performances obtained by the models when using different walk-forward setups (for reference, see Figure 3). Specifically, we fixed the development period to 3 years and chose two different lengths (in days) for validation and test. For the first setup, we chose a period of study from January 2003 to January 2016, with 3 years of training, 1 year (252 days) of validation, and 1 year of test, which amounts to 9 walks. For the second setup, we chose a study period from July 2003 to January 2016, with 3 years of training and we reduce the validation and test periods to 6 months (126 days), thus resulting in 18 walks. The models corresponding to the two setups will be assessed in an overlapping trading/test period, *i.e.* March 2007 to January 2016. We report the models performances in terms of root mean square error (RMSE) between the realized return and forecasted return as average across companies and each walk. Also, we present, for each of the models, the portfolio performance in terms of annual returns (Return p.a.) and risk metrics (MaxDD), when using a strategy that trades $k = 5$ pair. The maximum drawdown (MaxDD) quantifies the maximum amount of wealth reduction that a cumulative return has produced from its maximum value over time. Figure 4 depicts the obtained results and Table 2 summarizes the performances of the models resulting after the model selection phase (for each stock) and their corresponding ensemble.

In terms of RMSE, Figure 4a shows a consistent decrease when the length of validation and test decreases. This can be attributed to the decreasing number of samples. Model SVR with data at industry level data and TI as features exhibits the worst performance in both setups. At the opposite

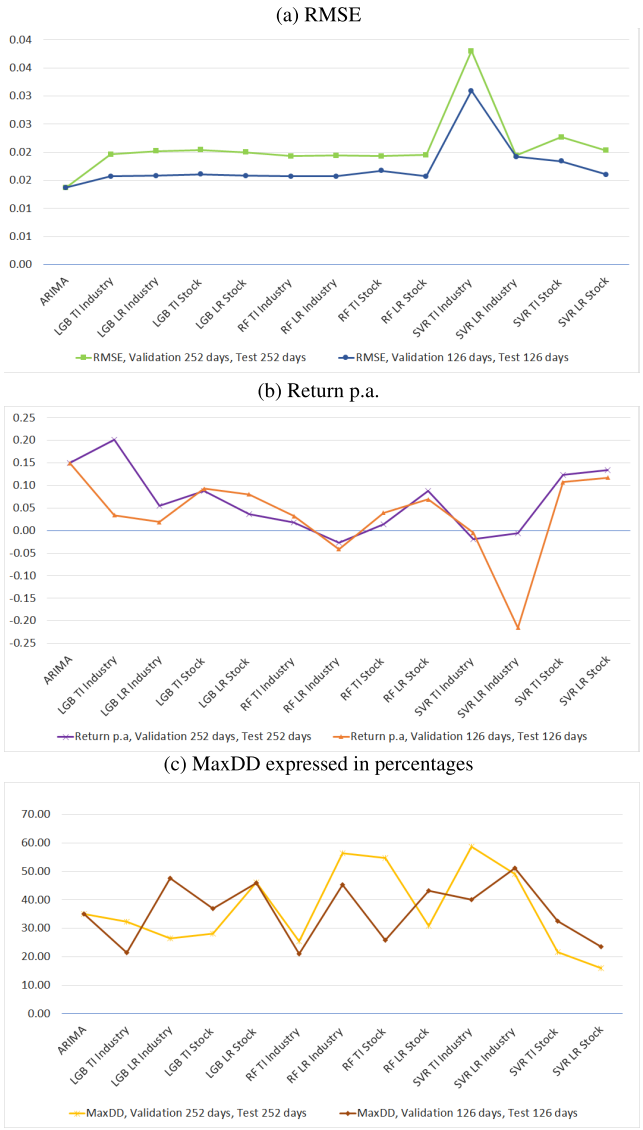

FIGURE 4. Model performances in terms of RMSE, return p.a., and MaxDD, developed from 2007 to 2016 when varying the validation and test period lengths.

pole we find ARIMA with the lowest RMSE. In terms of returns (Figure 4b) the best performing model is the LGB with data at industry level and TI as features in a year of validation and another year of test setup. The second best performing is ARIMA with annual returns close to 1.5%. The riskiest strategy (Figure 4c) is the SVR with data at industry level data and TI as features. Table 2 reveals for each forecasting algorithm the predictive performances and risk-return characteristics, respectively. With the reduction of validation length a deterioration in performance can be observed here as well, both in terms of return and MaxDD. The most surprising result it is represented by the small returns (0.0877) of the ensemble obtained by averaging models with a 6 month validation period. The ensemble is outperformed by most of the models. As a concluding remark, according to Table 2, we can state that the ENS with 1 year

TABLE 2. Models performances resulting after model selection phase and their corresponding ensemble (ENS). Each model is a combination of type of features and data level for each stock. The trading period is March 2007 and January 2016.

Method	Validation length and Test length	RMSE	Return p.a.	MaxDD[%]
ARIMA	-	0.0137	0.1493	35.04
RF	Validation 252 days, Test 252 days	0.0193	0.0761	23.23
LGB	Validation 252 days, Test 252 days	0.0195	0.1796	33.29
SVR	Validation 252 days, Test 252 days	0.0197	0.1835	28.90
ENS	Validation 252 days, Test 252 days	0.0190	0.3126	13.76
RF	Validation 126 days, Test 126 days	0.0157	0.0192	40.85
LGB	Validation 126 days, Test 126 days	0.0157	0.1109	28.55
SVR	Validation 126 days, Test 126 days	0.0159	0.1165	33.45
ENS	Validation 126 days, Test 126 days	0.0155	0.0877	31.78

TABLE 3. Return characteristics of StatArb strategies compared to the baselines (5-DAY and Buy&Hold) before transaction costs over a period between March 2007 to January 2016. Portfolio was constructed using $k = 5$ pairs. The best return performances are highlighted in bold.

