Research in International Business and Finance 70 (2024) 102336

Contents lists available at ScienceDirect

Research in International Business and Finance

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

Full length article

S&P 500 stock selection using machine learning classifiers: A look into the changing role of factors

Antonio Caparrini [a,*], Javier Arroyo [b], Jordi Escayola Mansilla [c]

[a] Faculty of Statistics, Universidad Complutense de Madrid, Madrid, 28040, Spain
[b] Institute of Knowledge Technology, Universidad Complutense de Madrid, Madrid, 28040, Spain
[c] Department of Statistics and Operational Research, Universitat Oberta de Catalunya, Barcelona, 08018, Spain

ARTICLE INFO

Keywords:
Stock selection
Machine learning
XGBoost
Factor investing
Backtesting
Explainability

ABSTRACT

This study examines the profitability of using machine learning algorithms to select a subset of stocks over the S&P 500 using factors as features. We use tree-based algorithms: Decision Tree, Random Forest, and XGBoost for their white model capabilities, allowing feature importances extraction. We defined a backtest to train the models with recent data and rebalance the portfolio. Despite incurring more risks, the selected assets of the portfolio outperform the index by using machine learning. Furthermore, we show that the feature importance that determines the best-performing assets changes at different times. Such models providing the evolution of the importance of factors can provide profitability insights while keeping explainability.

1. Introduction

The stock market provides a reliable and regulated environment where market participants can efficiently transact financial instruments allocating capital. Markets are complex and highly influenced by political, economic and psychological factors. Researchers and investors seek to detect anomalies in the market that can be exploited to make a profit. There are ever-growing researched and discovered anomalies (Cochrane, 2011) with explanatory capacity commonly known as factors.

One of the first factor models (Sharpe, 1964) uses the β (beta) as the factor chosen to explain the excess of returns. This factor correlates one specific asset's returns with the overall market returns. Fama and French (1992) propose that the excess return of stocks is explained by three factors, later extended to five factors (Fama and French, 2015).

Factors are typically used as valuation metrics to rank a universe of stocks and invest in those that rank higher. However, more sophisticated approaches can be used to exploit the factor's information to select the stocks better. One of these approaches is the use of machine learning.

The first public fund to adopt machine learning technologies for actively selecting stocks was AIEQ, established on October 18, 2017. Since then, similar funds have emerged, and the effectiveness of these methods and technologies has continued to improve. Moreover, research has shown that artificial intelligence-powered mutual funds outperform their human-managed counterparts (Chen and Ren, 2022), which can be attributed to their lower transaction costs, superior stock-picking capabilities, and reduced behavioral biases.

The list of factors considered in the literature is lengthy. After a critical examination, most of them were deemed statistically insignificant in the q-factor model (Hou et al., 2018).

However, using more sophisticated selection methods could capture interactions among factors. In particular, the advantage of machine learning methods over linear statistical methods (such as linear regression models or logistic regression) is that they can

* Corresponding author.
E-mail address: acaparri@ucm.es (A. Caparrini).

https://doi.org/10.1016/j.ribaf.2024.102336
Received 23 July 2021; Received in revised form 30 August 2023; Accepted 24 March 2024
Available online 27 March 2024

capture both linear and nonlinear relations between the factors and returns that traditional models cannot address. In other words, they can better grasp subtle anomalies and complex interactions. On the other hand, they are typically black-box models; hence, it is impossible to understand how they work.

Thus, we propose a machine-learning-based approach that exploits the relationships among factors to classify the stocks into an ordered categorical variable according to their profitability. We will consider three well-known tree-based algorithms because such methods make it possible to analyze feature importance. In this way, we overcome one of the major problems of machine learning methods, by revealing which factors contribute more to the classification.

As input features, we will consider a list of 20 factors that measure different aspects (momentum, profitability, value-vs-growth, investment, trading frictions, and intangibles). Since these profitable patterns may be short-lived, re-training the algorithm is crucial to learn the new profitable patterns.

In particular, we regularly re-train the algorithm to learn the new patterns, better adjust the importance of features and re-balance the portfolio accordingly. We will also use genetic algorithm optimization for the hyper-parameter tuning of our machine learning algorithms since we observe that hyper-parameter adjustment is typically overlooked in the literature and is, at best, performed using a coarse grid search over one or a few parameters.

We analyze the performance of our strategy to select stocks that belong to the S&P 500 from 3 January 2005 to 30 September 2019 and show that all of them beat the index.

We also analyze the evolution of the importance of features over time, that is, as we retrain the algorithm. The ability of machine learning algorithms to adapt to the changes and learn new patterns explains their success.

The paper is organized as follows: Section 2 introduces related work using machine learning and artificial intelligence in finance on similar topics. Section 3 presents the data, the machine learning techniques and their optimization by genetic algorithm. In Section 4 the strategy (or backtest) is described including the different windows, how the classifiers are trained and how the portfolio is formed. Section 5 shows the results of the three classifiers, the importance of variables and returns on the strategy. Finally, Section 6 summarizes the conclusions and discusses future research and improvements.

2. Related work

According to Ahmed et al. (2022), the literature on the application of artificial intelligence and machine learning in finance shows a rising trend in publications, particularly in the areas of bankruptcy prediction, stock price prediction, portfolio management, oil price prediction, anti-money laundering, behavioral finance, big data analytics, and blockchain. Among these, predicting stock prices is crucial for investors and has been a popular research topic over the last decade (Ghosh et al., 2022).

Several studies have been conducted in this last area. For example, Novak and Velušček (2016) applied support vector machines (SVMs) to predict stock prices and reported that machine learning techniques significantly improve prediction accuracy compared to traditional models. In another study, Chen and Ge (2021) investigated portfolio optimization using neural networks and found that this approach was more effective than classic solutions. Hanauer et al. (2022) applied linear regression and tree-based machine learning methods to estimate monthly peer-implied fair values of European stocks. In cryptocurrencies, Liu et al. (2023) employed machine learning models to predict returns for 3703 cryptocurrencies for the 2013–2021 period. Leippold et al. (2022) built and analyzed a comprehensive set of return prediction factors using various machine learning algorithms.

Similarly, Moghaddam et al. (2016) investigated the ability of artificial neural networks in forecasting the daily NASDAQ, Ghosh et al. (2022) applied both long short-term memory networks (LSTM) and random forest to forecast directional movements is constituent stocks of the S&P 500, and Liu et al. (2021) used LSTM combined with online social network data to forecast stock prices.

In our work, we apply machine learning algorithms to stock selection using factors, and there are recent precedents on the topic. For example, Rasekhschaffe and Jones (2019) use machine learning to forecast the cross-section of stock returns. Abe and Nakagawa (2020) uses deep learning on 33 factors and compares its performance in portfolio management against random forests and ridge regressions to forecast the next 5 day's stock return. However, this approach focuses on predicting returns, while our purpose is classification (e.g. outperformers versus underperformers). Regarding the use of classification algorithms, Jidong and Ran (2018) applies XGBoost to stock selection and Fu et al. (2020) applies random forest for the same purpose. We will consider both algorithms in our approach and a classification approach to limit the risk of overfitting and to include the investment decision in the problem.

