Doctoral Dissertation
Doctoral Program in Computer and Control Engineering (35thcycle)

Machine Learning-Driven Decision Making based on Financial Time Series

Exploding the Myth of Machine Learning in Finance

By

Jacopo Fior

******

Supervisor(s):
Prof. Luca Cagliero, Supervisor

Politecnico di Torino
2023

Declaration

Abstract

The field of financial time series analysis has witnessed remarkable advancements in recent years, driven by the integration of data science and machine learning techniques. This PhD thesis investigates the application of machine learning techniques in financial time series analysis to enhance decision-making processes and improve financial practices. The research objectives of this thesis encompass various research topics, including time series classification, forecasting, clustering, embedding, and summarization, all tailored to the financial domain. The research addresses several fundamental research questions, guiding the investigation into key aspects of financial time series analysis. These questions cover topics such as the determination of optimal time frequencies for trading, the usefulness of stock chart patterns, the unique behaviors of cryptocurrencies, the selection of locally optimal stock portfolios, the extraction of domain-related knowledge from financial time series, the feasibility of plug-and-play machine learning-based trading systems, and the trustworthiness of machine-generated signals.

The thesis encompasses multiple studies, each contributing to the overall understanding of financial time series analysis and addressing specific aspects of the research questions.

The findings from the studies provide valuable insights into financial time series analysis. The optimal time frequency for stock trading is not necessarily the most fine-grained or coarse-grained, emphasizing the importance of selecting appropriate time frequencies based on the data and trading objectives. Stock chart patterns prove to be valuable in filtering machine learning-generated signals, proving their complementary nature. Cryptocurrencies exhibit distinct behaviors (e.g., the momentum effect) that can be leveraged by Machine Learning-based systems to enhance trading strategies. Data-driven taxonomies outperform domain-specific taxonomies in terms of portfolio diversification, and the integration of an optimization step over a heuristic based portfolio generator proves to be able to enhance the selected portfolios profitability. Extracting meaningful insights from financial time series is achievable through time series embedding, enabling effective comparison of stocks behaviors even in the form of human readable summaries. And, finally, eXplainable AI techniques can guide feature selection and increase trust in data-driven systems.

These findings collectively demonstrate the potential of machine learning techniques in uncovering hidden patterns, making accurate predictions, and generating informative signals for financial decision-making. The research highlights the value of machine learning in enhancing financial practices by leveraging its strengths in identifying complex temporal dependencies and capturing underlying dynamics within financial markets.

Importantly, the thesis emphasizes the need for collaboration between machine learning algorithms and domain experts. While machine learning techniques can offer powerful tools for analyzing financial time series, the involvement of domain experts is crucial for interpreting the generated signals, providing contextual knowledge, and making informed decisions based on the recommendations. Machine learning-based trading systems should not be considered as standalone, *plug-and-play* solutions, but rather as complementary tools that require the expertise and judgment of seasoned investors.

The research presented in this thesis contributes to a deeper understanding of the complexities of financial markets and provides insights for practitioners, researchers, and investors seeking to leverage machine learning for financial decision-making.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background and motivation

In recent years, the fields of data science and machine learning have witnessed remarkable advancements, which have had a profound impact on the way we analyze and extract valuable insights from complex and vast datasets. Among the various domains benefiting from these advancements, one area that has experienced significant progress is the analysis of time series data. Time series data, characterized by their sequential and temporal nature, are abundant in numerous fields, with the financial domain standing out as a particularly prominent one.

Financial time series data encompasses a wide range of information, including historical stock prices, exchange rates, economic indicators, and other relevant financial metrics. Such data plays a vital role in the decision-making processes associated with investment strategies, risk management techniques, and portfolio selection and optimization. By leveraging the insights derived from analyzing financial time series, individuals and organizations can make informed decisions, mitigate risks, and optimize their financial outcomes.

The integration of machine learning and data science techniques into the analysis of time series data has revolutionized the way to approach and extract information from these complex temporal datasets. Machine learning algorithms and methodologies provide powerful tools for uncovering patterns, trends, and relationships within financial time series data that may not be immediately apparent to human analysts. These techniques enable the identification of intricate temporal dependencies, capturing hidden patterns that could contribute to the overall understanding of the financial markets.

Additionally, the employment of machine learning approaches allows the development of predictive models that can forecast future trends and market movements based on historical time series data. This capability has significant implications for financial decision-making, as timely and accurate predictions can lead to more effective investment strategies and risk management practices. By leveraging machine learning algorithms, researchers and practitioners can gain a deeper understanding of the underlying dynamics of financial markets, enabling them to make informed and data-driven decisions in real-time.

1.2 Research objectives

The intent of this PhD thesis is to contribute to this rapidly evolving field by focusing on the development of innovative approaches and methodologies for leveraging machine learning in financial time series analysis with the ultimate aim of enhancing decision-making processes and enabling more effective and efficient financial practices. The subsequent chapters of this thesis will delve into and explore various research topics, including time series classification, forecasting, clustering, embedding, and summarization. The key research activities include the study, design, and development of effective representations of univariate and multivariate time series data to tackle domain-specific tasks.

1.3 Research questions

To achieve the research objectives, this thesis focuses on addressing several fundamental data science problems related to time series analysis and adapting them to the financial domain. The following research questions will guide the investigation:

1. Trend is a friend. Ok, but... at which time frequency?
2. Can Stock Chart Patterns transcend the myth and prove their utility?
3. Cryptocurrencies vs. stocks. What's new?

4. *Locally* optimal stock portfolios selection. Possible or not?
5. Extracting useful domain related knowledge from financial time series. Really achievable or just wishful thinking?
6. Can Machine Learning-based Trading Systems actually be *plug-and-play*? What role should domain experts play?
7. Can traders and investors trust machine generated signals?

These research questions guided the different studies that will be presented in this thesis.

1.4 Main contributions

The application of data driven techniques in the financial domain, while seemingly trivial at first, poses significant challenges. The research questions introduced represent only a subset of the numerous problems that arise when attempting to employ these techniques to guide trading and investment strategies. The research presented in this thesis consists of multiple studies with a common objective: to assist traders and investors in navigating the complexities inherent in data-driven and machine learning-based methods, while also maximizing the potential of these approaches.

Each of the studies presented in this thesis aims to address one or more of those research questions.

Chapter 2 presents three distinct studies focused on stocks and cryptocurrencies trading. The objective of these studies is to identify the optimal model settings and data characteristics in short-term trading scenarios. Specifically:

- The first study (See Section 2.1) aims to determine the most effective time granularities and model settings for leveraging the available data (*Research Question 1*).
- The second study (See Section 2.2) explores the combination of stock chart patterns (a staple of technical analysis-based trading) with data-driven trading

signal generation. The goal is to assess the complementarity between these two approaches (*Research Questions 2 and 7*).

- The third study (See Section 2.3) delves into the use of different underlying assets and explores how specific properties of cryptocurrencies can enhance trading in this relatively new market (*Research Question 3*).

Chapter 3 focuses on long-term portfolio selection and introduces various improvements over a state-of-the-art approach. First, time series clustering is employed to create a data-driven taxonomy (*Research Question 5*), aiming to determine whether a data-driven technique can outperform domain-specific taxonomies in terms of portfolio diversification (See Section 3.3). Subsequently, in Section 3.4, the state-of-the-art approach is integrated into a larger system that includes an optimization step, investigating the potential for selecting optimal portfolios (*Research Question 4*). This study also investigates what kind of decisions are better left to the investor (*Research Question 6*).

Chapter 4 explores different methods to assist investors in gaining a better understanding of both the behavior of underlying assets and the rationale behind signal generation in data-driven approaches (*Research Question 7*). Specifically:

- The first study (See Section 4.1) investigates the value of extracting information from time series embedding to generate human-readable summaries (*Research Question 5*).
- The second study (See Section 4.2) explores the application of eXplainable AI techniques to cryptocurrency trading, with goals such as guiding feature selection and increase investors' trust in data-driven systems (*Research Question 3*).

Finally, Chapter 5 provides a comprehensive discussion of the findings from the different studies and their contributions to answering the research questions that guided this research.

Chapter 2

Market forecasting and data driven stock trading

Market prediction and data-driven stock trading have become crucial aspects of the financial industry, with accurate forecasts and intelligent decision-making having a significant impact on investment outcomes. The emergence of machine learning and artificial intelligence has sparked a growing interest in utilizing these technologies to enhance market analysis, prediction, and trading strategies. This chapter aims to explore various aspects of applying machine learning techniques in market forecasting and stock trading.

Forecasting the stock market is a popular yet challenging problem in finance, particularly as the accessibility of stock-related data has increased through trading platforms, online newspapers, and social communities. Consequently, the research community has increasingly focused on leveraging machine learning techniques to design automated trading strategies [1]. The exponential growth of Information and Communication Technologies has further revolutionized financial services, replacing discretionary traders' investments with orders triggered by quantitative trading systems. In fact, High-Frequency Trading (HFT) systems currently account for approximately 70% of the volume exchanged in the U.S. stock exchange [2]. As a result, forecasting stock markets has become particularly challenging for retail investors.

The abundance of historical stock-related data, now readily available through web-based and mobile services, has attracted the attention of the Artificial Intelligence and Data Science communities. Researchers aim to explain stock market movements by extracting significant patterns from stock price series, financial news flows, macroeconomic indicators, financial reports, and content posted on blogs and social networks.

Machine learning-based stock trading strategies have shown promise, particularly those based on classification techniques [3]. These strategies focus on predicting the most likely short-term stock direction (e.g., up, down, flat) based on the analysis of past trends. Predicting discrete outputs has proven more effective than forecasting actual stock prices [4]. Previous studies extensively investigated the integration of different price series-related features such as technical indicators, oscillators, volatility indices [5–7], and candlestick patterns [8, 9]. Various algorithms have been employed, including Support Vector Machines, Decision Trees, K-Nearest Neighbors [10, 11], Artificial Neural Networks [12, 13], Rough Cognitive Networks [14], Genetic Algorithms [15, 16], Itemset mining [17, 18], Ensemble methods [19, 20], and Deep Learning (DL) Architectures [21, 22]. A comprehensive review of the literature on this topic has been presented in [23].

The state of the art of machine learning-based stock trading can be summarized as follows:

- Technical indicators have emerged as the most predictive features [23].
- Support Vector Machines (SVMs) and Artificial Neural Network (ANN) models are the most popular algorithms due to their ability to handle non-linear relationships despite their inherent simplicity [23].
- DL models show great potential for effectiveness, provided a sufficiently large training set, ample computational power, and appropriate tuning of model hyperparameters [24].

Despite the significant progress made in forecasting market movements, integrating ML algorithms into real trading systems requires addressing several open issues, including:

- Determining the most appropriate time granularity for analyzing stock-related data.

- Identifying the optimal portion of historical data to consider when training ML models.
- Establishing recommended settings for traders seeking to integrate ML-based approaches into real trading system architectures.
- Exploring the potential improvement of data-driven systems' performance by incorporating standard trading rules based on technical indicators.

Furthermore, cryptocurrencies have recently gained significant attention from financial institutions, resulting in a surge of speculative interest in Bitcoin, Ethereum, and other cryptocurrencies [25]. Cryptocurrencies possess unique price trends and exchange volumes due to their medium of exchange and ownership policies, often exhibiting considerably higher price volatility compared to conventional assets [26]. This distinction raises the research question of whether the predictive models commonly applied to traditional markets, such as stock and Forex exchanges, also hold true for the cryptocurrency market or if different behaviors in this particular scenario can be leveraged to enhance cryptocurrency trading performance.

2.1 Exploring the use of data at multiple granularity levels in ML-based stock trading

The first investigation undergone addresses the first three unsolved issues mentioned above and is closely related to the research published in [27]. Specifically it focuses on the employment of Machine Learning (ML) algorithms for guiding intraday stock trading. The objective is to determine: (i) the optimal time granularity to employ, (ii) the most valuable data subset to utilize, and (iii) the best use of a trading system.

This research solely relies on analyzing historical stock price data and extracting features based on technical analyses [28].

Stock price data can be obtained and analyzed at different levels of granularity. The choice of granularity depends on factors such as the prediction horizon, the variability of the data series, and the presence of noise or errors. Selecting an appropriate time granularity is particularly challenging due to the unpredictability of market factors and the need to address the curse of dimensionality. The primary objective of this investigation is to examine the relationship between the time series data granularity and the performance of ML-based approaches.

ML models require training on historical data. Incorporating past values in the training set can enhance the models with additional knowledge. However, it can also introduce bias due to outdated trends in the series. To address this issue, the models are periodically retrained, albeit at the cost of additional computational time and memory. This exploration also aims at studying the impact of varying the frequency of ML retraining steps.

Apart from data-driven analyses, the outputs of classifiers are managed by dedicated platforms, specifically the trading systems. These systems, based on the allocated equity for stock trading in the capital budget and a list of trade recommendations from discretionary traders or data-driven approaches, execute trades by effectively managing the size and timing of the operations. To configure the trading system appropriately, the money management strategy needs to be aligned with the input signals [29]. In addition to the previous investigations, this work delves into setting up the most established aspects of the ML-based trading system, such as the stop loss value and the money management strategy.

Empirical analysis was conducted on the U.S. based NASDAQ-100 stock index. The performance of various ML pipelines and classification algorithms was compared at different time granularities. Additionally, the influence of key characteristics of the trading system on the profitability of the adopted strategy was analyzed. The results enabled the identification of specific configurations in which ML algorithms are particularly effective in supporting intraday stock trading.

2.1.1 Related works

In recent times, there have been significant advancements in incorporating machine learning (ML) approaches into quantitative trading systems. Various surveys, such as those conducted by [23], have categorized existing works based on the modeling techniques used, the considered features, and the evaluation metrics. [30] offers a comparison between machine learning and time series modeling, while [24] provide an overview of the latest deep learning models developed for financial applications.

[31] present a review that compares three popular methodologies: technical analyses, textual analyses, and high-frequency trading. [32] investigates the impact of investors' mood on the performance of algorithmic trading systems.

In contrast to [23, 24, 31, 32], the objective of this work is to explore a complementary aspect, specifically identifying the most suitable time granularity for ML-based stock trading. The relationship between the time granularity of price series and stock market predictions has been previously examined in [33]. However, their research focused on predicting stock trade volumes based on Yahoo! Finance User Browsing Behavior.

2.1.2 Methodology

This study focuses on a stock trading architecture that comprises the following components:

- *Data preparation*: This stage involves gathering historical stock-related price data, performing data cleaning to eliminate errors and inconsistencies, and transforming the raw data into a suitable format for conducting multivariate data analyses.
- *Machine Learning model training and application*: In this module, the collected data is analyzed to generate reliable trade recommendations. For example, it may provide recommendations like "Buy stock X now because the price is expected to rise in the next Y minutes". These recommendations are generated by classifiers that have been trained on historical data.
- *Trade and money management*: Given an initial equity, this module is responsible for executing trading positions in the stock market based on the ML-based trade recommendations. It handles the opening and closing of positions according to the generated recommendations while managing the allocation of funds.

Moving forward, a detailed explanation will be provided for each step involved in the pipeline of ML-based stock trading.

2.1.2.1 Data preparation

In conducting the empirical analyses, the historical price series data of the 100 stocks in the NASDAQ-100 index were employed. The data spanned over a one-year period, from October 24, 2018, to October 24, 2019, with samples taken every 5 minutes. The data was then aggregated into coarser intervals, ranging from 5 minutes to 1 day, to examine the impact of time granularity on trading performance.

Each stock was characterized by the Volume-Open-High-Low-Close raw data, treated as multivariate data streams represented by key-vector pairs. At each sampling time denoted as t, the price vector ($x_s^t = [x_{v(s)}^t, x_{o(s)}^t, x_{h(s)}^t, x_{l(s)}^t, x_{c(s)}^t]$) for a given stock series consisted of the volume, open, high, low, and close prices within the time interval $(t-1, t)$.

To ensure data quality for the machine learning (ML) models, data cleaning was performed to remove inconsistencies and errors. Additionally, values sampled during stock market closures were excluded as they were irrelevant for the research purposes.

To capture temporal correlations between past and current stock prices and learn predictive patterns, feature engineering was employed. In addition to considering the past series values themselves, various features from technical analysis [28] were incorporated. These features, which have shown relevance in forecasting short-term market movements in previous studies [23], included trend descriptors, price and volume momentum indicators, and volatility indices.

The considered features are as follows:

- *Price history*: recent past values of the stock price series.
- *Price trend*: summaries of the current series trends at various time spans, including Moving Average Convergence/Divergence, Aroon Oscillator, Average Directional Index, and the Difference between Positive Directional Index (DI+) and Negative Directional Index (DI-).
- *Price Momentum*: indicators of the rate of acceleration of the stock's price, such as the Percentage Price Oscillator, Relative Strength Index, Money Flow Index, True Strength Index, and Stochastic Oscillator.

- *Volatility*: expressions of the potential spread of the stock's prices, encompassing the Average True Range Percent and Chande Momentum Oscillator.
- *Volume Momentum*: measures of the rate of acceleration of the exchanged volume of a stock, including the Percentage Volume Oscillator, Force Index, Accumulation Distribution Line, and On Balance Volume.

To prevent bias in the subsequent ML training phase, all feature domains were normalized to a range of [-1, 1] using a min-max scaler [3].

A window-based data model was employed to organize the feature values extracted from the raw price series data. Each record in the model stored the feature values corresponding to the multivariate data at a fixed time point. For a given candidate stock s_z at time point t_k, the current and past values of the data features within a predefined sliding time window (SW) were considered. This window slid over the historical data used to train the ML model, allowing for the extraction of predictive patterns.

If the window size was sz, the k-th record included the feature values associated with multivariate data at t_k, t_{k-1},..., and so on, up to $t_{k-SW_{sz}+1}$. The time span (sp) between consecutive sampling times was fixed. Depending on the time span value (e.g., 15 minutes, 60 minutes, 120 minutes), the dataset consisted of sz samples at finer or coarser granularity levels. The primary objective of the work was to investigate the influence of time granularity (sz value) on the performance of the trading system.

Considering that the dataset included both the current and past sz values for each feature describing the time series, the dimensionality of the analyzed dataset was $F \times (sz+1)$, where F represented the cardinality of the feature set.

2.1.2.2 Machine Learning model training and application

Machine Learning models are trained on stock related historical data with the aim of learning predictive patterns. Trained models are then applied to current multivariate data for each individual stock to forecast the stock's future price direction.

Let t_c represent the current time and t_p denote the prediction time. The time gap between t_c and t_p is referred to as the *prediction horizon*. For simplicity, the prediction horizon is set to *sp*, indicating that the value to predict is the upcoming one in the stock price series.

In the training dataset specific to each stock, every record at an arbitrary time point t_k is labeled with the current stock direction. The label assigned to the record at t_k for stock s is determined as follows:

$$label(s,t_k) = \begin{cases} up & \text{if } \frac{x_{c(s)}^{t_{k+1}} - x_{c(s)}^{t_k}}{x_{c(s)}^{t_k}} \cdot 100 \geq upthr, \\ down & \text{if } \frac{x_{c(s)}^{t_{k+1}} - x_{c(s)}^{t_k}}{x_{c(s)}^{t_k}} \cdot 100 \leq downthr, \\ flat & \text{otherwise} \end{cases}$$

Here, $x_{c(s)}^{t_k}$ represents the closing price of stock s at time t_k, and $upthr$ and $downthr$ are analyst-provided thresholds. The label *up* is assigned when the relative closing price variation from t_k to the upcoming sampling time t_{k+1} exceeds $upthr$, $down$ when it falls below $downthr$, and *flat* otherwise.

To determine appropriate threshold values for the experimentation, the cumulative distribution of closing price variations between consecutive time points $(x_{c(s)}^{t_{k+1}} - x_{c(s)}^{t_k})$ in the training data is examined. Percentiles of the distributions are identified to ensure a balanced representation of the three classes: *up*, *down*, and *flat*. It is important to note that the distribution may vary across different stocks. However, for the majority of stocks and time granularities, setting the thresholds $downthr$ and $upthr$ based on the 30th and 70th percentiles of the distribution maintains a balanced representation among the classes, typically resulting in 30% *up*, 30% *down*, and 40% *flat* cases.

At any given current time t_c, a customized ML model is trained to forecast the price movements of a specific stock s. The model is trained on a dataset consisting of records corresponding to previous time points, such as t_{c-1}, t_{c-2}, ..., and so on. The trained model is then applied to the record at the prediction time t_p (typically, $t_p = t_{c+1}$).

Strategies for building the historical training dataset involve using various descriptors to describe stock-related price movements. The time span considered in constructing the dataset determines whether the model captures short-term, mediumterm, or long-term price trends. Including a larger time span incorporates older information into the model, influencing stock price forecasting.

Determining which portion of historical data is worth including in the training dataset can be challenging due to the high variability of the analyzed time series. To explore the effect of different data model settings, two complementary strategies are employed: the *hold-out approach* and the *expanding window approach* [3].

In both strategies, the data is divided in *training*, *validation* (used to fine-tune algorithm parameters), and *test* (used to backtest the model including the trading system).

In the *hold-out approach*, a fixed portion of historical multivariate data is used to predict future stock direction. The ML model is trained once per stock and then applied multiple times to forecast stock price directions on consecutive days. Conversely, in the *expanding window approach*, the training dataset is iteratively updated by including new multivariate data related to past days. For example, at time t_c, the training dataset includes records corresponding to times t_{c-1}, t_{c-2},.... At the next time point t_{c+1}, the training dataset is updated to include the new record corresponding to t_c (since all the relevant information is known). This approach requires training a new, updated model for each prediction time point, resulting in significantly higher computational effort.

The following established classification algorithms have been considered: (i) Decision Tree Classifier (DT), (ii) Support Vector Machines (SVMs), (iii) Multiple Layer Perceptron (MLP), and (iv) XGBoost (XGB) [34]. These models represent different linear and non-linear methods. Alternative models, including ARIMA and Naive Bayes classifiers, were considered but ultimately not employed due to their reliance on the data distribution for both setup and performance. The variance in data distribution across different granularities, as relevant to this study, would negatively affect the reliability of comparisons among diverse data granularities if these models were employed. Nevertheless, these models have been integrated in later studies, such as 2.2, where the focus is placed on a specific data granularity.

To adapt the algorithm configuration settings to the actual distribution of multivariate data, a grid search is performed on the validation set before applying the model in the trading simulations.

2.1.2.3 Trade and money management

At each time point t_c, the trading system updates the forecasts for individual stocks to provide automatically generated buy/sell orders for the stock market. Whenever the forecast predicts an upward movement in stock price, a long-selling trading signal is generated for the relevant stock s at the current closing price $x^{t_c}_{c(s)}$. Conversely, if the forecast predicts a downward movement, a short-selling signal is generated.

A trade remains open until one of the following conditions is met:

- The system receives an opposite trading signal from the machine learning (ML) model for the stock.
- The stock price moves in the wrong direction, reaching the predetermined maximum acceptable loss per trade, referred to as the Stop Loss value (SL).
- The trade reaches the *maximum duration*. Set to one day for intraday trades.

The determination of the cut-off threshold SL, required to enforce the second condition, depends from the selected time granularity. In order to automate the configuration of the trading system, SL is empirically established based on the observed distribution of closing price variations. Specifically, to eliminate excessive price fluctuations, SL is set as $SL = k \cdot \sigma$, where σ denotes the standard deviation of the distribution of percentage closing price variations, and k represents a non-negative scaling factor. An experimental analysis of the impact of k on trading system performance is part of the experiments.

To effectively manage the portfolio equity[1], several established money management strategies have been implemented:

- The *Fixed fractional strategy* which limits the amount of equity per trade[2] to a fixed fraction of the current overall equity.

[1]Portfolio equity refers to the total value of investments held by an individual or entity in various financial instruments such as stocks, bonds, and other assets. It reflects the overall worth of a portfolio at a specific point in time, considering the cumulative value of all assets and their performances.

[2]Equity per trade denotes the specific amount of funds allocated to an individual trade within the portfolio. It determines the size or proportion of the portfolio's value that is invested in a single transaction or trade.

- The *Fixed amount with max trades opened strategy* which restricts the number of concurrently opened trades and allocates a fixed percentage amount for each trade.
- The *Residual fixed fractional strategy* which restricts the amount of equity per trade to a fixed fraction of the currently available equity.

2.1.3 Experiments

An empirical evaluation was conducted on U.S. NASDAQ-100 stock market data to examine the utilization of multiple time granularities in machine learning-based stock trading. In addition to the main objective, other research questions were addressed by this work:

- What is the most capable classification algorithm? (See Section 2.1.4)
- Which training strategy is more effective? (See Section 2.1.5)
- What is the impact of the stop loss value and the money management strategy? (See Section 2.1.6)

Based on the outcomes achieved, Section 2.1.7 discusses the selection of time granularity that ensures optimal performance of the trading system.

The effectiveness of the expanding window strategy was tested by varying the training period from one month to 10 months. The resulting models were then validated on the subsequent month, and an automated trading session was simulated for the remaining time period.

In the experiments conducted using the hold-out strategy, the training data set consisted of the first 70% of the stock-related data, while the remaining 30% was used for validation and testing.

To evaluate the classification algorithms, the ability to correctly assign class labels was measured using average classifier accuracy [3]. Accuracy represents the percentage of correctly classified records, computed as the percentage of accurate stock price forecasts among all the predictions made. The weighted accuracy metric [34] was utilized to take into account class imbalance.

Furthermore, the profitability and robustness of the trading systems were assessed using the following metrics:

(i) *Overall return*: This metric quantifies the profits or losses generated by an investment over a specific time period, considering capital gains and losses, a fixed percentage transaction cost of 0.03%, and excluding dividends and interest incomes.

(ii) *Equity line*: A graphical representation of the variations in equity value over time.

(iii) *Max drawdown of the return*: This metric indicates the maximum observed loss from a peak to a dip of a portfolio before reaching a new peak.

Metrics (i) and (ii) indicate the profitability of the trading strategy, while metric (iii) serves as a quantitative risk indicator.

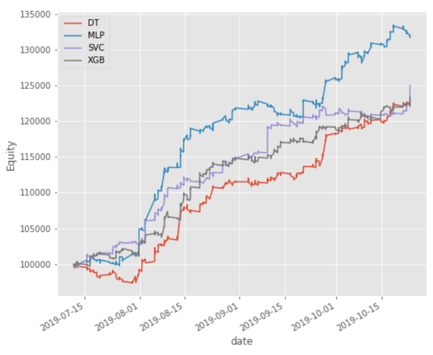
2.1.4 Comparison between different classification algorithms

As previously indicated, a comparison was made between multiple classification algorithms to determine the best-performing one across different levels of granularity. Specifically, four classification models were tested: Decision Tree, Support Vector Machine Classifier, Multiple Layer Perceptron, and XGBoost. All classifiers were implemented using the Scikit-learn library [34].

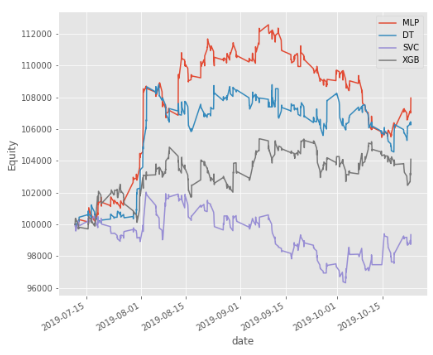
Given that varying data granularities present distinct challenges for classification, the grid search for identifying the best parameters for each method was incorporated directly into the pipeline. This enabled the model to adapt its parameters to each dataset effectively. The grid search encompassed the following value ranges:

- Support Vector Machine Classifier (SVC). *kernel* {Rbf, linear}, C [0.01, 1, 10, 100], γ: [0.01, 1, 10] (only for the linear kernel)
- Decision Tree (DT). *min_samples_split* [2, 10], *criterion*: {gini, entropy}, *max_depth*: [5, 20]
- MultiLayer Perceptron (MLP). *hidden_layer_sizes*: [(20,), (100,), (100, 100)], solver: lbfgs, max_iter: 200

To identify the optimal performing model, relying solely on conventional machine learning metrics like accuracy is inadequate. This limitation arises from



(a) 30-minute time granularity.



(b) 60-minute time granularity.

Fig. 2.1 Comparison between the equity lines achieved using different classification algorithms. Hold-out strategy. $SL = \sigma$. Max opened strategy.

inherent constraints within the financial sector, where the mere distinction between up-trends and down-trends falls short. For instance, predicting an up-trend when the actuality is a down-trend (or vice versa) is more detrimental than forecasting a stable trend. Furthermore, a classifier that accurately categorizes the majority of samples but concentrates errors on those samples featuring the most significant market fluctuations would yield unsatisfactory performance.

For this purpose, the performance evaluation conducted involved fixing to default values all other variables, such as the training strategy (Hold-out), the stop loss ($SL = \sigma$), and the trading strategy (Max opened). Subsequently, the equity lines of the models have been analyzed to determine the best-performing classifier.

Consistency in the results was observed across different time granularities, with the MLP classifier outperforming the others. Its proficiency in capturing non-linear series trends led to the highest overall return (+30% with 30-minute time granularity), highlighting its effectiveness.

Two representative time granularities were chosen to be plot, namely 30 minutes (Figure 2.1a) and 60 minutes (Figure 2.1b).

2.1.5 Effect of the ML training strategy

The selection of the training strategy has implications for both the profitability and complexity of the quantitative trading system.

Profitability overview Both strategies demonstrated notable returns with limited drawdown during the test periods, as indicated in Table 2.1. The expanding window approach using a 120-minute temporal aggregation exhibited the highest average effectiveness. The same granularity also yielded favorable results employing the hold-out strategy, while achieving a quite modest drawdown.

Table 2.1 Hold-out vs. Expanding Window. NASDAQ-100 index. Test period: 11/07/2019 to 23/10/2019.

Time	**Hold-out**		**Expanding window**	
Granularity	Over. Ret. (%)	Max. Drawdown (%)	Over. Ret. (%)	Max. Drawdown (%)
30min	35.23	-2.41	24.47	-1.72
60min	8.62	-4.60	20.98	-1.78
120min	19.91	-1.63	31.24	-1.67
1day	4.86	-5.11	3.47	-4.71

Complexity overview The complexity associated with the data-driven approach to stock trading primarily pertains to the training phase of machine learning (ML). This phase can be time-consuming. Among the various training algorithms, the most computationally intensive one is the Multilayer Perceptron (MLP).

In the hold-out scenario, training a model for a single stock using MLP takes a maximum of 0.45 seconds, considering 30-minute data. Thus, the overall time complexity of the training phase for all 100 stocks (45 seconds) is manageable.

In the expanding window scenario, instead, the training phase needs to be repeated at every trading step for each stock. The time complexity, with 30-minute granularity, varies from approximately 0.1 seconds per model when the expanding window size is relatively small (i.e., few historical price series samples) to around 0.55 seconds when dealing with larger time windows. On average, the training time is approximately 0.35 seconds per stock and time point which remains feasible in real trading contexts.

2.1.6 Influence of the trading system setup

The Stop Loss Value An example of the distribution of percentage closing price variations in the training data is depicted in Figure 2.2. The prices observed exhibited characteristics of Platykurtic normal distribution, i.e., normally distributed while presenting negative excess kurtosis. Consequently, the determination of the Stop

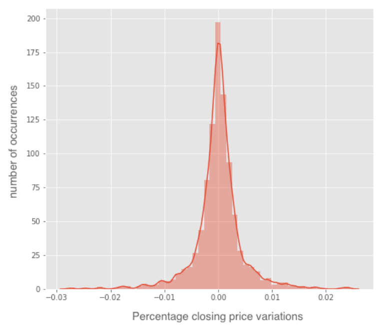


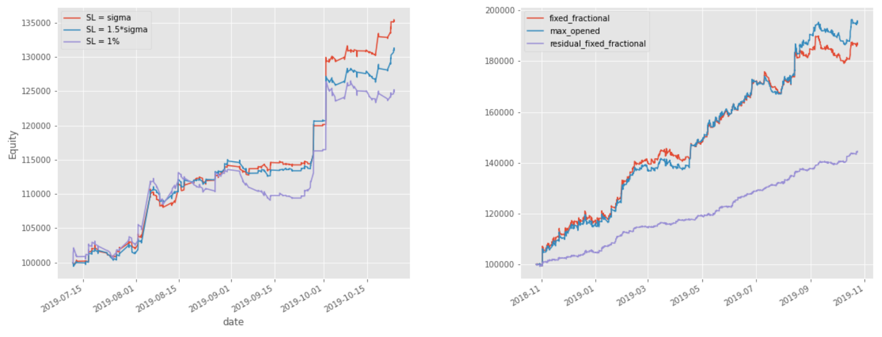
Fig. 2.2 Setup of the stop loss value. Distribution of the closing price variations.

Loss value (*SL*) involved considering scaling factor (*k*) values ranging from 1 to 1.5. The range of values [-*SL*, *SL*] encompassed approximately 90% of the observations. The outcomes, as shown in Figure 2.3a, indicate that setting *k* to 1 yields optimal trading performance.

The Money Management Strategy Figure 2.3b provides a comparison of equity lines obtained using different money management strategies. The experimental results revealed that the max opened strategy demonstrated slightly higher profitability compared to the fixed fractional strategy. However, it is important to note that the impact of this parameter is significantly influenced by the volatility of the underlying market. Therefore, adjusting the value of this parameter to suit the characteristics of the underlying stock exchange is recommended.

2.1.7 Comparison between different time granularities

In order to conduct a thorough analysis of the impact of time granularity on the performance of machine learning-based stock trading, the trading performance in the two considered training scenarios (i.e., hold-out and expanding window) is presented, respectively, in Tables 2.2 and 2.3. In the expanding window scenario, the most profitable time granularity observed was 120 minutes, demonstrating both a significant overall return (+148%) and a minimal maximum drawdown (-1.27%). In contrast, in the hold-out scenario, the highest overall returns (+35%) were obtained using 30-minute data samples. Nonetheless, the 120-minute granularity demonstrated competitiveness in both return (+20%) and especially drawdown (-1.63%).



(a) Comparison between different Stop Loss values.

(b) Comparison between money management strategies.

Fig. 2.3 Effect of the trading system setup: comparison between the equity lines achieved using different time granularities. SL=σ. Max opened strategy.

The influence of time granularity on the number of opened trades is illustrated in Table 2.4, focusing specifically on the hold-out strategy. As anticipated, finer granularity levels corresponded to a higher number of trades, as the machine learning-based process was triggered more frequently on average. Conversely, the percentage of winning trades (gains) increased as coarser granularity levels were used, as the models became less susceptible to biases caused by unpredictable small price variations. At the most profitable granularity level (120-minute data), the trading system opened 960 trades across 100 stocks and approximately 240 testing time points. Consequently, during active hours of the U.S. stock market, the trading system generated approximately 4 new trades every couple of hours.