	ARIMA	RF	LGB	SVR	ENS	ENS-DS $T = 30$	ENS-DS $T = 40$	ENS-DS $T = 60$	5-DAY	Buy & Hold
Return p.a	0.1493	0.0761	0.1796	0.1835	0.3126	0.3192	**0.3330**	0.3147	0.2155	0.0755
Excess return p.a.	0.1422	0.0695	0.1723	0.1762	0.3045	0.3111	**0.3247**	0.3066	0.2068	0.0680
Standard dev p.a.	0.2188	0.1776	0.1648	0.1380	0.1624	**0.1606**	0.1606	0.1606	0.2522	0.2227
Mean	0.0006	0.0004	0.0007	0.0007	0.0011	0.0012	**0.0012**	0.0011	0.0009	0.0004
Min	-0.0934	-0.0816	-0.0846	-0.0912	**-0.0754**	**-0.0754**	**-0.0754**	**-0.0754**	-0.1256	-0.0984
Q1	-0.0047	-0.0041	-0.0034	-0.0033	-0.0036	-0.0036	**-0.0034**	-0.0036	-0.0049	-0.0048
Median	0.0004	-0.0002	0.0005	0.0004	0.0006	0.0007	0.0007	0.0006	0.0003	**0.0008**
Q3	0.0057	0.0042	0.0046	0.0044	0.0050	0.0050	0.0049	0.0050	**0.0057**	0.0062
Max	0.1126	0.1387	0.1045	0.0577	0.1006	0.1006	0.1006	0.1006	**0.1830**	0.1452
Skewness	0.3613	1.4405	0.4219	-0.1524	0.8356	1.0659	1.1033	1.0121	**1.7274**	0.2227
Kurtoisis	9.4476	23.6159	17.1415	13.2800	15.1069	15.0316	14.9208	14.8477	**25.9812**	13.3642
Standard error	0.0003	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0003	0.0001
t-statistic	2.2093	1.4863	3.2157	3.8264	5.1979	5.3566	5.3385	5.2898	2.6648	1.9690

of validation and 1 year of test setup is the best performing model.

B. RETURN CHARACTERISTICS ASSESSMENT

As presented in Section VIII-A the best model performance is achieved when using the following setup: 3 years of development, 1 year for validation, period that serves the purpose for model selection and 1 year of test where we record the result and asses the proposed approach performance. Given these considerations we are focusing subsequent analyses on this setup.

Tables 3 and 4 report daily return characteristics from March 2007 to January 2016, before and after the transaction costs, respectively. As expected, the equal weighted ensemble, ENS, and the dynamic asset selection, ENS-DS, outperform both baselines and the underlying individual regressors. The annual returns are almost four times the level of the Buy&Hold and roughly two times the return of individual regressors, (*e.g.*, LGB). The same ratios are maintained even after the transactions costs. Furthermore, we observe a mean daily return of 0.11% for the ENS and 0.12% for ENS-DS ($T = 30, T = 40$) before the transaction costs, that is trice the market. After the transaction costs are applied, the returns deteriorate but are still considerably higher than the baselines. Analyzing the statistical moments, we observe that apart from SVR all the strategies expose a positive skewness, which indicates a longer tail for gains.

TABLE 4. Return characteristics of the StatArb strategies compared to the baselines (5-DAY and Buy&Hold) after transaction costs over a period between March 2007 to January 2016. Portfolio was constructed using $k = 5$ pairs. The best return performances are highlighted in bold.

	ARIMA	RF	LGB	SVR	ENS	ENS-DS $T = 30$	ENS-DS $T = 40$	ENS-DS $T = 60$	5-DAY	Buy & Hold
Return p.a	0.0569	0.0410	0.1032	0.0924	0.2198	0.2268	**0.2278**	0.2228	0.1579	0.0755
Excess return p.a.	0.0503	0.0346	0.0964	0.0856	0.2103	0.2192	**0.2202**	0.2153	0.1507	0.0681
Standard dev p.a.	0.1931	0.1775	0.1647	0.1381	0.1624	0.1606	**0.1606**	0.1607	0.2519	0.2227
Mean	0.0003	0.0002	0.0004	0.0004	0.0008	**0.0009**	**0.0009**	0.0008	0.0007	0.0004
Min	-0.0936	-0.0817	-0.0848	-0.0916	**-0.0756**	**-0.0756**	**-0.0756**	**-0.0756**	-0.1257	-0.0984
Q1	-0.0047	-0.0042	-0.0037	-0.0037	**-0.0038**	**-0.0038**	**-0.0038**	**-0.0038**	-0.0051	-0.0048
Median	0.0001	-0.0003	0.0002	0.0000	0.0005	0.0006	0.0005	0.0005	0.0001	**0.0008**
Q3	0.0052	0.0040	0.0044	0.0041	0.0049	0.0048	0.0049	0.0049	0.0055	**0.0062**
Max	0.0830	0.1385	0.1043	0.0575	0.1005	0.1005	0.1005	0.1005	**0.1827**	0.1452
Skewness	-0.0863	1.4347	0.4205	-0.1378	0.8461	1.0756	1.1136	1.0221	**1.7294**	0.2227
Kurtoisis	9.7357	23.6643	17.1581	13.2533	15.1040	15.0343	**15.5535**	14.8518	**25.5818**	13.3642
Standard error	0.0003	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0002	0.0003	0.0001
t-statistic	1.1363	0.9319	2.0127	2.1021	4.5401	4.6844	4.6711	4.6199	3.2354	1.9690

Last but not least, we assessed the statistical significance of returns by performing a Newey-West t-statistics with the null hypothesis that the mean return is equal to zero (the critical value at 5% significance level is 1.9600). The test resulted in the returns being statistically significant before and after transaction costs.