3. Methods

In this section, we introduce the dataset that will be used along with the machine learning algorithms for the model and the genetic algorithm that seeks to optimize their hyper-parameters.

3.1. Dataset

Quandl is, based on their description, the premier source for financial, economic, and alternative datasets, serving investment professionals. Quandl's platform is used by over 400,000 people, including analysts from the world's top hedge funds, asset managers and investment banks.

Table 1
Factor features. These are the factors used in the training of classifiers. The tables show the formal name of the factor, the category, and the variable name used in some figures. Section 3.1 provides more details about the dataset.

Factor	Category	Label
6-month prior return	Momentum	momentum_6 m
11-month prior return	Momentum	momentum_11 m
Return on equity	Profitability	roe
Beta 3Y covariance	Intangible	beta_3Y
Beta 3Y linear reg	Intangible	beta_3Y_coef
Price-to-book	Value-vs-growth	price-to-book
Earnings-to-price	Value-vs-growth	earnings-to-price
Price-to-sales	Value-vs-growth	price-to-sales
Market capitalization	Trading frictions	marketcap
Enterprise-value	Value-vs-growth	enterprise-value
Evebit	Value	evebit
Evebitda	Value	evebitda
12 month lagged return	Intangibles	returns_12m_lagged_12 m
24 month lagged return	Intangibles	returns_12m_lagged_24 m
Operating cash flow-to-price	Value-vs-growth	operating cashflow-to-price
Investment-to-price	Investment	investment-to-price
Earning per share	Profitability	earnings-per-share
Current ratio	Intangibles	current-ratio
Operating cashflow-to-equity	Profitability	operating cashflow-to-equity
Capex	Intangible	capex

Table 2
The target variable. The 12-month return is split into ranges to construct a label variable. This constructed variable allows using supervised machine learning algorithms. The universe considered in each date is the stocks on the S&P 500. On each rebalancing date, the 15 assets with more probability of belonging to label 4 are selected for the portfolio.

12-month return	Target variable
$return \geq 0.15$	4
$0.05 \leq return < 0.15$	3
$0 \leq return < 0.05$	2
$-0.15 \leq return < 0$	1
$return < -0.15$	0

This paper uses the Sharadar US Equities dataset.[1] This dataset provides fundamental indicators, financial ratios and stock prices for US public companies from January 1998.

From all the features available, we selected 18 and computed the 3 years beta, using the price data, covariance and linear regression adding 2 more features. Beta is computed against the S&P 500 daily returns. The final features are shown in Table 1. Most of these features are selected based on Hou et al. (2018) where the most relevant anomalies (or factors) are selected over 452 that are present in the literature.

The target variable is selected based on the 12-month returns as explained in Table 2. This labeling approach is similar to the one used in Fu et al. (2020); nonetheless, in our case, the yearly rate of returns is selected instead of a monthly rate.

3.2. Machine learning algorithms

The task we must solve is a classification task. On the one hand, we have the features that are the factors described in Table 1 and, on the other hand, the target variable which is a class or label assigned to the financial asset regarding its returns as defined in Table 2.

This task is solved using supervised machine learning algorithms. Given a dataset of labeled individuals with describing features, supervised machine learning algorithms aim to predict the class or label of an unlabeled new individual. The algorithm infers a function from the input labeled data to predict the labels on new data. Within the several supervised machine learning algorithms, we limit this research to the tree-based techniques described below for our experiments.

Tree-based techniques are chosen because they can extract the feature importance giving insights about the information used to make a prediction.

Decision Tree, Random Forest and XGBoost are the machine learning techniques taken into account. These methods have been applied to stock selection and portfolio formation in Creamer and Freund (2010), Sugitomo and Minami (2018), Jidong and Ran (2018), Abe and Nakagawa (2020), Fu et al. (2020), Brogaard and Zareei (2022) among others.

[1] More information: https://www.quandl.com/databases/SFA/data.

3.2.1. Decision trees

Decision Trees (DT) are non-parametric supervised machine learning algorithms (Breiman et al., 1984) used for classification and regression. The result of a Decision Tree is a hierarchical tree representation, where leaves represent the target class predicted and the nodes represent rules over the features.

Decision trees recursively partition the data into subsets based on the values of the input features. In this way, decision trees are able to model complex relationships between the input features and the target variable, including non-linear relationships and interactions between features.

Unlike other machine learning methods that are black-box models, the tree structure of a decision tree can be understood and analyzed. However, they are prone to over-fitting and typically perform worse than other more complex techniques, such as random forest or XGBoost.

The implementation of the Decision Tree used in this paper is available in the open source package scikit-learn (Pedregosa et al., 2011).[2]

3.2.2. Random forest

Random forest is an ensemble learning method that is made up of a set of decision trees whose predictions are aggregated to identify the most likely result.

Random forest utilizes both bagging and feature randomness. Bagging is the abbreviation of bootstrap aggregation. It works by training multiple models on different subsets of the training data. The subsets of the training data are created by random sampling with replacement from the original dataset. Feature randomness generates a random subset of features, which ensures low correlation among decision trees. By combining both methods, each decision tree learns a different part of the data set.

In sum, random forest fits a number of decision tree classifiers on random sub-samples of the dataset. In this way, it creates an uncorrelated forest of decision trees. The aggregation of the predictions of the individual decision trees is supposed to improve the predictive accuracy and control over-fitting. While random forests are not as interpretable as decision trees, it is possible to analyze the feature importance of a random forest by aggregating the importance of the feature in all its trees. In this work we use the scikit-learn implementation (Pedregosa et al., 2011) of Random Forest.[3]

3.2.3. XGBoost

Gradient boosting is a machine learning technique that combines multiple weak predictive models, typically decision trees, to create a strong predictive model. The key idea behind gradient boosting consists of training models in a sequential manner, where each subsequent model is built to correct the mistakes made by the previous models. This is achieved by minimizing a loss function that measures the difference between the predicted and actual values. It works on the principle that many weak learners (e.g. shallow trees) can together make a more accurate predictor.

The machine learning algorithm eXtreme Gradient Boosting (XGBoost) (Chen and Guestrin, 2016) efficiently implements the gradient tree boosting approach.

XGBoost is a reinforcement algorithm that iterates the generation of models, trees in this case, minimizing new algorithms' errors based on the errors detected at previous iterations, that is, each new tree aims to reduce the error left over by the previous learner. This allows the algorithm to iteratively improve its predictions and produce a more accurate model. However, this kind of algorithm tends to overfit, so it is important to use techniques to prevent overfitting. XGBoost is an advanced gradient-boosting technique that prevents overfitting by weighting the decrease of the objective function and the complexity of the model.