Table 2.2 Trading simulation performance. Hold-out strategy. NASDAQ-100. Test period: from 2019-07-11 to 2019-10-23. *SL*=σ. Max opened strategy.

granularity	overall return (%)	max drawdown (%)	# trades per day
5min	-25.98	-26.48	132
10min	-13.08	-13.95	84
20min	-9.94	-14.37	44
30min	**35.23**	-2.41	18
60min	8.62	-4.599	16
120min	19.91	**-1.63**	9
1day	4.86	-5.11	6

Table 2.3 Trading simulation performance. Expanding Window strategy. NASDAQ-100. Test period: from 2018-12-11 to 2019-10-23. *SL*=σ. Max opened strategy.

granularity	overall return (%)	max drawdown (%)	# of trades per day
30min	86.96	-2.93	23
60min	83.47	-3.92	12
120min	**148.97**	**-1.27**	10
180min	130.61	-4.25	9
240min	45.12	-6.66	9
1day	4.93	-6.84	6

Table 2.4 Trading simulation statistics. Hold-out strategy. NASDAQ-100. Test period: from 2019-07-11 to 2019-10-23. *SL*=σ. Max opened strategy.

granularity	# trades	gains (%)	short-sellings	long-sellings
5min	13693	25.3	6507	7186
10min	13693	26.3	4747	2286
20min	4599	26.6	2627	1225
30min	1826	31.4	687	574
60min	1702	31.1	974	529
120min	960	34.5	538	331
1day	726	36.4	406	264

2.2 Shortlisting ML-based stock trading recommendations using candlestick pattern recognition

The second investigation aims at answering *Research Question 2* and solving the fourth open issue listed in the chapter opening. Specifically, whether incorporating standard trading rules based on technical indicators in data-driven systems can improve their performances. This section is based on the work presented in [35].

In traders' decision-making process, common factors driving their choices include (i) the application of *fundamental analysis*, which involves assessing a stock's intrinsic value by analyzing relevant economic and financial factors such as production, earnings, employment, housing, manufacturing, and management [36], (ii) the consideration of *news and social content analysis*, which entails monitoring news trends or opinions expressed by domain experts on platforms like online newspapers, blogs, and forums [31], or (iii) the leveraging of *technical analysis*, which involves analyzing the historical stock price series as it is believed to encompass all external influences [6]. This work specifically focuses on the latter.

A significant aspect of technical analysis pertains the identification of graphical patterns from Japanese candlestick charts [28]. Discretionary traders commonly rely on candlestick patterns to anticipate future stock price movements. To leverage domain-specific knowledge within data-driven approaches, there has been a growing research interest in integrating pattern recognition techniques applied to the candlestick chart with machine learning models utilizing stock-related data [37–39].

However, existing hybrid solutions suffer from two main limitations: (i) machine learning models often generate an excessive number of trading signals, resulting in a relatively high rate of false positives [23], (ii) models trained on a combination of candlestick patterns and stock price-related features are susceptible to the curse of dimensionality problem [40].

To address both of these issues, this study proposes a decoupling of the pattern recognition and machine learning stages to generate profitable trading signals. The underlying idea is to shortlist machine learning-based recommendations through pattern recognition, thereby reducing the number of generated trading signals to a more reliable and validated subset. To implement this methodology, the work presented explores various pattern-based filtering approaches and combines them with different machine learning models, including shallow and deep supervised models, as well as autoregressive techniques. Candlestick patterns are selectively employed to filter machine learning-based recommendations based on their technical characteristics and the confidence level of domain experts [28].

The developed trading systems have been subjected to back-testing using data from two stock market indices, namely the U.S. Standard & Poor 500 index (S&P500) and the Italian FTSE MIB40. The experiments evaluated the performance of the systems under different market conditions.

The results validate the effectiveness of pattern-based filtering in terms of return on investment and maximal drawdown. The former metric quantifies the profitability of the trading strategy compared to traditional approaches that do not incorporate pattern-based filtering, while the latter indicates the strategy's resilience during unfavorable market conditions.

2.2.1 Related works

Machine learning-based approaches Extensive research has been conducted on the prediction of next-day stock direction using machine learning techniques. Various models used for this prediction task include shallow classification models such as SVMs, Decision Trees, and distance-based classifiers (e.g., [11, 10]), Artificial Neural Networks (e.g., [13, 12]), Genetic Algorithms (e.g., [16, 15]), Rough

Cognitive Networks (e.g., [14]), associative models (e.g., [41]), frequent pattern mining (e.g., [17, 18]), ensemble methods (e.g., [20, 19]), and Deep Learning Architectures (e.g., [21, 22]). Systematic reviews of the related literature can be found in [23, 31, 24, 30]. Specifically, the review by [23] classifies the existing solutions based on model category, input features, and assessment strategy. [24] focus on Deep Learning solutions, whereas the overviews presented by [30] and [31] compare machine learning-based approaches with those relying on time series forecasting methods and textual analysis, respectively.

A parallel research effort has been dedicated to integrating fundamentals of technical analysis into the machine learning process. Technical analysis encompasses a wide range of graphical and statistics-based heuristics commonly used by discretionary traders to invest in the stock market [28]. The objective is to base traders' decisions solely on the analysis of historical stock price movements, assuming they incorporate all external market influences. Technical analysis aims to identify underlying price trends by detecting repetitive patterns in the raw series. For instance, [42] and [43], respectively combined feed-forward Neural Networks and K-Nearest neighbor classifiers with popular technical oscillators. [11] proposed driving Artificial Neural Network training with more complex technical rules based on traders' experience, while [44–46] focused on combining the use of technical analysis with meta-heuristic algorithms and reinforcement learning methods. A comprehensive overview of state-of-the-art works on technical analysis applied to stock trading was provided by [6].

Pattern recognition-based approaches The use of Japanese candlestick charts, which summarize opening, closing, maximum, and minimum prices of a given stock within predefined time bins, is widespread among discretionary traders for making stock trading decisions [47]. Traders, typically, manually identify predefined graphical patterns from candlestick charts, as their occurrence is believed to predict price direction and market turning points [48]. Several automated solutions have been proposed to assist traders in pattern recognition from candlestick charts. For example, [49] developed a candlestick chart interpreter capable of recognizing and classifying uptrend, downtrend, trend-continuation, and trend-reversal patterns. Similarly, [50–52] focused on recognizing patterns indicating strong trend continuation signals. [53] combined flag pattern recognition with multiple filtering strategies for monitoring stock price variations across different timeframes. [54] applied image processing techniques to extract texture features from candlestick charts, while [55] exploited dynamic time warping to generate real-time trading signals. To handle the uncertainty and vagueness of candlestick pattern matching, [56, 57, 48] employed fuzzy modeling, considering fuzzy rules as a means to incorporate human cognition into the recognition of technical patterns.

Approaches that combine machine learning with pattern recognition To leverage the strengths of both machine learning and pattern recognition, the research community has explored the joint use of these techniques for next-day stock price prediction. [58] proposed a gating network incorporating fuzzy logic, where candlestick pattern information serves as input. Similarly, [37] and [38] extract features derived from candlestick charts to train feed-forward Neural Network and Support Vector Machines classifiers, respectively. [39] integrated ordered fuzzy candlestick patterns into an auto-regressive time series forecasting model.

Previous attempts to combine pattern recognition with machine learning primarily focus on incorporating specific features to enhance the prediction model's accuracy. In contrast, this study focuses on applying machine learning models to a reduced subset of stocks identified through candlestick pattern recognition.

2.2.2 Methodology

In this section, the methodology employed to integrate candlestick pattern recognition with machine learning for the prediction of the next-day stock price direction is presented. The subsequent step involves a multi-class classification task, wherein a predictive model is learned from historical data to forecast whether the next-day stock price direction will exhibit an increase, decrease, or remain flat [3].

The principal stages of the devised methodology are outlined as follows:

- **Data acquisition and preparation**. This initial step involves obtaining and gathering historical price data for a user-defined set of stocks, potentially of significant scale. The collected data is then stored in separate input datasets for each stock, which are suitable for both pattern recognition and classification purposes.

Algorithm 1: Proposed methodology

```
Input  : $S$: set of stocks;
         $\mathscr{P}$: historical price series of stocks in $S$;
         $d$: target date;
         $w$: historical window size;
         $\mathscr{J}$: Japanese candlestick patterns;
         $R$: reliability constraint (strong, weak, none);
         $F$: freshness constraint (expressed in days);
         $A$: agreement strategy (conservative, speculative, mixed);
         $Al$: Machine Learning algorithm;

Result: $TR^{ML+PR}$: trading recommendations for stocks $s \in S$

1 foreach $s \in S$ do
      /* Data acquisition and preparation                           */
2     $D_s \leftarrow$ ExtractFeatures($\mathscr{P}$, $d$, $w$)
3     $D_s^l \leftarrow$ LabelData($D_s$, $\mathscr{P}$, $d$)
      /* Candlestick pattern recognition                            */
4     $TR^{PR} \leftarrow$ ExtractGraphicalPatterns($s$, $\mathscr{P}$, $\mathscr{J}$, $R$, $F$)
      /* Classification                                             */
5     $M_s \leftarrow$ MLTraining($D_s^l$, $Al$)
6     $TR^{ML} \leftarrow$ ApplyMLModel($M_s$, $\mathscr{P}$, $d+1$)
      /* Combine pattern recognition with Machine Learning          */
7     $TR^{PR+ML} \leftarrow$ Agreement($A$, $TR^{ML}$, $TR^{PR}$)
8 end
9 return $TR^{ML+PR}$: Trading recommendations (buy, sell) for each stock $s \in \mathscr{S}$
```

Algorithm 2: ExractGraphicalPatterns

```
Input  : $S$: set of stocks;
         $\mathscr{P}$: historical price series;
         $\mathscr{J}$: set of (expert-provided) Japanese candlestick patterns;
         $R$: reliability constraint (strong, weak, none);
         $F$: freshness constraint (expressed in days);

Result: $TR^{ML+PR}$: trading recommendations for stocks $s \in S$

/* Scan the candlestick pattern $\mathscr{J}$ and select those
   satisfying constraint $R$                                     */
1 $\mathscr{J}^r$ = SelectReliablePatterns($\mathscr{J}$, $R$)
2 foreach $s \in S$ do
      /* Extract Open-High-Low-Close price values from the
         stock price series in $\mathscr{P}$ within the time range
         satisfying $F$                                          */
3     $OCLH_s$ = ExtractRecentSeriesValues($\mathscr{P}$, $s$, $F$)
      /* Match patterns in $\mathscr{J}^r$ with price values in $OCLH_s$  */
4     $M_s$ = MatchValues($OCLH_s$, $\mathscr{J}^r$)
      /* Generate trading recommendations based on the
         observed patterns                                       */
5     $TR^{PR}_s \leftarrow$ GenerateTradingRecommendations($M_s$)
6 end
/* Combine trading recommendations for all the considered
   stocks                                                        */
7 $TR^{PR} \leftarrow \Uplus_{s \in S} TR^{PR}$
8 return $TR^{PR}$: Trading recommendations (buy, sell) for each stock $s \in \mathscr{S}$
    based on candlestick pattern recognition
```

- **Candlestick pattern recognition**. This module takes the daily candlestick chart representing the historical stock prices as input. It employs technical analysis fundamentals to generate per-stock direction recommendations, namely *uptrend*, *downtrend*, and *flat*.

- **Classification**. The focus of this step is to apply machine learning models to predict the next-day direction for each stock in the input datasets. Specifically, a three-class classification model is trained for each stock to forecast the probability of an *uptrend*, *downtrend*, or *flat*, trend. The generated machine learning predictions are then utilized to determine per-stock direction recommendations.

- **Trading system**. The recommendations produced by the pattern recognition and machine learning steps are collectively processed by a specialized stock trading system. This system generates daily trading signals, such as "Open a long-selling position on stock *s*", by combining the recommendations derived from pattern recognition and machine learning. The pattern-based filtering step considers domain-specific pattern properties, including the reliability of recognized candlestick patterns (referred to as *reliability*), the time gap between pattern recognition events and ML predictions (referred to as *freshness*), the agreement between machine learning predictions and candlestick patterns regarding the most likely next-day stock direction (referred to as *agreement*), and the integration of a money management strategy into the trading system.

The detailed steps of the presented methodology are provided in Algorithm 1, while a more comprehensive explanation is presented below.

2.2.2.1 Data acquisition and preparation

This step involves the acquisition and preparation of historical data relevant to supervised machine learning and candlestick pattern recognition for each individual stock.

Data acquisition The data is acquired for a predefined set of stocks $S = \{s_1, s_2, \dots, s_{|S|}\}$. These stocks are typically selected based on various criteria such as belonging to the same stock market index (e.g., the U.S. Standard & Poor 500 index), geographical area (e.g., South East markets), market sector (e.g., Technology sector), capitalization level (e.g., high capitalization stocks), or a combination of these factors. For the purpose of the experimental evaluation, two representative stock sets are considered: (i) the 500 stocks in the U.S. Standard & Poor 500 index and (ii) the 40 stocks in the Italian FTSE MIB40 index. These sets have different characteristics in terms of the number of stocks and their distribution across sectors, making them suitable for evaluation.

For each stock s_i ($1 \leq i \leq |S|$), the following daily statistics are acquired from historical data: Closing Price (CP), Opening Price (OP), Maximum price (MAXP), Minimum price (MINP), and Exchange Volume (EV). The acquisition process generates separate time series for each stock-related statistic. To ensure the robustness of the subsequent machine learning models, a sufficiently large historical dataset referred to as *history* (H) is acquired. It is important to note that these statistics are useful for both candlestick pattern recognition and machine learning.

Data modeling In this step, the data is transformed to make it suitable for subsequent analysis. The goal of this paper is to forecast the next-day direction (i.e., uptrend, downtrend, or flat) of the closing price series for a given stock. Let s_i be the stock under consideration and d be the day corresponding to the prediction target (i.e., the day after the current date $d-1$). We denote the series of closing prices for the considered stock on the $|H|$ days preceding d as $cp_{d-|H|}, cp_{d-|H|+1}, \ldots, cp_{d-1}$, where $|H|$ is the history size in days.

The closing price direction cpd_{d-j} ($1 \leq j \leq w$) on a specific day $d-j$ indicates the marked next-day price variation and is determined as follows:

$$cpd_{d-j} = \begin{cases} uptrend \text{ if } \frac{cp_{d-j}-cp_{d-j-1}}{cp_{d-j-1}} \geq 1\%, \\ downtrend \text{ if } \frac{cp_{d-j}-cp_{d-j-1}}{cp_{d-j-1}} \leq -1\%, \\ flat \text{ otherwise} \end{cases}$$

Similar definitions apply to the other series (OP, MAXP, MINP, EV). In the classical univariate forecasting problem, the prediction of the next value in a series depends only on the preceding values of the same series. Therefore, to predict the direction of the next closing price of a stock, only the history of past closing prices of the same stock is considered and the other information are disregarded.

However, to improve prediction models, it is more effective to correlate sample values across multiple time series. For instance, the next-day stock price direction is likely to be correlated with the exchange volumes in the preceding days. Therefore, incorporating information from multiple series in a data model is preferred for accurate predictions [59].

The structured data model This model is suitable for training popular classification models [3]. It consists of a separate relational dataset for each stock in S. Each dataset collects per-stock key statistics over the considered history, with each statistic represented as a separate feature in the dataset schema. Each record in the dataset corresponds to a different trading day within the history and contains the feature values for that particular day.

The dataset schema includes information about stock s_i on the current day $d-1$ as well as on the w preceding days. Here are the features in the dataset schema, where feature names are indicated in capital letters and feature values in small letters:

- The current and past closing price directions within the sliding window (CPD_{d-w}, CPD_{d-w+1}, ..., CPD_{d-1})
- The current and past opening price directions within the sliding window (OPD_{d-w}, OPD_{d-w+1}, ..., OPD_{d-1})
- The current and past maximum price directions within the sliding window ($MAXPD_{d-w}$, $MAXPD_{d-w+1}$, ..., $MAXPD_{d-1}$)
- The current and past minimum price directions within the sliding window ($MINPD_{d-w}$, $MINPD_{d-w+1}$, ..., $MINPD_{d-1}$)
- The current and past exchange volume directions within the sliding window (EVD_{d-w}, EVD_{d-w+1}, ..., EVD_{d-1})

Each record in the dataset has a class label, which represents the target for prediction. Let s_i be the stock under consideration, and let r_d be the record in the dataset D_i corresponding to day d. The label assigned to r_d is the direction of the next-day closing price (cpd_d) of stock s_i.

Candlestick pattern recognition is based on the matching of static rules. The results of the recognition step cannot be quantified since they do not rely on machine learning or advanced image processing. Candlestick pattern recognition tools, such as TA-lib (https://www.ta-lib.org/), search for the preselected patterns in the Open-High-Low-Close stock price series and provide a list of retrieved patterns.

The candlestick chart model Technical analysts commonly make investment decisions based on the analysis of the graphical representation of the Open-High-Low-Close stock price series, known as the Japanese candlestick chart [60]. The technical analysts' community has established a short list of patterns that are considered predictive of upcoming stock price directions [28].

In the Japanese candlestick chart, each candle represents the high, low, open, and close prices of the stock for a specific period. In this study, stock data is analyzed at a daily granularity, so each candle corresponds to a different date. Candles consist of a body, which is white-colored if the closing price is higher than the opening price and black-colored otherwise. They also have upper and lower shadows that indicate the daily price excursions above and below the body, respectively. The candle body represents the price range from the opening to the closing price, while the shadows represent the daily minimum and maximum prices if they fall outside the open-close price range.

An example of a candlestick chart is shown in Figure 2.4. It illustrates the daily price variations of an arbitrary stock from February 8, 2021, to February 20, 2021 (excluding February 13 and 14 due to the stock market closure). For instance, the candle corresponding to February 16, 2021, is a long black candle without shadows. This indicates that the closing price (7) was below the opening price (28), and the daily minimum and maximum prices are within the opening-close range. The subsequent black candle (corresponding to February 17, 2021) has an upper shadow, indicating that the maximum daily price (13) was above the opening price (10).

2.2.2.2 Pattern recognition

The trading decisions made by discretionary traders involve manual exploration of individual candlestick charts to inform their actions [48]. These traders commonly search for specific graphical patterns that are popular among technical analysts due to their perceived ability to predict the direction of stock prices for the following day and identify market turning points. While the predictive power of candlestick patterns

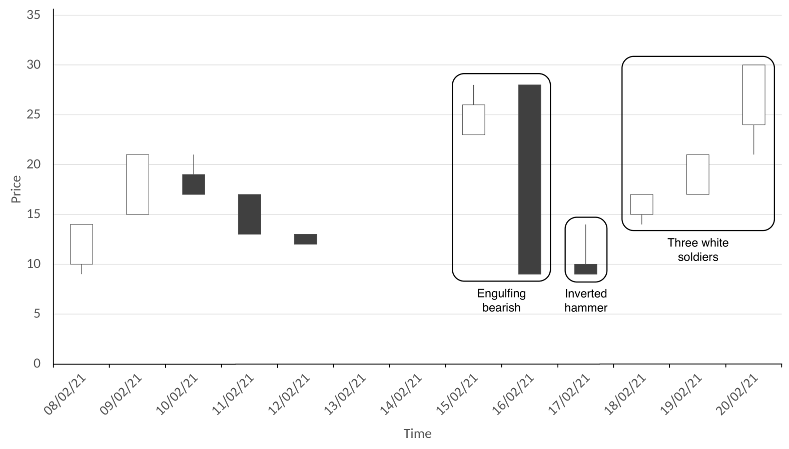


Fig. 2.4 Example of Japanese candlestick chart.

is a subject of debate [6, 61], they continue to be widely used for generating online stock trading signals. This motivates an investigation into incorporating candlestick patterns into machine learning-based trading systems, aiming to effectively and automatically combine data-driven models with domain-specific knowledge derived from past traders' experience.

Candlestick patterns consist of predefined sequences of candles, typically two or three in a row, each exhibiting specific characteristics. Figure 2.4 illustrates a candlestick chart with three highlighted patterns. The left pattern, known as the *engulfing bearish* pattern, suggests a likely downtrend in stock prices. It comprises a small white candle followed by a long black candle. Notably, the black candle's body fully overlaps the price range of the preceding white candle, indicating a clear dominance of sellers over buyers. The central pattern, referred to as the *inverted hammer* also signifies a downtrend and consists of a single black candle with a long upper shadow. The presence of an upper shadow implies that market participants initially tested the highest price of the day but ultimately rejected it. Lastly, the right pattern, named the *three white soldiers* indicates an uptrend and consists of three consecutive white candles. In this pattern, the highest price of each candle is lower than the subsequent one's lowest price, suggesting a prevalence of stock buyers over sellers.

According to the principles of technical analysis [28], candlestick patterns are most likely to have predictive power immediately after their occurrence, while their predictive ability diminishes over time. Additionally, based on the experience of past traders, certain patterns are considered more reliable than others. Technical analysts commonly classify candlestick patterns into categories based on their confidence level. It is important to note that this classification is widely accepted among the technical analysis community, except for rare exceptions. Table 2.5 presents a classification of candlestick patterns as either *lowly reliable*, *fairly reliable* or *highly reliable*.

To automate the recognition of candlestick patterns, the open-source Python-based Technical Analysis Library has been employed[3]. This library employs static rules to match a selection of technical patterns with input stock data, without employing machine learning-based inference or advanced image processing.

Algorithm 2 outlines the procedure used to extract graphical patterns from price series related to stocks. Firstly, we filter the list of expert-provided patterns based on the enforced reliability constraint (Line 1). Then, for each stock, we extract the Open-High-Low-Close price values that fall within an acceptable time range according to the freshness constraint. We subsequently search for a match between each considered pattern and the raw series values (Lines 2-4). Finally, for each stock, we generate a set of pattern-based recommendations, such as identifying an *uptrend* for stock s_i on day d or a *downtrend* for stock s_j on day d.

2.2.2.3 Classification

The objective of this step is to predict the future direction (i.e., *uptrend*, *downtrend* or *flat*) of each stock considered, utilizing machine learning techniques. Similar to the pattern recognition step, it generates a set of recommendations for each stock (e.g., *uptrend* of stock s_i on day d, *downtrend* of stock s_j on day d). It should be noted that machine learning-based recommendations may either agree or differ from those generated by the pattern recognition step.

For a given trading day $d-1$, this module trains an individual classifier for each stock $s_j \in S$ using the corresponding historical dataset. The per-stock classification models are then applied to forecast the direction of the closing price for the respective stock on the next day (d). The training and application of the classification models are repeated daily at the market's close to generate profitable stock recommendations for the following trading day.

[3]https://github.com/quantopian/ta-lib

Table 2.5 List of Japanese candlestick patterns.

Pattern (duration in candles)		
Low reliability	*Fair reliability*	*High reliability*
Doji (1)	BreakAway (5)	Closing Marubozu (1)
Gap Side Side White Three Crows (3)	Doji Star (2)	ConcealBabysWall (4)
Hammer (1)	Dragonfly Doji (1)	Counter Attack (2)
Hanging Man (1)	Engulfing (2)	Dark Cloud Cover (2)
Harami (2)	Gravestone Doji (1)	Evening Doji Star (3)
Harami Cross (2)	Hikkake (3)	Evening Star (3)
High Wave (1)	Hikkake Mod (3)	Falling Three Methods (3)
Inverted Hammer (1)	Homing Pigeon (2)	Identical Three Crows (3)
Long Leggend Doji (1)	Ladder Bottom (5)	In Neck (2)
Rickshaw Man (1)	Long Line (1)	Kicking By Length (2)
Separating Lines (2)	Matching Low (2)	Kicking (2)
Shooting Star (1)	On Neck (2)	Maruzobu (1)
ShortLine (1)	Piercing Line (2)	Mat Hold (5)
SpinningTop (1)	StalledPattern (3)	Morning Doji Star (3)
Takuri (1)	Stick Sandwich (3)	Morning Star (3)
Three Line Strike (3)	Tasuki Gap (3)	Rising Three Methods (3)
Thrusting (2)	Three River (3)	Three Black Crows (3)
	Three Stars in the South (3)	Three Inside Up (3)
	Three White Soldiers (3)	Three Outside Up (3)
	Tri-Star (3)	Upside Gap Two Crows (3)
	Two Crows (3)	
	Upside Gap Three Methods (3)	

To ensure up-to-date stock recommendations, the training data and classification models are incrementally updated using an expanding window approach [3]. Specifically, on each trading day $d-1$, the structured training dataset D_j associated with stock s_j is updated by including only the records that pertain to the current date (d) and the recent past days within the training window ($d-1$, $d-2$, …, $d-w$), following the procedure outlined in Section 2.2.2.1.

The current version of the classification module combines multiple classification algorithms [3], which are implemented in the Scikit-learn library [34]. Additionally, it incorporates an established Deep Learning architecture (Long Short-Term Memories) available in Keras[4], as well as two popular autoregressive methods, namely ARIMA and Exponential Smoothing, which are available in StatsModels[5]. It should be noted that neural network-based approaches require a sufficiently large amount of

[4]https://keras.io/
[5]https://www.statsmodels.org/

training data to avoid overfitting, in this study a minimum of 6 months of past data has been deemed suitable. The complete list of classification algorithms is provided below.

- Traditional classification models
  - Fully Connected Neural Networks: MultiLayer Perceptron (MLP)
  - Support Vector Machines: Support Vector Classifier (SVC)
  - Bayesian models: Gaussian Naive Bayes classifier (GNB)
  - Tree-based models: Decision Tree Classifier (DT)
  - Ensemble methods: Random Forest (RF)
- Deep Learning models: Long Short-Term Memories (LSTM)
- Autoregressive models
  - Autoregressive Integrated Moving Average (ARIMA)
  - Exponential Smoothing (ExpSmooth)

2.2.2.4 Trading systems

This step involves the combination of recommendations generated separately by candlestick pattern recognition and machine learning-based data-driven forecasts. The outcome is a condensed list of dependable trading signals, such as the decision to BUY stock s_i on day d or SELL stock s_j on day d.

On any given trading day, the following scenarios can occur for each stock: (i) no recommendations, (ii) a single recommendation from machine learning, (iii) one or more recommendations based on the recognition of candlestick patterns, or (iv) a combination of scenarios (ii) and (iii).

The trading systems designed prioritize the stock trading recommendations derived from machine learning, and the pattern recognition is employed to validate their reliability. In order to achieve this, we evaluate the stock trading recommendations produced by machine learning for each day, considering the following aspects:

1. The agreement between the recommendations made by machine learning and those produced by candlestick pattern recognition.

2. The reliability and freshness of the patterns used to generate stock recommendation from the candlestick chart.

3. The adopted money management strategy.

Agreement between machine learning and candlestick pattern recognition Regarding the agreement between machine learning and candlestick pattern recognition, the recommendations can either align or differ on a daily basis. In the former case, the reliability of the recommendation is strengthened, while in the latter case, the generation of trading signals should be avoided as the recommendation appears to be controversial.

In the proposed methodology, we rely on the following agreement strategies:

1. *Conservative approach.* follow the stock trading recommendation produced by machine learning if and only if there exist pattern-based recommendations and they are all in agreement with the machine learning-based one. Adopting this strategy entails opening a trade only when a machine learning recommendation is confirmed by candlestick pattern analysis.

2. *Speculative approach.* follow the stock trading recommendation produced by machine learning unless *all* pattern-based recommendations are opposed to it. Adopting the aforesaid strategy prevents trading systems from opening a stock trade when candlestick pattern analysis is strongly discordant.

3. *Mixed approach.* follow the stock trading recommendation produced by machine learning unless *at least one* pattern-based recommendation is opposed to it. Adopting the aforesaid strategy prevents trading systems from opening a stock trade even when candlestick pattern analysis is weakly discordant.

The conservative approach tends to generate fewer but more reliable signals, while the speculative approach tends to generate a larger number of signals while avoiding conflicting situations. The mixed approach falls between the two approaches mentioned above.

Reliability of the candlestick patterns The reliability of the candlestick patterns can be filtered based on their confidence level, which is determined by domain experts. The following reliability constraints can be enforced:

1. *Strong reliability constraint.* Consider only the pattern-based recommendations corresponding to highly reliable patterns.
2. *Weak reliability constraint.* Consider only the pattern-based recommendations corresponding to either highly or fairly reliable patterns.
3. *No reliability constraint.* Consider all the pattern-based recommendations independently of the reliability score of the corresponding pattern.

Freshness of the candlestick patterns The *freshness* of a candlestick pattern refers to the temporal proximity (in terms of days) between the pattern's end and the prediction date. For instance, in Figure 2.4, if the prediction date is 20/02/2021, the *three white soldiers* pattern has maximum freshness, indicating its high predictive power. Conversely, the freshness values for the *engulfing bearish* and *inverted hammer* patterns are 4 and 3, respectively.

To ensure that all pattern-based recommendations have reasonably high predictive power, a maximum freshness level constraint is imposed on the candlestick patterns. For example, setting the freshness level constraint to 0 means considering only patterns with the maximum freshness level, while disregarding others.

Money management Money management plays a strategic role in managing financial portfolios. To diversify investments across multiple assets, a limited portfolio amount is invested in each stock to protect the equity from significant losses. The trading system used in this work employs a fixed amount strategy [36], where the same percentage of the current equity is invested in each trade (e.g., a fixed 5% amount was used in the experiments). It is important to note that as the equity changes due to gains or losses, the absolute amount invested in each operation also changes. Additionally, the number of simultaneously opened operations is limited (20 in the simulations presented).

When faced with a potentially large set of stock recommendations, the trading system must decide which trading signals to generate each day. The decisions are based upon the following rules:

1. *Number of currently open trades.* If a multi-day trading strategy is applied, then part of the equity cannot be used to open new trades until the previously

opened trades have been closed. To reduce the negative effects due to market volatility, in case of abundance of trading signals, the system considers first the stocks with larger capitalization (which are known to be less subject to temporary price fluctuations).

2. *Weekly/monthly budget.* To prevent significant losses due to temporary market instability, a maximal budget can be allocated to stock investments within a restricted time period. In case the weekly or monthly cumulative loss generated by a trading system exceeds a predefined amount, the system does not consider any further recommendations within the same period.

3. *Risk/reward.* Opening a trading operation entails following a future trend prediction that is likely to happen. The risk of a single trade should be limited by setting a stop loss (1% in this work) and the expected reward should be higher than the risk.

The trading systems integrated in the proposed methodology operate as follows:

1. Open new long-selling trade (betting on a significant stock price increase within the considered day) for every stock for which a BUY signal is generated. The long-selling operation is opened at the opening price and closed at the closing price.

2. Open new short-selling trade (betting on a significant stock price decrease within the considered day) for every stock for which a SELL signal is generated. The short-selling operation is opened at the opening price and closed at the closing price.

Trading systems Various trading system variants have been implemented in the context of the study. For the sake of brevity, hereafter they will be denoted with the following notation: *AgreementType-ReliabilityConstraint-FreshnessLevel*. The *AgreementType* can be *Conservative*, *Speculative* or *Mixed*, depending on the agreement strategy used to combine machine learning and candlestick pattern recognition. Similarly, the *ReliabilityConstraint* can be *Strong*, *Weak* or *NoRel*, indicating the enforcement of a specific reliability constraint in the system. Finally, the *FreshnessLevel* can take values such as *Fresh0*, *Fresh1*, and so on, representing the enforced freshness level constraint.

For example, the trading system *Mixed-Strong-Fresh0* employs a mixed approach with a strong reliability constraint and a freshness level constraint set to zero. On the other hand, the system *Conservative-NoRel-Fresh2* is based on a conservative approach, without any reliability constraint, and a maximal freshness level constraint set to 2.

2.2.3 Experiments

An extensive empirical evaluation was conducted to assess the performance of trading systems on real datasets. In the subsequent sections, a selection of the most significant findings will be presented. The experiments encompassed several aspects, namely: a comparison of performance between various strategies for combining machine learning with pattern recognition, an examination of the influence of the pattern reliability constraint, an analysis of the impact of the pattern freshness constraint, a performance comparison of different machine learning algorithms, and an assessment of the execution time required by the proposed approach.

All the experiments were run on a machine equipped with Intel® Xeon® X5650, 32 GB of RAM and running Ubuntu 18.04.1 LTS

2.2.3.1 Stock-related data

Stock-related data was collected by utilizing the AlphaVantage service's APIs[6]. The analysis of trading system performance encompassed two distinct global stock indices, namely the american Standard & Poor 500 (S&P500) index and the italian FTSE MIB40 index. The S&P500 index consists of 500 stocks from various sectors, making it one of the largest stock indices. On the other hand, the FTSE MIB40 index is relatively smaller, comprising 40 stocks from diverse sectors. Notably, the trading volumes for S&P500 stocks are considerably higher than those of FTSE MIB40 stocks.

Considering the dynamic nature of markets, the system's performances have been analysed using market data obtained during different time periods. Specifically, separate analyses were conducted for three reference years characterized by distinct market conditions. The year 2011 was representative of a bearish market, while the

[6]www.alphavantage.co

year 2013 served as an example of a bullish market. Additionally, the year 2015 encompassed a combination of both bullish and bearish sub-periods. Subsequently, individual one-year datasets were examined, with each dataset corresponding to a specific market index and yearly period. To ensure brevity, the S&P500 datasets were assigned the labels S&P500 2011, S&P500 2013, and S&P500 2015, respectively. A similar naming convention was implemented for the FTSE MIB40 datasets.

Data exploration The distributions of various types of Japanese candlestick patterns in the analyzed datasets are examined in Table 2.6. The analysis focuses on the average pattern count per day and per stock, with the freshness level set to *Fresh0*. For instance, in the FTSE MIB40 2013 dataset, the average number of candlestick patterns (overall) per day and per stock is approximately 1.85. The pattern counts exhibit similar averages for the U.S. and Italian market indices.

Out of the available patterns, approximately 15% of them are considered highly reliable, 25% are moderately reliable, while the remaining patterns are deemed less reliable. As anticipated, the number of long-selling patterns exceeds the number of short-selling patterns, given the longer durations of uptrend market periods compared to downtrend periods.

Over a one-year timeframe, the average daily pattern counts for the entire market indices amount to 57 for the 40 Italian stocks and 80 for the 500 U.S. stocks.