C. RISK EXPOSURE ASSESSMENT

Tables 5 and 6 offer an in depth analysis of risk incurred by the trading strategies, before and after transaction costs, respectively. We start by analyzing the tail risk. According to Value at Risk (Var 1%) before transaction costs (Table 5), ARIMA is the riskiest strategy of all, with -4.1% exceeding both baselines. At the opposite pole we find the ensemble strategies, with -2.6% for ENS, and approximately -2.3% for ENS-DS. After the transaction costs (Var 1%, Table 6) the same picture is maintained, but now, ARIMA (-3.80%) is almost on par with 5-DAY baseline (-3.78%). Compared to [17], that reports values between -5.9 and -6.9 percent after transaction costs, our strategies have significantly lower values. Under the conditional value at risk (CVar 1%, Table 5), positions have slightly changed, in the sense that ARIMA (-5.7%) incurs the highest risk out of regression models, but lower than 5-DAY strategy (-6.2%).

Sharpe ratio is a risk metric, defined as the excess return per unit of risk measured in standard deviations [56]. A Sharpe ratio greater than one usually signifies a portfolio of superior performance as opposed to a portfolio with a Sharpe ratio less than one. In general, a portfolio with a larger Sharpe ratio will outperform one with a smaller ratio. In Table 5, it can be noticed that Sharpe ratio started from 1.72 for the simple ensemble and turned into 1.85 for the proposed ENS-DS ($T = 40$), before transaction costs.

By the same token, another metric that measures the reward-to-risk ratio is Sortino Ratio which considers the risk expressed as downside deviations. By checking results in Tables 5 and 6, we can realize that downside deviations are less expressed for the proposed strategies. This naturally leads to a more favorable Sortino ratio: for ENS 2.89 before transaction costs and 2.11 after transaction costs, and for for ENS-DS strategies approximately 3 before transaction costs and approximately 2.2 after transaction costs. This amounts to twice the values for 5-DAY baseline (1.49 before transaction costs and 1.16 after transaction costs). MaxDD offers an outlook on how sustainable an investment loss can be, where lower is better. Also for this metric we notice the better performance of ENS-DS strategies compared to the Buy&Hold and 5-DAY baselines. ENS produces 13.76% value decreasing to 11.46% for ENS-DS, T = 40 (Table 5), that is less than one fourth of the Buy-and-Hold (55%) and one third compared to the 5-DAY. After transaction costs (Table 6), as expected, the values of MaxDD have increased for both ENS (28.42%) and ENS-DS, T = 40 (26.06%), nevertheless lower than the baselines. With respect to Calmar Ratio, before applying the transaction costs (Table 5), we find that its value has increased from 2.27 for ENS to 2.90 for ENS-DS, T = 40. After transaction costs (Table 6), the ENS strategy registers a value of 0.77, whereas for the ENS-DS, $T = 40$, the value is 0.87, compared to 0.16 of the Buy & Hold strategy. Calmar Ratio scales annualized returns by the value of MaxDD and determines

TABLE 5. Risk assessment of the trading strategies before transaction costs compared to the baselines (5-DAY and Buy&Hold) over a period between March 2007 and January 2016. Portfolio was constructed using $k = 5$ pairs. The best performances are highlighted in bold.

	ARIMA	RF	LGB	SVR	ENS	ENS-DS $T = 30$	ENS-DS $T = 40$	ENS-DS $T = 60$	5-DAY	Buy & Hold
Var 1%	-0.0418	-0.0297	-0.0306	-0.0225	-0.0262	**-0.0220**	-0.0229	-0.0228	-0.0377	-0.0354
Var 5%	-0.0181	-0.0129	-0.0126	-0.0105	-0.0111	-0.0110	**-0.0106**	-0.0111	-0.0179	-0.0171
CVar 1%	-0.0569	-0.0458	-0.0444	-0.0457	-0.0393	**-0.0370**	-0.0403	-0.0372	-0.0624	-0.0512
CVar 5%	-0.0327	-0.0241	-0.0236	-0.0208	-0.0206	**-0.0199**	-0.0202	-0.0201	-0.0318	-0.0290
Downside dev. p.a.	0.1460	0.1129	0.1090	0.1094	0.0987	0.0960	**0.0954**	0.0960	0.1514	0.1347
Sharpe Ratio	0.7167	0.4663	1.0468	1.2454	1.7185	1.7677	**1.8575**	1.7452	0.8539	0.3152
Sortino Ratio	1.1171	0.7887	1.6403	1.9690	2.8908	3.0303	**3.0404**	2.9850	1.4922	0.7812
MaxDD	35.04%	23.23%	33.29%	28.90%	13.76%	11.63%	**11.46%**	13.69%	36.68%	55.19%
Calmar	0.4261	0.3278	0.5394	0.6350	2.2711	2.7445	**2.9048**	2.2982	0.5876	0.1646
Omega	1.1633	1.1131	1.2508	1.2888	1.4230	1.4220	**1.4636**	1.4348	1.2248	1.1139

TABLE 6. Risk assessment of the trading strategies after transaction costs compared to the baselines (5-DAY and Buy&Hold) over a period between March 2007 and January 2016. Portfolio was constructed using $k = 5$ pairs. The best performances are highlighted in bold.