XGBoost generally reduces bias and improves accuracy with respect to simpler classifiers such as decision trees. As in the case of random forests, it is possible to analyze the feature importance of an XGBoost algorithm by aggregating the feature importance of the individual trees.

The implementation used in this work is the open-source XGBoost Python package.[4]

3.3. Time series KFold cross-validation

When performing the first training of a machine learning algorithm, we will tune the hyperparameters using only the training data with a genetic algorithm. In order to avoid overfitting, we will perform k-fold cross-validation. Since we are dealing with time-series data, we use a time-series split.[5] This cross-validation is a variation of k-fold. The kth split returns the first k folds as the train set and the (k+1)th fold as the validation set. Note that, unlike standard cross-validation methods, successive training sets are supersets of those that come before them. Fig. 1 shows how the splits are performed in our research, where we have taken $k = 5$.

[2] https://scikit-learn.org/stable/modules/tree.html.

[3] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html.

[4] https://xgboost.readthedocs.io/en/latest/.

[5] https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.TimeSeriesSplit.html.

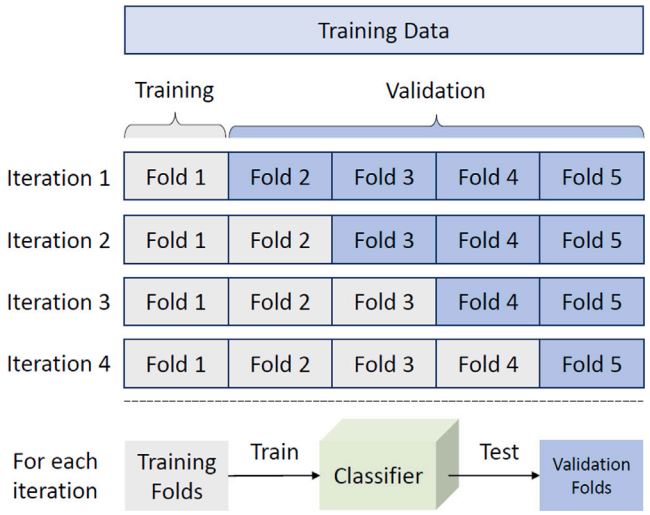

Fig. 1. KFolds for the training data on the first date. Only on the first rebalancing date is a genetic optimization algorithm performed to search for the best hyperparameters for an algorithm. This genetic algorithm needs a fitness value for the search, which for our research is balanced accuracy. In order to avoid overfitting, instead of using a training metric, a temporal k-fold technique is used (with k = 5). This genetic search is performed using the first Training Data (Fig. 1). When each individual (classifier) of the genetic search is evaluated, four classifiers are trained, as shown in the iterations splitting the Training Data. The mean of the balanced accuracy of the four classifiers is used as the fitness metric for the individual.

3.4. Genetic algorithms

A Genetic Algorithm is a heuristic search and optimization technique based on natural selection. Machine learning algorithms used in this work have several parameters that can be tuned. The parameters depend on the implementation (mostly related to the learning capacity or avoiding over-fitting). We use deap[6] (Fortin et al., 2012) to carry out the genetic search: each set of parameters is an individual, and the target metric to optimize is balanced accuracy. Our research has used this approach in other fields, such as credit risk (Ariza-Garzon et al., 2020) or automatic music genre classification (Caparrini et al., 2020).

The balanced accuracy[7] is a metric used in binary and multiclass classification problems to deal with imbalanced datasets. It is defined as the average of recall obtained in each class. For each individual in the population, i.e. for each parameter combination, we use as fitness value the inverse of the mean classification error in the validation of stratified (k-fold) cross-validation setting with $k = 5$. Balanced accuracy increases the importance of the error in the minority classes. However, this fitness value is a balanced accuracy obtained in (k-fold) cross-validation.

The parameters used in the genetic algorithm search are:

- Epochs: 4
- Initial population: 14
- Mate method: two-point crossover
- Selection: Select the best individual among four randomly chosen individuals, repeated 100 times
- Mutation: Mutate an individual by replacing attributes, with probability 0.35 by a number uniformly drawn between the lower and upper bounds of the attribute.

The search for hyper-parameters is time-consuming with more extensive datasets and boosted classifiers like XGBoost. Ideally, we would conduct a genetic search on each rebalanced date. However, this could take a non-viable amount of time. Only one search is performed on the first rebalanced date for each algorithm. The same hyper-parameters as the first search are used for consecutive rebalancing dates. Nevertheless, the classifier is trained again using the window of data according to the strategy defined in Section 4.

4. Backtest

In this section, we introduce the framework used to assess the performance of the proposed strategies. The methodology considers the practices we thought achieve less over-fitting and avoid look-ahead bias.

In Arnott et al. (2018), the authors warn about the use of machine learning in backtesting, a large amount of data allows for multiple layers of cross-validation, but in finance, there are fewer data and a lower information-to-noise ratio. Besides, Fabozzi and

[6] https://deap.readthedocs.io.

[7] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.balanced_accuracy_score.html.