Table 2.6 Average number of Japanese candlestick patterns per stock and trading day

	FTSE MIB40			**S&P500**		
	2011	**2013**	**2015**	**2011**	**2013**	**2015**
Long-selling	1.056	1.244	1.110	1.042	1.117	1.085
Short-selling	0.666	0.604	0.592	0.595	0.572	0.606
Low reliability	1.039	1.126	1.017	0.989	1.040	1.046
Fair reliability	0.423	0.446	0.422	0.415	0.422	0.420
High reliability	0.261	0.276	0.264	0.232	0.227	0.225
Overall	1.723	1.848	1.703	1.637	1.689	1.691

2.2.3.2 Algorithm settings

A grid search is conducted, individually for each algorithm, to fit the classification models to the analyzed data distributions. The grid search parameters and the corresponding value ranges are outlined below. Further information regarding the algorithm parameters can be found in [34].

- Support Vector Machines Classifier (SVC). *kernel* {Rbf, linear}, C [0.01, 1, 100], γ: [0.01, 10] (only for the linear kernel)
- Bayesian classifier (Gaussian Naive Bayes).
- Decision Tree (DT). *min_samples_split* [2, 10], *criterion*: {gini,entropy}, *max_depth*: [5, 20]
- MultiLayer Perceptron (MLP). *hidden_layer_sizes*: [(20,), (100,), (100, 100)], solver: lbfgs, max_iter: 200
- Ensemble method (Random Forest). *n_estimators*: [10, 150], *criterion*: {gini,entropy}, *max_depth*: [5, 20]
- ARIMA. p [1, 3], d [1, 2], q [1, 2]
- Exponential Smoothing. a: [0.1, 0.5], b: [None, 0.1, 0.5]
- LSTM[7]. optimizer='adam', loss='mse', activation='relu'

2.2.3.3 Backtest configuration

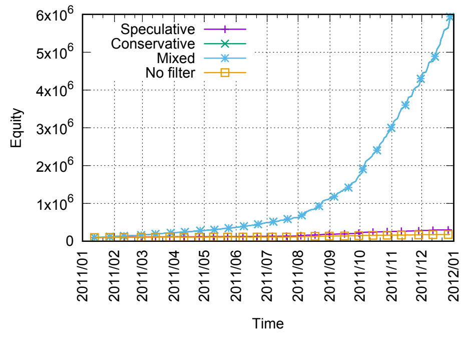
Trading sessions were simulated by conducting backtesting on the analyzed datasets [36]. The performance of the trading systems was evaluated from two complementary perspectives: (i) the profitability of the investment strategy over time, quantified as the *return of investment* (referred to as *return* for brevity), and (ii) the measurement of the largest decline in equity value from a peak to a trough, known as the *maximum drawdown*. This metric represents the volatility of the strategy. The primary objective of a trading strategy is to maximize the return of investment while minimizing the maximum drawdown.

To visually illustrate both of these performance measures, the equity line plot will be leveraged. This plot demonstrates the fluctuations in the overall equity value over time, assuming an initial virtual investment.

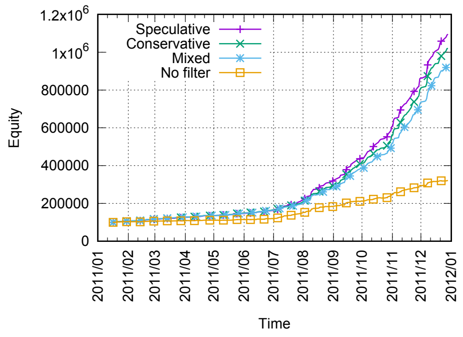
[7]Recommended Keras setup available provided by [62].

For the experiments of this study, certain assumptions were made: (i) an initial portfolio of 100,000 USD, (ii) a fixed transaction cost of 10 USD, (iii) a fixed 5% trade amount, (iv) a fixed stop loss of 1% (without trailing stop), and (v) no take profit or stop profit orders were considered.

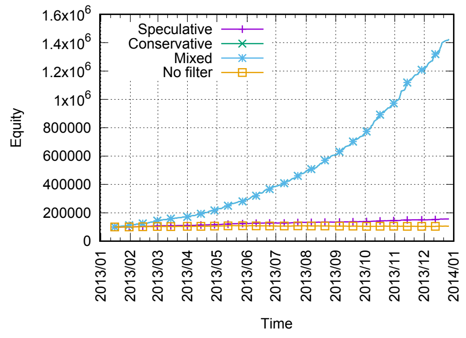
Fig. 2.5 Equity lines: Impact of the agreement strategy.



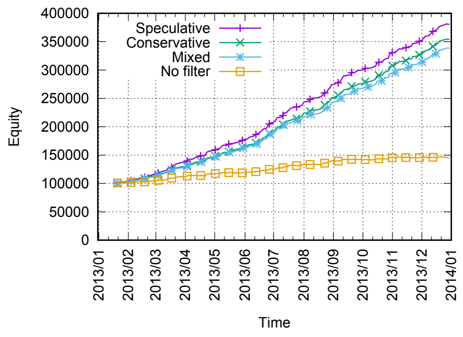
(a) S&P500 2011 dataset. Random Forest classifier, Reliability=Strong, FreshnessLevel=Fresh0.



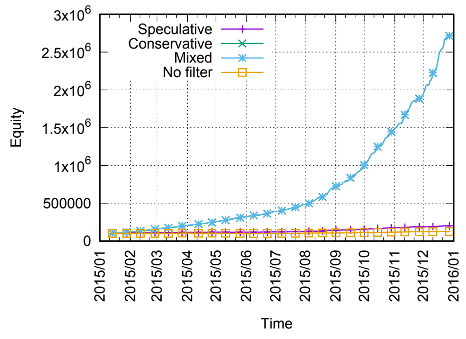
(b) FTSE MIB40 2011 dataset. Random Forest classifier, Reliability=NoRel, FreshnessLevel=Fresh0.



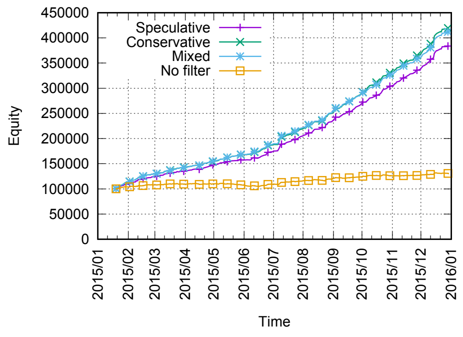
(c) S&P500 2013 dataset. Random Forest classifier, Reliability=Strong, FreshnessLevel=Fresh0.



(d) FTSE MIB40 2013 dataset. Random Forest classifier, Reliability=NoRel, FreshnessLevel=Fresh0.



(e) S&P500 2015 dataset. Random Forest classifier, Reliability=Strong, FreshnessLevel=Fresh0.



(f) FTSE MIB40 2015 dataset. Random Forest classifier, Reliability=NoRel, FreshnessLevel=Fresh0.

Table 2.7 Trading statistics: S&P500 2011 dataset. Random Forest classifier.

Configuration	**Avg num trades x day**	**Avg return x trade (%)**	**1Y max draw-down (%)**	**1Y max gain (%)**	**Num trades**	**Tot return (%)**
Mixed-Strong-Fresh0	15.2	**2.3**	**0.0**	**5840.0**	3662	**5840.0**
Conservative-Strong-Fresh0	15.2	**2.3**	**0.0**	5827.8	3663	5827.8
Mixed-Weak-Fresh0	19.9	1.7	-0.5	4556.8	4845	4556.8
Mixed-NoRel-Fresh0	**20.0**	1.5	**0.0**	2782.3	**4860**	2782.3
Conservative-Weak-Fresh0	**20.0**	1.4	-0.5	2396.5	4852	2396.5
Conservative-NoRel-Fresh0	**20.0**	1.2	-0.2	1305.5	**4860**	1305.5
Speculative-NoRel-Fresh0	**20.0**	1.0	-0.2	667.7	**4860**	667.7
Speculative-Weak-Fresh0	**20.0**	0.7	-0.4	275.4	**4860**	275.4
Speculative-Strong-Fresh0	**20.0**	0.6	-0.6	207.0	**4860**	207.0
No filter	**20.0**	0.4	-0.8	80.8	**4860**	80.8

2.2.3.4 Comparison between different strategies

The objective of this analysis is to empirically compare different approaches that combine pattern recognition with machine learning. The equity lines of the trading systems achieved through various agreement strategies are compared in Figure 2.5. These experiments specifically focus on the Random Forest classifier, with the reliability constraint set to *Strong* and the freshness level set to zero. The impact of the algorithm and these parameters will be discussed later.

All pattern-based strategies have outperformed the underlying market significantly, such as a yearly gross return of 207% achieved by the least performing

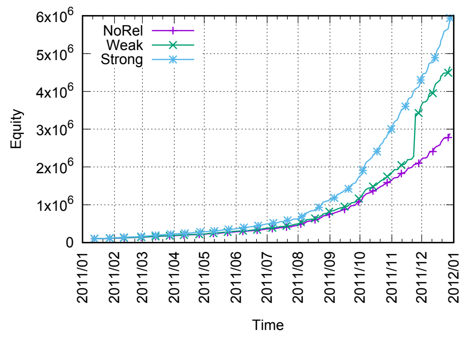
Table 2.8 Trading statistics: FTSE MIB40 2011 dataset. Random Forest classifier.

Configuration	**Avg num trades x day**	**Avg return x trade (%)**	**1Y max draw-down (%)**	**1Y max gain (%)**	**Num trades**	**Tot return (%)**
Speculative-NoRel-Fresh0	**14.79**	1.44	-0.38	**995.22**	3624	**995.22**
Conservative-NoRel-Fresh0	11	1.85	-0.12	920.15	2696	920.15
Mixed-NoRel-Fresh0	8.3	2.3	**-0.1**	829.8	2020	829.8
Speculative-Weak-Fresh0	17.2	1.0	-0.5	559.1	4220	559.0
Conservative-Weak-Fresh0	6.5	2.6	-0.2	534.8	1524	534.8
Mixed-Weak-Fresh0	5.8	2.8	-0.2	466.8	1339	466.8
Speculative-Strong-Fresh0	18.5	0.9	-0.5	431.0	4535	431.0
Conservative-Strong-Fresh0	4.0	**3.7**	**-0.1**	261.6	732	261.6
Mixed-Strong-Fresh0	4.0	**3.7**	**-0.1**	261.6	732	261.6
No filter	19.0	0.7	-0.7	221.0	**4659**	221.0

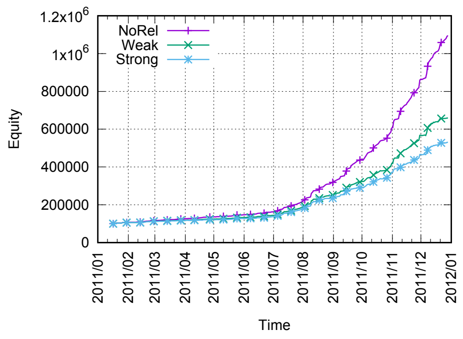
strategy on the S&P500 2011 dataset. Among the S&P500 datasets, the mixed strategy, which lies between a speculative and a conservative approach, performed the best, while the approach that disregards all recommendations from the pattern recognition modules performed the worst (5840% vs. 80%). These results demonstrate the benefits of utilizing candlestick pattern recognition to validate machine learning-based recommendations.

Conservative approaches exhibited approximately one order of magnitude better performance than speculative approaches on the S&P500 datasets, likely due to generating a significantly lower number of partly unreliable signals. For further

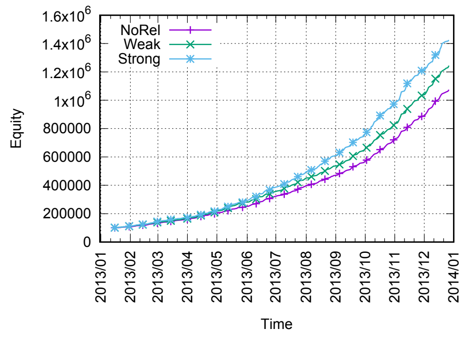
Fig. 2.6 Equity lines: Impact of pattern reliability constraint.



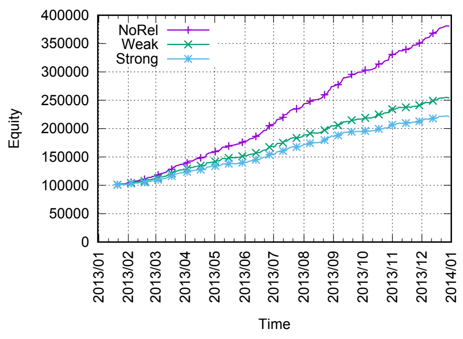
(a) S&P500 2011 dataset. Random Forest classifier, AgreementType=Mixed, FreshnessLevel=Fresh0.



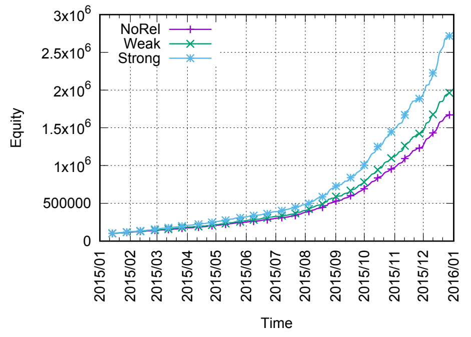
(b) FTSE MIB40 2011 dataset. Random Forest classifier, AgreementType=Speculative, FreshnessLevel=Fresh0.



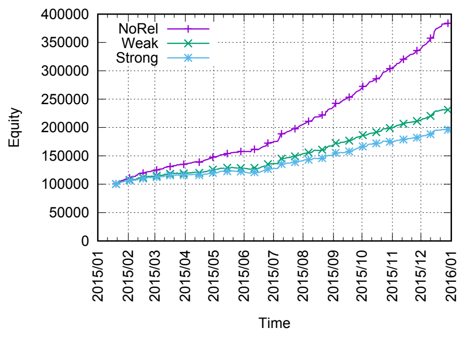
(c) S&P500 2013 dataset. Random Forest classifier, AgreementType=Mixed, FreshnessLevel=Fresh0.



(d) FTSE MIB40 2013 dataset. Random Forest classifier, AgreementType=Speculative, FreshnessLevel=Fresh0.



(e) S&P500 2015 dataset. Random Forest classifier, AgreementType=Mixed, FreshnessLevel=Fresh0.



(f) FTSE MIB40 2015 dataset. Random Forest classifier, AgreementType=Speculative, FreshnessLevel=Fresh0.

insights into the results, Tables 2.7 and 2.8 provide key statistics summarizing the trading sessions conducted on the S&P500 2011 and FTSE MIB40 2011 datasets, respectively.

When trading in the U.S. stock market, the speculative approach resulted in a notably higher number of daily trades. However, this negative effect was partially mitigated by the money management strategy, which limits the system to a maximum of 20 trades per day. Conversely, on the FTSE MIB40 datasets, speculative approaches slightly outperformed conservative ones and significantly outperformed mixed approaches. This opposing trend is likely attributed to the significantly smaller number of stocks (40 stocks in the FTSE MIB40 vs. 500 in the S&P500). It is worth noting that the number of trades per day on the FTSE MIB40 datasets is significantly lower than those on the S&P500 datasets (e.g., *Speculative-Strong-Fresh0* with 13.7 trades on FTSE MIB40 2015 compared to 20 trades on S&P500). As discussed later, reliability and freshness constraints had a considerable impact on the performance of the trading system when trading in the Italian stock market.

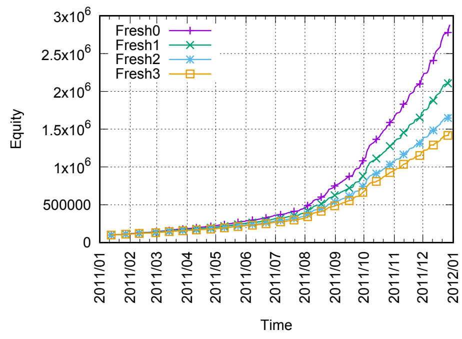
In summary, conservative or mixed strategies appear suitable for large market indices like the S&P500, as they prevent machine learning-based systems from generating unreliable signals. However, a speculative approach is still advisable for small market indices where conservative approaches fail to capture all profitable market opportunities.

2.2.3.5 Impact of the reliability constraint

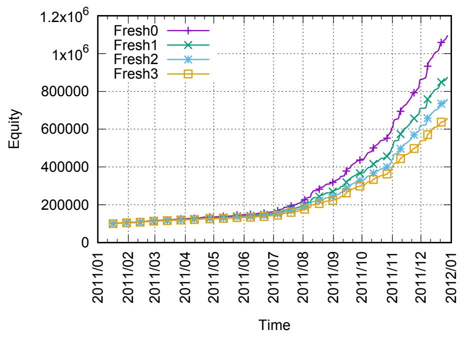
The effect of considering candlestick pattern reliability in the stock recommendation process was empirically analyzed. The equity lines of the trading systems, utilizing the Random Forest classifier and the best performing approach (mixed for S&P500 datasets, speculative for FTSE MIB40 datasets), were examined under different reliability constraints (strong, weak, and no constraint), as depicted in Figure 2.6.

In the case of S&P500 datasets, filtering patterns based on their reliability score appears to be advantageous. However, for FTSE MIB40 datasets, solely selecting weakly/strongly reliable patterns seems to be detrimental. This discrepancy can be attributed to the fact that the reliability constraint prevents the opening of unfavorable trades on the S&P500 datasets. On the other hand, due to the effect of the mixed strategy, most of the unreliable recommendations were already eliminated on the FTSE MIB40 datasets. Hence, enforcing a reliability constraint has a negligible impact in this scenario.

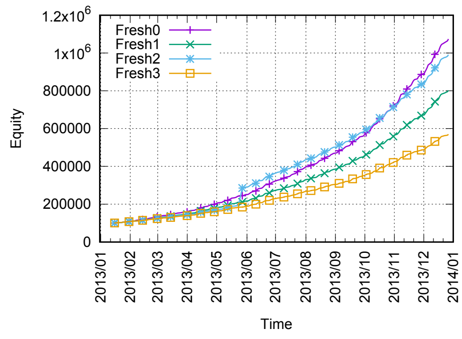
Fig. 2.7 Equity lines: Impact of the freshness level constraint.



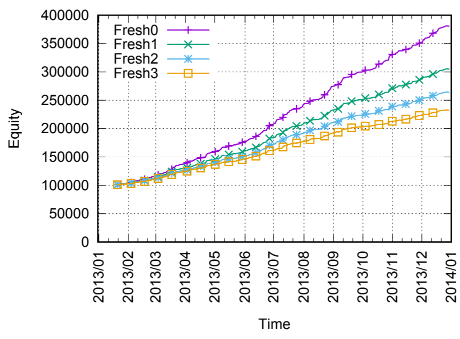
(a) S&P500 2011 dataset. Random Forest classifier, AgreementType=Mixed, Reliability=Strong.



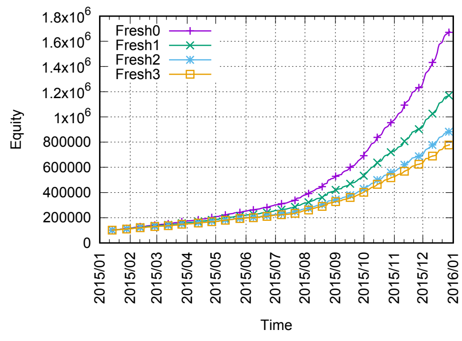
(b) FTSE MIB40 2011 dataset. Random Forest classifier, AgreementType=Speculative, Reliability=NoRel.



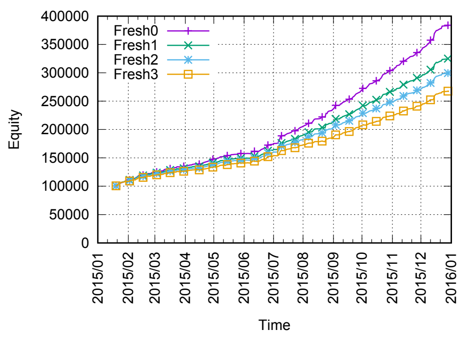
(c) S&P500 2013 dataset. Random Forest classifier, AgreementType=Mixed, Reliability=Strong.



(d) FTSE MIB40 2013 dataset. Random Forest classifier, AgreementType=Speculative, Reliability=NoRel.



(e) S&P500 2015 dataset. Random Forest classifier, AgreementType=Mixed, Reliability=Strong.

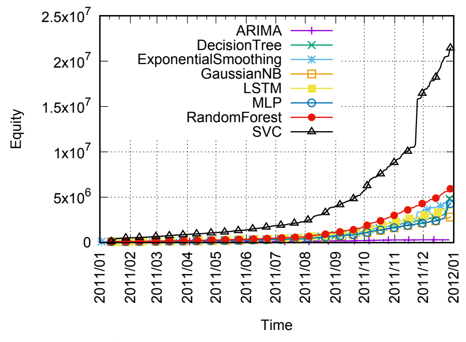


(f) FTSE MIB40 2015 dataset. Random Forest classifier, AgreementType=Speculative, Reliability=NoRel.

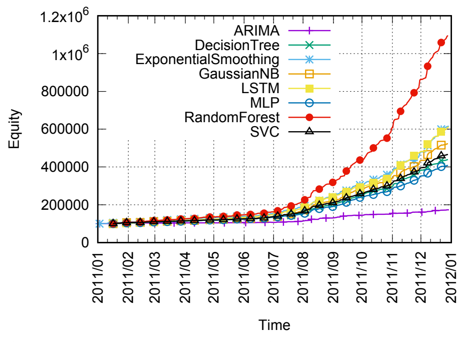
2.2.3.6 Impact of the freshness level constraint

The effect of pattern freshness on trading system performance was also investigated. The equity lines obtained by varying the freshness level constraint are presented

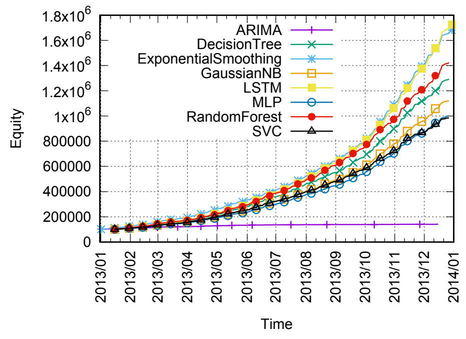
Fig. 2.8 Equity lines: comparison between classification algorithms.



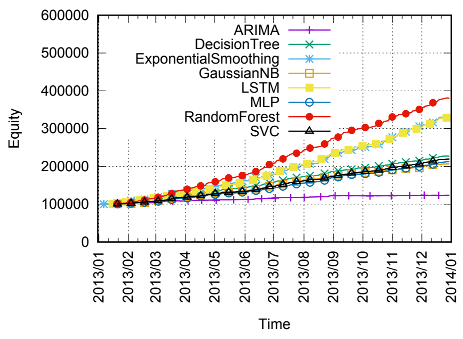
(a) S&P500 2011 dataset. AgreementType=Mixed, Reliability=Strong, FreshnessLevel=Fresh0.



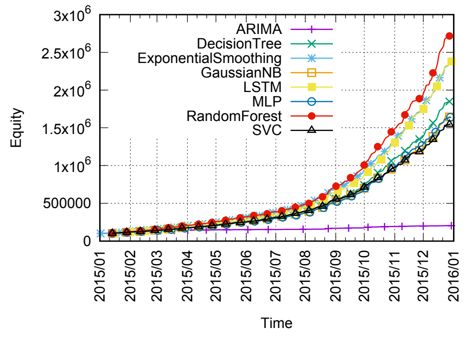
(b) FTSE MIB40 2011 dataset. AgreementType=Speculative, Reliability=NoRel, FreshnessLevel=Fresh0.



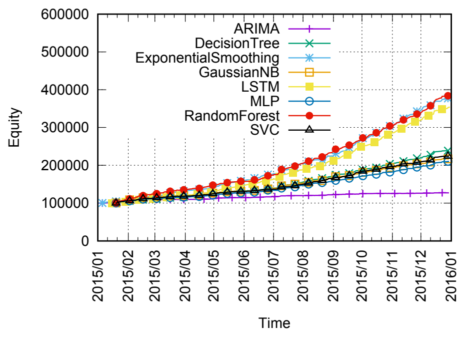
(c) S&P500 2013 dataset. AgreementType=Mixed, Reliability=Strong, FreshnessLevel=Fresh0.



(d) FTSE MIB40 2013 dataset. AgreementType=Speculative, Reliability=NoRel, FreshnessLevel=Fresh0.



(e) S&P500 2015 dataset. AgreementType=Mixed, Reliability=Strong, FreshnessLevel=Fresh0.



(f) FTSE MIB40 2015 dataset. AgreementType=Speculative, Reliability=NoRel, FreshnessLevel=Fresh0.

in Figure 2.7. Setting the freshness level to zero indicates considering only the patterns that end precisely on the prediction date. Across all conducted experiments, this specific configuration demonstrated the best performance. For instance, on the S&P500 2011 dataset, the gross return with a freshness level of zero was 3×10^6, whereas with a level of 3, it was 1.5×10^6. These results align entirely with the principles of technical analysis [28].

2.2.3.7 Comparison between classifiers

The performance of the system was compared by modifying the setup of the machine learning module. In Figure 2.8, the equity lines achieved using different algorithms are plotted, with each algorithm's best configuration being set separately for each dataset. With the exception of ARIMA and Exponential Smoothing, all lines exhibit relatively high overall return and limited drawdown across all tested datasets.

The performance of the autoregressive method displayed significant fluctuations depending on the analyzed market and underlying market conditions. In contrast, ensemble methods consistently demonstrated stable performance throughout all experiments. Specifically, Random Forest performed the best on four out of six datasets, while ranking second on the remaining datasets. Notably, on the FTSE MIB40 2011 dataset, its overall return surpassed all other methods by an order of magnitude.

2.2.3.8 Statistical significance tests

The statistical significance of performance gaps between the different trading system variants was validated using the Friedman test. This test is commonly employed for the statistical comparison of classifiers across multiple datasets [63].

The following procedure was applied for each classifier and market:

1. Separately for each dataset of the considered market, the system variants are sorted by decreasing value of the reference evaluation metric (i.e., the overall return (%)).
2. For each variant, its average ranking over all the considered market datasets is computed.
3. The observed differences between the average rankings of the considered variants are compared with the critical difference threshold CD that establishes whether the difference is statistically significant. By setting the significance level to 95%, the corresponding value of CD is 0.55.

Table 2.9 Average rankings and Friedman test (CD=0.55). S&P500 datasets.

	Mean rank based on 1Y total return (%)			
Configuration	**Random Forest**	**LSTM**	**Exponential Smoothing**	**Decision Tree**
Mixed-Strong-Fresh0	**1.3**	**1.3**	**1.3**	1.7
Conservative-Strong-Fresh0	1.7	1.7	1.7	**1.3**
Mixed-Weak-Fresh0	3.0	3.0	3.0	3.3
Mixed-NoRel-Fresh0	4.0	4.3	4.3	4.0
Conservative-Weak-Fresh0	5.0	4.7	4.7	4.7
Conservative-NoRel-Fresh0	6.0	6.0	6.0	6.0
Speculative-NoRel-Fresh0	7.0	7.0	7.0	7.0
Speculative-Weak-Fresh0	8.0	8.0	8.0	8.0
Speculative-Strong-Fresh0	9.0	9.0	9.0	9.0
No filter	10.0	10.0	10.0	10.0

	Mean rank based on 1Y total return (%)			
Configuration	**Gaussian NB**	**MLP**	**SVC**	**ARIMA**
Mixed-Strong-Fresh0	1.7	**1.7**	**1.3**	6.7
Conservative-Strong-Fresh0	**1.3**	2.0	1.7	6.3
Mixed-Weak-Fresh0	3.0	2.3	3.0	5.0
Mixed-NoRel-Fresh0	4.0	4.3	4.0	**1.0**
Conservative-Weak-Fresh0	5.0	4.7	5.0	4.0
Conservative-NoRel-Fresh0	6.0	6.0	6.0	2.0
Speculative-NoRel-Fresh0	7.0	7.0	7.0	3.0
Speculative-Weak-Fresh0	8.0	8.0	8.0	8.0
Speculative-Strong-Fresh0	9.0	9.0	9.0	9.0
No filter	10.0	10.0	10.0	10.0

Table 2.9 provides a summary of the results for the S&P500 datasets, with the one-year total return used as the reference evaluation metric. The method that achieved the best average mean rank value for each classifier is highlighted in bold. The majority of the differences between the mean ranks of the strategies were statistically significant at the 95% confidence level. It is worth noting that the *Mixed-Strong-Fresh0* variant performed significantly better than all other methods in terms of total return for six out of eight classification algorithms, except for ARIMA and Decision Tree.

Table 2.10 presents similar statistics obtained for the FTSE MIB40 datasets. The *Speculative-NoRel-Fresh0* variant performed the best, which confirms the findings discussed in the previous sections.

Table 2.10 Average rankings and Friedman test (CD=0.55). FTSE MIB40 datasets.

	Mean rank based on 1Y total return (%)			
Configuration	**Random Forest**	**LSTM**	**Exponential Smoothing**	**Decision Tree**
Speculative-NoRel-Fresh0	**1.7**	**1.0**	**1.3**	**1.0**
Conservative-NoRel-Fresh0	**1.7**	2.0	1.7	2.0
Mixed-NoRel-Fresh0	2.7	3.0	3.0	3.0
Speculative-Weak-Fresh0	4.7	5.0	5.3	4.7
Conservative-Weak-Fresh0	4.7	4.3	4.3	5.0
Mixed-Weak-Fresh0	5.7	5.7	5.3	6.0
Speculative-Strong-Fresh0	7.7	7.0	7.0	6.3
Conservative-Strong-Fresh0	7.7	8.3	8.0	8.7
Mixed-Strong-Fresh0	8.7	8.7	9.0	9.0
No filter	10.0	10.0	10.0	9.3

	Mean rank based on 1Y total return (%)			
Configuration	**Gaussian NB**	**MLP**	**SVC**	**ARIMA**
Speculative-NoRel-Fresh0	**1.7**	**1.7**	**1.7**	**1.3**
Conservative-NoRel-Fresh0	2.0	**1.7**	**1.7**	2.0
Mixed-NoRel-Fresh0	2.3	2.7	2.7	2.7
Speculative-Weak-Fresh0	4.7	5.0	4.7	6.7
Conservative-Weak-Fresh0	5.0	4.7	5.0	4.7
Mixed-Weak-Fresh0	6.3	6.0	6.3	5.7
Speculative-Strong-Fresh0	7.0	7.0	6.3	6.3
Conservative-Strong-Fresh0	8.0	8.7	8.7	7.7
Mixed-Strong-Fresh0	9.0	8.3	9.0	8.7
No filter	9.0	9.3	9.0	9.3

2.2.3.9 Execution times

The phase that demanded the most computational resources was the training of the classifiers. The average time required per trading day for learning and applying traditional classification models varied between 3s and 8s for the FTSE MIB40 datasets (40 stocks), and between 5s and 260s for the S&P500 datasets (500 stocks). In comparison, neural networks and autoregressive models took approximately ten times longer.

The time spent on pattern matching and combining stock recommendations was negligible compared to the time invested in classifier training. Training LSTMs required a different approach as it necessitated dedicated Graphical Processing Units (GPUs). However, the execution times of all the tested algorithms were deemed suitable for generating daily buy and sell signals for both the S&P500 and FTSE MIB40 market indices.

2.3 Leveraging the momentum effect in ML-based cryptocurrency trading

The conventional finance theory models necessitate investors to be risk-averse and capable of making rational decisions. The objective is to maximize profits by disregarding external influences and having complete access to all market information. Additionally, these models rely on an effective arbitrage mechanism, which plays a crucial role in determining security prices [64].

The arbitrage mechanism allows investors to profit from buying or selling the same asset at lower or higher prices, respectively. When an arbitrage opportunity arises, it is crucial for investors to promptly exploit it. This ensures that market prices quickly return to their equilibrium, preventing assets from being overvalued or undervalued for extended periods. However, these assumptions are partially unrealistic in the context of the cryptocurrency market [25].

These factors contribute to anomalous phenomena observed in cryptocurrency markets, such as the momentum and reversal effects. The momentum effect describes the positive correlation between prices, where rising asset prices tend to continue rising and falling prices tend to keep falling. Conversely, the reversal effect describes the negative correlation between prices, where assets that deviate from their fundamental values eventually revert back after a prolonged period. In this study, these effects are leveraged to design a profitable trading strategy for cryptocurrencies.

Recent research on cryptocurrency markets [65, 66] has empirically demonstrated the following:

- The behavior of cryptocurrency hourly returns differs on overreaction days compared to normal days.
- A momentum effect is observed on days with significant price changes (overreaction days).
- The momentum effect continues the day after an overreaction day.

These findings form the basis for new and profitable trading strategies for cryptocurrencies based on momentum indicators. For instance, [65] propose initiating a new trading position (buying or selling) on a cryptocurrency asset when the momentum level surpasses a user-specified threshold. However, the volatility of equities generated by this strategy tends to be excessive due to the challenges in identifying the overreaction days.

To mitigate the negative effects of highly volatile financial markets, machine learning techniques have proven to be more robust than traditional rule-based trading systems, particularly under challenging market conditions [23]. The primary reason is that machine learning models, through comprehensive data exploration, can tailor trading strategies to the observed market conditions.

This work aims at enhancing existing cryptocurrency trading models based on the momentum effect by leveraging machine learning techniques. Unlike previous approaches, the method presented combines market forecasting with supervised momentum analysis. Specifically, for each cryptocurrency, the model estimates the probability of being affected by the momentum effect on the next trading day, as well as the momentum direction. It initiates a new trading position for a cryptocurrency asset only when the machine learning model's output aligns with the observed momentum-based signal.

Experiments were conducted using historical data from three prominent cryptocurrencies to evaluate the effectiveness of a machine learning-based approach compared to a heuristic approach. The results of the machine learning-based approach outperformed the heuristic approach in terms of both the F1-score of the classification model and the return on investment from simulated trades. This suggests that machine learning is particularly effective in mitigating the negative effects associated with cryptocurrency assets.

2.3.1 Related works

2.3.1.1 Machine Learning-based approaches to cryptocurrency trading

Although machine learning has been widely used for stock market forecasting [23, 31, 24, 30], there have been limited efforts to adapt current trading systems for the cryptocurrency market. Specifically, [67] examined the application of traditional classification techniques like Support Vector Machines and Decision Trees to predict next-day cryptocurrency prices, while [68, 69] explored the use of Deep Learning techniques. Different data types, such as macro-financial indicators and blockchain information, have been investigated. Additionally, social media data, such as Twitter [70] or GitHub and Reddit [71], have been utilized to forecast cryptocurrency prices. Another research direction explores ensemble methods, including both standard shallow approaches like Random Forests and Stochastic Gradient Boosting Machine [72], and deep learning models as component learners [73].