	ARIMA	RF	LGB	SVR	ENS	ENS-DS $T = 30$	ENS-DS $T = 40$	ENS-DS $T = 60$	5-DAY	Buy & Hold
Var 1%	-0.0380	-0.0299	-0.0308	-0.0227	-0.0262	**-0.0223**	-0.0231	-0.0231	-0.0378	-0.0354
Var 5%	-0.0154	-0.0130	-0.0129	-0.0107	**-0.0112**	**-0.0112**	**-0.0112**	**-0.0112**	-0.0183	-0.0171
CVar 1%	-0.0535	-0.0459	-0.0446	-0.0359	-0.0394	**-0.0372**	-0.0375	-0.0375	-0.0625	-0.0512
CVar 5%	-0.0290	-0.0242	-0.0238	-0.0191	-0.0208	**-0.0201**	-0.0203	-0.0203	-0.0324	-0.0290
Downside dev. p.a.	0.1341	0.1138	0.1107	0.0926	0.1006	0.0980	**0.0974**	0.0980	0.1539	0.1347
Sharpe Ratio	0.3511	0.2793	0.6411	0.6640	1.2711	1.3118	**1.3145**	1.2936	0.6812	0.3152
Sortino Ratio	0.5515	0.4901	1.0100	1.0576	2.1142	2.2279	**2.2319**	2.1852	1.1560	0.7812
MaxDD	35.92%	30.09%	36.53%	50.56%	28.42%	27.09%	**26.06%**	27.10%	37.15%	55.19%
Calmar	0.1584	0.1362	0.2826	0.1827	0.7769	0.8407	**0.8730**	0.8225	0.4250	0.1646
Omega	1.0794	1.0694	1.1503	1.1492	1.3003	1.3104	**1.3106**	1.3043	1.1727	1.1139

how many average annual returns are needed to recover from a maximum drawdown. Thus, the ENS-DS strategy needs 1.15 years to recover from the maximum drawdown, whereas the Buy & Hold would need approximately 6 years. The Omega metric, introduced by [57], divides expected returns into two parts: gains and losses or returns above the expected rate (upside) and those below it (downside). In simple terms, Omega can be considered as the ratio of upside (good) returns relative to downside (bad) returns. Before transaction costs (Table 5), this ratio has increased also from 1.42 (ENS) to 1.46 in our proposed approach (ENS-DS, T = 40). After transaction costs (Table 6), results show the same positive trend, i.e., from 1.3003 for simple ensemble, ENS, to 1.3106 for ENS-DS, T = 40. This translates into higher chances of achieving daily positive returns.

We can conclude that, in terms of risk, the dynamical asset selection strategy, ENS-DS, improves the results of simple ensemble, ENS, irrespective of the look-back period, T. In particular, when fixing $T = 40$ we see the best improvement.

D. COMPARISON WITH STATE OF THE ART TRADING STRATEGIES

To finally assess the performances of our proposed StatArb approach in a real-world trading scenario, we

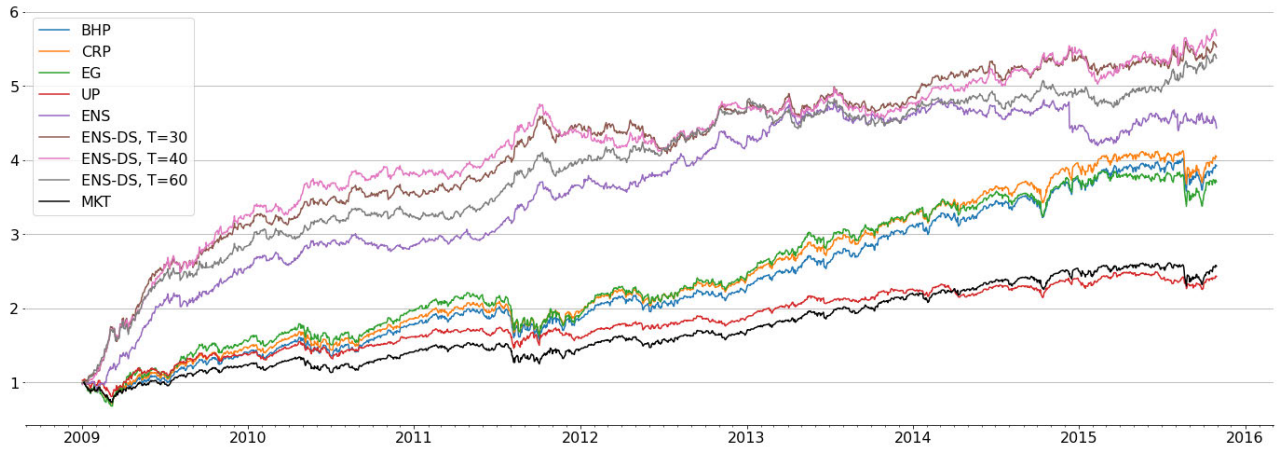

FIGURE 5. Equity curves for the proposed strategies compared to BHP, CRP, EG, UP, and MKT within a period from January 2009 to November 2015, before transaction costs.

TABLE 7. Risk assessment of the trading strategies before transaction costs compared to the BHP, CRP, EG, UP, and MKT from a period from January 2009 to November 2015. Best values are highlighted in bold.

	BHP	CRP	EG	UP	ENS	ENS-DS $T = 30$	ENS-DS $T = 40$	ENS-DS $T = 60$	MKT
MaxDD[%]	28.6751	29.7377	29.5396	22.4178	13.6908	13.3987	**11.4293**	11.7018	27.131
Sharpe Ratio	1.0794	1.1037	1.0994	0.8644	1.6701	1.9902	**2.0045**	1.9218	0.8687
Sortino Ratio	1.5591	1.6085	1.5989	1.2387	2.5439	3.1163	**3.1176**	2.9901	1.2377

compared it with a set of state-of-the-art portfolio strategies, well-established in the literature [58], by following the same validation process as in [29]. Specifically, we considered the competitors listed below (for more in-depth details on these algorithms the reader is referred to [58]):

BHP - Buy&Hold-based Portfolio, i.e. a portfolio implementation of the standard Buy&Hold strategy described in Section VII-E, where rather than buying a single asset (e.g., an *ETF* or a *stock*), the investor buys shares of all the index companies proportionally to their prices;

CRP - Constant Rebalanced Portfolio, i.e. a variation of the BHP strategy, where the portfolio weights are periodically rebalanced, according to the price changes of the underlying assets;

UP - Universal Portfolio, which is a parameterized CRP strategy over the whole simplex domain. The algorithm learns adaptively from historical data and maximizes the log-optimal growth rate in the long run;

EG - Exponential Gradient, i.e. a momentum strategy that focuses on the best performing asset of the index, in the last time period.