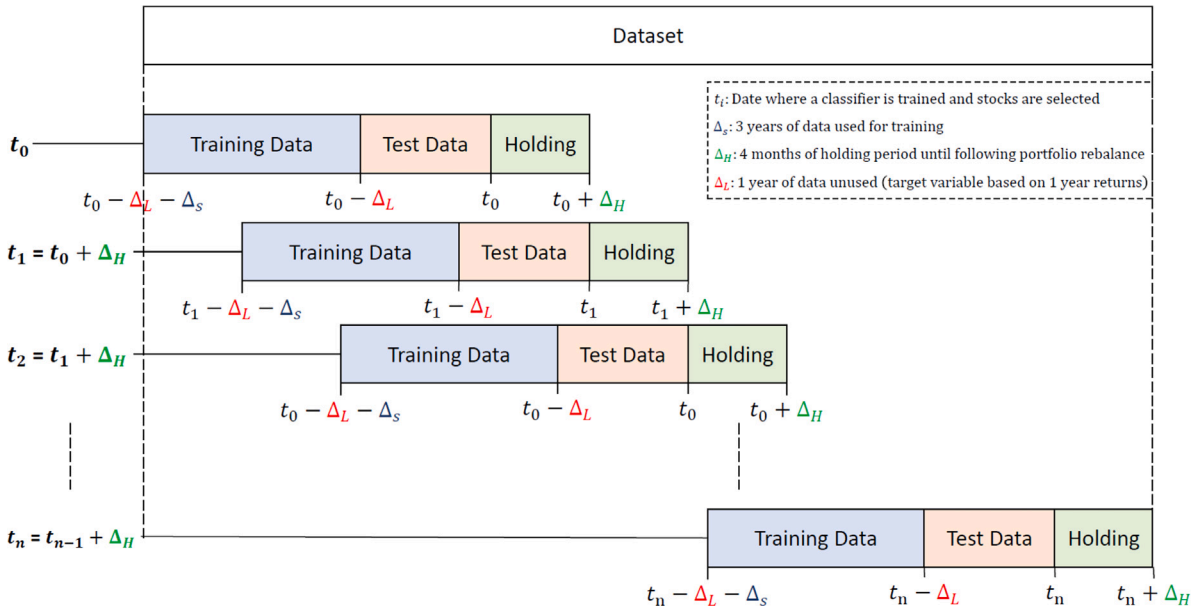

Fig. 2. Backtest Design. The figure shows the ranges taken into account to perform the training of algorithms and the selection of stocks. The Training Data is three years, the Test Data is one year, and the holding period is four months. There are 44 rebalancing dates. On each of them, first, a classifier is trained using the training data; second, the classifier is tested using the Test Data to obtain the accuracy of the classifier; third, the classifier is used to predict the classes of the S&P 500 stocks, and finally, the 15 stocks with more probability to be label 4 are selected to form the portfolio. t0 is a particular case where a hyperparameter optimization is performed using the Training Data, as described in Fig. 1. In the 43 following dates, the hyperparameters remain with the same values.

López De Prado (2018) acknowledge that only strategies that perform well are published while thousands tested never make it to the public. This pick-the-pleasing outlier strategy certainly will generate disappointment when implemented in a productive trading system.

At any given date, it has to be asserted that the model's training is performed only considering data known at that time. Furthermore, the first date considered must have enough data to train the model.

A machine learning algorithm able to predict the best-performing stocks over a long period would decrease performance as new data is available and not used in the model. The most current market conditions could generate some features to vary in importance. As a result, the most recent data should be used to retrain the algorithm to achieve better predictability. We opt not to use the history of past returns but the most recent ones, arguing that the whole history could be misleading and the training time would increase.

A trading strategy transforms a signal into asset weights. Our signal is the predicted probability of class 4 (returns ≥15%) for a given stock. We apply the equal-weighted policy, selecting 15 assets, each contributing to 1/15 of the portfolio. To describe the strategy, first, we define t_i, $\triangle_S$, $\triangle_H$ and $\triangle_L$

- t_i: Re-balance date where a classifier is trained, and stocks are selected
- $\triangle_S$: Range of time that contains the data used for training (**3 years**)
- $\triangle_H$: Holding period between rebalanced dates (**4 months**)
- $\triangle_L$: Period of data not used in training equal to the horizon in which the label is computed (**1 year**)

The backtest is conducted following a series of steps:

- First, a rebalanced date is selected (t_i). On this date, a classifier is trained using the data available from the range ($t_i - \triangle_S - \triangle_L, t_i - \triangle_L$). We avoid the use of the data between ($t_i - \triangle_L, t_i$); otherwise, we would incur look-ahead bias.
- Then, the classifier is used to predict the classes for the data in t_i. These results are used to rank the stocks, and the best-ranked ones are selected to form the portfolio. The weights of the stocks selected are the naive $1/N$ since it is simple and usually hard to beat (DeMiguel et al., 2009). Returns are computed using the pricing data for the stocks in the portfolio on t_{i+1} and t_i.
- Finally, we conduct the same process in t_{i+1}, being $t_{i+1} = t_i + \triangle_H$, until the last rebalanced date is simulated.

Fig. 2 shows how the training is performed on each rebalanced date.

5. Results

In this section, the results are discussed. First, the performance of the models in terms of accuracy, then the overall returns of the backtest strategy and finally the importance of the variables through time.

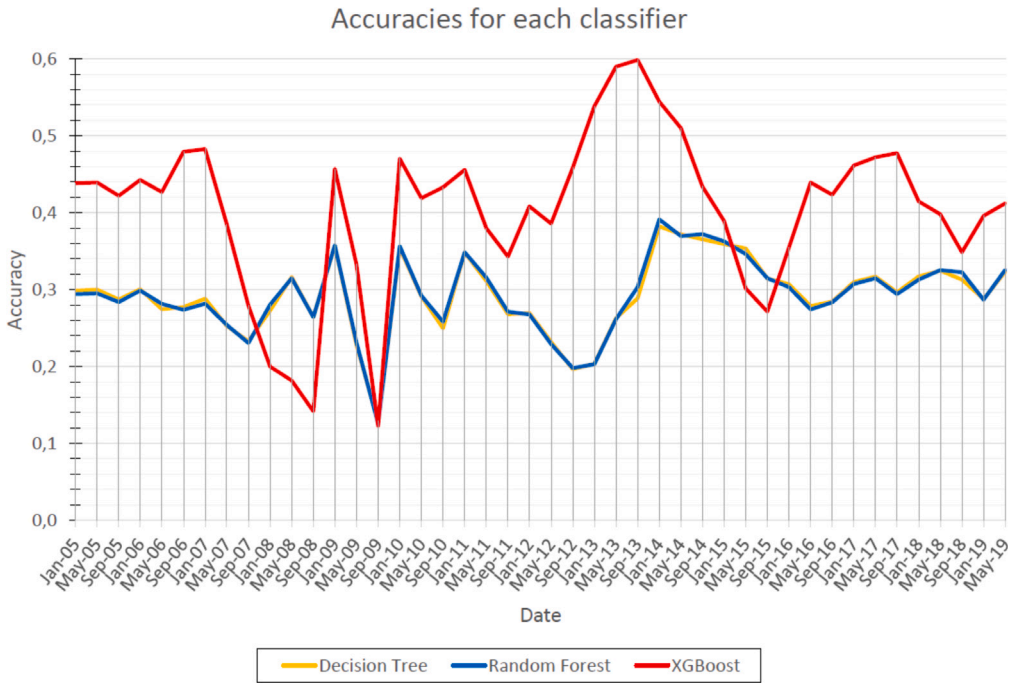

Fig. 3. Classifier accuracies. One classifier is trained on every rebalancing date to select the portfolio (44 of each of the three classifiers). The figure shows the accuracy of the classifiers on each rebalancing date. These are test metrics using the Test Data described in Fig. 2.

Table 3
Classifier accuracy. One classifier is trained on every rebalancing date to select the portfolio (44 of each of the three classifiers). Mean accuracy and standard deviation for all the classifiers trained of each type are presented. This metric is taken using the test data as it is shown in Fig. 2.