[74] combined daily price data from 42 cryptocurrencies with key economic indicators from stock and Fiat markets to train a Gradient Boosting Decision Tree algorithm. However, these features are often noisy, making the inference process complex and lacking easy interpretability. To address this challenge, [75] proposed the use of an Attentive Memory module, which combines a Gated Recurrent Unit with a self-attention component to establish attentive memory for each input sequence. The results demonstrated that the raw sequence already contains most of the relevant information, while contextual information is less useful for building accurate predictive models.

In contrast to previous studies that focused on short-term price forecasting [68, 69, 75], this study introduces a new machine learning-based strategy specifically designed to predict the overreaction effect. It incorporates the market properties highlighted by recent empirical evidence [65] to leverage the predictive power of machine learning models on historical prices.

2.3.1.2 Applications of the momentum effect in the financial domain

The cryptocurrency market represents a relatively new and unexplored case of a highly volatile market, making it susceptible to overreactions. This is in contrast to more traditional markets like forex, commodities, and stocks. Recent studies [76, 77] have examined various aspects of the cryptocurrency market, including its efficiency, long-memory properties, price persistence [78], the existence of price bubbles [79], market competitiveness [80], price predictability [66], and the presence of anomalies [81].

The momentum effect in the cryptocurrency market is influenced by its similarity to emerging markets, characterized by low regulation, trading barriers, limited information availability, and asset complexity. As institutional investors often lack authorization or interest in participating in the cryptocurrency market, small investors are the main participants, accessing the market directly through wallets and exchanges. While exchanges have become more user-friendly, they still pose a significant hurdle for market access. Additionally, the nature of cryptocurrency as an asset is more complex to comprehend compared to traditional assets like stocks, commodities, and forex. Consequently, investors in the cryptocurrency market, often individuals, are drawn to its high volatility and speculative opportunities, making decisions driven more by common sentiment than rational pricing analysis.

Only a limited number of studies have specifically examined momentum and overreactions in the cryptocurrency market. For example, [82], utilizing a quantile autoregressive model, found that deeply negative returns are often followed by subsequent periods of negative returns, while abnormally positive weekly returns tend to be followed by price increases. These findings suggest that investors tend to overreact during days with negative returns and during weeks with positive sentiment and rising prices. Notably, no evidence of momentum was found at the monthly frequency, indicating that the cryptocurrency market exhibits faster momentum dynamics compared to traditional asset markets like stocks.

[65] have investigated price patterns following overreactions in the cryptocurrency market. They observed that after an overreaction, price movements are more significant than on normal days, suggesting the profitability of a trading strategy based on the momentum effect following overreactions. Empirical results revealed that hourly returns during days of positive or negative overreactions are significantly higher or lower, respectively, compared to average days. Furthermore, abnormal days can be identified before the end of the day, as the price trend tends to align with the direction of the overreaction until the day concludes. However, applying a reactive trading strategy may result in potential losses due to the generation of false trading signals.

To mitigate such negative effects, the objective of this study is to develop a machine learning-based system capable of identifying both positive and negative overreaction conditions in cryptocurrency price series.

2.3.2 Preliminaries

The overreaction detection recently proposed in [65] is a heuristic approach that examines hourly cryptocurrency price series. The method involves triggering trading operations (buy or sell) when the current price surpasses predefined thresholds.

The thresholds used in this heuristic method are computed based on the daily average return and standard deviation of the cryptocurrency. Daily returns ($\mathrm{R_i}$) are calculated using the formula:

$$R_i = (Close_i / Open_i - 1) * 100\%$$

where:

- $\mathrm{R_i}$: returns on the i-th day in %
- $\mathrm{Open_i}$: open price on the i-th day
- $\mathrm{Close_i}$: close price on the i-th day

The calculated returns are divided into two sets: positive returns and negative returns. This division allows for the separate determination of positive and negative thresholds for each trading day.

A trading day is classified as a positive overreaction day if:

$$R_i > (R_n + k * \sigma_n)$$

And it is classified as a negative overreaction day if:

$$R_i < (R_n - k * \sigma_n)$$

Where:

- $\mathrm{R_n}$: average daily returns for period n
- σ_n: standard deviation for period n
- k: number of standard deviations

The average daily return (R_n) and standard deviation (σ_n) for a period of n days are computed as follows:

$$R_n = \sum_{i=1}^{n} \frac{R_i}{n}$$

$$\sigma_n = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (R_i - R_n)^2}$$

The heuristic method compares the current price with the thresholds in order to assign the current trading day to one of three categories: positive overreaction, negative overreaction, or normal day. By monitoring the hourly price series, it detects if the asset price exceeds any of the two thresholds. If the price surpasses a threshold, the current day is labeled as an overreaction day. A prediction is considered correct if the daily closing price remains beyond the threshold level.

The trading system proposed in the study utilizes the momentum effect to open or close trading positions. Specifically, it opens a long-selling (short-selling) position when a positive (negative) overreaction is identified. Each trading position is closed at the end of the day, without the use of a stop loss, following an intraday trading scenario.

2.3.3 Methodology

A new machine learning-based approach is presented to address the limitations of the heuristic method, specifically the generation of inaccurate trading signals.

2.3.3.1 Problem statement

An indicator function, denoted as O_c, is defined for cryptocurrency c on trading day d_i. It takes the following values:

$$O_c(d_i) = \begin{cases} 1 \; d_i \text{ positive overreaction} \\ -1 \; d_i \text{ negative overreaction} \\ 0 \text{ normal day} \end{cases}$$

The goal is to model the relationship between the presence of an overreaction condition on the subsequent trading day d_{i+1} and the historical feature values $\mathscr{S}_c$ that describe cryptocurrency c on the current trading day d_i and preceding days d_{i-1}, d_{i-2}, ..., d_{i-W+1}. This is achieved by finding an arbitrary classification function f_c:

$$O_c(d_{i+1}) = f_c(\mathscr{S}_c(d_i), \mathscr{S}_c(d_{i-1}), \dots \mathscr{S}_c(d_{i-W+1}))$$

Here, $f_c(\cdot)$ represents the prediction function that we seek to discover, W denotes the size of the historical time window considered by the classification model, and $O_c(d_{i+1})$ corresponds to the value of the target variable.

2.3.3.2 Method proposed

The main steps of the proposed methodology are as follows:

1. **Data acquisition and preparation** In this step, historical data associated with one or more cryptocurrencies is collected, and feature engineering is conducted using well-known technical analysis indicators [28].
2. **Dataset labeling** During this second step, the dataset samples, which represent descriptions of cryptocurrency price series on specific trading days, are assigned labels using the indicator function O_c.
3. **Classification** In this last step, a classification model is trained and applied to predict the presence and direction of an overreaction condition on the next trading day.

Data acquisition and preparation A sufficiently large amount of historical prices is initially collected for the cryptocurrencies under consideration. In the experiments, historical data for three well-known cryptocurrencies, namely Bitcoin (BTC), Ethereum (ETH), and Litecoin (LTC), is retrieved. The data is obtained from Crypto Data Download[8], a service that collects cryptocurrency data from major crypto exchanges, specifically from the Kraken exchange. Each dataset for the three cryptocurrencies includes daily price and volume data. Each sample in the dataset is

[8]www.cryptodatadownload.com

Feature	Description	Category
Open	Open price of the current day	Candlestick
High	Highest price of the current day	
Low	Lowest price of the current day	
Close	Close price of the current day	
Volume	Trading volume of the current day	
SMA5-20	Relative difference between SMA(5) and SMA(20)	Trend
SMA8-15	Relative difference between SMA(8) and SMA(15)	
SMA20-50	Relative difference between SMA(20) and SMA(50)	
EMA5-20	Relative difference between EMA(5) and EMA(20)	
EMA8-15	Relative difference between EMA(8) and EMA(15)	
EMA20-50	Relative difference between EMA(20) and EMA(50)	
MACD	Moving Average Convergence/Divergence	
AO14	Aroon Oscillator (14 periods)	
ADX14	Average Directional Index (14 periods)	
WD14	Difference between Positive Directional Index (DI+) and Negative Directional Index (DI-) (14 periods)	
PPO12-26	Percentage Price Oscillator (12 and 26 periods)	Volatility
RSI14	Relative Strength Index (14 periods)	
MFI14	Money Flow Index (14 periods)	
TSI	True Strength Index	
SO14	Stochastic Oscillator (14 periods)	
CMO14	Chande Momentum Oscillator (14 periods)	
ATRP14	Average True Range Percentage: ratio between Average True Range and Close (14 periods)	
PVO12-26	Percentage Volume Oscillator (14 and 26 periods)	Volume
ADL	Accumulation Distribution Line	
OBV	On Balance Volume	
FI13	Force Index (13 periods)	
FI50	Force Index (50 periods)	

Table 2.11 Technical indicators and their corresponding category.

characterized by a timestamp, open, high, low, and close prices, as well as traded volume in both cryptocurrency and USD amounts.

Following the approach of [83], the daily price variations are described using a set of established technical indicators based on the work of [28]. These indicators serve to summarize the status and trends of the cryptocurrency, including trend direction, strength, and potential reversals due to overbought or oversold conditions. In line with [83], a total of 22 technical indicators and oscillators are considered in the experiments, as presented in Table 2.11.

Dataset labeling To exploit the momentum effect resulting from overreactions, the dataset samples are labeled using the indicator function described previously. Following the approach of [65], the presence of overreactions is leveraged to selectively initiate trading positions on the analyzed cryptocurrency.

To determine the values of k and W, the indications provided by [65] are followed. Different values of k are tested within the range of [0, 2], and W values are tested within the range of [50, 360]. It should be noted that setting k to 0 generates a high number of incorrect trading signals, while setting k to 2 significantly reduces the number of overreaction days. The best performance is achieved by setting k to 1. Additionally, the size of the historical window (W) cannot be lower than 50 to ensure the computation of the technical indicators. Setting larger values for W entails considering long-term price trends, which prove to be detrimental for daily cryptocurrency trading.

Classification The machine learning module is responsible for performing a 3-class classification task. The objective is to predict whether a positive overreaction, a negative overreaction, or a normal condition will occur. Generating the appropriate trading signal based on the prediction follows a similar approach to the rule-based model outlined in [65]

2.3.4 Experiments

The current study examines both the baseline approach suggested by [65] and various machine learning classifiers as alternative methods for generating trading decisions. To assess the performance of these approaches, extensive back-testing experiments are conducted.

2.3.4.1 Experimental environment

Experiments are conducted on a single-node setup within an HPC facility. The node utilizes Ubuntu 20.04.2 LTS and is equipped with an Intel(R) Xeon(R) Gold 6140 CPU @ 2.30GHz, which has 8 CPU threads, and 40 GB of RAM. Preliminary LSTM tests are performed on a NVIDIA V100 GPU with 16GB of VRAM.

The training time for each machine learning model is typically in the order of seconds, even in the worst cases. Therefore, it is not a limiting factor of the system proposed. Even the times required for running a grid search are sustainable (in the order of tens of minutes).

2.3.4.2 Experimental design

In the back-testing experiments, a series of steps are followed. Firstly, cryptocurrency data is gathered from online sources, encompassing both daily and minutely data. This data serves as the foundation for the analysis. Subsequently, the data is partitioned into training and back-test (evaluation) sets. The training set is employed to train machine learning models, enabling them to ascertain the relationship between the descriptive features and overreaction conditions. Finally, the back-test is conducted on the remaining days, during which the machine learning models utilize the acquired knowledge from the daily data to predict overreactions. It is important to note that, akin to the approach described in [65], the original heuristic method identifies overreaction conditions using hourly data prices, whereas the machine learning models make predictions based on daily data.

2.3.4.3 Models

In the experiments conducted, the original heuristic approach (HE) proposed by [65] remains unchanged. The baseline method is then extended by incorporating the machine learning-based approach introduced. As previously discussed, the problem is formulated as a three-class classification task aimed at predicting positive or negative overreaction scenarios, as well as neutral days. To tackle this task, well-known machine learning models are utilized [84]. Specifically, Support Vector Machine (SVM), Gaussian Naive-Bayes (GNB), Multinomial Naive-Bayes (MNB), K-Nearest Neighbors (KNN), Logistic Regressor (LG), Random Forest (RFC), and Feed-forward fully-connected Neural Network (MLP) models are tested. The implementations available in the SK-Learn library [85] are utilized for the traditional classifiers.

Additionally, a series of preliminary tests are conducted using the Deep Learning-based LSTM architecture provided by the PyTorch library [86]. However, due to the relatively new nature of cryptocurrencies as financial markets and the analysis focusing on daily cryptocurrency prices, the available data collection appears to be inadequate for training Deep Learning models.

2.3.4.4 Back-testing setup

The approach is evaluated on three different cryptocurrencies, namely Bitcoin (BTC), Ethereum (ETH), and Litecoin (LTC). The experimentation covers the time span from September 2, 2015, to December 31, 2020. The training data for machine learning classifiers consists of data from September 2, 2015, to December 31, 2019, while the remaining year is used for back-testing each model.

Hyper-parameter optimization is performed for each model using exhaustive grid-search. The configurations are validated using time-aware k-fold cross-validation, with the number of folds set to 5[9]. The best performing configuration, determined by the F1-measure weighted by class frequency, is selected.

To address class imbalance in the training sets, we explore over-sampling techniques such as SMOTE [87] and ADASYN [88]. However, no classification model benefits from over-sampling, except for LSTM. For LSTM, the loss is also rescaled by class frequency. Table 2.12 provides the details of the remaining hyperparameters tested during the validation process.

Regarding the heuristic method, a manual grid-search is conducted on two core parameters: the number of past days and the number of standard deviations used to compute the high and low thresholds. The number of past days is tested within the range of $[365, 50]$, while the number of standard deviations k is tested within the range $[0, 1, 2]$. The best results are obtained with a threshold definition period of 50 past days and 2 standard deviations.

2.3.4.5 Classification results

The classification performance in detecting one-day ahead overreaction conditions is presented in Tables 2.13-2.15. The F1 measure weighted by class frequency is reported, categorized by the number of standard deviations (k). The first row represents the results of the baseline heuristic approach proposed by [65].

[9]The scikit-learn Python implementation, specifically *sklearn.model_selection.TimeSeriesSplit*, is used.

Model	Hyper-parameter	Grid values
RFC	criterion	"gini", **"entropy"**
	min_samples_split	**0.01**, 0.05
	min_samples_leaf	**0.005**, 0.01
	max_depth	None, 5, **10**, 20
	class_weight	"balanced", **"balanced_subsample"**
KNN	weights	**"uniform"**, "distance"
	n_neighbors	**3**, 5, 7
	algorithm	**"ball_tree"**, "kd_tree"
MLP	hidden_layer_sizes	(10,), (30,), **(10, 10)**, (512), (512, 256)
	activation	"relu", "logistic", **"tanh"**
	solver	**"lbfgs"**, "sgd", "adam"
	learning_rate	**"constant"**, "invscaling"
	learning_rate_init	**2e-5**, 1e-4, 1e-5, 1e-2, 1e-1
	tol	1e-4, **1e-5**
SVC	kernel	"linear", **"poly"**, "rbf"
	degree	3, **4**, 5
	C	0.001, 0.01, **1**, 10, 50
MNB	alpha	0.01, 0.1, 1, **10**
LSTM	n_layers	**2**, 3, 4, 5
	bidirectional	True, **False**
	sequence length	3, 5, **7**, 10
LG	solver	"newton-cg", "lbfgs", **"liblinear"**, "sag", "saga"
	penalty	**"l1"**, "l2"
	C	1e-4, 1e-3, 1e-2, 1e-1, **1**, 10

Table 2.12 List of hyper-parameters used for validation. We report each hyper-parameter name and values using SK-learn notation. The best configuration is highlighted in bold.

The results clearly indicate that almost all machine learning methods outperform the heuristic approach under all circumstances. On average, the K-NN model demonstrates the best performance. However, there is no clear winner across all three cryptocurrencies.

In addition to the 3-class classification (*positive overreaction*, *negative overreaction*, *normal day*), the classification performance is also assessed using a binary class, where the distinction is made only between *overreaction* and *normal day*. As the 2-class classification problem is inherently simpler compared to the 3-class problem, it is expected to yield higher F1-measure scores. However, it is important to note that the binary classification outcome does not provide differentiation between positive and negative overreactions. Therefore, additional actions would be necessary to determine the direction of trading positions.

2.3.4.6 Trading results

Trading simulations were conducted to further assess the applicability of machine learning-based momentum detection methods. The heuristic-based trading strategy involved hourly price monitoring, where long (short) positions were opened when the price reached the positive (negative) threshold. In contrast, the machine learning-based trading strategy relied on the classifier's labels to open a long (short) position at the beginning of each trading day if the label was positive (negative). Both strategies closed their positions at the end of each trading day.

The results, presented in Tables 2.16-2.18, compared the performance of the heuristic-based strategy with the two best performing machine learning-based strategies in terms of percentage of profitable trades, total return, and average return per trade. The findings indicated that the machine learning-based strategies generated fewer positions compared to the heuristic strategy, but achieved a higher percentage of profitable trades. This demonstrates that the machine learning-based approach was more effective in reducing excessive signal generation. Furthermore, on average, the machine learning strategies exhibited higher total returns and average returns per trade. This was attributed to the ability of machine learning methods to open positions at the beginning of each overreaction day, without waiting for a match with the threshold level. As a result, they entered the market from more advantageous positions compared to the heuristic trading strategy. Among the machine learning models, K-NN showed potential by consistently delivering favorable results and exhibiting a reliable total return on BTC, which was found to be a less responsive asset for momentum effect strategies.

2.4 Findings and discussion

This chapter introduced three studies [27, 35, 89] that explore the application of machine learning (ML) techniques in the financial domain, specifically focusing

	# of labels	
Model	**3**	**2**
HE	*0.70*	*0.74*
GNB	0.67	0.71
KNN	**0.77**	**0.77**
LG	0.71	0.68
MLP	0.70	0.74
MNB	**0.77**	**0.77**
RFC	0.75	0.75
SVC	0.73	0.75

Table 2.13 BTC back-testing: F1 (weighted) scores for each classification model and number of classes. Best performers for each setup are reported in bold. Baseline results are italicized.

	# of labels	
Model	**3**	**2**
HE	*0.68*	*0.72*
GNB	**0.74**	**0.76**
KNN	0.69	0.70
LG	**0.74**	**0.76**
MLP	0.68	0.68
MNB	**0.74**	0.74
RFC	0.71	**0.76**
SVC	0.73	**0.76**

Table 2.14 ETH back-testing: F1 (weighted) scores for each classification model and number of classes. Best performers for each setup are reported in bold. Baseline results are italicized.

on short-term trading. These works approach the field from various angles, including determining optimal time granularities for fine-grained data, establishing the appropriate amount of data for model training, designing a trading system based on ML-generated signals, leveraging standard technical analysis patterns to enhance ML trading signals, and investigating the unique characteristics of cryptocurrencies as assets compared to stocks and which particular behaviors could be leveraged to improve trading results.

The first study introduced delved into the influence of time granularity on the performance of classification-based stock trading. By empirically evaluating trading systems at different time frequencies, optimal settings have been identified. Specifically, the use of MultiLayer Perceptron models trained on 120-minute data,

	# of labels	
Model	**3**	**2**
HE	*0.69*	*0.73*
GNB	0.73	0.77
KNN	0.77	**0.78**
LG	**0.78**	0.69
MLP	0.71	0.76
MNB	0.77	0.77
RFC	0.77	**0.78**
SVC	0.75	**0.78**

Table 2.15 LTC back-testing: F1 (weighted) scores for each classification model and number of classes. Best performers for each setup are reported in bold. Baseline results are italicized.

Model	Trades	Profitable trades	profitable trades %	total return	return per trade
HE	*40*	*19*	*47.50%*	*2.47%*	*0.06%*
KNN	11	6	**54.55%**	**8.63%**	**0.78%**
SVC	48	26	54.17%	3.61%	0.08%

Table 2.16 BTC trading results. Best performers are reported in boldface. Baseline results are italicized.

Model	trades	profitable trades	profitable trades %	total return	return per trade
HE	*84*	*45*	*53.57%*	*59.49%*	*0.71%*
KNN	52	33	63.46%	97.44%	1.87%
LG	30	27	**90.00%**	**126.23%**	**4.21%**

Table 2.17 ETH trading results. Best performers are reported in boldface. Baseline results are italicized.

Model	trades	profitable trades	profitable trades %	total return	return per trade
HE	*83*	*33*	*39.40%*	*14.46%*	*0.17%*
KNN	24	12	50.00%	36.36%	1.52%
SVC	39	24	**61.54%**	**68.13%**	**1.75%**

Table 2.18 LTC trading results. Best performers are reported in bold. Baseline results are italicized.

employing an expanding window train-test strategy, yielded significant improvements in overall return and maximum drawdown. The selected granularity strikes the best balance between the accuracy of predictions and the number of trades opened, providing superior results in terms of return and maximum drawdown. It is important to note that, while adjusting the granularity within reasonable bounds can be beneficial, extreme changes such as using granularities finer than 20 minutes or coarser than 240 minutes were found to be unsatisfactory in this research. These findings highlight the importance of selecting an appropriate time granularity that captures the essential dynamics of stock price movements. Even though the study revealed that simple forecasting methods did not yield satisfactory results when using daily data, obtaining fine-grained data is often challenging. Hence, the second study aims to enhance the performance of ML-based trading signal generation by incorporating established technical analysis techniques.

Building upon the insights gained from the first study, this research work introduced a novel approach that combined ML techniques with candlestick pattern recognition. By decoupling pattern recognition and ML steps and incorporating domain-specific knowledge from technical analysis, unreliable trading recommendations can be filtered. The results of the empirical evaluation revealed the importance of tailoring the trading strategy to the size of the market index. For large indices with a vast number of stocks, a conservative approach that adheres to both data-driven and technical analysis guidelines is recommended. In contrast, smaller market indices benefit from more speculative strategies that retain a higher number of trading signals capturing more profitable market opportunities than a conservative one. Furthermore, the study emphasized the significance of considering pattern reliability and freshness, particularly in complex market conditions, to mitigate the influence of false signals.

The third study explored the trading of cryptocurrencies and its difference with stock trading. Cryptocurerncies are a quite particular financial asset presenting unique characteristics. This works, specifically, study the application of ML techniques to enhance momentum-based cryptocurrency trading. By predicting overreaction conditions in cryptocurrency prices, the performance of heuristic approaches have bees outperformed, highlighting the potential of ML in this domain. Notably, K-Nearest Neighbors (KNN) emerged as the most effective classifier across various cryptocurrencies and settings. The study also emphasized the importance of the number of overreaction days available in the historical data, indicating that the benefits of ML are more pronounced when a sufficient number of these days is present.

By integrating the insights obtained from these three studies, a comprehensive understanding of the primary factors impacting the performance of ML-based trading systems can be gained. Time granularity emerges as a critical consideration, as it directly impacts the ability of models to capture the underlying dynamics of stock price movements. Additionally, incorporating specific knowledge about the underlying assets is consistently valuable. This knowledge can take the form of leveraging established technical analysis patterns or capitalizing on the unique behaviors exhibited by the assets themselves. Finally, the findings of the third study presented indicate that cryptocurrencies display distinct behaviors that set them apart from stocks, thereby presenting novel opportunities for successful ML-based trading strategies.

The findings of these studies pave the way for further research and development in the field of ML-based trading systems. Future work could explore advanced deep learning architectures, encompassing a broader range of historical data and incorporating additional sources of information such as news and social content. The extension of these studies to even more markets and assets could provide valuable insights into their applicability and performance. Moreover, considering the encouraging outcomes achieved through the integration of machine learning and technical analysis in stock trading, a potential avenue for striking a balance between heuristic approaches and ML techniques in momentum based cryptocurrencies trading could encompass the development of a hybrid approach combining heuristic rules based on technical analysis with machine learning methods.

In conclusion, the collective findings from these studies contribute to the understanding of the intricacies of ML-based trading systems. These insights have practical implications for traders and researchers alike, and they lay the groundwork for future advancements in the field of ML-based trading strategies.

Chapter 3

Portfolio Allocation Based on Time Series Analysis

The buy-and-hold strategy is widely employed for stock market investments. It involves purchasing stocks and holding them in the portfolio for an extended period, with the expectation of significant price appreciation [36]. This strategy stands in contrast to intraday and high-frequency trading, which focus on predicting short-term market movements.

Profitability and diversification of selected assets are key considerations in buy-and-hold investments. On one hand, the chosen stocks should yield high returns to compensate for investment risks. On the other hand, investments should be diversified across multiple assets to mitigate losses when certain assets depreciate [90]. A common approach to diversification is spreading investments across different sectors, assuming that stocks within the same industry sector are likely to be correlated due to shared economic influences [91].

3.1 Related works

The availability of market-related data over the past two decades has led to the development of machine learning-based trading systems for financial market investments (e.g., [92–94]). Many of these systems take stock diversification into account. For instance, researchers have used methods such as fractal dimension parameters [95], a combination of ordinary least squares and autoregressive models [96], subspace factorization [97], genetic algorithms [98, 99], and traditional clustering techniques [100, 94] to measure and achieve portfolio diversification. Clustering has proven to be promising in generating profitable yet diversified portfolios by identifying similar trends in stock-related data prior to advanced portfolio optimization models [101, 102].

Stock portfolio optimization is a process that involves allocating funds to a selected set of equities [103]. The traditional mean-variance model, introduced by [104], focuses on finding the optimal balance between investment returns and risk. Return is measured as the mean of historical stock prices, while risk is quantified as the variance. Various extensions to this model have been proposed in the literature, including incorporating advanced risk measures, limiting the number of selected stocks, and considering transaction costs [105].

Another commonly used approach based on the mean-variance model is the Capital Asset Pricing Model (CAPM), which is utilized for portfolio construction [106]. However, it has faced criticism for its empirical performance and simplifications that limit its practical application. In its classic Sharpe-Lintner version, the expected return on an asset is calculated by adding the risk-free interest rate to a risk premium, determined by the market beta of the asset multiplied by the premium per unit of beta risk.

Instead of seeking a portfolio that strictly meets specific conditions, portfolio optimizers can be integrated into financial Decision Support Systems, allowing end-users to customize their preferences, targets, and risk attitudes [107–109]. This work presents a decision support system that combines a hybrid strategy for portfolio selection.

Efficiency and estimation concerns have been the focus of attention within the research community. Linear and quadratic mixed-integer programming solvers often encounter challenges when dealing with a large number of securities. Additionally, allowing end-users to personalize the portfolio selection typically involves imposing specific constraints, which further complicates the optimization problem [110]. To tackle these NP-hard problems effectively, heuristic approaches are commonly adopted to narrow down the list of candidate stocks [111–113]. This work addresses the selection of a profitable subset of stocks for purchase, while introducing additional customized constraints on the candidate portfolio, following a similar approach to [111–113].

Researchers have also explored the combination of optimization strategies with data mining and machine learning techniques to heuristically select the most advantageous stocks for investment. Hybrid approaches employ machine learning techniques to forecast future stock prices and then identify stocks with higher expected returns for portfolio creation. For instance, [114] and [115] utilize an investment decision model to predict stock price movements and only stocks projected to achieve the expected return are considered in the Markowitz optimization model. [116] employs a genetic algorithm for the initial stage of selecting high-quality assets. This study proposes a hybrid approach that integrates established data mining techniques, such as itemset mining, with optimization techniques. Unlike [116, 114, 115], the proposed approach is completely unsupervised.

Other hybrid approaches adopt a two-stage process involving stock evaluation and scoring. For example, [117] evaluate individual stocks by predicting their returns in the next period and then compute a scoring function considering fundamental factors like net profit margin and cash flow ratio. Alternative strategies for stock evaluation and scoring include clustering methods to identify groups of similar stocks [118], genetic algorithms [119], and swarm intelligence methodologies [120–122] for portfolio optimization. Additionally, itemset mining is utilized to generate candidate portfolios satisfying global constraints on expected portfolio returns [123]. [123] employ a greedy selection approach for candidate portfolios based on user-specified constraints related to portfolio size and diversification level. In contrast to [117, 118], the approach presented includes not only single stock selection, but also global portfolio evaluation using a parallel itemset mining method. Unlike [123], in addition to the itemset mining phase, the best candidate portfolio are narrowed down using an adapted Markowitz logic that incorporates various additional constraints, including those based on fundamental analysis. Furthermore, a parallel itemset mining process is implemented to handle large sets of stocks effectively. Finally, [123] employs a standard taxonomy that is replaced by a data/driven one in this study.

3.2 Planning stock portfolios by means of weighted frequent itemsets

Frequent itemset mining is a well-established unsupervised technique utilized for discovering recurring item correlations from transactional data [124]. A frequent itemset refers to an arbitrary set of l items ($l \geq 1$) whose observed frequency of occurrence (support) surpasses a specified threshold. In the context of this work, itemsets represent diverse stock portfolios composed of l stocks.

Conventional itemset mining algorithms like Apriori [125] and FP-Growth [126] do not take into account the weights associated with the items found in each transaction. However, in this scenario, item weights indicate the percentage closing price variation relative to the preceding trading day.

Several algorithm extensions have been proposed to integrate item weights into the itemset mining process, as demonstrated in works such as [127, 128]. Simultaneously, significant efforts have been dedicated to parallelizing the extraction of frequent itemsets using the Hadoop-Spark framework, enabling scalability to handle large datasets [129].

[17] introduced DISPLAN, an unsupervised approach for generating diversified stock portfolios. DISPLAN employs a weighted itemset mining algorithm [128] to generate potentially profitable portfolios. It considers a minimum diversification level specified by domain experts and a taxonomy that categorizes stocks into sectors to prevent excessive exposure to specific sectors. DISPLAN recommends candidate portfolios that meet the diversification level and maximize expected returns.

The primary idea of DISPLAN is to filter out combinations whose average least return among the constituent stocks falls below a specified threshold.

This approach utilizes a weighted transactional dataset and a taxonomy describing relationships between the stocks of the market considered. The process involves the following steps:

- **Extracting portfolios**. All stock portfolios that meet certain criteria are extracted. These criteria include:
  - The average daily return of all stocks in the portfolio must exceed a specified minimum return threshold (*minret*).

  - The percentage of stocks from different categories (based on the taxonomy provided) should be above a specified diversified threshold (*mindiv*).

- **Selecting the optimal portfolio**. From the extracted portfolios, the methodology chooses the portfolio that maximizes both the average return and the number of contained stocks. This selection process aims to identify a portfolio that strikes a balance between high returns and diversification.

This approach present multiple limitations. First, the DISPLAN approach [17] relies solely on sector information for portfolio diversification, neglecting correlations in price series among stocks. This could be problematic as sector-based stock classification is conventional and potentially biased. Stocks within the same sector may exhibit variable and uncorrelated trends, while stocks from different sectors may display strong correlations with each other (see Section 3.3). Nonetheless, this model also demonstrates great potential and could be incorporated into a larger system to generate a subset of candidate portfolios for further in-depth analysis (see Section 3.4). Finally, due to its sequential implementation, the original form of this heuristic method is not suitable for handling a very large initial stock set (see Section 3.5).

3.3 Price Series Cross-Correlation Analysis to Enhance the Diversification of Itemset-based Stock Portfolios

The study presented in this Section, based on the work published in [130], addresses the first limitation of DISPLAN as identified in Section 3.2. It introduces an automated stock diversification method that overcomes the limitation identified by incorporating a scalable time series clustering step. The clustering process utilizes time series cross-correlation analysis [131] to generate tailored clustering outcomes for the candidate stock set. These outcomes are iteratively processed to automate the creation of an input taxonomy that captures the similarities and differences among stocks, enabling the computation of portfolio diversification.

Through empirical exploration of the initial clustering results, the study reveals the presence of unpredictable inter-sector stock correlations and intra-sector dissimilarities in stock price series. Additionally, the extended version of DISPLAN demonstrates significantly improved performance compared to its baseline version in terms of both average return and drawdown control.

3.3.1 Methodology

The methodology developed for this work aims at enhancing the effectiveness of the DISPLAN stock portfolio planner by employing time series correlation analysis. The methodology consists of the following steps:

- **Data acquisition and preparation**. In this step, the price series data associated with a large set of stocks is obtained, stored, and prepared for analysis.
- **Cross-correlation analysis**. This phase involves examining the cross-correlation among multiple price series to identify groups of stocks that exhibit similar temporal trends. Additionally, clustering techniques are applied to generate a taxonomy that categorizes the input stocks based on their time series cross-correlation analysis.
- **Itemset-based portfolio generation**. This step focuses on generating profitable and diversified stock portfolios using a state-of-the-art itemset-based strategy. The taxonomy derived from the previous step is utilized to guide the portfolio generation process, ensuring effective diversification of stocks.

By leveraging time series correlation analysis, this methodology aims to improve the performance and efficiency of DISPLAN in creating stock portfolios.

Data acquisition and preparation The daily historical prices of a large set of stocks $\mathscr{S}$ are obtained. In the subsequent analysis, the focus will be on the closing price series. These price series are represented in two ways: (i) a time series representation, where the closing prices cp_i^x and cp_i^y of stock pairs s_x and s_y at timestamp t_i are aligned to assess their relative displacement, and (ii) a transactional data representation, comprising a set of weighted transactions. Each transaction corresponds to a unique timestamp t_i and includes pairs $\langle s_x, cp_i^{s_x}\rangle$ for every stock in $\mathscr{S}$.

Cross-Correlation Price Series Analysis Homogeneous groups of time series are created based on their pairwise time-based similarity. To compare the time series, the price series are first normalized using z-score normalization. Then, a cross-correlation distance is employed to determine their similarity and capture series invariances.

Cross-correlation is a statistical measure commonly employed in signal and image processing to compare two time series point by point [132]. Given two N-length series TS_x and TS_y corresponding to stocks s_x and s_y, TS_y remains fixed while TS_x is slid over TS_y to compute their inner product for each shift of q steps ahead. The cross-correlation function $CC(TS_x, TS_y)$ produces a value for each shift of TS_x over TS_y. At step w, the cross-correlation takes the value $CC_w(TS_x, TS_y) = \sum_{i=1}^{k} cp_i^{s_x} \cdot cp_i^{s_y}$ for non-zero series values. Importantly, this measure is shift-invariant, enabling time series comparison even when the analyzed series contain gaps.

With a predefined number of clusters k, a scalable iterative refinement procedure [131] is applied to partition the input series based on their pairwise cross-correlation. It is worth noting that while the optimal value of k is not known in advance, a reasonable choice is to set k as the number of industrial sectors in the analyzed stocks, according to the classification provided by the Global Industry Classification Standard[1]. An empirical analysis of the impact of this parameter will be presented in Section 3.3.2.