For this analysis, we have taken into account a long period spanning between 2009 and 2015. The period encompasses both bull and bear markets regimes, i.e. the market recovery post-global financial crisis (2009-2013), as well as low volatility periods (2013-2015), where statistical arbitrage strategies do not fare particularly well, as noticed by several works in the literature [59]. Figure 5 presents the cumulative returns (or *equity* curves) of the proposed methods (ENS and ENS-DS), against the aforementioned competitors and the behaviour of the general market (MKT). Moreover, in Table 7, we present risk characteristics (in terms of Maximum Drawdown, Sharpe Ratio, and Sortino Ratio) for all these strategies. By analyzing Figure 5, we notice that the proposed methods clearly outperform the baselines and the market, with our ENS-DS ($T = 40$) providing the best overall performance. Such a perception is then confirmed by looking at the results in Table 7, where the proposed approaches show significantly lower values of risk exposure (with a MaxDD ranging between 11.43% to 13.69%, against the 22.41% of the best alternative), and better return-over-risk metrics (e.g., by reaching a considerable Sharpe Ratio of 2.00).

Overall, our results also compare favorably to other statistical arbitrage strategies, such as classical pairs trading results in [7], where Sharpe ratio has a value of 0.59 for the top 20-pairs from 1962 to 2002. In [5], for a generalized pairs trading, the authors report a Sharpe ratio of 1.44 from 1997 to 2007. In [24], the author proposes a method that uses Elman neural networks and ELECTRE III with a Sharpe ratio of approximately 1.5 within a time-span from 1992 to 2006.

Similarly, researchers from the Union Bank of Switzerland (UBS), in their work [60], leverage on RF models to form monthly portfolios. The trading signals are constructed based on the top quantile of the forecasted monthly returns, for which stocks are bought long, and the flop quantile for which the corresponding stocks are sold short. The authors test the methodology on constituents of Asia and Pacific indexes (including the Australia SP300 index), on a period from 1997 to 2015. Comparing the cumulative return in time (ours - ENS-DS, $T = 40$ versus theirs - MSCI AC Asia Pacific ex-Japan), we can observe a similar behavior during the financial crisis of 2007-2009, where both approaches recorded their best performance. Afterward, they report a declining performance that reaches a plateau in 2011-2015. As per Figure 5, our proposed method developed its highest earnings in early 2009, a period of high turmoil. After that period, though the method registers moderate returns, it exhibits an increasing trend. Finally, in terms of risk, they report a MaxDD slightly lower (approximately 10% compared to our MaxDD values of 11% before transaction costs and 26% after transaction costs) and quite surprisingly registered after the global financial crisis, in 2012.

IX. CONCLUSION AND FUTURE WORK

This study aims to extend the existing literature on algorithmic trading based on machine learning and statistical arbitrage by proposing a general approach for risk-controlled trading. We are following the well-established steps of a statistical arbitrage trading system, that is forecasting the price returns, then ranking followed by trading of a number of pairs. The forecasting is performed by a heterogeneous ensemble and, subsequently, we mitigate the risk by employing a dynamic asset selection strategy that prunes assets if they had a decreasing performance in the past period. The proposed approach can be regarded as general and applicable to a wide variety of assets, as well as all of its underlying components.

To test our hypothesis, we created an instance out of it, where we focused our studies on the S&P500 Index, using its constituents as tradeable assets and statistical arbitrage as a trading strategy. For forecasting, we have used three machine learning algorithms, that is Light Gradient Boosting, Random Forests, and Support Vector Machines to which we appended a statistical tool, Auto-regressive moving average, widely used for time-series forecasting. Subsequently, their output is ensembled. To create a heterogeneous ensemble we also proposed to use a set of heterogeneous features that can be used to train the models. By performing a walk-forward procedure, for each stock and walk, we tested all the combinations of features and internal parameters of each regressor to select the best model for each of them. Based on this diverse pool of forecasting methods, our experiments show that the ensemble strategy reaches significant returns of 0.113% per day or 31% per year. When enhancing the trading process with the dynamical asset selection strategy, we managed to successfully increase returns from 31% of the common ensemble to 33%. As many strategies can provide very high returns, they often are also very volatile. The relation of profit and the amount of risk to achieve this result have to be in a reasonable proportion, thus we investigate our strategies considering various risk metrics. To this end, the most remarkable result is that the risk hedged to 14% maximum draw-down for the ensemble to 11% for our proposed dynamic asset selection method, compared to 55% of the market.

With respect to the computational cost of our approach, we may notice that the most intensive step is represented by the training stage. Indeed, in terms of *Big O* notation, this step entails an $O(s \times h)$ cost for each sequential model considered, thus proportional to the number of stocks s and model hyperparameters h. Vice versa, the most computationally expensive process of the testing stage involves the calculation of the rolling mean directional accuracy only, which can be considered negligible in the general mechanics of the proposed method. Hence, although we retain that the integration of such an architecture in a production environment is feasible, to reduce the computational burden, more elaborate training implementations, such as model parallelism, must be taken into account.

Overall, we believe that such promising results open up several important research directions. In this regard, our future work first aims at enriching the current approach with new types of assets like bonds or futures (ETFs). In the second stage, we will consider different strategies for asset grouping and build group models. Finally, a further challenge will involve the employment of neural networks designed for time series forecasting.

NOMENCLATURE

5-DAY	Statistical arbitrage strategy based on five-day cumulative return of each asset
ARIMA	Auto-Regressive Integrated Moving Average
B&H	Buy-and-hold trading strategy based on long-term investment on a single asset
ENS	The proposed ensemble of models
ENS-DS	The proposed ensemble of models, enhanced by the dynamic asset selection strategy
LGB	Light Gradient Boosting
LR	Lagged intra-day price returns
MDA	Mean Directional Accuracy
RF	Random Forests
SVR	Support Vector Regression
TI	Technical Indicators

REFERENCES

[1] S. Carta, D. R. Recupero, R. Saia, and M. M. Stanciu, "A general approach for risk controlled trading based on machine learning and statistical arbitrage," in *Proc. 6th Int. Conf. Mach. Learn., Optim., Data Science (LOD)* in Lecture Notes in Computer Science, vol. 12565, 2020, pp. 489–503.