Algorithm	Mean accuracy	Accuracy deviation
Decision Tree	0.292	0.05
Random Forest	0.293	0.051
XGBoost	0.404	0.104

5.1. Classification performance

We have conducted 3 backtest trading strategies, each using a specific machine learning technique and maintaining the exact details. Due to the retrain classifiers on each rebalanced window ($\triangle_H$) we obtained several different classifiers with their respective accuracy. Aiming to show an aggregate metric of these accuracies, the mean accuracy and standard deviation for each algorithm are presented in Table 3. XGBoost is the best-performing classifier; nonetheless, its deviation implies that the accuracy is unstable across time. For better visualization, in Fig. 3 accuracies for each classifier in each rebalanced date are shown. Even though the accuracies seemed low, only the 15 stocks selected on each date had a real effect on the final return. As a result, we considered that the backtest results in Section 5.2 are more relevant indicators of the performance.

5.2. Strategy backtest returns

Summarizing, strategies create the portfolio selecting the 15 predicted best-performing stocks from the S&P 500 list on the selection date using the machine learning model. They are held until the next date when the portfolio is rebalanced using the same method. Table 4 presents the returns obtained for each strategy and the S&P 500 from 3 January 2005 to 30 September 2019.

All the strategies outperform in returns the S&P 500, but the DT and RF perform slightly better than the S&P 500, 1%–2% annualized, at the cost of doubling volatility. Only the strategy selecting assets with XGBoost performs well considering the volatility increase.

It is important to remark that transaction costs and fees are not considered in this research. Updating portfolio compositions is not free, and the cost is related to the times it is performed, the number of assets, and the weight change. They could diminish, hinder or cancel profitability in implementing the approach. Furthermore, interactions with the market have an impact on it. This could diminish the effect of the anomalies detected and the returns obtained. Fig. 4 shows the comparison between the strategies and the S&P 500 Index.

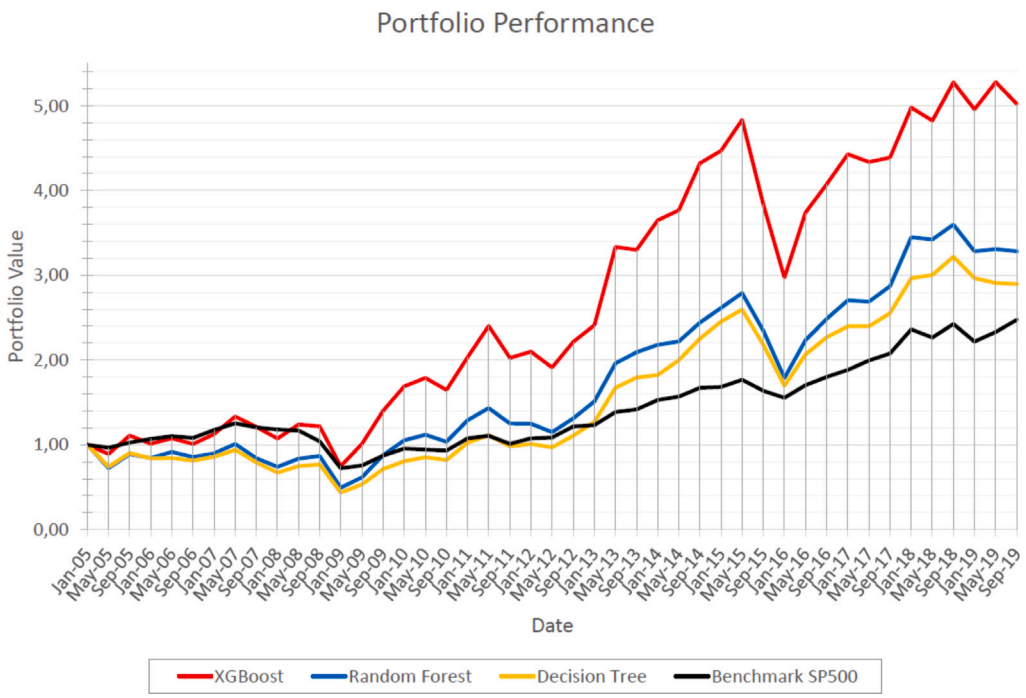

Fig. 4. Returns comparison. The graph shows the evolution of a theoretical 1-dollar investment using the stocks selected by the machine learning algorithms and the S&P 500 for benchmarking purposes. For the machine learning portfolios, on each rebalancing date, a subset of the 15 assets with more probability to be label 4 (as described in Table 2) form the portfolio with equal weights. It is important to notice that our research does not consider trading costs.

Table 4
Portfolio returns. The table shows the returns of each portfolio selecting 15 stocks on the S&P 500 rebalancing every four months and the Benchmark S&P 500 from January 2005 to September 2019. The overall returns consider the total return from the first to the last rebalancing date; this is transformed into an equivalent compounded annual return. The annual volatility is added to the Sharpe ratio to assess the return on risk, noticing that the portfolios selected using the machine learning algorithms have greater returns involving more risk.

Portfolio	Overall returns	Annualized returns	Annual volatility	Sharpe ratio
Benchmark S&P 500	247.63%	6.18%	17.81%	0,35
Decision Tree	289.63%	7.29%	33.54%	0,22
Random Forest	328.2%	8.18%	38.7%	0,21
XGBoost	502.16%	11.26%	39.42%	0,29

5.3. Feature importance

Tree-based machine learning algorithms offer an importance score based on the reduction in the criterion used to select the split points. For each classifier, a vector of feature importances is obtained. In Figs. 5, 6, 7 the feature importances for each set of algorithms is shown. We show the mean and standard deviation of the feature importances of the algorithms retrained in all the rebalancing dates.

Fig. 5 for Decision Trees and Fig. 6 Random Forests show almost identical feature importances even in mean and deviation. These models most informing features are the 2 calculated betas explaining almost 20% of the splits. They are faster and simpler models so the decision-making using betas makes sense given that a greater beta implies more returns with the market in an upwards trend. In these two algorithms, the least important features on average have been those related to the company's market value (evebit and evebitda). The standard deviations show that the importance of some variables greatly varies over time, as is the case of the momentum variables. On the other hand, the importance of the variability of variables such as the enterprise value almost no varies.

Fig. 7 shows the aggregated importance of the features in the XGBoost algorithms. A significant difference is that the average importance of the features is quite similar between them and that the standard deviation is slight, with a few exceptions, such as the case of two of the most important features (12 months lagged returns and the beta 3Y coefficient). With smaller variations, market capitalization is one of the essential features over time.

The variability of the feature importances because market conditions change and so do the features that determine the best-performing assets. Detailed feature importances time evolution is shown in Figs. 8, 9, 10 that we are going to seek insights.