To ensure convergence of the iterative procedure, the clustering process is repeated multiple times (with 100 iterations, as per the authors' recommendations). If two arbitrary stocks s_x and s_y are assigned to the same cluster in the majority of runs, it suggests a strong correlation between them. Consequently, the output of the clustering phase is represented by a co-occurrence matrix M of size $|S| \times |S|$, indicating the relative frequency of co-occurrences for each stock pair in the clustering outcomes. For example, if cell m_{xy} contains the value 0.9, it means that stocks s_x and s_y have been assigned to the same cluster 90% of the time. To convert the co-occurrence matrix M into a binary similarity matrix M^{01}, a discretization threshold t is applied:

$$\mathrm{m}_{xy}^{01} = \begin{cases} 1 \; m_{xy} \geq t \\ 0 \text{ otherwise} \end{cases}$$

[1]https://www.msci.com/gics

The taxonomy of stocks is then generated based on the similarity matrix M^{01}. Specifically, considering the arbitrary stocks s_x and s_y, the row vectors m_{x*} and m_{y*} in the similarity matrix are used. A single-linkage clustering algorithm [133] is employed to measure pairwise stock similarity using the cosine similarity between m_{x*} and m_{y*}. The agglomerative clustering process is stopped by enforcing a maximum cutoff threshold on the pairwise cluster similarity. Additionally, small clusters with fewer than 3 points are merged together using the traditional sector-based categorization.

Itemset-based portfolio generation To generate candidate stock portfolios, the methodology employs the DISPLAN itemset mining approach [17], detailed in Section 3.2, using the taxonomy produced for all diversification purposes.

3.3.2 Experimental Results

The effectiveness of the proposed methodology was evaluated empirically by simulating separate trading sessions on the Standard & Poor 500 (S&P 500) stock market across three diverse time periods. These periods include: (i) 2007-2009, characterized by bearish market conditions; (ii) 2011-2013, characterized by bullish market conditions; and (iii) 2014-2016, characterized by a mixed trend.

For each period, time series clustering was performed on the first two years of S&P 500 data, while the trading sessions were simulated over the last year. Specifically, DISPLAN-CrossCorr and its baseline version were backtested by training on the first half of the third year and testing on the second half. During the validation phase, the clustering parameters (k, t, p) were varied, i.e., k from 5 to 15, t from 70% to 85%, p from 60% to 70%, along with the parameters of the portfolio generator, i.e., the minimum support threshold from 1% to 6% and the diversification threshold from 50% to 80%.

To test the performance of the trading systems with varying stock compositions, they were separately backtested on two subsets of S&P 500 stocks: (i) the NASDAQ-100 index, excluding financial sector stocks, and (ii) the DOW JONES index, consisting of the 30 largest companies listed on US stock exchanges.

In the trading simulation, it was assumed that the trading system opened fixed-fractional trades on all selected portfolio stocks, without stop loss or stop profit limits.

Trades were initiated at the opening price of the first day of the test period and closed at the closing price of the last day. Per-trade transaction costs were approximated at 0.15%, and the trading performance was analyzed in terms of average percentage return/loss in the test period (compared to the beginning) and maximum drawdown (the highest potential loss) in the test period, which measures the investment risk.

3.3.2.1 Cluster Exploration

The outcomes of the time series clustering conducted on the bullish two-year period 2011-2012 are presented in Table 3.1. The table displays the number of stocks in each cluster, the stock similarity as a percentage, and the sector to which each stock belongs according to the Global Industry Classification Standard[2]. The standard assigns stocks to specific industries (level 2), industry groups (level 3), and sectors (level 4).

The clustering process, which relied on cross-correlation, enabled the identification of groups of stocks exhibiting significant temporal correlation. Several patterns were identified as a result.

Inter-sector stock similarity This particular pattern reveals groups of stocks from different sectors that display unexpected similar trends based on the clustering results. For instance, cluster 9 primarily consists of stocks from the *Energy* sector, with 25 out of 27 stocks belonging to this sector. These selected stocks are further categorized into the *Energy equipment and services* industry (6 stocks) and the *Oil, gas, and consumable fuels* industry (19 stocks). The remaining 2 stocks in cluster 9 are classified under the *Materials* sector, specifically in the *Metal and Mining* industry, which is closely related to the energy sector. Similar analyses were conducted on data from periods other than 2011-2012, resulting in clusters resembling the ones described earlier. However, in most of the other periods, the cluster also include two additional stocks: NRG Energy, Inc. and FLR, Inc. Although formally categorized as *Utilities* and *Industrial* respectively, their business operations are strongly reliant on energy assets.

[2]https://www.msci.com/gics

Intra-sector stock dissimilarity This pattern highlights groups of stocks from the same sector that are assigned to different clusters. The automatic separation of stocks from the same sector presents an interesting area for investigation by domain experts. For instance, cluster 3 consists of 16 stocks from the *Industrial* sector, all belonging to the *Capital goods* industry group. However, these stocks are associated with different industries within the sector, such as 2 stocks from *Construction & Engineering*, 4 stocks from *Electrical Equipment*, 2 stocks from *Industrial Conglomerates*, and 8 stocks from *Machinery*. Similarly, cluster 6 includes 3 additional stocks from the *Industrial* sector, however these stocks belong to a distinct industry group (i.e., *Transportation*). These clustering outcomes reveal a more detailed categorization of stocks, identifying cross-correlations between subsets of industries within the same sector while capturing their time-related differences. Based on the automatically inferred taxonomy, industries within the same sector can be considered different by the proposed approach and included in the same portfolio. Since these industries exhibit different temporal behaviors, including them in the same portfolio enhances diversification.

Clusters 6 and 14 group a subset of stocks again from the *Industrial* sector. In particular, cluster 6 contains 3 stocks from the *Airlines* industry (industry group *Transportation*), while cluster 14 comprises 3 stocks from another industry group (i.e., *Capital goods*), all belonging to the *Aerospace and defense* industry. Hence, the clustering process refines the conventional classification by capturing temporal correlations among stock price series.

Comparison with standard sector-based categorization The clusters generated in this study were compared to the standard GICS sector-based stock categorization using the metrics of homogeneity and completeness [133]. The objective was to determine the extent to which time series clustering produces different stock categories compared to the standard model. The sector-based categorization was used as the reference model, and the clustering outcome was empirically compared to it. Homogeneity refers to the clustering outcome being entirely composed of stocks from the same sector within each cluster, while completeness indicates that all stocks from the same sector are grouped together in a single cluster. The generated clusters demonstrated high homogeneity (e.g., 97% for the outcomes reported in Table 3.1), but the completeness was moderate (76%). This suggests that the clusters generally align with the given sector-based categorization, although there are significant

Table 3.1 Example of clustering outcome: Stock distribution over industrial sectors. Period 2011-2012.

Cluster Id	Stocks (#)	Similarity (%)	SECTORS										
			S01	S02	S03	S04	S05	S06	S07	S08	S09	S10	S11
1	19	94.1	0	0	0	0	15	2	0	0	2	0	0
2	24	99.8	0	0	0	0	0	0	0	0	0	0	24
3	22	92.3	0	5	16	1	0	0	0	0	0	0	0
4	31	94.7	0	0	0	0	0	0	0	31	0	0	0
5	26	99.6	0	0	0	0	0	0	0	0	0	26	0
6	3	99.3	0	0	3	0	0	0	0	0	0	0	0
7	15	96.5	0	0	0	0	0	0	15	0	0	0	0
8	25	92.8	0	0	0	25	0	0	0	0	0	0	0
9	27	96.1	25	2	0	0	0	0	0	0	0	0	0
10	4	93.4	0	0	0	0	0	4	0	0	0	0	0
11	3	91.3	0	0	0	0	0	3	0	0	0	0	0
12	3	99.3	0	0	0	3	0	0	0	0	0	0	0
13	3	96.0	0	0	0	0	0	0	0	0	0	0	3
14	3	95.3	0	0	3	0	0	0	0	0	0	0	0
15	3	92.0	0	0	0	0	0	3	0	0	0	0	0
16	4	97.5	0	0	4	0	0	0	0	0	0	0	0

Sectors legend:
- S01 = Energy
- S02 = Materials
- S03 = Industrials
- S04 = Consumer Discretionary
- S05 = Consumer Staples
- S06 = Health Care
- S07 = Financials
- S08 = ICT
- S09 = Communication Services
- S10 = Utilities
- S11 = Real Estate

changes due to either sector-based category fragmentation or the reassignment of specific stocks from their original sector to a more correlated cluster. The results obtained in other periods are consistent with these findings.

3.3.2.2 Analysis of the equities

For cross-correlation analysis in time series comparison, certain parameters need to be set, including (i) the initial number of clusters, denoted as k, (ii) the discretization threshold, denoted as t, and (iii) the similarity threshold, denoted as p. The parameter k serves as a constraint for time series clustering [131]. In the case of input stocks with predefined categorization into industrial sectors, it is recommended that domain experts set k to the total number of sectors present in the analyzed index. This initial estimation will be further refined by the taxonomy generation procedure, which combines the clustering outcomes. Empirical evidence depicted in Figure 3.1

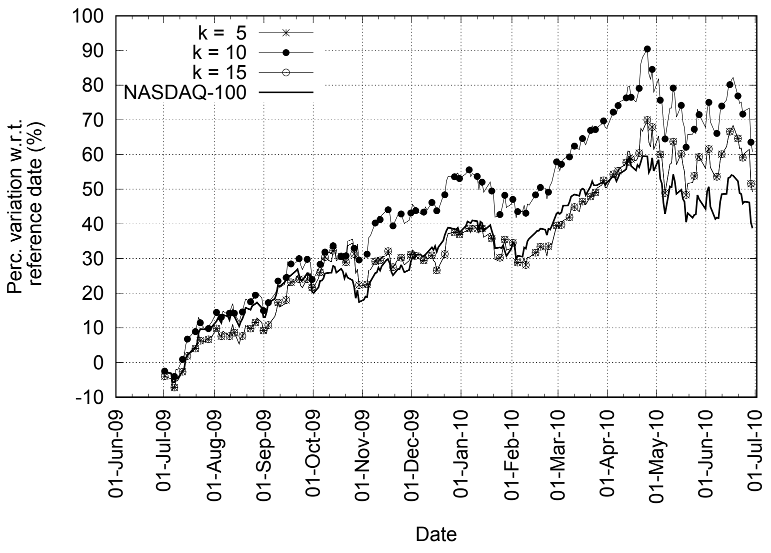


Fig. 3.1 Effect of parameter *k*. *t*=85%. *p*=70%. NASDAQ-100 index. Period: 2007-2009. Percentage variation w.r.t. July 1, 2009.

demonstrates the impact of setting *k* values close to the number of standard sectors (11) on the NASDAQ-100. The observed results align with the expected outcomes.

Regarding parameters *t* and *p*, these represent cut-off thresholds used to selectively filter combinations of correlated stocks. Lower values for these parameters result in coarser aggregation. Figure 3.2 illustrates the effect of setting the similarity threshold *t* to 85% and the cosine similarity threshold to 70%, which have shown promising performance.

3.4 Early Portfolio Pruning: a Scalable Approach to Hybrid Portfolio Selection

Given the extremely promising results obtained by DISPLAN in portfolio selection, this work studies the incorporation of the model into a bigger system. The DISPLAN model is used here to generate a list of candidate portfolios over which heavier and more complex analysis can be made.

The mean-variance model introduced by [104] is a well-known approach for optimizing portfolios. It quantifies the return and risk of stocks based on historical price distributions. However, there are scalability issues and concerns about the

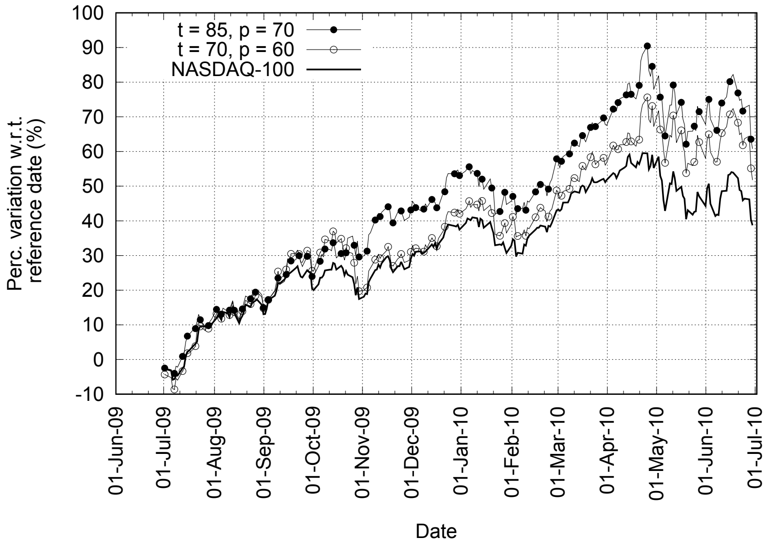


Fig. 3.2 Effect of parameters *t* and *p*. *k*= 10. NASDAQ-100 index. Period: 2007-2009. Percentage variation w.r.t. July 1, 2009.

reliability of estimated values in Markowitz's model. The information required to estimate expected value and higher-order moments increases exponentially with the number of candidate stocks.

In addition to scalability issues, investors often need to consider additional constraints such as transaction costs and stock diversification strategies. These constraints make the portfolio selection problem NP-hard, leading to the use of heuristic methods to identify relevant stocks. Hybrid solutions combining data mining and machine learning techniques have emerged to address these challenges. They involve a two-step process where a subset of the most relevant stocks is selected using data-driven models, followed by an optimization step applied to the shortlisted stocks.

Previous hybrid methods have focused on selecting portfolios from a shortlist of individual stocks, limiting the efficiency and scalability of the optimization step. To overcome this limitation, it would be beneficial to prune part of the portfolio candidates at an early stage while integrating complex portfolio-level constraints. This work introduces a scalable hybrid method called Early Portfolio Pruning (EPP), which first generates a set of candidate portfolios using an itemset-based heuristic. The selection problem is then solved on a portfolio shortlist instead of individual stocks. The analytical complexity is shifted to the itemset-based heuristic phase, which can employ scalable algorithms. EPP emphasizes its ability to discard less interesting portfolios from the search space early on.

To address the limitations of the Markowitz approach, this method quantifies stock interactions only within a restricted number of previously shortlisted portfolios, reducing the model complexity. It also provides a configurable system that allows the incorporation of other metrics instead of solely relying on the first two moments, which are often considered unstable and subject to variation in the financial world.

The proposed hybrid portfolio generation method offers investors the flexibility to customize the selection process even when dealing with a large set of stocks. The optimization problem by Markowitz is reformulated to suit the modified task, and a scalable implementation of EPP is developed. The method is further integrated into a financial Decision Support System (DSS) that leverages fundamental data analysis and user preferences to choose the portfolio.

Performance evaluations are conducted on stocks from the U.S. market. The back-testing simulations demonstrate the effectiveness of EPP in generating profitable yet low volatile portfolios, even in adverse market conditions like the COVID-19 pandemic.

3.4.1 Problem statement

Table 3.2 report the notation that will be adopted.

The Mean-Variance (MV) model, introduced by [104], is a well-established strategy for stock portfolio optimization. The main concept is to treat the return of each individual stock as a random variable and use the expected return and variance to model profitability and risk, respectively. The MV model calculates distribution descriptors based on historical stock prices (H) to quantify the return on investment and risk level of each stock [134].

In the MV model, the return of a candidate stock s_i is modeled as a random variable R_i, with an associated expected return $\mu_i = E(R_i)$. By arranging these expected returns into a vector μ, the expected portfolio return can be calculated as:

$$\mu^{\mathsf{T}}\mathbf{w} = \sum_{i=1}^{|S|} w_i \mu_i. \tag{3.1}$$

Table 3.2 Summary of notations and their meanings.

F	Financial statements of the candidate stocks
S	Set of candidate stocks
$\mathbb{P}$	Power set of S that represents all the possible portfolios
P_q	Stock portfolio consisting of a selection of candidate stocks, indexed by $q \in \{1,\ldots,\lvert\mathbb{P}\rvert\}$
H	Historical price series of the candidate stocks within the reference time period
w_i	Proportion of the total amount available for investment applied to stock $s_i \in S$
x_q	Binary vector in $\{0,1\}^{\lvert S\rvert}$ with 1 when the stock s_i is selected in portfolio P_q and 0 otherwise
$E[\cdot]$	Expected value function
$E_{min}[\cdot]$	Lower-Bound estimate of the Portfolio Return (LBPR)
R_i	Random variable return of stock $s_i \in S$ over the holding period
$\mu_i = E[R_i]$	Expected return of the individual stock s_i over the holding period
$\mu^{\mathbb{P}} : \mathbb{P} \to \mathbb{N}^+$	Generalized expected return of the portfolio P_q over the holding period
μ	Vector with the expected return of all the stock in S
$\sigma_{ij} = Cov(s_i, s_j)$	Covariance of the returns for the pair of stocks s_i and s_j
Σ	Covariance in matrix form for all the stock in S.
$\Sigma^{\mathbb{P}} : \mathbb{P} \to \mathbb{N}^+$	Generalized risk measure for the portfolio P_q over the holding period
$c^{\mathbb{P}} : \mathbb{P} \to \mathbb{N}^+$	Technical and fundamental analysis constraint function for the portfolio P_q over the holding period
$\mathbf{1}, \mathbf{0}$	Vectors with all elements set to 1 and 0

Here, the participation weights of the candidate stocks are stored in the vector $\mathbf{w} \in \mathbb{R}^{|S|}$, where w_i, i=1,2,...,$|S|$ represents the weight of stock $s_i \in S$ in portfolio P.

In addition to maximizing the expected return, the MV model incorporates portfolio diversification by estimating the return dispersion as:

$$\mathbf{w}^{\mathsf{T}}\Sigma\mathbf{w} = \sum_{i=1}^{|S|}\sum_{j=1}^{|S|} w_i \sigma_{ij} w_j. \tag{3.2}$$

According to the MV model, the stock portfolio optimization problem can be formulated as a linear combination of these objectives [135]:

$$\begin{aligned} \textbf{maximize } & \mu^{\mathsf{T}}\mathbf{w} - \lambda \cdot \mathbf{w}^{\mathsf{T}}\Sigma\mathbf{w} \\ \textbf{s.t. } & \mathbb{1}^{\mathsf{T}}\mathbf{w} = 1 \qquad , \\ & \mathbf{w} \geq \mathbf{0} \end{aligned} \tag{3.3}$$

where

- $\lambda \in \mathbb{R}^{+}$ is the risk aversion coefficient, i.e., the larger the coefficient the more risky the generated portfolio,
- $\mathbf{w} \geq \mathbf{0}$ defines as positive (short-selling operations are not permitted) value each weight w_i.

In this context, the following assumptions are made:

- The total amount available for stock investments is allocated.
- The amount allocated to each stock is kept fixed until the end of the holding time.

In the traditional MV model formulation, the set of input stocks S is treated as a single, large portfolio, and each stock $s_i \in S$ is assigned a continuous weight w_i. In contrast, the current study focuses on the binary stock selection problem [136] and adopts a uniform investment strategy. This involves selecting an equally weighted portfolio from the power set $\mathbb{P}$ of the input stocks S. Additionally, a buy-and-hold strategy is implemented for stock market investment, which entails purchasing the securities and selling them at the end of the specified holding period.

3.4.2 The proposed Mean-Variance model adaptation

In this work, an adapted version of the traditional Mean-Variance (MV) model is presented, which focuses on identifying the best portfolio from a set of portfolio candidates, rather than shortlisting individual stocks.

To generate portfolio candidates, an itemset-based heuristic is employed. This heuristic approximates the expected return of a candidate portfolio by combining the daily returns of the least performing stock in the portfolio. It should be noted that the approach can be generalized to incorporate an arbitrary portfolio-level heuristic that can be computed in a scalable manner.

In the itemset-based heuristic, the Lower-Bound estimate of the Portfolio Return [123] (LBPR) is utilized to ensure that each candidate stock portfolio meets a certain criterion. The LBPR is defined and calculated as follows:

$$E_{\min}[P_q] = \mathsf{average}_{d \in H} \left\{ \mathsf{minret}(P_q, d) \right\}, \tag{3.4}$$

where P_q is a selected portfolio identified as an element of the power-set $\mathbb{P}$ of S, thus $q \in \{1, \ldots, |\mathbb{P}|\}$. The function minret$(\cdot)$ returns the least daily return over all the portfolio stocks on a given day d.

The objective is to identify the portfolio $P \in \mathbb{P}$ that optimally aligns with a rank-based objective function. This function takes into account various expert-driven decision criteria, enabling a comprehensive evaluation of the portfolio. The fundamental concept is to combine the portfolio-level return, which is assessed using a specific performance measure such as LBPR, with other performance measures generated independently through different strategies. These additional measures can encompass factors like volatility or the wisdom of crowds, allowing for a comprehensive assessment of candidate portfolios. By combining these diverse measures, the aim is to identify the portfolio that performs best across multiple evaluation criteria.

The proposed selection model examines a subset of portfolios $\mathbb{P}$ and utilizes a vector[3] $x_q \in \mathbb{P}$ to store binary choices associated with each candidate portfolio

$$\begin{aligned} \textbf{minimize } & (1-\lambda)\mu^{\mathbb{P}}(x_q) + \lambda \cdot \Sigma^{\mathbb{P}}(x_q) \\ \textbf{s.t. } & \mathsf{c}^{\mathbb{P}}(x_q) \geq \mathsf{Th} \quad \mathsf{Th} \in \mathbb{R}^T \\ & x_q \in \mathbb{P} \end{aligned} \quad , \tag{3.5}$$

where

- The return ranking function $\mu^{\mathbb{P}}\colon \mathbb{P} \to \mathbb{N}^+$ directly interfaces with the adopted portfolio-level heuristic by returning the rank of a candidate portfolio and, in this work, it boils down to a ranking based on the lower bound estimate of the return $E_{\min}[P_q]$.
- The risk ranking function $\Sigma^{\mathbb{P}}\colon \mathbb{P} \to \mathbb{N}^+$ returns the rank of a candidate portfolio based on its risk measure, avoiding the estimation of statistical measures for all the combination of available stocks.
- Constraints $\mathrm{c}^{\mathbb{P}}\colon \mathbb{P} \to \mathbb{R}^{\mathrm{T}}$ allow end-users to set up multiple decision criteria based on a variety of T factors through a threshold Th, among which the stock diversification over sectors, the observed trends in the historical stock prices, and the fundamentals behind the considered assets. Constraint enforcement will be discussed later on (see Section 3.4.3.2).
- $\lambda \in [0,1]$ is the *risk aversion* of the end-users, which allows us to make a combination of portfolio pay-off and risk.

The ranking strategy serves as a substitute for a traditional combinatorial optimization approach. It provides a meaningful way to compare the trade-off between risk and return, which may not be directly comparable through a purely quantitative method. It is worth noting that in Equation (3.5) the dependency on the portfolio family $\mathbb{P}$ is made explicit and the set of additional constraints can be conveniently adapted to the end-users' needs.

[3]Notice that for each portfolio P_q holds a biunivocal relation with the binary vector x_q with 1 when the stock s_i is selected in portfolio P_q and 0 otherwise.

3.4.3 The Early Portfolio Pruning method

The hybrid method proposed involves a two-step procedure:

- **Candidate portfolio generation**. This step involves analyzing historical stock-related data using a parallel itemset mining approach. Its objective is to create a collection of candidate stock portfolios by identifying a subset of promising portfolios based on a global trend analysis of the constituent stock prices.
- **Portfolio selection**. From the candidate portfolios generated in the previous step, this stage determines the optimal choice based on both end-users' preferences and an analysis of additional stock-related data, such as fundamental analysis. To accomplish this, a solver is employed, which applies the adapted Mean-Variance philosophy described in Section 3.4.2..

OHLCV data, which encompass open, high, low, close, and volume information, are extensively utilized for analyzing stock price and volume trends through technical analysis. These data are considered to inherently encompass all the underlying effects and are examined in the initial phase of the hybrid method, known as candidate portfolio generation.

The taxonomy employed in this context consists of a collection of aggregation hierarchies constructed based on stocks. Its primary purpose is to facilitate diversification of fund allocation across various sectors, thereby reducing overall risk exposure [96].

In fundamental analysis, financial reports are commonly leveraged to assess the intrinsic value of equities by examining relevant economic and financial factors [137]. Both aggregation hierarchies and financial reports are employed to guide the portfolio selection step.

Transactional stock price model To generate the candidate portfolios based on itemsets, the daily closing prices of each considered stock are extracted from the OHLCV data. These closing prices are then stored in a transactional dataset [138]. Each transaction, denoted as tr_x, corresponds to a specific trading day, d_x, within the reference time period $[d_{start}, d_{end}]$. The transaction tr_x consists of pairs $\langle s_i, r_i^x \rangle$, where r_i^x represents the percentage variation in the closing price of stock s_i between days d_x and d_{x-1}.

An example of a transactional dataset is illustrated in Table 3.3. It comprises six transactions, each capturing the closing price variations (compared to the previous day) for stocks A, B, C, D, and E on different trading days. For instance, on day d_1, the closing price of stock A has increased by 5% relative to the preceding day. It should be noted that in the transactional data model, the temporal order of the transactions (i.e., days d_1-d_6) is not significant.

Table 3.3 Transactional data representation. Reference time period $[d_1, d_6]$

Time stamp	**Transaction**
d_1	⟨A,5%⟩, ⟨B,5%⟩, ⟨C,-1%⟩, ⟨D,7%⟩, ⟨E,5%⟩
d_2	⟨A,2%⟩, ⟨B,6%⟩, ⟨C,0%⟩, ⟨D,2%⟩, ⟨E,2%⟩
d_3	⟨A,4%⟩, ⟨B,5%⟩, ⟨C,-2%⟩, ⟨D,4%⟩, ⟨E,5%⟩
d_4	⟨A,4%⟩, ⟨B,2.5%⟩, ⟨C,-4%⟩, ⟨D,10%⟩, ⟨E,4%⟩
d_5	⟨A,1%⟩, ⟨B,4%⟩, ⟨C,-2%⟩, ⟨D,7%⟩, ⟨E,1%⟩
d_6	⟨A,-1%⟩, ⟨B,6%⟩, ⟨C,0%⟩, ⟨D,1%⟩, ⟨E,-1%⟩

Taxonomy over stocks A taxonomy is established over stocks to incorporate the information regarding stock membership into specific financial sectors. Each stock is mapped to its corresponding sector. In order to avoid dependences in the results, in the experiments conducted for this part of the work, the hierarchical stock relationships are derived from the standard GICS sector-based stock categorization[4]. Alternatively, the relationships can be automatically inferred by end-users using ad hoc clustering-based methods, such as those mentioned in Section 3.3, or even of other derivation such as subspace factorization or genetic algorithms, as discussed in [98, 99].

Financial statements Fundamental analysis involves the examination of economic and financial factors related to a stock, such as production, earnings, employment, housing, manufacturing, and management. In this study, the focus is on a subset of fundamental factors identified by [139] as predictors of stock performance. These selected factors are: (i) rate of sales growth over the past year (SGI) [140], (ii) gross margin (GMG) [141], (iii) earning surprise (CHGEPS) [142], (iv) total capital expenditures (CAPX), (iv) revenues earned or expenses incurred (ACCRUAL) [143], and (v) level of research and development investments (R&D)

[4]https://www.msci.com/gics

It should be noted that other factors can also be considered as constraints based on specific analysis requirements.

3.4.3.1 Candidate portfolio generation

To generate candidate stock portfolios, the methodology employs the DISPLAN itemset mining approach [17], presented in Section 3.2.

3.4.3.2 Portfolio selection

Modern financial decision support systems enable end-users to express their preferences for portfolio selection with varying levels of insight, thereby extending beyond Markowitz's original work, which solely relied on the first two moments of return distribution. Decisions in portfolio selection are typically influenced by: (i) current market conditions, (ii) economic investors' preferences and risk attitudes, and (iii) intrinsic economic value of the considered assets.

In order to identify the optimal portfolio, EPP employs the adapted Mean-Variance model described in Section 3.4.2. This model allows for the incorporation of user-specified constraints at the portfolio level. Specifically, the following constraints are considered:

- **Fundamental analysis**. Portfolios are evaluated based on the relative strength of the financial stock fundamentals. A ranking strategy is employed to shortlist portfolios that include top-ranked stocks across a range of established financial indicators.
- **Diversification**. Portfolios are expected to include stocks that are well diversified across the given taxonomy. Portfolios that fail to meet a sufficient level of diversification are pruned at an early stage.
- **Trend**. Stock price trends are commonly used to formulate trading strategies. Portfolios are shortlisted based on the long-term price trends of the constituent stocks, which are estimated using established technical analysis indicators.

Follows a more detailed description of the supported constraints.

Portfolio-level constraints based on fundamental factors A fundamental score is assigned to each portfolio based on the characteristics of the constituent stocks. The process begins by evaluating the financial and economic strength of each stock using a range of fundamental factors, as described in [144]. These individual stock scores are then combined to determine the portfolio-level score.

To obtain a summary of key financial information, a subset of fundamental indicators and ratios is extracted from the initial set available in the fundamental reports. This selection process follows the guidance provided by [139], which focuses on indicators that are effective for stocks with extreme returns. This allows for the early pruning of stocks in the initial stage of the algorithm based on returns. The choice of indicators is made by an expert-driven approach, where the end-user selects the indicators deemed relevant from a default pool.

The per-stock score is an integer value ranging from zero to the number of activated indicators/ratios. It considers the number of factors that rank in the upper percentile in the overall stock ranking. In line with [144], the stocks are ranked in descending order based on the summarizing factor score, and only those within the upper 80% percentile are included. The purpose of this score is to assess the relative strength of a stock's financial fundamentals in terms of global quality by considering the distribution of each indicator among the pruned stocks.

Portfolio-level constraints based on diversification The level of risk in a portfolio is evaluated by ensuring that the selected candidates meet a minimum requirement for stock diversification, as determined by the stock categorization specified in the input taxonomy. The diversification level of the portfolio is measured as the percentage of stocks belonging to distinct categories. It should be noted that the domain expert has the flexibility to manually define the minimum level of diversification in order to effectively manage risk exposure.

Portfolio-level constraints based on technical analyses The analysis of stock prices using classical technical analysis indicators and oscillators, such as Simple Moving Average (SMA) and Exponential Moving Average (EMA) as described in [28], provides valuable insights into underlying price trends and trading volumes. To ensure that portfolios selected do not include stocks with negative trends, EPP incorporates a portfolio evaluation based on technical analysis. One example is the comparison between current portfolio prices and the simple or exponential moving averages calculated at a fixed periodicity (e.g., if the price is above the SMA with a period of 50, it indicates a likely price uptrend). By considering such indicators, EPP helps to assess and incorporate information about stock price trends into the portfolio selection process.

Risk aversion To accommodate the preferences of end-users, the value of risk aversion λ in the adapted Mean-Variance logic can be determined by the users themselves within the range of [0,1]. A higher value of λ indicates a higher level of risk aversion, as discussed in Section 3.4.2. It is worth noting that alternative and more advanced approaches for assigning risk aversion levels to stock portfolios, such as the method proposed by [145], can also be integrated into the system. This allows for greater flexibility in incorporating various risk aversion strategies based on the specific requirements and preferences of end-users.

3.4.3.3 The Decision Support System

Algorithm 3 outlines the proposed Decision Support System. It begins by applying the LBPR algorithm to the dataset of daily stock prices $\mathbf{D_s}$. Subsequently, the system ranks the resulting set of portfolios shortlisting the top-$\mathbf{k}$ portfolios based on specific criteria. Portfolios with a diversification level below the threshold $\mathbf{th_d}$ and a fundamental score lower than $\mathbf{th_f}$ are filtered out. Finally, the system chooses the stock portfolio that achieves the optimal balance between expected return and risk.

For a clearer understanding of the decision-making process, Figure 3.3 provides a visual representation of the key steps employed by the decision support system.

3.4.4 Experiments

3.4.4.1 Experimental design

Stock-related data was crawled from Yahoo! Finance[5].

The experiments were run on a hexa-core 2.67 GHz Intel Xeon with 32GB of RAM, running Ubuntu Linux 18.04.4 LTS.

[5]https://finance.yahoo.com

Algorithm 3: EPP: pseudocode of the decision support system

```
Input  : D_s: Dataset containing daily stock prices of traded companies;
         D_f: Dataset containing financial statements of traded companies;
         D_d: Dataset containing sectors of traded companies;
         sup: Minimum support threshold for LBPR (default: 8%);
         l: Maximal portfolio size for LBPR (default: 7);
         k: number of portfolios to maintain after LBPR step (default: 100);
         th_d: diversification threshold (default: th_d = 70%);
         tic: Defined condition on technical indicator[s] (default: daily closing price
         above/under SMA-50 periods);
         f: Fundamental indicator[s] to be employed;
         th_f: Fundamental indicators scoring threshold (default: 20th percentile);
         λ: Risk-based weight between rankings [0, 1] (default: 0.5);
Output : ranking: Ranking of suggested portfolios;

   /* Lower Bound Portfolio Return Estimation                          */
1  $\mathbf{R_{lbpr}}, \mathbf{ranking_{lbpr}} \leftarrow$ LBPR($\mathbf{D_s}$, sup, $l$);
2  $\mathbf{R_{div}} \leftarrow$ filterDiversification($\mathbf{R_{lbpr}}$, $\mathbf{th_d}$, $\mathbf{D_d}$);
3  $\mathbf{R_{topk}} \leftarrow$ filterTopK($\mathbf{R_{div}}$, $\mathbf{ranking_{lbpr}}$, k);
   /* Filtering portfolios                                             */
4  foreach portfolio p $\in \mathbf{R_{topk}}$ do
5      if p satisfies tic then
6          $\mathbf{R_{ti}}$.insert(p);
7  $\mathbf{L_{stocks}} \leftarrow$ extractStocks($\mathbf{R_{ti}}$);
8  foreach stock s $\in \mathbf{L_{stocks}}$ do
9      $\mathbf{score_s} \leftarrow$ scoringStock(s, f, $\mathbf{D_f}$);
10     $\mathbf{L_{scores}}$.insert($\mathbf{score_s}$);
11 foreach portfolio p $\in \mathbf{R_{ti}}$ do
12     $\mathbf{score_p} \leftarrow$ scoringPortfolio($\mathbf{L_{scores}}$);
13     if $\mathbf{score_p} > \mathbf{th_f}$ then
14         $\mathbf{R_f}$.insert(p);
   /* Ranking result set                                               */
15 $\mathbf{ranking_{risk}} \leftarrow$ rankByRisk($\mathbf{R_f}$);
16 $\mathbf{ranking} \leftarrow (1-\lambda) \cdot \mathbf{ranking_{lbpr}} + \lambda \cdot \mathbf{ranking_{risk}}$;
17 return ranking;
```

The framework is written in the Python and Spark languages.