[2] B. M. Damghani, "The non-misleading value of inferred correlation: An introduction to the cointelation model," *Wilmott*, vol. 2013, no. 67, pp. 50–61, Sep. 2013, doi: 10.1002/wilm.10252.

[3] J. Jayko, "A demon of our own design: Markets, hedge funds, and the perils of financial innovation," *J. Pension Econ. Finance*, vol. 7, no. 3, p. 363, 2008.

[4] A. W. Lo, *Hedge Funds: An Analytic Perspective*. Princeton, NJ, USA: Princeton Univ. Press, 2010.
[5] M. Avellaneda and J.-H. Lee, "Statistical arbitrage in the US equities market," *Quant. Finance*, vol. 10, no. 7, pp. 761–782, Aug. 2010.
[6] A. E. Khandani and A. W. Lo, "What happened to the quants in Aug. 2007? Evidence from factors and transactions data," *J. Financial Markets*, vol. 14, no. 1, pp. 1–46, 2011.
[7] E. Gatev, W. N. Goetzmann, and K. G. Rouwenhorst, "Pairs trading: Performance of a relative-value arbitrage rule," *Rev. Financial Stud.*, vol. 19, no. 3, pp. 797–827, 2006.
[8] G. Vidyamurthy, *Pairs Trading: Quantitative Methods and Analysis*. Hoboken, NJ, USA: Wiley, 2004.
[9] C. Kaufman and D. T. Lang, "Pairs trading," *Data Sci. R: A Case Stud. Approach to Comput. Reasoning Problem Solving*, vol. 1, pp. 241–308, Apr. 2015.
[10] B. M. Henrique, V. A. Sobreiro, and H. Kimura, "Literature review: Machine learning techniques applied to financial market prediction," *Expert Syst. Appl.*, vol. 124, pp. 226–251, Jun. 2019.
[11] E. F. Fama, "Efficient capital markets: A review of theory and empirical work," *J. Finance*, vol. 25, no. 2, pp. 383–417, May 1970.
[12] A. Lo and J. Hasanhodzic, *The Evolution of Technical Analysis: Financial Prediction from Babylonian Tablets to Bloomberg Terminals*. New York, NY, USA: Bloomberg, 2011.
[13] S. Carta, A. Ferreira, D. R. Recupero, M. Saia, and R. Saia, "A combined entropy-based approach for a proactive credit scoring," *Eng. Appl. Artif. Intell.*, vol. 87, Jan. 2020, Art. no. 103292, doi: 10.1016/j.engappai.2019.103292.
[14] M. Atzeni and D. R. Recupero. (2019). *Multi-Domain Sentiment Analysis With Mimicked and Polarized Word Embeddings for Human–Robot Interaction*. FGCS. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X19309719
[15] Y. Li and Z. Yang, "Application of EOS-ELM with binary jaya-based feature selection to real-time transient stability assessment using PMU data," *IEEE Access*, vol. 5, pp. 23092–23101, 2017.
[16] M. Khalaf, H. Alaskar, A. J. Hussain, T. Baker, Z. Maamar, R. Buyya, P. Liatsis, W. Khan, H. Tawfik, and D. Al-Jumeily, "IoT-enabled flood severity prediction via ensemble machine learning models," *IEEE Access*, vol. 8, pp. 70375–70386, 2020.
[17] C. Krauss, X. A. Do, and N. Huck, "Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500," *Eur. J. Oper. Res.*, vol. 259, no. 2, pp. 689–702, Jun. 2017.
[18] L. Takeuchi, "Applying deep learning to enhance momentum trading strategies in stocks," Stanford Univ., Stanford, CA, USA, Tech. Rep., 2013. [Online]. Available: http://cs229.stanford.edu/proj2013/TakeuchiLee-ApplyingDeepLearningToEnhanceMomentumTradingStrategiesInStocks.pdf
[19] D. Enke and S. Thawornwong, "The use of data mining and neural networks for forecasting stock market returns," *Expert Syst. Appl.*, vol. 29, no. 4, pp. 927–940, Nov. 2005.
[20] M. T. Leung, H. Daouk, and A.-S. Chen, "Forecasting stock indices: A comparison of classification and level estimation models," *Int. J. Forecasting*, vol. 16, no. 2, pp. 173–190, 2000. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0169207099000485
[21] G. S. Atsalakis and K. P. Valavanis, "Surveying stock market forecasting techniques—Part II: Soft computing methods," *Expert Syst. Appl.*, vol. 36, no. 3, pp. 5932–5941, Apr. 2009.
[22] S. Carta, A. Corriga, A. Ferreira, D. R. Recupero, and R. Saia, "A holistic auto-configurable ensemble machine learning strategy for financial trading," *Computation*, vol. 7, no. 4, p. 67, Nov. 2019.
[23] R. C. Cavalcante, R. C. Brasileiro, V. L. F. Souza, J. P. Nobrega, and A. L. I. Oliveira, "Computational intelligence and financial markets: A survey and future directions," *Expert Syst. Appl.*, vol. 55, pp. 194–211, Aug. 2016.
[24] N. Huck, "Pairs selection and outranking: An application to the S&P 100 index," *Eur. J. Oper. Res.*, vol. 196, no. 2, pp. 819–825, Jul. 2009.
[25] N. Huck, "Pairs trading and outranking: The multi-step-ahead forecasting case," *Eur. J. Oper. Res.*, vol. 207, no. 3, pp. 1702–1716, Dec. 2010.
[26] M. Dixon, D. Klabjan, and J. H. Bang, "Classification-based financial markets prediction using deep neural networks," *Algorithmic Finance*, vol. 6, nos. 3–4, pp. 67–77, Dec. 2017.