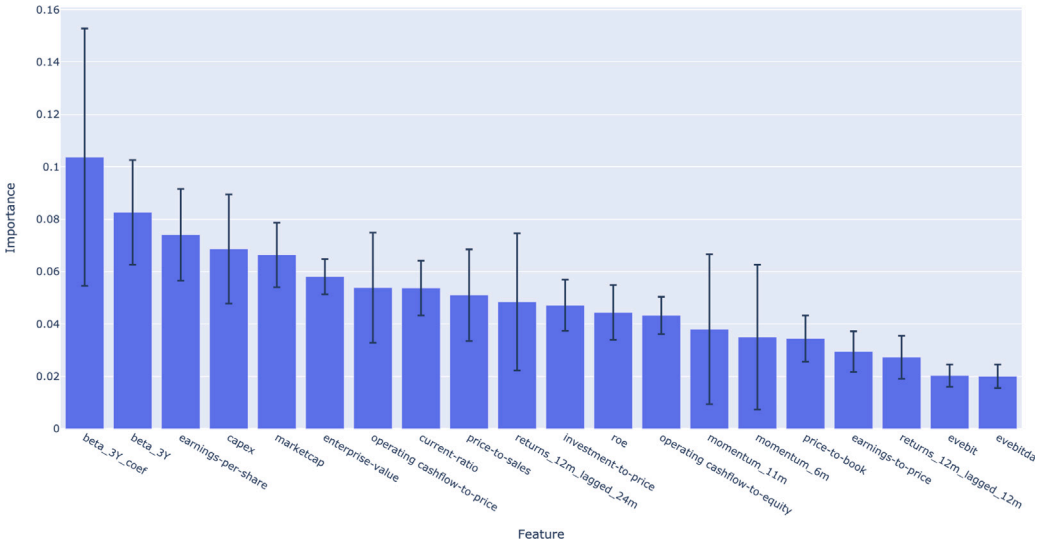

Fig. 5. Feature importances ranking Decision Tree. 44 Decision Tree classifiers are trained (one on each rebalancing date). The figure shows the aggregated ranked mean of those importances with their standard deviation.

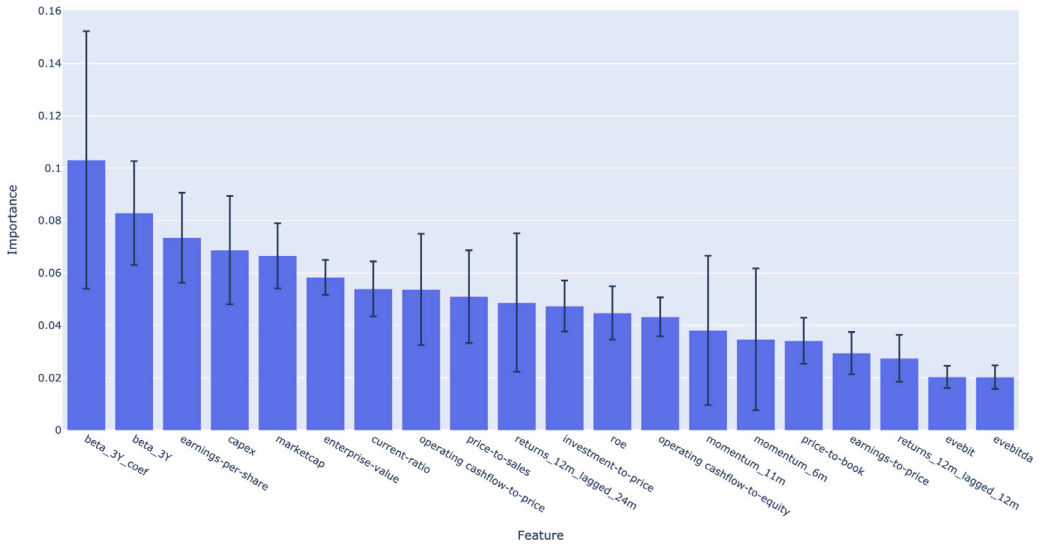

Fig. 6. Feature importances ranking Random Forest. 44 Random Forest classifiers are trained (one on each rebalancing date). The figure shows the aggregated ranked mean of those importances with their standard deviation.

To correctly interpret the figures, it is essential to notice that each feature's scale of the X-axis is different. Clearly, feature importances change in the timeline, meaning that the algorithms adapt to the changing market conditions by assigning different importances to the predictors.

The period between 2009 and 2013 is a remarkable example. In this period features relevant in other periods like market capitalization, evebit, roe, price to book and capex abruptly drop in importance while other features like momentum, enterprise value and lagged returns spikes in importance. This insight is interesting considering the financial crisis of 2009, the features that predict returns changes based on the market conditions. If we leave aside the 2009–2013 period, the importance as predictors of some factors has changed before 2009 and after 2013. In some cases, it has increased, as is the case of capex, while in others, it has decreased, as is the case of the operating cashflow-to price.

The feature importances of the XGBoost, best-performing strategy and classifier, are similar in shape to the importances of the Decision Tree and Random Forest. However, the range of deviation is typically lower in comparison. This means that for a given period, the importance of the factors analyzed (that is, their respective effectiveness as predictors) evolves similarly regardless of the algorithm used. However, the differences in the performance of the algorithms evidence that XGBoost is generally better at capturing successful relationships among them and the target variable.

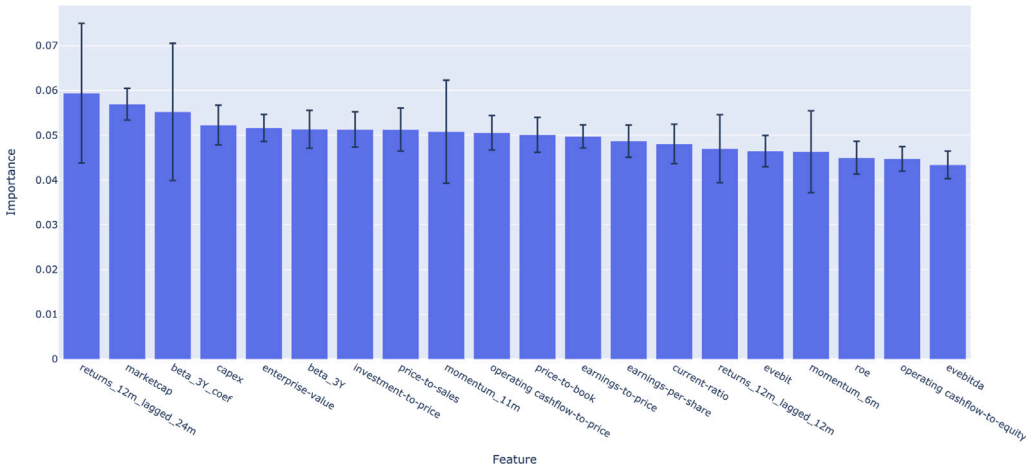

Fig. 7. Feature importances ranking XGBoost. 44 XGBoost classifiers are trained (one on each rebalancing date). The figure shows the aggregated ranked mean of those importances with their standard deviation.