A set of back-testing trading simulations is conducted to evaluate the profitability and riskiness of EPP. The test periods are defined based on the bearish and bullish market states, as introduced in a previous study (bear-bull). The raw price series of

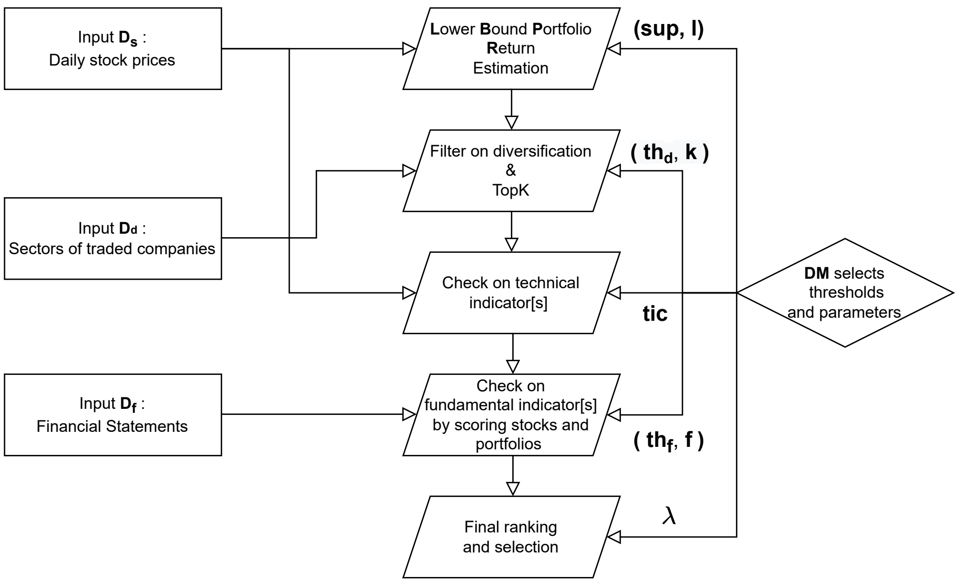

Input Ds : Daily stock prices
Input Dd : Sectors of traded companies
Input Df : Financial Statements
Lower Bound Portfolio Return Estimation
Filter on diversification & TopK
Check on technical indicator[s]
Check on fundamental indicator[s] by scoring stocks and portfolios
Final ranking and selection
(sup, l)
(thd, k)
tic
(thf, f)
λ
DM selects thresholds and parameters

Fig. 3.3 Graphical representation of the main decision support system steps.

the analyzed market index is segmented into bearish and bullish market states, as shown in Figure 3.4, and reference time periods are selected accordingly.

- **Bearish period**. The bearish period covers the time span from 2008 to 2009. It was characterized by a bearish market condition primarily caused by the global financial crisis that originated from the subprime mortgage crisis.
- **Bullish period**. The bullish period spans from 2012 to 2015. It was characterized by a bullish market condition driven by global economic growth. This period is a subset of a larger 10-year bullish period, and the selected subsection represents the period with the fastest market growth.
- **Covid-19 pandemic period**. The Covid-19 pandemic period ranges from 2018 to 2020. It was characterized by a mix of bearish and bullish market states. This case study focuses on the outbreak of the Covid-19 pandemic, the implementation of restrictions, and the end of the first epidemic wave. It serves as a real-life and challenging scenario for analysis.

For each period, a series of back-testing simulations are conducted to evaluate the effectiveness and robustness of the portfolio optimization strategies using historical stock-related data relative to the NASDAQ-100 index. The itemset-based model is

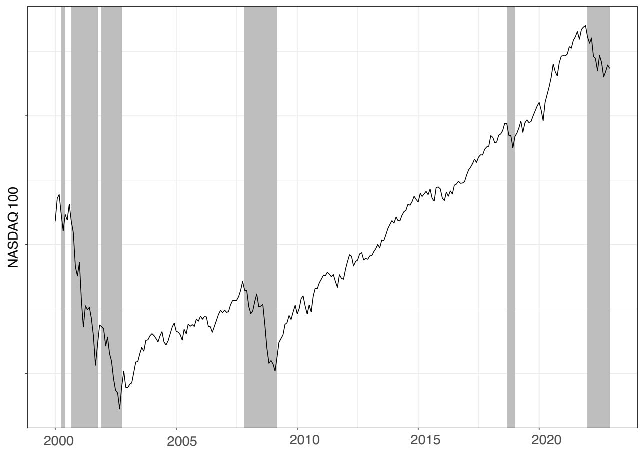


Fig. 3.4 The NASDAQ-100 index. Bearish periods are colored in gray whereas bullish ones are in white.

trained using a six-month period (e.g., from July 1, 2007, to December 31, 2007, for the bearish period), and then applied to the following 12 months (e.g., for the year 2018). The simulations employ a buy-and-hold strategy, where the portfolio stocks are purchased at the beginning of the period and sold at the end.

In all the conducted simulations, an initial equity of 100,000 USD is considered. A fixed-fractional money management strategy is adopted, without any stop loss or stop profit limits. Additionally, per-trade transaction costs are approximated to be 0.15% based on [36].

The performance of the proposed methodology have been compared with:

- The **NASDAQ-100 benchmark**, which replicates the NASDAQ-100 index with no leverage.
- The established Mean-Variance model by Markowitz [36]. The portfolio optimization follows the strategy presented by [146] and selects the optimal portfolio on the efficient frontier based on the Sharpe ratio [147], which measures the reward-to-variability ratio of a portfolio compared to a risk-free asset. This strategy is referred to as **Markowitz-Sharpe**.
- A set of recently proposed Deep Reinforcement Learning (DRL) strategies to stock portfolio allocation available in the **FinRL** library, namely **A2C, TD3, and DDPG** [148].
- The itemset-based heuristic for portfolio generation called **DISPLAN** [123].

- Three established **U.S. hedge funds** (only for the most recent Covid-19 pandemic period 2018-2020) investing on the same assets, i.e., MSEGX-Morgan Stanley Inst Growth A[6], OLGAX-JPMorgan Large Cap Growth A[7], PIODX-Pioneer Fund Class A[8].

Markowitz portfolios are generated using the MATLAB function *estimateMaxSharpeRatio*.

The minimum support threshold for DISPLAN and EPP is adjusted within the range of 3% to 12%, while the diversification threshold remains constant at 70%.

FinRL is trained using a decade's worth of historical data to mitigate the adverse impact of data overfitting.

During each trading simulation, an analysis is conducted on the following metrics:

- The **equity line plot**. This graph visually depicts the temporal fluctuations of the equity throughout the test period [28].
- The **payout**. It quantifies the overall percentage return or loss of the portfolio at the conclusion of the test period [135].
- The **volatility**. This metric measures the standard deviation of the overall portfolio value in relation to the daily returns [135].

To provide a graphical representation of these metrics, time series plots are used. These plots illustrate the percentage variation of the equity concerning the initial investment value (e.g., refer to the equity line plot shown in Figure 3.8a) and the daily percentage change in the portfolio's price (e.g., see the volatility plot displayed in Figure 3.8b).

3.4.4.2 Results of the backtesting simulations

The following results are presented in this section:

[6]https://www.morganstanley.com/im/en-us/intermediary-manager-research/product-and-performance/mutual-funds/us-equity/growth-portfolio.shareClass.A.html

[7]https://am.jpmorgan.com/us/en/asset-management/adv/products/jpmorgan-large-cap-growth-fund-a-4812c0506

[8]https://www.amundi.com/usinvestors/Products/Mutual-Funds

- The comparison between the equity lines of the portfolios generated by EPP and those of the tested competitors (see Figures 3.5a, 3.6a, 3.7a, 3.8a, 3.9a, and 3.10a). These comparisons provide a high-level view of the overall performance achieved by different approaches. For the sake of clarity, the comparisons with the Reinforcement Learning strategies are reported in separate plots (see Figures 3.5b, 3.7b, and 3.9b).
- The comparison between the equities selected by EPP with those selected by the real hedge funds (see Figure 3.11) with the aim of showing the applicability of the proposed system in a real scenario.
- The volatility of EPP compared with those of the other approaches (see Figures 3.6b, 3.8b, and 3.10b).

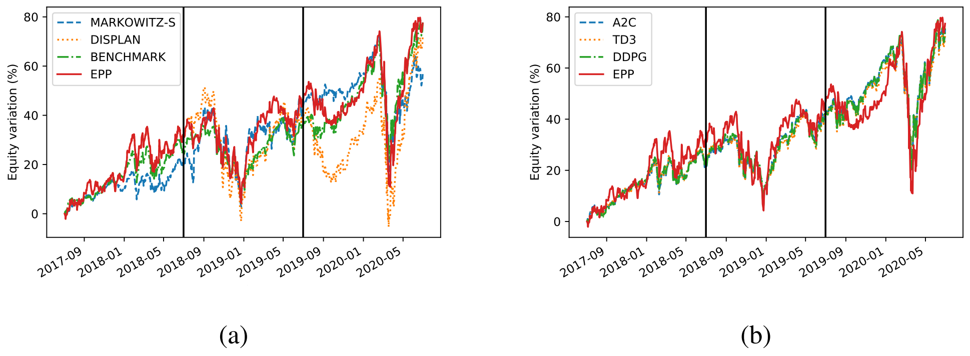


Fig. 3.5 Percentage variation of the equities. Covid-19 pandemic period. NASDAQ-100 index. (a) Comparison with benchmark, DISPLAN, and Markowitz-Sharpe. (b) Comparison with the Deep Reinforcement Learning strategies.

Each market period will be separately analysed.

Covid-19 pandemic period Figures 3.5a and 3.5b present a comparison of equities during the period of the Covid-19 pandemic (2018-2020 including training). EPP demonstrates strong resilience against negative market trends. This can be observed in the frequency of declines in DISPLAN values (indicated by orange dots) during different time periods (e.g., 09/2019-12/2019, 01/2020-05/2020), while EPP maintains profitable values. EPP outperforms both DISPLAN and Markowitz-Sharpe, while achieving comparable results to DRL-based methods.

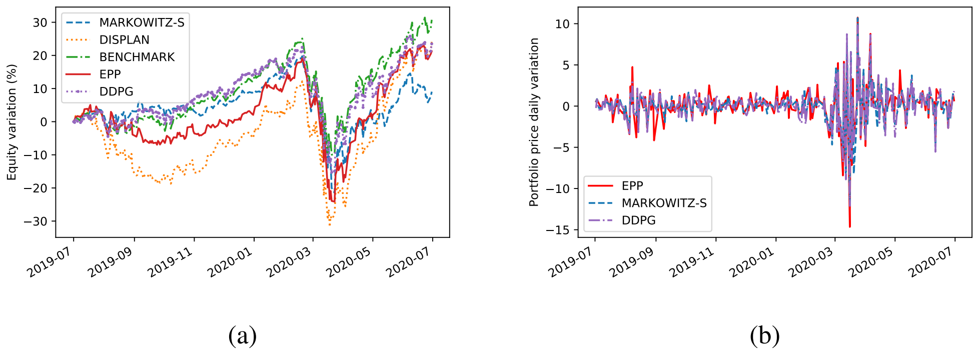
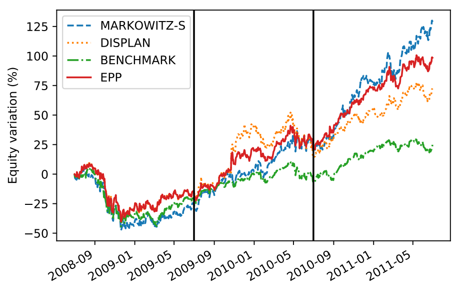


Fig. 3.6 Performance comparison during the outbreak of the Covid-19 pandemic. (a) Percentage variation of the equities. (b) Volatility plot.

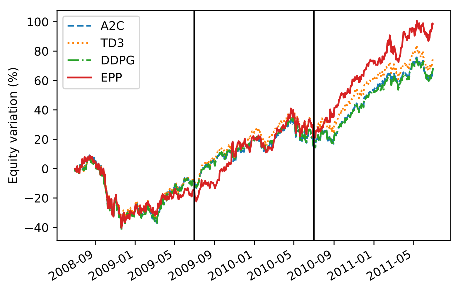
A closer examination of the Covid-19 pandemic outbreak period reveals that EPP and DRL-based methods exhibit better ability to counteract market drawdowns compared to DISPLAN and Markowitz-Sharpe, particularly during the peak of the epidemic wave. DRL agents inherently possess adaptability against adverse market movements, while the combination of a static itemset-based model with an adapted Markowitz model demonstrates the empirical property of being adaptive to market changes. This analysis is supported by the equity lines in Figure 3.6a and the volatility plot in Figure 3.6b.

Bearish market period During the bearish period from 2008 to 2011, which corresponds to the 2008 financial crisis and subsequent market recovery, there were notable performance differences between various models. In this challenging situation, EPP initially performed well in the first two years (2008-2010) but was later surpassed by Markowitz-Sharpe during the market rally after the financial crisis. On the other hand, when compared to DRL-based models, there were several instances of changes in ranking within the first two years, with EPP eventually overtaking it during the rally phase (see Figures 3.7a and 3.7b).

However, when specifically examining the period of the financial crisis, the EPP portfolio demonstrated lower volatility compared to Markowitz-Sharpe, while its draw-down and payout were similar to those of the DRL-based methods (see Figures 3.8a and 3.8b).

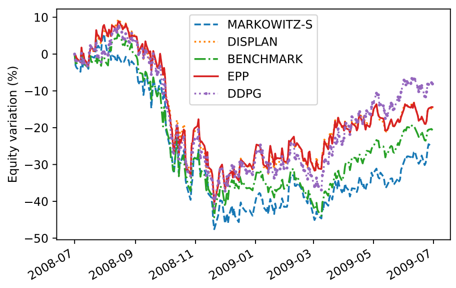


(a)

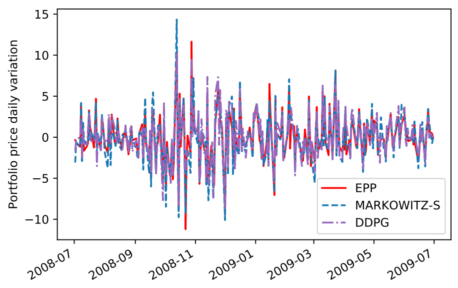


(b)

Fig. 3.7 Percentage variation of the equities. Bearish period. NASDAQ-100 index. (a) Comparison with benchmark, DISPLAN, and Markowitz-Sharpe. (b) Comparison with Deep Reinforcement Learning strategies.



(a)



(b)

Fig. 3.8 Performance comparison during the outbreak of the financial crisis. (a) Percentage variation of the equities. (b) Volatility plot.

Bullish market period In the bullish period under consideration, EPP outperforms the other competitors tested (See Figures 3.9a and 3.9b).

When focusing on the period of maximum market growth, specifically depicted in Figure 3.10a, it becomes apparent that the payout of Markowitz-Sharpe surpasses that of EPP. However, it is important to note that the volatility associated with Markowitz-Sharpe is significantly higher, as illustrated in Figure 3.10b.

The reason behind this discrepancy lies in the fact that EPP, due to its portfolio-level constraints, adopts a more conservative approach compared to Markowitz-Sharpe. Even in bullish market conditions where risky strategies relying on a small number of stocks tend to be rewarded, EPP prioritizes a more cautious investment strategy.

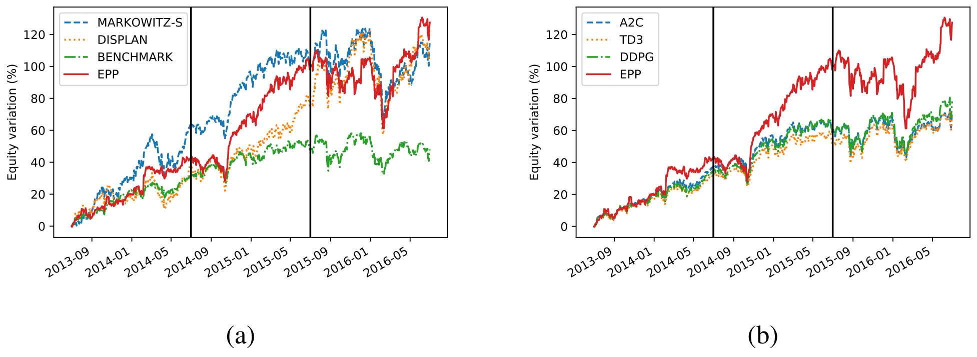


Fig. 3.9 Percentage variation of the equities. Bullish period. NASDAQ-100 index. (a) Comparison with benchmark, DISPLAN, and Markowitz-Sharpe. (b) Comparison with Deep Reinforcement Learning strategies.

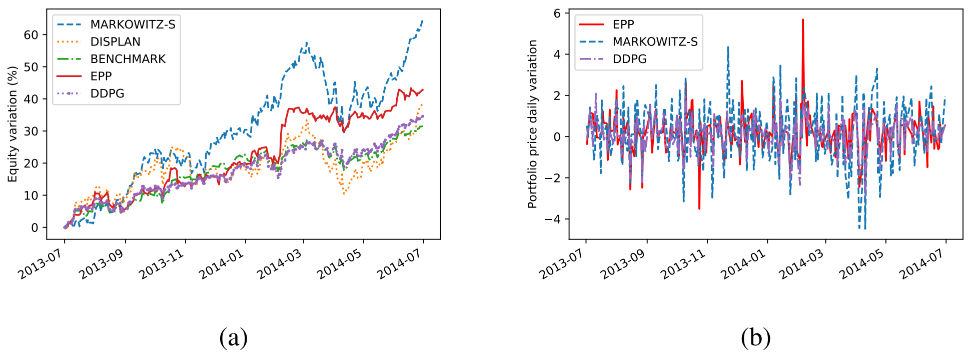


Fig. 3.10 Performance comparison during the period of most significant market growth (2013-2014). (a) Percentage variation of the equities. (b) Volatility plot.

Comparison with hedge funds Figure 3.11 shows the comparison of the system performances against well established hedge funds in the Covid-19 pandemic period. The results shown confirm the usability of the proposed system in real-world scenarios.

3.4.4.3 Effect of the risk aversion

The risk exposure of the EPP portfolio can be customized by end-users through the convenient adjustment of the risk aversion parameter, represented as λ. A higher value of λ signifies a greater emphasis on the risk-based ranking of candidate portfolios.

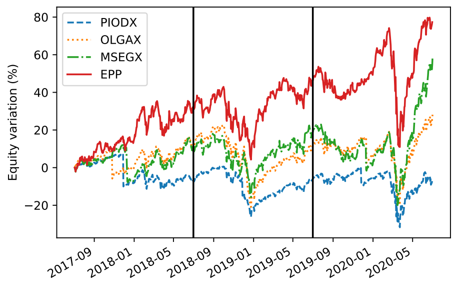
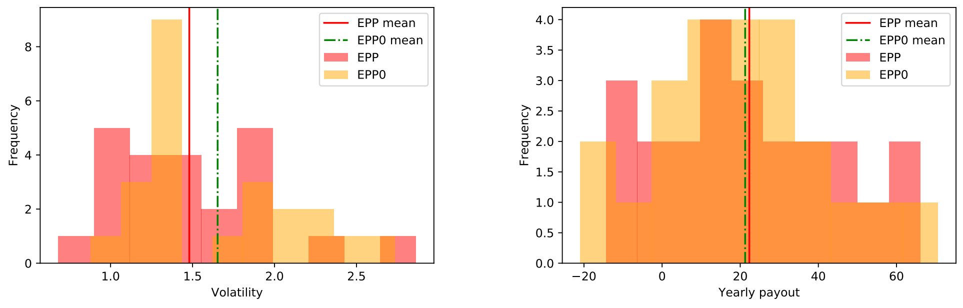


Fig. 3.11 Percentage variation of the equities. Covid-19 pandemic period. Comparison with real funds.



(a)

(b)

Fig. 3.12 EPP standard configuration with medium risk aversion (λ=0.5) vs. EPP with no risk aversion (λ=0). Years 2008-2020. (a) Distributions of the daily volatility statistic. (b) Distributions of the yearly payouts.

A set of experiments was conducted to analyze how risk aversion affects the performance of the portfolio. Two scenarios were evaluated: one with moderate risk aversion (λ=0.5) and another with no risk aversion (λ=0). The impact of risk aversion was examined by comparing the daily volatility and payout distributions across the analyzed years (2008-2020), as shown in Figures 3.12a and 3.12b.

The average payout values between the two scenarios are relatively similar, while the volatility consistently remains higher in the absence of risk aversion. However, it is not advisable to adopt an extreme configuration. Completely disregarding risk rankings would expose investors to significant market fluctuations without yielding substantial returns. Conversely, relying solely on risk rankings would diminish the importance of the selected heuristic.

3.5 Scalability

The last limitation of DISPLAN, and its evolutions introduced in this chapter, concerns the limited scalability of the system. The computational complexity of the algorithm is primarily influenced by the itemset mining step, which is utilized to generate candidate stock portfolios in a heuristic manner. For every version, the subsequent steps, applied to a restricted subset of portfolios, have minimal impact on time and memory complexity.

The enumeration of all possible frequent itemsets in a large dataset is recognized as NP-Hard [149]. Specifically, the number of generated candidates grows linearly with the dataset size and combinatorially with the number of input items. However, as discussed in [123], the optimal portfolio size is typically an order of magnitude smaller than the number of candidate stocks. Consequently, its influence on maximal itemset mining is less critical.

The scalable version here presented employs a parallel implementation of a maximal itemset mining algorithm [129], ensuring a time complexity of $O(\frac{|D_s|}{P})$, where $|D_s|$ represents the dataset size and P denotes the number of partitions employed in the parallel and distributed computation. The scalability of the algorithm is supported by empirical evidence.

The parallel and distributed itemset mining techniques introduced by [150] are currently supported by the ML-Lib library [151]. For this work, the parallel mining process is adapted to effectively handle transactional data that includes item weights.

3.5.0.1 Evaluation

The scalability of the system is evaluated by examining two factors: the number of stocks considered and the size of the historical time window used in the learning phase to generate the itemset-based model.

To assess scalability with the number of stocks, stocks from the Standard & Poor 500 (S&P 500) index are randomly added to the initial stock set. The simulations are then iteratively rerun until the entire S&P 500 index is covered.

Several backtesting simulations were conducted to evaluate scalability in terms of the number of stocks considered and the size of the training window

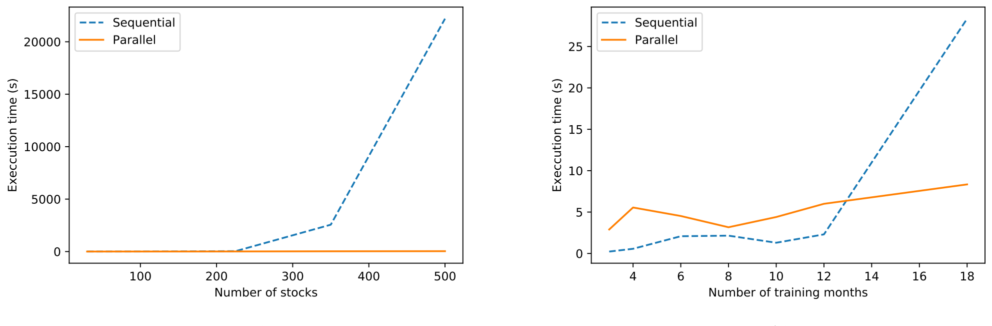


Fig. 3.13 Time complexity analysis. Comparison between DISPLAN with Parallel itemset mining and the DISPLAN variant with sequential itemset mining. (a) Scalability with the number of initial stocks. (b) Scalability with the training window size.

The scalability of the system was first tested by varying the number of initial stocks from 10 to 500, as shown in Figure 3.13a. The objective was to examine the impact of the parallel itemset mining phase on the time complexity of DISPLAN. To achieve this, both a parallel and a sequential version of the system were tested. The sequential variant employs an efficient, although not parallel, implementation of the FP-Growth algorithm[9]. Conversely, the parallel version of the model make use of the PFP algorithm [129].

As expected, an increase in the number of initial stocks resulted in a super-linear growth in execution time, primarily due to the generation of a large number of candidate itemsets that were subsequently processed sequentially. Conversely, in the parallel version, the workload was distributed among multiple workers, leading to a roughly linear increase in execution time with the number of stocks.

To further assess scalability, a similar test was conducted while keeping the number of initial stocks fixed at 100 (i.e., the stocks in the NASDAQ-100 index) and varying the number of training months from 3 to 18, as depicted in Figure 3.13b. An analysis of the time complexity revealed a non-linear increase when the training window size exceeded 12 months. However, this scalability issue appears to be less critical compared to the previous one, as the standard configuration for the window size typically ranges between 6 and 12 months.

[9]http://fimi.uantwerpen.be/src/

3.6 Findings and discussion

The studies presented in this chapter aim at overcoming the limitations of DISPLAN [17], an unsupervised approach for generating diversified stock portfolios.

By integrating parallel itemset mining, the system enables the analysis of large sets of stocks that would be challenging to handle using a sequential approach. The results confirm the improvement of computational performances.

Then, the application of time series cross-correlation analyses for enhancing the stock categorization process of the stock portfolio generator is analysed. The experimental findings have revealed instances where the conventional sector-based classification of stocks has overlooked the correlations between stocks within or across sectors. The developed trading strategy has demonstrated improved performance compared to the standard approach in various market conditions in the U.S. markets.

Finally, a hybrid financial decision support system designed for the selection of stock portfolios is introduced. This framework integrates two complementary approaches to portfolio selection: (i) an optimization-based approach and (ii) a heuristic approach. Specifically, it leverages the parallel itemset mining process introduced by DISPLAN before introducing specific constraints and risk-averse adjustments. The main concept is to simplify the complexity associated with stock-based approaches by filtering a portion of candidate portfolios during the initial itemset mining phase. The itemsets obtained represent potential stock portfolios, which are then analyzed using traditional Markowitz's philosophy. Furthermore, portfolio-level constraints based on supplementary knowledge from taxonomies and financial reports can be enforced. The selected portfolios demonstrate favorable performance in terms of payout and risk exposure compared to alternative methods, including Deep Reinforcement Learning approaches (which dynamically adapt models to current market conditions), Markowitz-Sharpe models, and established U.S. hedge funds.

Chapter 4

Time Series Embeddings and Summarization for Explainability

In the dynamic and complex world of financial markets, where billions of dollars are traded daily, understanding the underlying factors that influence asset prices and making informed investment decisions is of paramount importance. Traditionally, financial analysts and traders have relied on various quantitative and qualitative methods to assess investment opportunities and predict market trends. However, as markets become increasingly interconnected, information-intensive, and influenced by technological advancements, there is a growing need for advanced techniques that not only generate accurate predictions but also provide meaningful explanations for those predictions. In this context, the field of eXplainable Artificial Intelligence (XAI) has emerged as a powerful tool to bridge the gap between complex financial data and human interpretability. This chapter delves into two distinct studies that share a common goal: aiding experts and traders in better understanding the underlying markets and the signals generated by the models they rely on. In other words, aiding them in making better and informed investment decisions.

The first study has been presented in [152] and focuses on addressing the challenges faced by financial analysts and traders in comprehending the intricate relationships between stocks. Private and professional investors have access to vast amounts of financial data, particularly in the form of temporal evolution of stock prices. However, interpreting and comparing the underlying assets can be challenging. This study aims to address this challenge by automatically generating summarized comparisons of financial stock series based on established fundamental indicators. The underlying premise is that by offering comprehensive and comparative explanations, financial analysts and traders can make more informed decisions, mitigate risks, and identify potential investment opportunities more effectively.

The second study, presented in [153], recognizes the emergence of cryptocurrencies as a disruptive force in the financial landscape. With the advent of cryptocurrencies, traders and investors face new challenges in understanding the underlying dynamics of these digital assets. The study aims to utilize explainable AI methods to assist cryptocurrency traders in gaining insights into the inner workings of machine learning forecasting models. By unraveling the black box of these models and shedding light on the most important features driving predictions, this study seeks to empower traders with a deeper understanding of what the models actually do and how they can be utilized effectively in the context of cryptocurrency trading.

Although these two studies diverge in terms of the specific financial instruments they address and their way to aid investors, they share the common underlying motivation of providing investors with transparent and comprehensible explanations to augment their decision-making processes. Both studies leverage state-of-the-art techniques to bridge the gap between complex financial data and human understanding. By enhancing transparency and interpretability, these studies contribute to the emerging field of explainable finance [154–156], which aims to demystify the decision-making processes behind sophisticated financial models and foster greater trust and comprehension among investors.

4.1 Generating comparative explanations of financial time series

One of the most labor-intensive tasks for financial investors is to analyze market data, including financial reports, stock prices, and macroeconomic indicators [157, 158]. Understanding the underlying assets is crucial for investors, and they often seek easy-to-understand explanations of the changes in stock prices. This study focuses on generating textual explanations for stock price series. By presenting summaries in natural language, human users can gain a better understanding of how the analyzed series evolves over time and make informed decisions.

The most commonly used method for summarizing time series in text form is through *standard protoforms*. These protoforms utilize explainable summary templates to showcase important patterns in databases [159]. For example, a protoform could be: "*The samples of Time Series T acquired in the last week are very similar to those observed in most of the previous 10 weeks*". Here, *last week* represents the time window, *very similar* is the protoform quantifier, and *most* is the protoform summarizer.

State-of-the-art approaches automatically generate protoforms generate from time series data using clustering and fuzzy modeling techniques [160, 161]. However, these approaches have several limitations:

- Standard protoforms are not suitable to conduct comparative analyses of time series data at multiple granularity levels. For example, they cannot generate statements like: "*The samples of Time Series T acquired in the last week are very similar to those observed in the time series group G*".
- Protoform quantifiers and summarizers are defined using static domain-specific rules, which means their values may not accurately reflect the underlying data distribution.
- Protoforms are not tailored to the financial domain, so they do not take into account domain-specific aggregations such as fundamental indicator levels, market sector, and sentiment.

The goal of this work is to generate comparative explanations for financial stock price series by employing a self-supervised time series embedding representation. The approach involves encoding the characteristics of stock series into a unified vector space and then summarizing the differences between individual stock vectors and the encoding of stocks belonging to a reference group (e.g., stocks from the same sector with the highest operating profit). The values of quantifiers and summarizers are dynamically defined based on a data-driven approach using the inferred latent space.

The self-supervised encoding procedure is applied to both historical stock series and stock-related news data, capturing key information about stocks on a daily basis. It encompasses stock price trends, seasonality, momentum, and market movers' sentiment.

To evaluate the proposed approach, both intrinsic and extrinsic evaluations are conducted. Intrinsic evaluation involves shortlisting the generated summaries using established protoform-based metrics. Extrinsic evaluation includes backtesting the reliability of the generated stock recommendations. Preliminary results on the U.S. stock market confirm the applicability of the proposed approach in supporting stock trading activities.

4.1.1 Related works

Explainable summaries can be generated using various approaches such as static domain-specific rules, statistic/probabilistic methods, or neural techniques [162]. The Deep Learning community has made efforts to automate the process by leveraging data and algorithms instead of relying on static rules defined by experts. However, even though probabilistic/statistical approaches and neural methods can automatically generate text, the quality and readability of the summaries may not always be guaranteed. As a result, many existing time series summarization techniques still rely partially on rule-based methods, which produce standard summary templates known as protoforms [160, 163, 161, 164]. For example, [163] and [164] generate narratives summarizing the key trends of the data (e.g., increasing, decreasing), while [160] use Evolutionary Genetic Algorithms to identify suitable summary templates based on user-defined constraints. More recently, [161] applies sequence pattern mining and clustering techniques to support protoform generation.

This study introduces a hybrid approach to time series summarization that combines the reliability of rule-based strategies with the flexibility of natural language processing (NLP) methods. It employs a high-dimensional vector representation of the time series, generated by a specialized embedding model [165], to dynamically construct summaries that provide comparative explanations.

4.1.2 Data overview

To generate explainable summaries of financial data, the analysis focuses on multiple aspects of stock-related information. These aspects include the raw time series data of historical prices and exchange volumes, well-established price trend and volatility indicators, news sentiment, and the values of major fundamental indicators.

The study focus on the time series T_s of the daily closing prices of each stock s belonging to the Standard & Poor 500 (S&P500) index. The historical stock data analyzed spans from 2007 to 2018, with available data for 468 out of the 500 firms[1].

As for the price-related indicators, several technical indicators that describe momentum, trend, and volatility of stock prices are considered based on [28]. Specifically:

- Exponential Moving Average (EMA) with 5, 20, 50, and 200 periods.
- Moving average convergence divergence (MACD) with the following EMA combinations: (5, 20), (20, 50), and (50, 200).
- Relative Strength index (momentum oscillator) with cutoff thresholds 30% (over-sold) and 70% (over-bought).
- Aroon oscillator (trend descriptor).
- Accumulation/distribution indicator (price and volume divergence).

The analysis also takes into account the sentiment of news related to each stock. The sentiment score (ss) is computed as an average per-day and per-stock score, ranging from -1 to 1. A positive score ($ss(s) > 0$) indicates a positive market sentiment about the stock, while a negative score ($ss(s) < 0$) reflects a negative sentiment. English-written news articles on S&P500 stocks from the period 2007-2018 are collected from Reuters, and VADER [166], a rule-based sentiment analysis tool, is applied to perform sentiment analysis. Approximately 5253 news articles per stock are considered, with the number of daily news varying due to the popularity of certain stocks.

Fundamental indicators, which encompass the economic and financial factors influencing the stock and underlying assets, are also examined. The study considers well-established indicators such as Earnings Before Interests Taxes Depreciation and Amortization (EBITDA), Return On Equity (ROE), Return On Assets (ROA), Research & Development investments (R&D), and Net Income [167].

[1]Data crawled from AlphaVantage (https://www.alphavantage.co/)

4.1.3 Financial data summarizer

The proposed method for summarizing financial time series is outlined in Figure 4.1. The summarization pipeline consists of the following steps:

1. **Financial data encoding**. This step involves transforming the raw time series and news data into a unified vector representation of the stocks. The encoding captures price-related trends, momentum, seasonality, and market sentiment.
2. **Quantifier/summarizer evaluation**. In this step, the values of summarizers and quantifiers for each stock and reference time period are estimated based on the encoded stock representation.
3. **Protoform generation**. The objective of this step is to compose the explainable summaries of the financial time series and calculate the corresponding quality indices.