[27] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," *Eur. J. Oper. Res.*, vol. 270, no. 2, pp. 654–669, Oct. 2018.
[28] N. Huck, "Large data sets and machine learning: Applications to statistical arbitrage," *Eur. J. Oper. Res.*, vol. 278, no. 1, pp. 330–342, Oct. 2019.
[29] J. Knoll, J. Stübinger, and M. Grottke, "Exploiting social media with higher-order factorization machines: Statistical arbitrage on high-frequency data of the S&P 500," *Quant. Finance*, vol. 19, no. 4, pp. 571–585, Apr. 2019. [Online]. Available: www.scopus.com
[30] S. J. Brown, W. Goetzmann, R. G. Ibbotson, and S. A. Ross, "Survivorship bias in performance studies," *Rev. Financial Stud.*, vol. 5, no. 4, pp. 553–580, Oct. 1992.
[31] A. W. Lo and A. C. MacKinlay, "Data-snooping biases in tests of financial asset pricing models," *Rev. Financial Stud.*, vol. 3, no. 3, pp. 431–467, Jul. 1990.
[32] D. Lou, C. Polk, and S. Skouras, "A tug of war: Overnight versus intraday expected returns," *J. Financial Econ.*, vol. 134, no. 1, pp. 192–213, Oct. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0304405X19300650
[33] Y. Kara, M. A. Boyacioglu, and Ö. K. Baykan, "Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul stock exchange," *Expert Syst. Appl.*, vol. 38, no. 5, pp. 5311–5319, May 2011.
[34] J. Patel, S. Shah, P. Thakkar, and K. Kotecha, "Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques," *Expert Syst. Appl.*, vol. 42, no. 1, pp. 259–268, Jan. 2015.
[35] J. Patel, S. Shah, P. Thakkar, and K. Kotecha, "Predicting stock market index using fusion of machine learning techniques," *Expert Syst. Appl.*, vol. 42, no. 4, pp. 2162–2172, Mar. 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417414006551
[36] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in *Proc. Adv. Neural Inf. Process. Syst.*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. New York, NY, USA: Curran Associates, 2017, pp. 3146–3154.
[37] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. New York, NY, USA: Springer, Feb. 2009.
[38] M. Jiang, J. Liu, L. Zhang, and C. Liu, "An improved stacking framework for stock index prediction by leveraging tree-based ensemble models and deep learning algorithms," *Phys. A, Stat. Mech. Appl.*, vol. 541, Mar. 2020, Art. no. 122272. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0378437119313093
[39] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.
[40] V. N. Vapnik, "An overview of statistical learning theory," *IEEE Trans. Neural Netw.*, vol. 10, no. 5, pp. 988–999, Sep. 1999.
[41] A. Chalimourda, B. Schölkopf, and A. J. Smola, "Experimentally optimal ν in support vector regression for different noise models and parameter settings," *Neural Netw.*, vol. 17, no. 1, pp. 127–141, 2004.
[42] G. E. P. Box and G. Jenkins, *Time Series Analysis, Forecasting and Control*. San Francisco, CA, USA: Holden-Day, 1990.
[43] T. Devezas, "Principles of forecasting. A handbook for researchers and practitioners: J. Scott Armstrong. Kluwer Academic Publishers, Norwell, MA, USA, 2001, XII and 849 pages. ISBN 0-7923-7930-6 (Hardbound); US$190," *Technol. Forecasting Social Change*, vol. 69, no. 3, pp. 313–316, 2002.
[44] A. A. Ariyo, A. O. Adewumi, and C. K. Ayo, "Stock price prediction using the ARIMA model," in *Proc. UKSim-AMSS 16th Int. Conf. Comput. Modeling Simulation*, Mar. 2014, pp. 106–112.
[45] G. Rogova, "Combining the results of several neural network classifiers," *Neural Netw.*, vol. 7, no. 5, pp. 777–781, 1994.
[46] V. Genre, G. Kenny, A. Meyler, and A. Timmermann, "Combining expert forecasts: Can anything beat the simple average?" *Int. J. Forecasting*, vol. 29, no. 1, pp. 108–121, Jan. 2013.
[47] A. Timmermann, "Chapter 4 forecast combinations," in *Handbook of Economic Forecasting*, vol. 1, G. Elliott, C. Granger, and A. Timmermann, Eds. Amsterdam, The Netherlands: Elsevier, 2006, pp. 135–196.
[48] J. H. Stock and M. W. Watson, "Combination forecasts of output growth in a seven-country data set," *J. Forecasting*, vol. 23, no. 6, pp. 405–430, Sep. 2004.
[49] C. Bergmeir, R. J. Hyndman, and B. Koo, "A note on the validity of cross-validation for evaluating autoregressive time series prediction," *Comput. Statist. Data Anal.*, vol. 120, pp. 70–83, Apr. 2018.
[50] O. Blaskowitz and H. Herwartz, "Adaptive forecasting of the EURIBOR swap term structure," *J. Forecasting*, vol. 28, no. 7, pp. 575–594, Nov. 2009.
[51] L. E. Farmer, L. Schmidt, A. Timmermann, F. Diebold, X. Gabaix, B. Paye, and H. Pesaran, "Pockets of predictability," SSRN, Tech. Rep. 3152386, 2019. [Online]. Available: https://ssrn.com/abstract=3152386