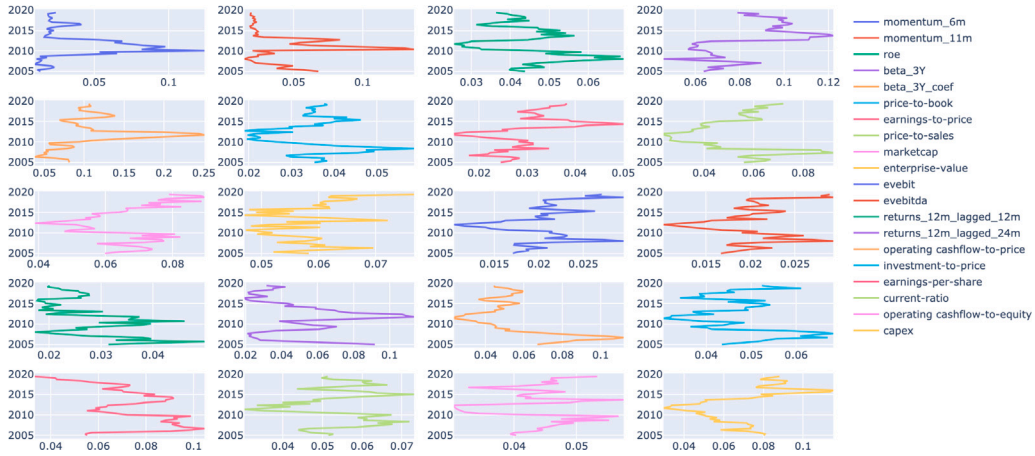

Fig. 8. Decision Tree Feature importances. 44 Decision Tree classifiers are trained (one on each rebalancing date). Each one of them has its feature importance for the 20 features. The figure shows how the importance of each of the features changes between classifiers.

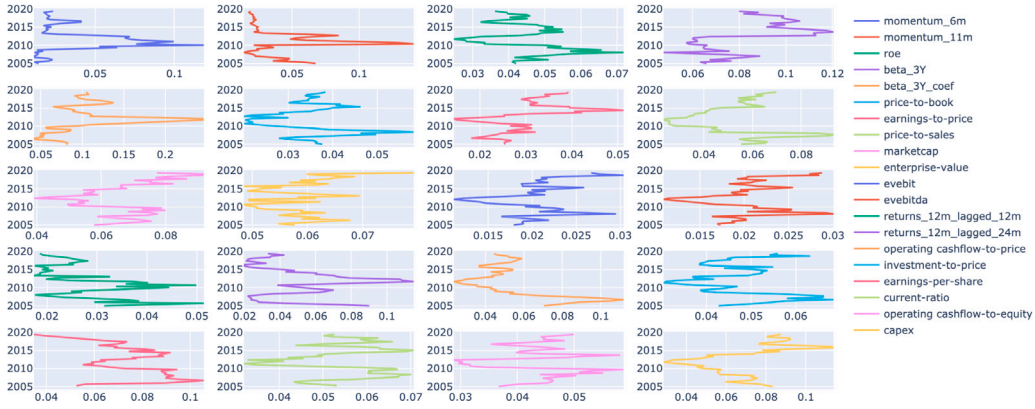

Fig. 9. Random Forest Feature importances. 44 Random Forest classifiers are trained (one on each rebalancing date). Each one of them has its feature importance for the 20 features. The figure shows how the importance of each of the features changes between classifiers.

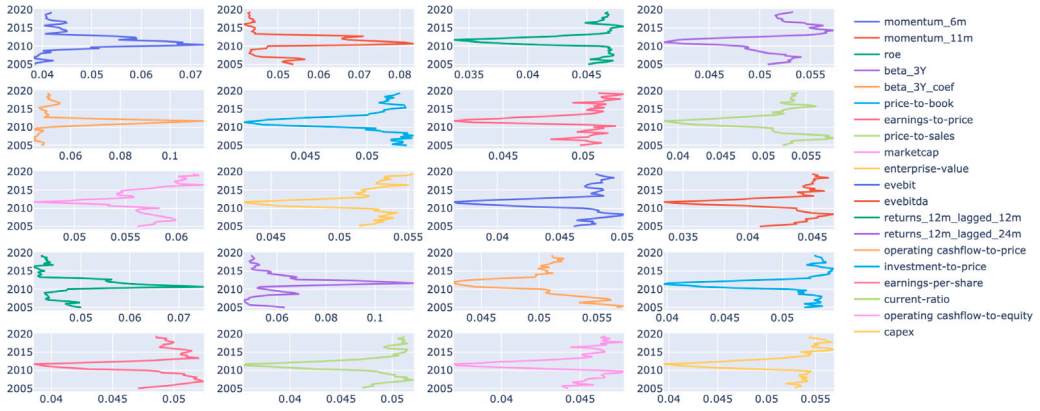

Fig. 10. XGBoost Feature importances. 44 XGboost classifiers are trained (one on each rebalancing date). Each one of them has its feature importance for the 20 features. The figure shows how the importance of each of the features changes between classifiers.

6. Conclusion

This article has shown that using machine learning algorithms for a subset stock selection from an index, such as S&P 500, using financial factors as predictors can outperform the index, as the machine learning algorithms adapt to market conditions by changing the importance of the factors as predictors. We followed what we considered are good practices for applying machine learning to the cross-section of stock returns:

- Using reliable input data.
- Training with the most recent data to perform predictions.
- Asserting neither survivor nor look-ahead bias.
- Performing time-series split in training to reduce overfitting.
- Fine-tuning the hyperparameters using a metaheuristic strategy, such as genetic algorithms.

Still, our work has some limitations that would improve the results and explainability.

- We have not taken into account transactional costs. The rebalancing is each four months, and depending on the results, it is possible to sell all the positions and buy new ones. The transactional costs on these could erase the exceeding profits and should be taken into account before using the approach.
- Our approach only can be performed when the factor data is available to obtain the most profitable assets. Since we depend on the data provider, bias on the data or missing data will penalize the portfolio formation.
- Ideally, the genetic optimization of the hyperparameters should be performed in each of the re-trained and rebalancing dates. We were not able due to time constraints and time-consuming experiments. Bringing the approach to production could be viable using parallelizing and better computing power.

According to our results, XGBoost performed substantially better than random forest and decision trees in terms of classification and financial performance. This is no surprise since gradient boosting tree methods are highly effective and often appear as the winning solutions at machine learning competitions and in state-of-the-art research that use machine learning classification.

The analysis of the feature importance over time showed that the algorithms adapt to the changing market conditions by changing the importance assigned to the features. This is an interesting insight since it points out that the effectiveness of the factors as predictors of good stocks changes over time and that the machine learning algorithms can successfully reflect these changes. As future work, it would be interesting to further analyze the role of the factors as predictors using explainable tools such as the SHAP values (Lundberg and Lee, 2017) that can quantify not only the feature importance but also the sign and the shape of the relationship among each feature and the target variable.