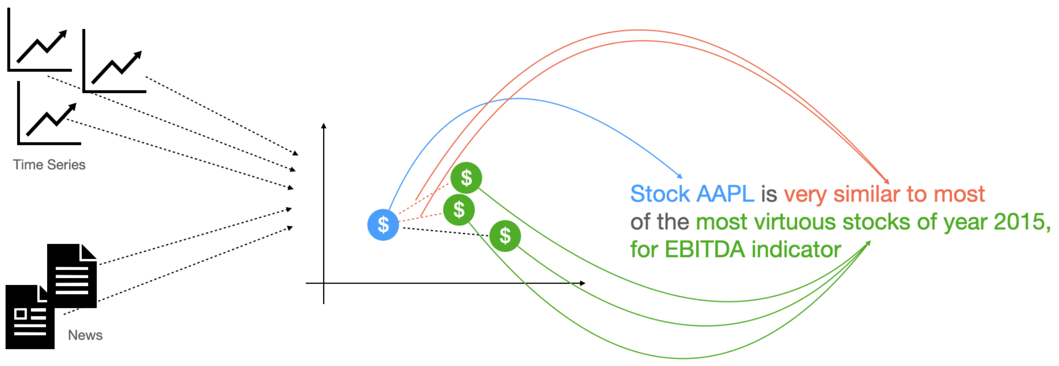

Time Series
News
$
Stock AAPL is very similar to most
of the most virtuous stocks of year 2015,
for EBITDA indicator

Fig. 4.1 Sketch of the time series summarization system.

Financial data encoding In this step, multimodal financial data is encoded into a unified and high-dimensional vector space. Each stock is represented by a vector in this latent space, which captures its essential price-related characteristics such as trends, seasonalities, and momentum, along with the sentiment derived from news articles.

To achieve this, the time series and news data undergo a two-step process [168, 169]. First, they are transformed into a discrete sequence of symbols using a SAX

(Symbolic Aggregate approXimation) representation [170]. Then, the established Signal2Vec encoder [165] is employed to encode the sequences.

The SAX representation condenses the daily multimodal information of stock prices, technical indicators, and news sentiment scores into a unique symbol. By using Signal2Vec, discrete sequences of various time periods (e.g., yearly periods) are encoded and associated with the corresponding stock identifier. This approach enables the description of the underlying behavior of the same stock by utilizing sequences that pertain to it.

Quantifier and summarizer estimation Quantifiers and summarizers play a crucial role in comparative summaries as they indicate the degree of agreement between the summary claim and the analyzed data.

To determine reliable values for quantifiers and summarizers, the multimodal stock vector space trained in the previous step is employed. Algorithm 4 describes the data-driven procedure employed for quantifier and summarizer estimation. In the algorithm, s_1 and s_2 denote the stocks under consideration for summary types *virtuos_stocks*, *year_to_stock*, and *virtuos_multivariate*. While $v(s_1)$ and $v(s_2)$, respectively, denote the corresponding vectors encoding the time series and news contents for each stock.

For every fundamental indicator, the top-ranked stocks (i.e., the stocks in the first quantile for the fundamental indicator) are shortlisted. Then, the distance between each stock in the vector space and each reference stock or group of stocks (such as the sector) is calculated. These vector distances serve to measure the level of similarity with the reference group. Finally, quantifiers and summarizers in natural language are obtained by uniformly dividing the similarity scores of each stock into discrete categories.

Algorithm 4: Quantifier and summarizers estimation

```
Input: stock set $S$, fundamental indicator set $I$, reference time period $T$
Output: $R^i_s$ for all $s \in S$ and $i \in I$
for $s \in S$ do
    for $i \in I$ do
        $d(s_1, s_2) \leftarrow$ compute-distance($v(s_1)$, $v(s_2)$);
        $v^i_s \leftarrow$ value of indicator $i$ for $s$ within the reference time period $T$;
        $q^i_s \leftarrow$ quantile of stock $s$ according to $i$, depending on $v^i_s$;
        if $q^i_s = 1st$ then
            $R^i \leftarrow R^i \cup \{s\}$;
            for $s \in S$ do
                $R^i_s \leftarrow$ Nearest neighbors of stock $s$ according to $i$ calculated
                  using set of distances $d$;
            end
        end
    end
end
```

Protoform generation The generation of the five different types of comparative summaries presented in Table 4.1 is carried out. These summaries, referred to as *protoforms* [161], consist of sentences that provide understandable comparisons between time series data using natural language. Each protoform comprises various fields that represent the following elements:

- *Stock.* The name of the stock under consideration.
- *Sector.* The market sector under consideration.
- *Indicator.* The fundamental indicator under consideration.
- *Quantifier.* A word or phrase indicating the frequency of the summarizer being true.
- *Summarizer.* A word or phrase indicating the level of similarity between the items being compared.
- *Time window.* A time window of interest for the given protoform.

Summary Type	Protoform Template
virtuous_stocks	Stock [stock] is [summarizer] to most of the most virtuous stocks, for [indicator]
sectors	Sector [sector] is [summarizer] to most of the most virtuous stocks of [sector] sector, for [indicator]
years_to_stock	In [quantifier] years the stock [stock_1] has been [summarizer] to the stock [stock_2]
year_to_years	In year [period] the stock [stock_1] has been [summarizer] to [quantifier] years of the stock [stock_2]
virtuous_multivariate	Stock [stock] is [summarizer] to [quantifier] of the most virtuous stocks, for [indicator_1]..[indicator_n]

Table 4.1 Proposed protoforms.

Table 4.2 presents the potential values associated with each field and the corresponding summaries in which they appear. Further details about the proposed protoforms are provided below.

Summaries of type *virtuous_stocks* compare a single stock to the most virtuous stocks, using a fundamental indicator as the reference metric for stock virtuosity. Conversely, summaries of type *sectors* compare market sectors instead of individual stocks. Additionally, summaries of type *virtuous multivariate* allow for the inclusion of multiple indicators to define virtuous stocks and specify a percentage of reference stocks with a given level of similarity.

The summaries of types *years_to_stock* and *year_to_years* perform time-based comparisons between time series. Specifically, summaries of the former type indicate for how many years one stock has been similar to all years of another stock. Whereas, summaries belonging to the latter compare a single year of one stock to all the years of another one, defining a level of similarity and indicating the corresponding number of years

4.1.4 Experiments

4.1.4.1 Intrinsic evaluation

The summaries generated are evaluated using a set of reference quality metrics originally introduced in [161]. These metrics are normalized to a range of zero to one. Below is a brief description of the metrics considered:

Field	Values	Summaries
[stock]	Stock ticker choosen for the comparison	virtuous_stocks years_to_stock year_to_years virtuous_multivariate
[summarizer]	very similar, fairly similar, not similar	all
[indicator]	EBITDA, ROE, ROA, R&D, Net Income	virtuous_stocks sectors virtuous_multivariate
[sectors]	Market sector (e.g. Energy, Healthcare, ...)	sectors
[quantifier]	none, few, many, all	years_to_stock year_to_years
[period]	reference year	year_to_years
[quantifier]	percentage of reference stocks with given level of similarity	virtuous_multivariate

Table 4.2 Fields of the protoforms in Table 4.1.

- *Degree of truth (T1)*: Measures the truthfulness of the quantifier-summarizer pair expressed by the summary. Applicable only to summary types year_to_years and years_to_stock.

- *Degree of Imprecision (T2)*: Evaluates the precision of the summary in relation to the entire data collection.

- *Degree of covering (T3)*: Indicates the percentage of data instances covered by the summary statement.

- *Degree of Appropriateness (T4)*: Quantifies the discrepancy between the observed values of summarizers and the expected values. This metric is relevant only for summaries of type virtuous_multivariate.

Table 4.3 presents representative examples of summaries from different types, along with their corresponding metric values. The generated summaries can be ranked based on decreasing coverage and precision to identify the most reliable stock explanations. For instance, the *sectors* summaries allow end-users to compare the market sector *Energy* with *Utilities* and *Industrial*, respectively. The Degree of covering (T3) indicates that these summaries are supported by approximately half of the covered data instances. According to the Degree of Imprecision (T2), their precision is nearly maximal (99%) in both cases.

Summary Type	**Summary**	**T1**	**T2**	**T3**	**T4**
virtuous_stocks	Stock DU is not similar to most of the most virtuous stocks, for the ROA indicator.		1	0.99	
virtuous_stocks	Stock AAPL is very similar to most of the most virtuous stocks, for the ROA indicator.		0.99	0.41	
sectors	Most of the stocks of Energy sector, has been not similar to most of to the most virtuous stocks of Utilities sector for the ROA indicator.		0.99	0.47	
sectors	Most of the stocks of Energy sector, has been very similar to most of to the most virtuous stocks of Industrial sector for the R&D indicator.		0.99	0.65	
years_to_stock	In few years the stock AAPL has been fairly similar to the stock SYF	0.83	0.90	0.17	
years_to_stock	In most years the stock AAPL has been very similar to the stock ANSS	0.77	0.74	0.42	
year_to_years	In year 2015 the stock HPE has been very similar to few years of the stock JEF	0.71	0.71	0.14	
year_to_years	In year 2015 the stock HPE has been fairly similar to most years of the stock FDX	0.93	0.75	0.33	
virtuous_multivariate	Stock FITB is fairly similar to 22% of the most virtuous stocks, for the ROE and the Net Income indicator.		0.93	0.22	0.24
virtuous_multivariate	Stock FITB is fairly similar to 32% of the most virtuous stocks, for the ROE and the ROA indicator.		0.9	0.32	0.17

Table 4.3 Examples of generated summaries.

4.1.4.2 Extrinsic summary validation

The usability of the per-stock summaries is validated through extrinsic evaluation.

Two examples of summaries from the *Sectors* type are presented in Table 4.4, comparing the performance of the *Industrials* sector with that of the *Communication Services* and *Materials* sectors, respectively. Figure 4.2 displays the corresponding temporal price variations. The observed trends in the price series align with the summaries: the *Industrials* sector shows a high similarity to the most virtuous stocks in the *Communication Services* sector, while exhibiting weak similarity to the *Materials* sector.

Summary	**T1**	**T2**	**T3**	**T4**
Most of the stocks of Industrials sector, has been very similar to most of to the most virtuous stocks of Communication Services sector for the ROE indicator.		0.99	0.44	
Most of the stocks of Industrials sector, has been not similar to most of to the most virtuous stocks of Materials sector for the ROE indicator.		0,99	0,41	

Table 4.4 Sectors-type summary examples.

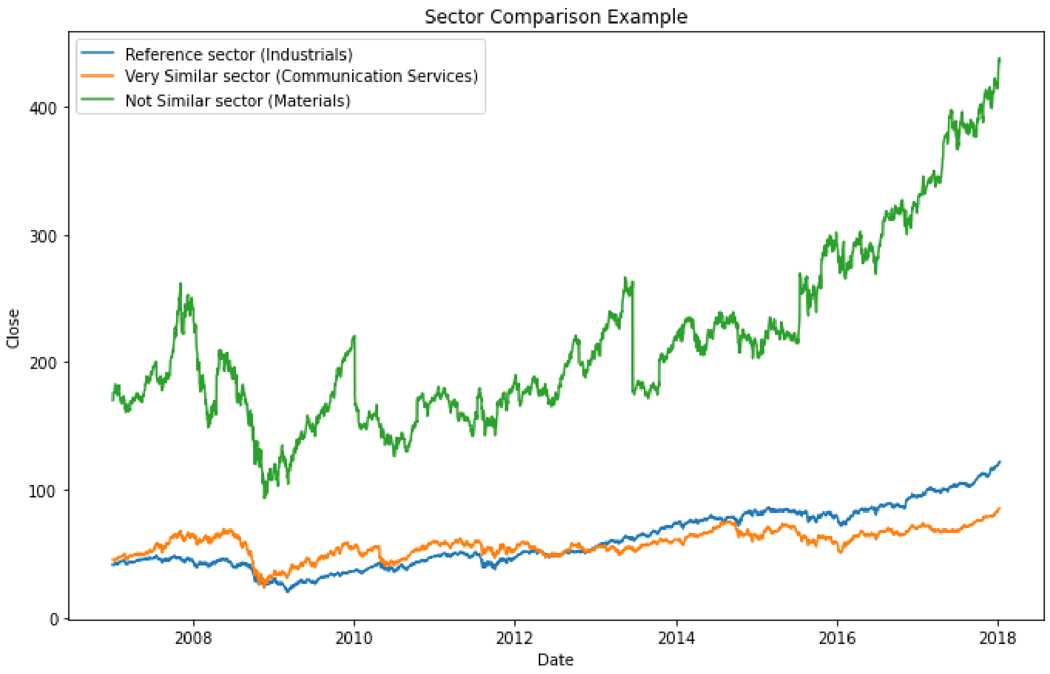


Fig. 4.2 Comparison between the Energy Sector and the most virtuous stocks for Industrial and Utilities Sectors.

Additionally, Table 4.5 includes two summaries of type *years_to_stock* that compare the performance of the Apple stock (AAPL) with that of Vertex Pharmaceuticals

(VRTX) and CME Group (CME) stocks, respectively. According to the generated summaries, the price movements of AAPL are expected to be more similar to VRTX than to CME. This expectation is supported by the historical time series depicted in Figure 4.3, particularly evident during the years 2014-2016.

Summary	**T1**	**T2**	**T3**	**T4**
In most years the stock AAPL has been very similar to the stock VRTX	0,77	0,74	0,42	
In most years the stock AAPL has been not similar to the stock CME	0.77	0.74	0.42	

Table 4.5 years_to_stock summary examples.

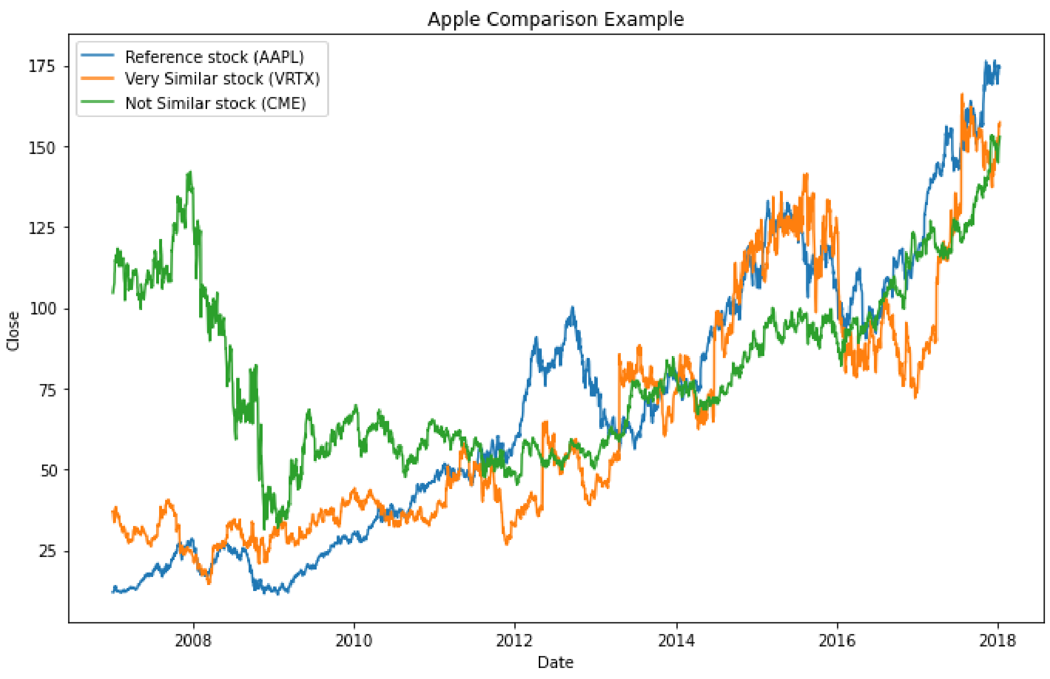


Fig. 4.3 Comparison between APPL, VRTX, and CME stocks.

4.2 Leveraging eXplainable AI to support cryptocurrency investors

Cryptocurrencies are digital assets whose transfers and accounting are cryptographically established through the Blockchain [171]. They have gained popularity as financial assets for online trading, despite not being backed by physical assets. After the introduction of Bitcoin as the first and most famous cryptocurrency [172], numerous other cryptocurrencies have been created, providing increased investment options in the cryptoasset market.

In response to the growing trend of algorithmic trading, researchers have focused on developing cryptocurrency trading systems based on Machine Learning (ML) and Artificial Intelligence (AI). These methods range from classical classification and regression algorithms [173–177] to Deep Learning architectures [178, 68, 179, 180]. The objective is to learn predictive models from historical data, including market information, blockchain-related data, and news, to forecast future price movements. A recent survey on cryptocurrency trading is available in [181].

Even though Machine Learning-based solutions have demonstrated superior performance compared to simpler heuristic methods [181], they often lack transparency. State-of-the-art classification models, including Neural Network-based approaches, are not explainable, limiting domain experts' understanding of the model's decisions. Cryptocurrency markets are influenced by various factors, such as market trends, blockchain characteristics, and investor sentiment. This highlights the need for new approaches that explain the reasoning behind ML in cryptocurrency trading.

This research focuses on employing an established eXplainable AI (XAI) method called SHapley Additive exPlanations (SHAP) [182] to provide domain experts with effective visualizations of ML reasoning in cryptocurrency trading. SHAP quantifies the contribution of different features to classifier predictions, enabling the identification of factors influencing ML-based trading decisions. The following research questions are addressed by this work:

- **Q1**. What are the most discriminative features for cryptocurrency price prediction?

- **Q2**. How can quantitative estimates of the influence of specific features and feature categories on machine learning-based cryptocurrency predictions be provided to cryptocurrency investors?
- **Q3**. How can the statistical dependency of ML feature ranks returned by SHAP be evaluated across different time periods and cryptocurrencies?

To address Q1, a wide range of features computed from the daily price series of 21 cryptocurrencies are examined. These features, established for cryptocurrency trading, include price- and volume-related series, technical indicators summarizing momentum, volatility, and moving averages, as well as blockchain-related features that are unique to cryptoassets.

To address Q2, eXplainable AI methods based on Shapley values [183] are used to provide evidence to cryptocurrency traders regarding the key factors influencing algorithmic trading. A new eXplainable AI tool, CryptoMLE (Cryptocurrency-based Machine Learning Explainer), is introduced for visualizing and monitoring machine learning-based systems, with a focus on blockchain-based features that impact the decision-making process.

To tackle Q3, the Rank Biased Overlap similarity measure [184] is applied to quantify pairwise agreement among the top-10 features selected by SHAP. Additionally, experiments are conducted using feature subcategories and categories rather than individual features.

4.2.1 Related works

Table 4.6 provides a summary of the key characteristics of various approaches to eXplainable AI (XAI) in finance, including the CryptoMLE tool introduced by this work. The analysis of existing and previous works is conducted based on the following aspects:

- *Considered assets*. This includes specific stocks, cryptocurrencies, or a combination of both.
- *Analyzed features*. These are the environmental and market characteristics considered by the classification models, including blockchain-related features for cryptocurrency assets.

- *Availability of a user interface.* This refers to the presence of a user interface that supports domain experts in their decision-making process.
- *Main model for explaining ML-based decisions.* The specific model employed to explain the decisions of ML-based systems (e.g., SHAP [182] for the proposed tool).
- *Resolution of the analyzed data.* Typically, one sample per trading day.
- *Goal of the approach.* The intended purpose of the approach (e.g., support decision-making with data-driven insights for the proposed tool).

The main objective of this research is to present a visual analytics tool that offers AI-based explanations for cryptocurrency investors. It should be noted that the goal is not to propose a new and more effective trading system, but rather to provide experts with an interactive tool based on XAI, enabling them to understand the decisions of algorithmic trading approaches and make informed choices.

Similar to previous works [185–187], CryptoMLE provides a graphical interface for domain experts. However, unlike prior studies on algorithmic trading, it allows interactive collection, analysis, and comparison of data models trained across multiple time periods. Additionally, although CryptoMLE analyzes a large number of cryptocurrencies similar to [187], it also incorporates blockchain-related data.

CryptoMLE relies on the use of SHAP [182] for explanation purposes, while other approaches [186, 188] adopt simpler explainable models such as partitional clustering and decision trees, which are known to be less robust to noise and model bias compared to SHAP. The work presented in [189] represents the first attempt to utilize SHAP in algorithmic trading. However, this current research differs from [189] in the following ways:

- It focuses on short-term cryptocurrency trading instead of long-term portfolio management, comparing classification models for predicting the next-day cryptocurrency price.
- It introduces a graphical tool that supports decision-making and allows experts to interact with the tool to gain insights into specific market trends.
- It analyzes a significantly larger set of cryptocurrencies, encompassing 21 different cryptocurrencies compared to the 8 considered in [189].

Paper	Asset	Features	User Interface		XAI model	XAI resolution	XAI Goal
			Graphical	Interactive			
CryptoMLE	21 cryptocurrencies	BC,MP,TA	Yes	Yes	SHAP [182]	Daily	Decision-making
[190]	S&P index	MA	No	No	Ablation, permutation, added noise, integrated gradients [191]	Daily	XAI model comparison
[192]	CHES120 China	MA	No	No	Custom LightGBM-based model [193]	10s-, 30s, 1min ticks	Matching testing and real-trading performances
[189]	8 crypto	MA	No	No	SHAP [182]	Daily	Portfolio management approach for crypto
[188]	The BTC crypto	BC,MA	No	No	K-means clustering, Decision Tree classifier [84]	Daily	Valuation method for cryptocurrency markets
[185]	The ETH crypto	MA,TA	Yes	No	Adversarial Deep Neural Networks [194]	Daily	Display reversal patterns on candlestick charts [28]
[186]	The S&P stocks	MA,TA,News	Yes	No	Decision Tree classifier	Daily	Identify the most impactful words in business-specific stock market sectors
[187]	18 crypto	MA,Reddit	Yes	No	Ensemble methods, co-occurrence analyses [84]	Daily	Correlation analysis between crypto
[195]	B6, SPF	MA, V	No	No	Decision trees [84], SHAP [182]	Daily	Adapt market data to the Machine Learning pipeline.

Table 4.6 Comparison with prior works. Legend: crypto = cryptocurrency/cryptocurrencies, BC = blockchain, MA = market data, V = Exchanged volumes, TA = technical analysis, B6 = CME Globex British Pound futures, SPF = S&P E-mini Futures.

4.2.2 Data overview and categorization

Historical data was collected for the 21 most popular cryptocurrencies from 2011 to 2018[2]. The data was sampled at a daily granularity for the experiments. However, the analyses conducted can be easily extended to other aggregation levels, whether finer or coarser.

Three main feature categories are considered in this study:

- The *BlockChain-related* (BC) features, which encompass the underlying characteristics of the distributed ledger technology that enables each cryptocurrency [196].
- The *Market Data* (MD) features, which represent the primary cryptocurrency Open-High-Low-Close-Volume (OHLCV) price series, as well as selected summarized features derived from the candlestick chart [197].
- The *Technical Analysis* (TA) features, which include various momentum indicators, volatility indices, and oscillators commonly used in Technical Analysis for both cryptocurrencies and regulated market assets [178].

[2]For cryptocurrencies introduced after 2011, data was collected from the time they became available.

The features are grouped into their respective categories and subcategories, following the hierarchy presented in Table 4.7. A wide range of features, among the most established ones for cryptocurrency trading according to [181], were considered.

4.2.2.1 BlockChain-related features

BC features were collected, these contained various specific properties of the underlying blockchain architecture and were aggregated on a daily basis. The BC category comprises 30 features that encompass different aspects addressed by the following subcategories: *Address*, *Economics*, *Exchange*, *Fees and Revenues*, *Market*, *Mining*, *Network usage*. These features are likely to exhibit direct or indirect relationships with the bid and ask prices of cryptocurrencies, thus potentially considered relevant by the Machine Learning model for accurate price predictions.

The diverse nature of the technologies supporting each cryptocurrency emphasizes the significance of cross-cryptocurrency analyses of BC features in understanding the reasoning behind Machine Learning predictions. For instance, a detailed examination of blockchain supply and mining features can reveal fluctuations in the level of interest among cryptocurrency investors towards specific virtual assets.

4.2.2.2 Market data features

The temporal patterns in cryptocurrency prices are captured by MD features [180]. The dataset gathered dataset consists of various data components, including the original Open-High-Low-Close-Volume (OHLCV) price series, the differences obtained from the Seasonal-Trend Decomposition using Loess (STL) [198] technique, and the properties related to the shapes of the candles in the candlestick chart [197].

4.2.2.3 Technical analysis features

Technical analysis offers a consolidated overview of trends related to price and volume [28]. These trends are derived from historical price and volume data using the TA-Lib Python library[3], which provides various tools for technical analysis.

[3]https://ta-lib.org/

Category	**Subcategory**	**Description**
Blockchain	Addresses	Metrics representing an index of network activity and interest.
	Economics	Metrics regarding the ratio of the USD network value divided by the adjusted transfer value (in USD).
	Exchange	Metrics representing the currency flow for known centralized exchange addresses, for both deposit and withdrawals.
	Fees and Revenues	Metrics covering the network's efficiency in terms of transfer costs, representing fees for doing operations on the blockchain such as transactions and smart contract execution.
	Market	Metrics covering the economic aspects of cryptocurrency markets such as capitalization, BTC exchange price, ROI and volatility returns.
	Mining	Metrics representing protocol-specific parameters.
	Network Usage	Metrics covering blockchain activity in the form of mined block and their size.
	Supply	Metrics that aim to explain token supply and its distribution among wallets.
	Transactions	Metrics addressing transferred value and throughput of the network.
Market data	Prices	Features directly derived from Open, High, Low, Close prices of the current timestamp.
	Volume	Features directly derived from the trading volume of the current timestamp.
	Volatility	Features directly derived from current volatility of the currency.
	History	Features derived from the historical time series of Open, High, Low, Close prices and Volume.
	Candlestick Analysis	Features concerning the analysis of the candlesticks shapes.
Technical Analysis	Trend Indicators	Trend-following indicators whose values help assess the direction and strength of established trends.
	Momentum Indicators	Indicators used to determine the strength or weakness of a stock's price.
	Volatility Indicators	Indicators measuring how far the security moves away from its mean price.
	Volume Indicators	Indicators representing a security's bull and bear power.

Table 4.7 Categories and subcategories of the features present in the dataset.

The TA features encompass significant price-related characteristics of cryptocurrencies, including momentum, volatility, and oversold/overbought conditions.

These features have been demonstrated to be pertinent in cryptocurrency trading as well [178].

4.2.3 SHapley Additive exPlanations values

SHapley Additive exPlanations (SHAP) [182] is a technique used to provide explanations for individual predictions. The method utilizes the concept of the Shapley value, whose applications to eXplainable AI rely on coalitional game theory [183].

4.2.3.1 The Shapley value

In a set of players denoted as $\mathscr{P}$=$\{P_1, P_2, \ldots, P_n\}$, a player coalition $\mathscr{C}$ is a subset of $\mathscr{P}$ that cooperate to achieve a specific task. The utility $\mathscr{U}(\mathscr{P})$ measures the payoff or benefit obtained by the coalition for the given task. The marginal utility $\mathscr{U}(P_j)$ quantifies the additional contribution that a new player P_j brings to the existing coalition $\mathscr{P}$, expressed as:

$$\mathscr{U}(P_j) = \mathscr{U}(\mathscr{P} \cup P_j) - \mathscr{U}(\mathscr{P})$$

The Shapley value [183] is defined as the expected value of the marginal contribution $\mathscr{U}(P_j)$ over all possible coalitions. It is computed as follows:

$$\mathscr{S}\mathscr{V}_i = \frac{1}{n} \sum_{\mathscr{S} \subseteq \mathscr{N} \setminus P_i} \frac{\mathscr{U}(\mathscr{P} \cup P_i) - \mathscr{U}(\mathscr{P})}{\binom{n-1}{|\mathscr{C}|}}$$

However, calculating the exact Shapley value requires enumerating all possible coalitions, which is computationally infeasible in real-world scenarios.

4.2.3.2 Additive feature attribution methods

In the context of a training dataset comprising a set of features $\mathscr{F}$=$\{F_1, F_2, \ldots, F_n\}$, each individual feature's value is considered as a player in a coalition. The total number of features, denoted as n, represents the maximum coalition size.

In the given context, a complex prediction model denoted as f is trained on a dataset consisting of instances $\mathscr{F}$. For simplicity, the financial forecasting model assumes the prediction of the next-day closing price direction (i.e., *Uptrend* or *Downtrend*) of a specific cryptocurrency based on the past samples observed within the last W days. It is important to note that, for the sake of simplicity, the *Stationary* class (neither Uptrend nor Downtrend) is disregarded in this scenario.

The objective is to obtain explanations for the prediction model f that provide insights into the effects of the features in $\mathscr{F}$. More specifically, our goal is to explain the prediction $f(x)$ for a given instance x of $\mathscr{F}$ by assessing the contribution of each individual feature.

Within this context, the Shapley value of a feature F_i indicates how the *payout* should be distributed fairly among the features. It quantifies the effect of the individual feature F_i on the prediction outcome. To generalize the players as sets of feature values, an additive feature attribution method is employed to linearly combine the individual Shapley values.

The explanation model g is defined as a linear combination of binary features associated with each feature F_i:

$$g(z') = \phi_0 + \sum_{i=1}^{n} \phi_i \cdot z'_i \ , \ z' \in 0,1^n$$

Where z'_i is a binary variable indicating whether a feature is present (z'_i=1) or absent (z'_i=0). The attribution value ϕ_i quantifies the effect of feature F_i on the prediction $f(x)$. The explanation model combines the effects of individual feature attributions to approximate the output.

where z'_i is a binary variable denoting either the presence of a feature (z'_i=1) or its absence (z'_i=0). ϕ_i is the F_i's attribution value, which quantifies the effect of F_i on $f(x)$. The explanation model sums the effect of all individual feature attributions approximating the output.

4.2.3.3 The SHAP explanation model

In [182] Shapley values are employed to provide explanations for Machine Learning models. This is achieved by using sampling approximations to estimate the original Shapley expression. Specifically, the effect of removing a variable from the model is approximated by integrating over samples obtained from the training dataset.

The generation process of the SHAP model involves several key steps:

1. Generate random sample coalitions z'' of $m<n$ features in $\mathscr{F}$, where $z'' \in 0,1^m$.
2. Sample coalitions to valid instances.
3. Train a regression model on the generated instances, whose target is the prediction for a coalition.

To get from coalitions of feature values to valid data instances (Step 2), instance values are taken from the instance x we want to explain for all features that are present in the coalition (z''=1), whereas the other features are randomly sampled from the training dataset instances for all the absent features (z_i''=0).

The regression function (Step 3) corresponds to the weighted linear explanation model g previously defined following the additive feature attribution method.

4.2.4 The CryptoMLE tool

Receiving advice from algorithmic advisors is increasingly popular among financial analysts [199]. However, relying on sophisticated Machine Learning models trained on extensive datasets poses a significant risk in financial market forecasting. These ML models often function as black boxes, and domain experts are reluctant to trust them.

eXplainable AI models offer insights into ML algorithms by indicating the importance of features and their impact on ML predictions [200]. They provide both local and global explanations. In the case of local explanations, the focus is on a specific instance x, estimating the effects of features in $\mathscr{F}$ on $f(x)$ [182]. Conversely, global models summarize the main patterns that drive ML decisions across instances. In this work, the combination of local explanations from SHAP is used to model the global influence of individual features, feature subcategories, and categories on ML models.

A visual eXplainable AI tool called CryptoMLE (Cryptocurrency-based Machine Learning Explainer) is presented. It supports cryptocurrency traders and investors in monitoring the performance of quantitative Machine Learning-based cryptocurrency predictions. The tool consists of an interactive dashboard that provides a summary of the main feature contributions to the ML price predictions.

Figure 4.4 displays a snapshot of the dashboard interface. The upper plot in the dashboard shows the SHAP time series of the ten most influential features in predicting the "uptrend" class. Its purpose is to explain the functioning of ML within a specific time period and the variation of ML decisions over time. Each time series value (referred to a day d) represents the mean Shapley value of a given feature F_i computed over the preceding W days. The mean Shapley value indicates the effect of F_i on the ML model trained on a specific day d using a sliding window approach.

For instance, according to the SHAP series plot in Figure 4.4, the MD feature *close_resid* appears to be the most influential between August 2017 and April 2018. In the period from May 2018 to December 2018, *close_resid* and *high_resid* share the top position. The SHAP series plot can be useful, particularly for discretionary traders who need to select and monitor a relatively small subset of visual features.

The lower bee-swarm summary plots in the dashboard snapshot are pop-up windows that analysts can activate when they seek insights into the characteristics of the ML model trained on a particular day. These plots display the Shapley values of all instances belonging to a training window of size W (i.e., W points per feature). For the sake of readability, only the top ten features in descending order of Shapley value are visualized.

For example, based on the left-hand side bee-swarm summary plot in Figure 4.4, during the training window from February 2017 to July 2017, *close_resid*, *volume_pct_tag8*, *macd_12_26*, and *low_close_dist_pct_d30* are the only features that receive a significant number of positive Shapley values. Comparing different summary plots over time can be helpful in detecting temporal changes in ML decisions. Traders can manually verify and potentially revise their current trading strategy based on the alarms triggered by the eXplainable AI tool.

To generate the plot, the procedure described in Algorithm 5 is applied, considering one cryptocurrency at a time. Firstly, the dataset D_c containing the data for the cryptocurrency c is split into a training set and a test set. The feature importance scores are then computed based on a general-purpose Machine Learning model trained on D_{train} (e.g., XGBoost [34]). Subsequently, a ranked feature list is generated based on the importance scores, and system hyperparameters are fine-tuned. This initial phase aims to perform feature selection and parameter tuning before training the subsequent models.

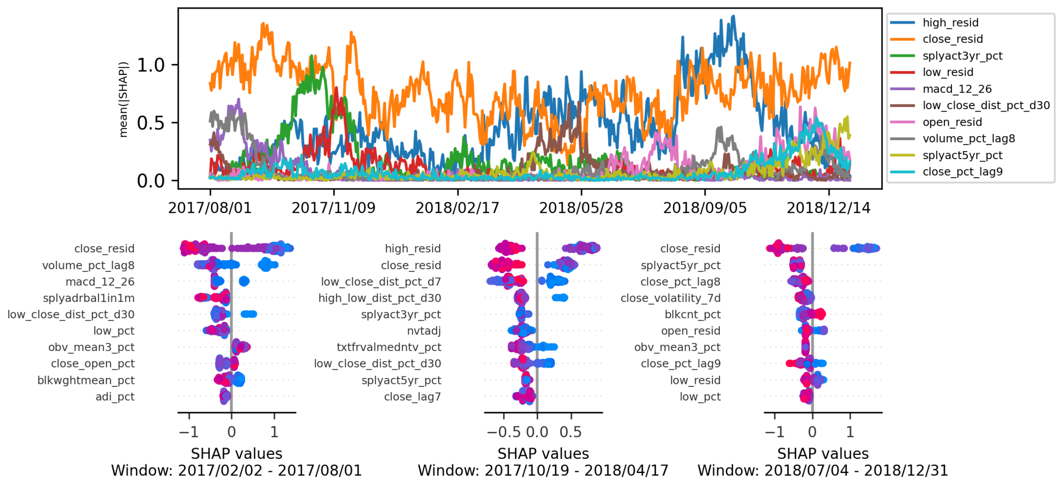


Fig. 4.4 Interactive dashboard snapshot. Uptrend class. LTCUSD. Training window size W=90

To ensure up-to-date and contextualized models, one model is retrained for each test date/time-step t, considering the latest W days preceding t and using the previously determined feature subset and hyperparameters. In other words, a sliding window approach is employed to train ML models tailored to the specific time-step t. Finally, the trained ML models (one per test time-step) are analyzed to compute the SHAP series and summary plots, enabling the visual exploration of ML reasoning at different time points. This process is repeated for all cryptocurrencies of interest.