[52] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Oct. 2011.
[53] P. F. Christoffersen and F. X. Diebold, "How relevant is volatility forecasting for financial risk management?" *Rev. Econ. Statist.*, vol. 82, no. 1, pp. 12–22, Feb. 2000, doi: 10.1162/003465300558597.
[54] J. Stübinger and S. Endres, "Pairs trading with a mean-reverting jump–diffusion model on high-frequency data," *Quant. Finance*, vol. 18, no. 10, pp. 1735–1751, Oct. 2018.
[55] *LightGBM Documentation*. Accessed: Feb. 2021. [Online]. Available: https://lightgbm.readthedocs.io/en/latest/Python-API.html
[56] W. F. Sharpe, "Mutual fund performance," *J. Bus.*, vol. 39, no. 1, pp. 119–138, Jan. 1966. [Online]. Available: http://www.jstor.org/stable/2351741
[57] C. K. William and F. Shadwick, "A universal performance measure," *J. Perform. Meas.*, vol. 6, pp. 59–84.
[58] B. Li and S. C. H. Hoi, "Online portfolio selection: A survey," *ACM Comput. Surv.*, vol. 46, no. 3, pp. 1–36, Jan. 2014, doi: 10.1145/2512962.
[59] H. Rad, R. K. Y. Low, and R. Faff, "The profitability of pairs trading strategies: Distance, cointegration and copula methods," *Quant. Finance*, vol. 16, no. 10, pp. 1541–1558, Oct. 2016, doi: 10.1080/14697688.2016.1164337.
[60] N. Baltas, D. Jessop, S. Jones, P. Winter, S. Wu, O. Antrobus, and P. Stoltz, "Quantitative monographs: 'Stock selection using machine learning,'" UBS, Zürich, Switzerland, Tech. Rep. d1U3L1n0hmdP, 2015. [Online]. Available: https://neo.ubs.com/shared/d1U3L1n0hmdP

SALVATORE M. CARTA (Member, IEEE) received the Ph.D. degree in electronics and computer science from the University of Cagliari, in 2003. He is currently an Associate Professor with the Department of Mathematics and Computer Science, University of Cagliari, Italy. In 2005, he joined the Department of Mathematics and Computer Science, University of Cagliari, as an Assistant Professor. In 2006 and 2007. he was a Guest Researcher with the Swiss Federal Institute of Technology as an invited Professor, hosted by Laboratoire des Systmes Intgrs-LSI. His research interests include clustering algorithms, social media analysis, and text mining for behavioral pattern identification and recommendation in group of users and in single users; AI algorithms for credit scoring, fraud detection and intrusion detection; AI algorithms for financial forecasting and robo-trading; and E-coaching platforms and AI algorithms for healthy lifestyles. He is author of more than 110 conference and journal papers in these research fields, with more than 1400 citations and is member of the ACM. He founded three hi-tech companies, spin off of the University of Cagliari, and is currently leading one of them.

SERGIO CONSOLI received the Ph.D. degree. He is currently a Scientific Project Officer at the European Commission, Joint Research Centre (DG-JRC), Ispra, Italy, working at the Centre for Advanced Studies on the project Big Data and Forecasting of Economic Developments. Previously, he was a Senior Scientist within the Data Science department at Philips Research, Eindhoven (NL), focusing on advancing automated analytical methods used to extract new knowledge from data for HealthTech applications. Other former experiences include the Italian Presidency of the Council of Ministers and the National Research Council of Italy. He also provided ICT consultancy services to Isab, the largest oil refinery in the Mediterranean area. His education and scientific experience fall in the areas of data science, operational research, artificial intelligence, knowledge engineering, semantic reasoning, and machine learning. He is the author of several research publications in peer-reviewed international journals, granted EPO and WIPO patents, edited books, and leading conferences in the fields of his work. He is the co-editor of the book S. Consoli, D. Reforgiato Recupero, and M. Petkovic *Data Science for Healthcare: Methodologies and Applications* (Springer Nature, 2019).

ALESSANDRO SEBASTIAN PODDA received the B.Sc. and M.Sc. degrees (Hons.) in informatics from the University of Cagliari, and the Ph.D. degree in mathematics and informatics, supported by a grant from the Autonomous Region of Sardinia, with a thesis entitled "Behavioural contracts: from centralized to decentralized implementations." He is currently a Postdoctoral Researcher with the Department of Mathematics and Computer Science, University of Cagliari. In 2017, he has been a Visiting Ph.D. Student with the Laboratory of Cryptography and Industrial Mathematics, University of Trento. He is currently a member of the Artificial Intelligence and Big Data Laboratory and the Blockchain Laboratory of the University of Cagliari, in which he has been technical supervisor and collaborator of numerous research projects. His current research interests mainly include deep learning, financial forecasting, information security, blockchains, and smart contracts. To date, he has been the coauthor of several journal articles and scientific conference papers, as well as reviewer for different top-tier international journals.

DIEGO REFORGIATO RECUPERO received the Ph.D. degree in computer science from the University of Naples Federico II, Italy, in 2004. He has been an Associate Professor with the Department of Mathematics and Computer Science, University of Cagliari, Italy, since December 2015. From 2005 to 2008, he has been a Postdoctoral Researcher with the University of Maryland College Park, USA. He won different awards in his career (such as Marie Curie International Reintegration Grant, Marie Curie Innovative Training Network, Best Research Award from the University of Catania, Computer World Horizon Award, Telecom Working Capital, Startup Weekend). He co-founded six companies within the ICT sector and is actively involved in European projects and research (with one of his companies he won more than 30 FP7 and H2020 projects). His current research interests include sentiment analysis, semantic web, natural language processing, human–robot interaction, financial technology, and smart grid. In all of them, machine learning, deep learning, big data are key technologies employed to effectively solve several tasks. He is the author of more than 140 conference and journal papers in these research fields, with more than 1600 citations.

MARIA MADALINA STANCIU received the B.Sc. and M.Sc. degrees in information and communication technology from the Military Technical Academy, Bucharest, Romania, in 2007. She is currently pursuing the Ph.D. degree with the Mathematics and Computer Science Department, University of Cagliari. She has a proven track record as a Software Engineer of building highly scalable and distributed systems on several fields like environmental, social and governance research. Her current research interest is machine learning applied to financial forecasting.