In order to improve the performance of the approach proposed, it is important to notice that risk management was not taken into account during the stock selection process. By adding risk management constrictions, the variance of the result could improve. This article has tested the stock selection approach explained by beating the benchmark index. Applying this approach in other markets and the previously mentioned improvements would lead to better and more stable results.

As another potential improvement, we have opted to use an equal weight policy on the stocks selected on each rebalancing date. Thus, optimizations in the weights could lead to improved portfolio returns and better ratios involving risk, such as Sharpe Ratio.

Our approach seems to perform better in a market environment without disruptions. A future line of research that could lead to improvement is to combine this model with another aimed to predict disruptive moments or market crashes. The graphical representation of the feature importances gives insights into the change in feature importances at different points in time. Thus, using other techniques for feature importance such as SHAP values could add value and interpretability. Considering the different

models for the different dates, the use of the returns generated by the previous model in the next ones with an ensemble or stack model could lead to improvements and better performance. Each model has outlooks about different moments and patterns that could be repeated in the future, they will be worthy of taking advantage of.

CRediT authorship contribution statement

Antonio Caparrini: Conceptualization, Methodology, Investigation, Software, Writing – original draft, Writing – review & editing. **Javier Arroyo:** Conceptualization, Methodology, Investigation, Writing – review & editing, Visualization. **Jordi Escayola Mansilla:** Conceptualization, Investigation.

Data availability

The authors do not have permission to share data.

Acknowledgment

Javier Arroyo acknowledges support from the European Union's H2020 Coordination and Support Actions CA19130.

References

Abe, M., Nakagawa, K., 2020. Cross-sectional stock price prediction using deep learning for actual investment management. In: Proceedings of the Proceedings of the 2020 Asia Service Sciences and Software Engineering Conference. ASSE '20, Association for Computing Machinery, Nagoya, Japan, New York, NY, USA, pp. 9–15.

Ahmed, S., Alshater, M.M., Ammari, A.E., Hammami, H., 2022. Artificial intelligence and machine learning in finance: A bibliometric review. Res. Int. Bus. Finance 61, 101646.

Ariza-Garzon, M.J., Arroyo, J., Caparrini, A., Segovia-Vargas, M.J., 2020. Explainability of a machine learning granting scoring model in peer-to-peer lending. IEEE Access 8, 64873–64890.

Arnott, R.D., Harvey, C.R., Markowitz, H., 2018. A backtesting protocol in the era of machine learning. SSRN Electron. J. 1–18.

Breiman, L., Friedman, J., Olshen, R., Stone, C., 1984. Classification and Regression Trees. Wadsworth and Brooks/Cole, Monterey, CA, USA.

Brogaard, J., Zareei, A., 2022. Machine learning and the stock market. J. Financ. Quant. Anal. 1–42.

Caparrini, A., Arroyo, J., Pérez-Molina, L., Sánchez-Hernández, J., 2020. Automatic subgenre classification in an electronic dance music taxonomy. J. New Music Res. 49, 269–284.

Chen, S., Ge, L., 2021. A learning-based strategy for portfolio selection. Int. Rev. Econ. Finance 71, 936–942.

Chen, T., Guestrin, C., 2016. XGBoost. In: Proceedings of the Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, New York, NY, USA, pp. 785–794.

Chen, R., Ren, J., 2022. Do AI-powered mutual funds perform better? Finance Res. Lett. 47, 102616.

Cochrane, J.H., 2011. Presidential address: Discount rates. J. Finance 66, 1047–1108.

Creamer, G., Freund, Y., 2010. Automated trading with boosting and expert weighting. Quant. Finance 10, 401–420.

DeMiguel, V., Garlappi, L., Uppal, R., 2009. Optimal versus naive diversification: How inefficient is the 1/N portfolio strategy? Rev. Financ. Stud. 22, 1915–1953.

Fabozzi, F.J., López De Prado, M., 2018. Being honest in backtest reporting: A template for disclosing multiple tests. J. Portf. Manag. 45, 141–147.

Fama, E.F., French, K.R., 1992. The cross-section of expected stock returns. J. Finance 47, 427–465.

Fama, E.F., French, K.R., 2015. A five-factor asset pricing model. J. Financ. Econ. 116, 1–22.

Fortin, F.A., De Rainville, F.M., Gardner, M.A., Parizeau, M., Gagné, C., 2012. DEAP: Evolutionary algorithms made easy. J. Mach. Learn. Res. 13, 2171–2175.

Fu, Y., Cao, S., Pang, T., 2020. A sustainable quantitative stock selection strategy based on dynamic factor adjustment. Sustainability 12, 3978.

Ghosh, P., Neufeld, A., Sahoo, J.K., 2022. Forecasting directional movements of stock prices for intraday trading using LSTM and random forests. Finance Res. Lett. 46, 102280.

Hanauer, M.X., Kononova, M., Rapp, M.S., 2022. Boosting agnostic fundamental analysis: Using machine learning to identify mispricing in European stock markets. Finance Res. Lett. 48, 102856.

Hou, K., Xue, C., Zhang, L., 2018. Replicating anomalies. Rev. Financ. Stud. 33, 2019–2133.

Jidong, L., Ran, Z., 2018. Dynamic weighting multi factor stock selection strategy based on XGboost machine learning algorithm. In: Proceedings of the 2018 IEEE International Conference of Safety Produce Informatization. IICSPI, pp. 868–872.

Leippold, M., Wang, Q., Zhou, W., 2022. Machine learning in the Chinese stock market. J. Financ. Econ. 145, 64–82.

Liu, Y., Li, Z., Nekhili, R., Sultan, J., 2023. Forecasting cryptocurrency returns with machine learning. Res. Int. Bus. Finance 64, 101905.

Liu, K., Zhou, J., Dong, D., 2021. Improving stock price prediction using the long short-term memory model combined with online social networks. J. Behav. Exp. Finance 30, 100507.

Lundberg, S.M., Lee, S.I., 2017. A unified approach to interpreting model predictions. Adv. Neural Inf. Process. Syst. 30.

Moghaddam, A.H., Moghaddam, M.H., Esfandyari, M., 2016. Stock market index prediction using artificial neural network. J. Econ. Finance Adm. Sci. 21, 89–93.

Novak, M.G., Velušček, D., 2016. Prediction of stock price movement based on daily high prices. Quant. Finance 16, 793–826.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. J. Mach. Learn. Res. 12, 2825–2830.

Rasekhschaffe, K.C., Jones, R.C., 2019. Machine learning for stock selection. Financ. Anal. J. 75, 70–88.

Sharpe, W.F., 1964. Capital asset prices: A theory of market equilibrium under conditions of risk. J. Finance 19, 425–442.

Sugitomo, S., Minami, S., 2018. Fundamental factor models using machine learning. J. Math. Finance 08, 111–118.