4.2.5 Experimental results

In this section, a session of machine learning-based forecasting of 21 cryptocurrency prices explained by CryptoMLE is simulated.

The remaining part of the section is structured as follows:

- Section 4.2.5.1 provides clarification on the experimental settings and the reproducibility aspects.
- Section 4.2.5.2 presents the main findings related to question Q1. It examines the most discriminative features for cryptocurrency price prediction. The empirical outcome compares the feature importance plots across different cryptocurrencies.

Algorithm 5: CryptoMLE: Procedure of dashboard generation for a cryptocurrency.

```
Input  : F: feature set;
         D_c: dataset associated with cryptocurrency c;
         W: sliding training window;
         f_pr: Chosen Machine Learning model for the prediction step;
         f_fs: Chosen Machine Learning model for the feature selection step;
output : SH: time series of average Shapely values per-feature;
         BS: bee-swarm summary plots for each point of the test-set;

   /* Train-test dataset split                                    */
1  $\mathbf{D_{train}}, \mathbf{D_{test}} \leftarrow$ SplitDataset($\mathbf{D_c}$)
   /* Feature selection                                           */
2  $\mathbf{M_{fs}} \leftarrow$ TrainFeatureSelectionModel($f_{fs}$, $\mathbf{D_{train}}$, $\mathbf{F}$)
3  $\mathbf{R} \leftarrow$ FeatureImportanceRankingForModel($\mathbf{M_{fs}}$)
4  $\mathbf{F_s} \leftarrow$ SelectFeaturesFromRanking($\mathbf{R}$, $\mathbf{F}$)
   /* Hyper-parameters tuning                                     */
5  $\mathbf{P} \leftarrow$ TuneHyperparameters($f_{pr}$, $\mathbf{D_{train}}$, $\mathbf{F_s}$)
   /* Dashboard generation                                        */
6  foreach time-step $t \in \mathbf{D_{test}}$ do
7      $\mathbf{M_{pr}} \leftarrow$ TrainPredictionModel($f_{pr}$, $\mathbf{D_{(t-W,t)}}$, $\mathbf{F_s}$, $\mathbf{P}$)
8      $\mathbf{BS_t} \leftarrow$ ProduceBeeswarmPlot($\mathbf{M_{pr}}$)
9  $\mathbf{SH} \leftarrow$ ProduceShapTimeSeries($\mathbf{BS_t}$)
10 return $\mathbf{SH}$, $\mathbf{BS_*}$
```

- Section 4.2.5.3 addresses question Q2. It focuses on providing cryptocurrency investors with quantitative estimates of the influence of specific features and feature categories on machine learning-based cryptocurrency predictions. The empirical outcome includes selected SHAP series and bee-swarm summary plots that highlight interesting trends in the analyzed cryptocurrencies.

- Section 4.2.5.4 investigates question Q3. It explores how to evaluate the statistical dependence of the ML feature ranks returned by SHAP in different time periods and for different cryptocurrencies. To address this, the pairwise agreement between the shortlisted feature ranks is assessed using the Rank Biased Overlap similarity measure [184].

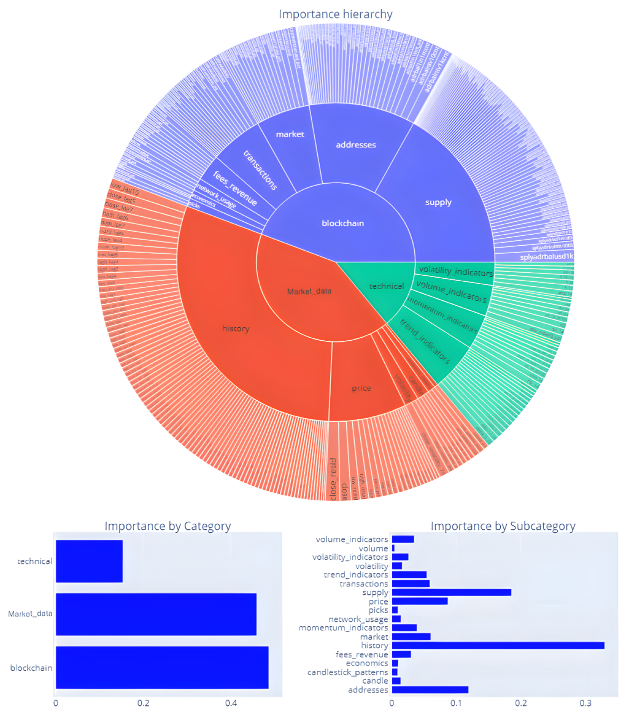


Fig. 4.5 Hierarchical mean feature importance over all the analyzed cryptocurrencies.

4.2.5.1 Experimental design

The experiments were conducted in a single-node setting on an HPC facility. The node used was equipped with Ubuntu 20.04.2 LTS, an Intel(R) Xeon(R) Gold 6140 CPU @ 2.30GHz with 8 CPU threads, and 40 GB of RAM.

For both classification and feature importance estimation, the XGBoost classifier from the SK-Learn library [34] was chosen as the representative ML model. This classifier is known for its efficiency and accuracy.

To execute SHAP [182], the publicly available code provided by the authors of the paper was utilized.

4.2.5.2 Feature importance across cryptocurrencies

The feature importance scores computed by the XGBoost ML model over all cryptocurrencies are presented in the pie charts shown in Figure 4.5. Similarly, Figures 4.6-4.8 separately show the feature importance scores computed over specific cryptocurrencies, respectively BTC, BCH, and ETH) . BTC is widely recognized

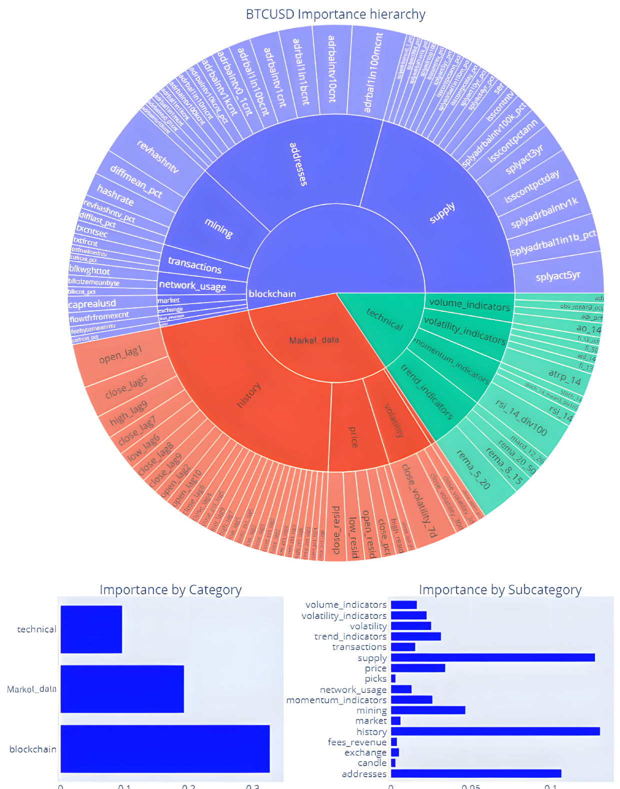


Fig. 4.6 Hierarchical mean feature importance for BTCUSD.

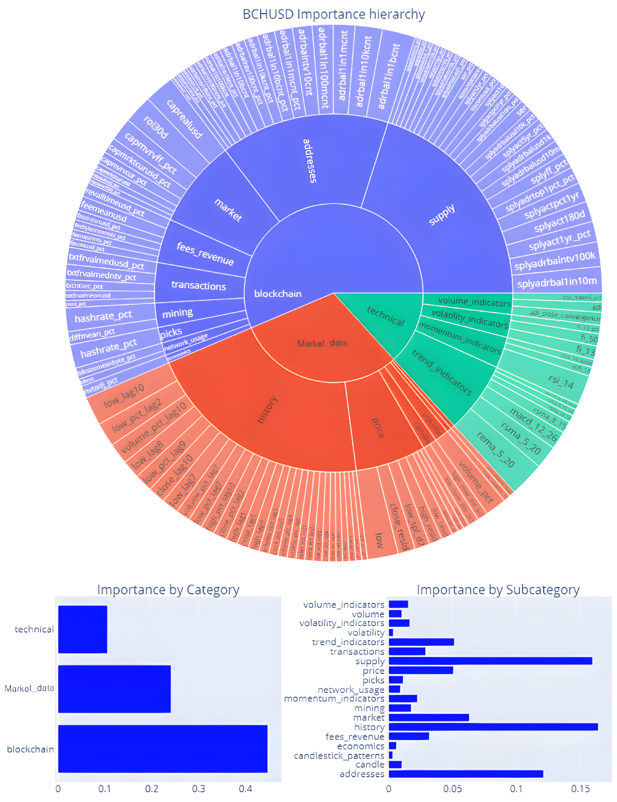


Fig. 4.7 Hierarchical mean feature importance for BCHUSD.

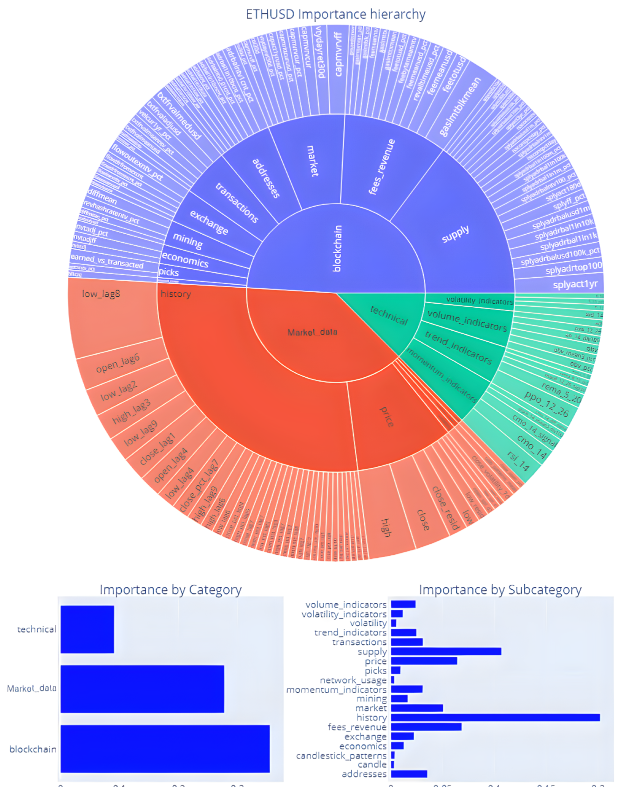


Fig. 4.8 Hierarchical mean feature importance for ETHUSD.

as the most prominent cryptocurrency, while BCH is a fork of BTC, and ETH is another highly popular cryptocurrency.

In Figure 4.5, the outermost circular crown of the pie chart represents the average importance scores per feature, considering all cryptocurrencies. Notably, price-related features like *close_resid* (which denotes the Seasonal-Trend decomposition using LOESS of the closing price series [198]) have demonstrated significant relevance in predicting future cryptocurrency prices. However, the importance of selected features varies across different cryptocurrencies and also encompasses blockchain-related features. For instance, *hashrate_pct*, which indicates the computational capacity of miners or mining networks, holds particular significance for BitCoin casH (BCH) due to its focus on addressing efficiency concerns in the established BTC cryptocurrency. Conversely, it is less relevant for Ethereum (ETH) since ETH has weaker correlations with BTC.

To provide a broader perspective on the discriminative features for each cryptocurrency, the feature importance scores are aggregated by subcategory and category, as shown in the inner crowns and bar charts in Figure 4.5. The most influential features belong to the *Blockchain* category (average score 0.48), followed by *Market data* (0.46) and *Technical analysis* (0.16). This implies that cryptocurrency traders should prioritize monitoring blockchain-related features rather than solely focusing on price-related features like moving averages and momentum.

The most influential subcategories include *Supply* (within the BC category), which pertains to supply chain properties, *History* (within the MD category), which relates to historical cryptocurrency prices, and *Addresses* (within the BC category), which represents blockchain network activity metrics. By analyzing these feature subsets, experts can effectively disregard nearly 70% of the original features. Additionally, the variability in feature importance across different cryptocurrencies is notable, as depicted in Figures 4.6-4.8. For example, in the case of ETH, the significance of blockchain-related features is slightly lower compared to BTC and BCH, possibly due to the primary influence of blockchain architecture on the price movements of Bitcoin-related assets. Ethereum (ETH) exhibits partial correlation with BC and weak dependency on blockchain-related properties such as hash rate and transaction counts.

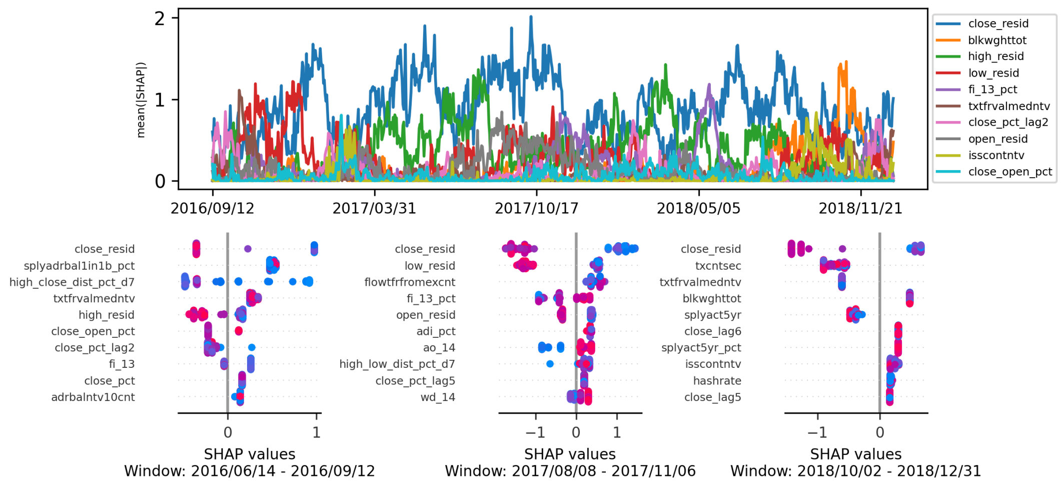


Fig. 4.9 Interactive dashboard snapshot. Uptrend class. BTCUSD. Sliding training window size W=90.

4.2.5.3 Visual explanations

The dashboard snapshots for six combinations of cryptocurrencies and prediction classes are presented in Figures 4.9, 4.11, and 4.13 for the Uptrend class, and Figures 4.10, 4.12, and 4.14 for the Downtrend class.

In Figure 4.9, the line chart displays the average Shapley value variations over time for the top-10 most influential features related to the Uptrend class. While certain features such as *close_resid* consistently demonstrate high relevance across all time periods, others exhibit varying levels of influence. These dynamically influential features can be incorporated into trading system models based on feedback obtained from the eXplainable AI tool. Traders can also utilize this information to assess the reliability of predictions. If features associated with the highest absolute Shapley values align with traders' prior knowledge of remarkable features, it enhances their confidence in the predictions and their incorporation into cryptocurrency trading strategies. Essentially, the visual explanation provided by CryptoMLE serves two purposes: (i) understanding the rationale behind ML decisions and (ii) identifying potentially interesting cryptocurrency-specific patterns that are worth considering in future trading activities.

The bottom charts in Figure 4.9, referred to as SHAP Summary Plots, present the Shapley values calculated for three representative time windows of size W. Each chart illustrates the Shapley values associated with the top-10 most influential fea-

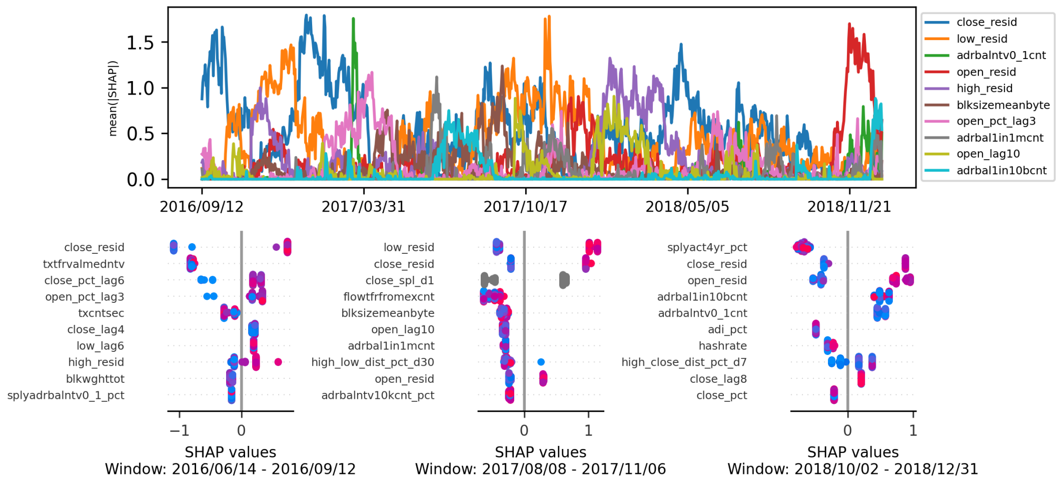


Fig. 4.10 Interactive dashboard snapshot. Downtrend class. BTCUSD. Sliding training window size W=90

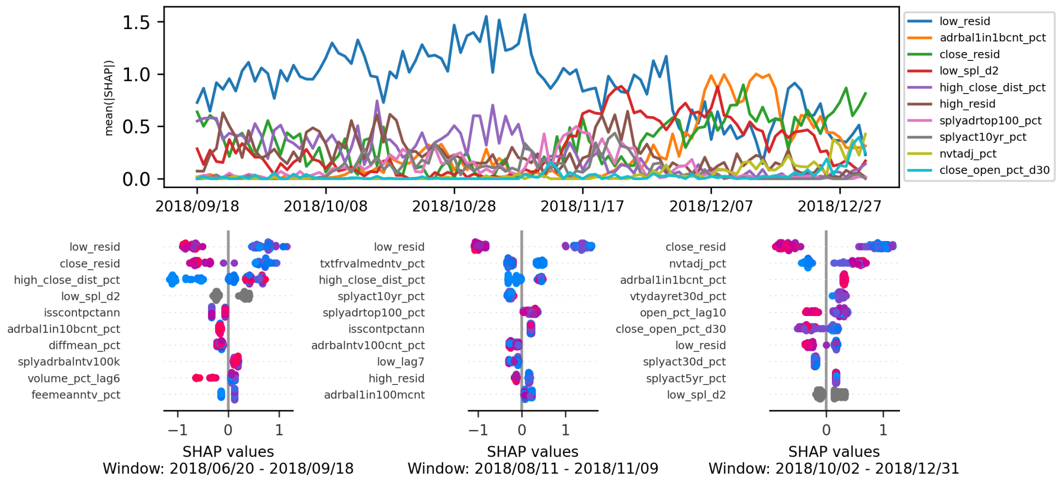


Fig. 4.11 Interactive dashboard snapshot. Uptrend class. BCHUSD. Sliding training window size W=90

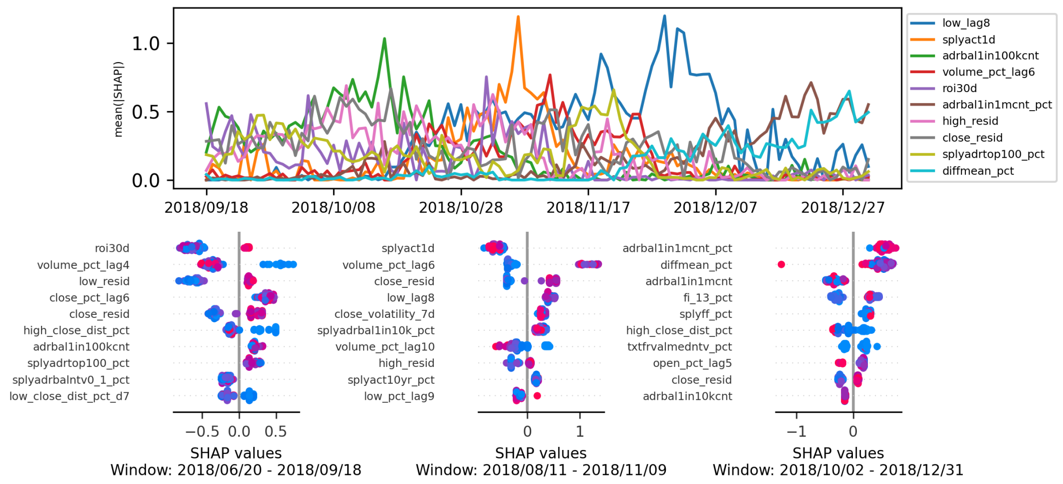


Fig. 4.12 Interactive dashboard snapshot. Downtrend class. BCHUSD. Sliding training window size W=90

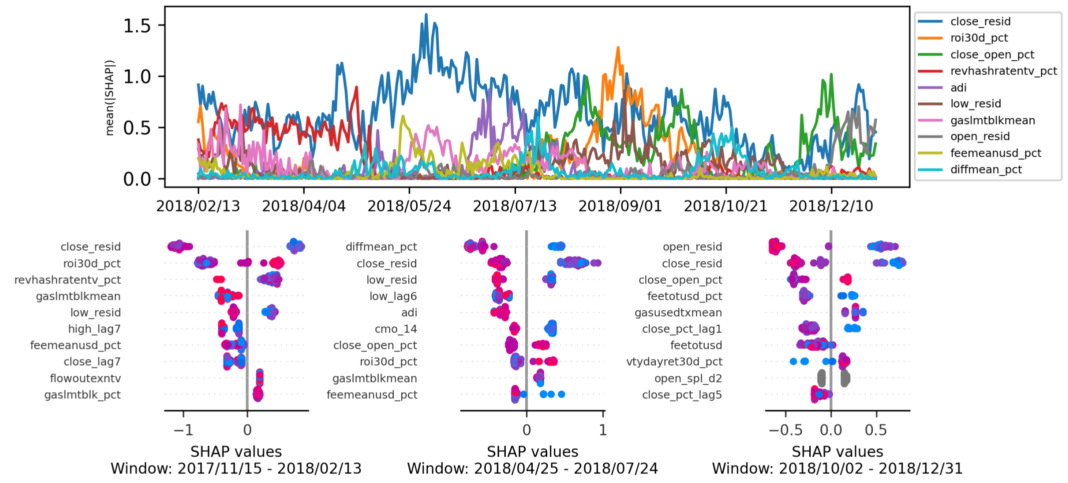


Fig. 4.13 Interactive dashboard snapshot. Uptrend class. ETHUSD. Sliding training window size W=90

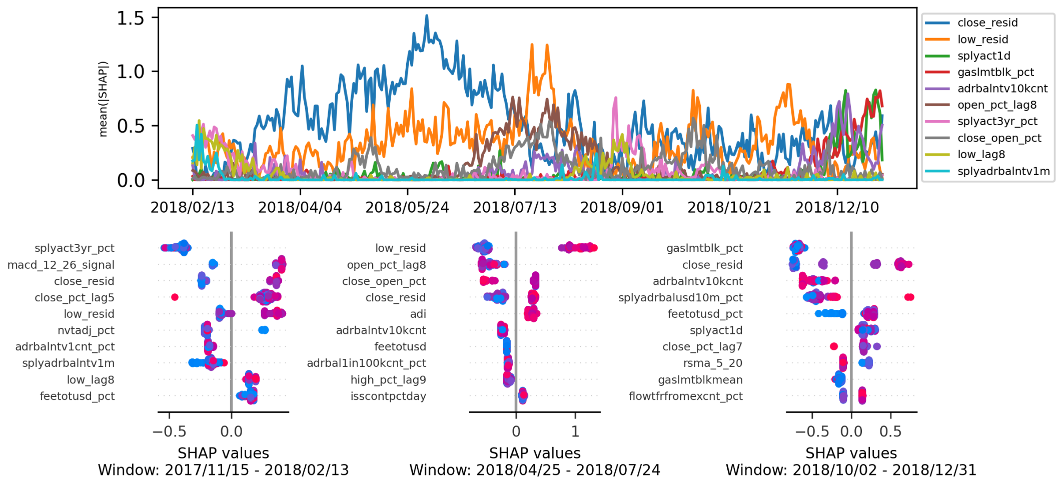


Fig. 4.14 Interactive dashboard snapshot. Downtrend class. ETHUSD. Sliding training window size W=90

tures for the predictions made within the respective time window. The absolute Shapley value indicates the strength of feature influence, while its sign indicates whether the feature has a positive or negative impact on the Uptrend label prediction. Shapley values close to -1 or 1 indicate that a feature significantly affects the prediction compared to others. The variability observed in the summary plots over time strongly depends on market conditions. For instance, in the last quarter of 2018, the AI model predictions were primarily influenced by historical price series, whereas in previous quarters of 2018, the influence of blockchain-related features was more prominent. Based on these findings, domain experts can investigate the reasons behind such a strategy change to evaluate the reliability of the algorithmic trading approach. Specifically, during the last quarter of 2018, all BitCoin-related assets experienced a significant market downturn, indicating the prevailing market downtrend for algorithmic trading systems.

Figure 4.10 provides similar information but focuses on the Downtrend class, explaining the features that had the most impact on the prediction of the Downtrend label. Some features are relevant for predicting both class labels, while others are specific to each class.

Figures 4.11-4.12 and Figures 4.13-4.14 present similar information for BCH and ETH, respectively. It can be observed that certain top features are shared between BTC and BCH, whereas ETH is more influenced by other blockchain-related features. Most of the top feature categories are consistent across all three cryptocurrencies.

Table 4.8 Most influential features for class Uptrend.

Crypto	top1_feature	top1_subcategory	top2_feature	top2_subcategory	top3_feature	top3_subcategory
ADA	close_resid	market_data_prices	splyact180d_pct	blockchain_supply	adi_pct	technical_analysis_volume
BCH	close_resid	market_data_prices	nvtadj_pct	blockchain_economics	adrbal1in1bcnt_pct	blockchain_address
BNB	close_open_pct_d30	market_data_candlestick_analysis	splyadrbalntv100k	blockchain_supply	capact1yrusd	blockchain_market
BTC	close_resid	market_data_prices	txcntsec	blockchain_transactions	txtfrvalmedntv	blockchain_transactions
BTG	close_resid	market_data_prices	high_low_dist_pct_d7	market_data_candlestick_analysis	low_pct_lag4	market_data_history
DASH	close_resid	market_data_prices	adrbal1in1mcnt_pct	blockchain_address	low_pct_lag3	market_data_history
DOGE	close_open_pct_d30	market_data_candlestick_analysis	close_resid	market_data_prices	txtfrvalmedntv	blockchain_transactions
EOS	close_resid	market_data_prices	open_pct_lag6	market_data_history	low_pct_lag3	market_data_history
ETC	adrbalntv0_01cnt	blockchain_address	gaslmtblk	blockchain_fees	gaslmttx	blockchain_fees
ETH	open_resid	market_data_prices	close_resid	market_data_prices	close_open_pct	market_data_candlestick_analysis
LINK	txtfrcnt	blockchain_transactions	splyadrtop1pct_pct	blockchain_supply	caprealusd	blockchain_market
LTC	close_resid	market_data_prices	high_resid	market_data_prices	txtfrcnt	blockchain_transactions
NEO	adi_pct	technical_analysis_volume	close_resid	market_data_prices	txtfrcnt	blockchain_transactions
QTUM	low_resid	market_data_prices	volume_pct_lag9	market_data_history	open_pct_lag5	market_data_history
TRX	high_resid	market_data_prices	close_resid	market_data_prices	high_pct_lag8	market_data_history
WAVE	txcntsec	blockchain_transactions	low_resid	market_data_prices	adi	technical_analysis_volume
XEM	cmo_14	technical_analysis_momentum	high_pct_lag2	market_data_history	close_resid	market_data_prices
XMR	close_resid	market_data_prices	high_lag2	market_data_history	rema_8_15_pct	technical_analysis_trend
XRP	close_resid	market_data_prices	close_open_pct_d3	market_data_candlestick_analysis	close_pct_lag7	market_data_history
ZEC	high_resid	market_data_prices	close_resid	market_data_prices	close_pct_lag3	market_data_history
ZRX	txtfrvalmeanusd	blockchain_transactions	high_pct_lag2	market_data_history	txtfrvaladjntv_pct	blockchain_transactions

Table 4.9 Most influential features for class Downtrend.

crypto	top1_feature	top1_subcategory	top2_feature	top2_subcategory	top3_feature	top3_subcategory
ADA	splyadrtop100_pct	blockchain_supply	splyact1yr_pct	blockchain_supply	close_resid	market_data_prices
BCH	adrbal1in1mcnt_pct	blockchain_address	diffmean_pct	blockchain_mining	adrbal1in1mcnt	blockchain_address
BNB	splyadrbalntylk_pct	blockchain_supply	splyadrbal1in1k_pct	blockchain_supply	low_close_dist_pct_d30	market_data_candlestick_analysis
BTC	splyact4yr_pct	blockchain_supply	close_resid	market_data_prices	open_resid	market_data_prices
BTG	close_resid	market_data_prices	low_resid	market_data_prices	txtfrvaladjusd	blockchain_transactions
DASH	close_resid	market_data_prices	isstotl_isstot365_pct	blockchain_supply	high_resid	market_data_prices
DOGE	volume_pct_lag3	market_data_history	high_resid	market_data_prices	txtfrvalmedntv	blockchain_transactions
EOS	close_pct_lag8	market_data_history	close_resid	market_data_prices	open_pct_lag4	market_data_history
ETC	close_resid	market_data_prices	splyactever_pct	blockchain_supply	nvtadj	blockchain_economics
ETH	gaslmtblk_pct	blockchain_fees	close_resid	market_data_prices	adrbalntv10kcnt	blockchain_address
LINK	high_close_dist_pct_d3	market_data_candlestick_analysis	splyadrbalusd1m	blockchain_supply	close_resid	market_data_prices
LTC	close_resid	market_data_prices	volume_pct_lag1	market_data_history	close_open_pct_d30	market_data_candlestick_analysis
NEO	close_resid	market_data_prices	low_pct_lag9	market_data_history	low_close_dist_pct	market_data_candlestick_analysis
QTUM	close_pct_lag10	market_data_history	low_resid	market_data_prices	open_lag9	market_data_history
TRX	fi_13_pct	technical_analysis_volatility	close_resid	market_data_prices	high_resid	market_data_prices
WAVE	close_pct_lag7	market_data_history	close_resid	market_data_prices	volume_pct_lag3	market_data_history
XEM	low_resid	market_data_prices	volume_pct_lag2	market_data_history	low_lag4	market_data_history
XMR	close_resid	market_data_prices	txcnt_pct	blockchain_transactions	close_volatility_7d	market_data_volatility
XRP	close_resid	market_data_prices	adrbalntv1mcnt_pct	blockchain_address	volume_pct_lag4	market_data_history
ZEC	close_resid	market_data_prices	close_pct_lag8	market_data_history	low_spl_d1	market_data_prices
ZRX	high_close_dist_pct_d3	market_data_candlestick_analysis	splyadrbal1in10k_pct	blockchain_supply	low_resid	market_data_prices

Tables 4.8 and 4.9 display the top-3 most influential features per cryptocurrency and class in terms of average Shapley value. The obtained results confirm that, for most analyzed cryptocurrencies, the subcategories of the most influential features are independent of the predicted class label.

4.2.5.4 Statistical dependence between feature ranked lists

The feature ranked lists associated with the 21 cryptocurrencies were assessed for agreement using the Rank Biased Overlap similarity measure [184]. The objective was to determine whether ML predictions on different cryptocurrencies are influenced by similar features, feature subcategories, or categories.

Tables 4.10 and 4.11 present the pairwise similarity matrices for the Uptrend and Downtrend classes, respectively. These matrices enable the identification of specific

Table 4.10 Pairwise similarity among cryptocurrencies. Class Uptrend.

	ADA	BCH	BNB	BTC	BTG	DASH	DOGE	EOS	ETC	ETH	LINK	LTC	NEO	QTUM	TRX	WAVE	XEM	XMR	XRP	ZEC	ZRX
ADA	1.00	0.84	0.98	0.84	0.89	0.94	0.94	0.72	0.80	0.78	0.68	0.94	0.85	0.72	0.76	0.90	0.81	0.90	0.88	0.91	0.86
BCH	0.84	1.00	0.85	1.00	0.74	0.89	0.84	0.61	0.66	0.94	0.84	0.84	0.69	0.61	0.59	0.74	0.63	0.71	0.77	0.81	0.78
BNB	0.98	0.85	1.00	0.85	0.90	0.96	0.96	0.74	0.82	0.80	0.70	0.96	0.84	0.74	0.78	0.88	0.79	0.88	0.89	0.93	0.87
BTC	0.84	1.00	0.85	1.00	0.74	0.89	0.84	0.61	0.66	0.94	0.84	0.84	0.69	0.61	0.59	0.74	0.63	0.71	0.77	0.81	0.78
BTG	0.89	0.74	0.90	0.74	1.00	0.84	0.91	0.83	0.65	0.79	0.52	0.91	0.79	0.83	0.87	0.79	0.83	0.96	0.96	0.94	0.76
DASH	0.94	0.89	0.96	0.89	0.84	1.00	0.95	0.77	0.76	0.84	0.74	0.95	0.80	0.77	0.75	0.84	0.73	0.82	0.87	0.92	0.89
DOGE	0.94	0.84	0.96	0.84	0.91	0.95	1.00	0.78	0.73	0.84	0.65	1.00	0.84	0.78	0.82	0.84	0.78	0.89	0.94	0.97	0.86
EOS	0.72	0.61	0.74	0.61	0.83	0.77	0.78	1.00	0.45	0.67	0.40	0.78	0.62	1.00	0.96	0.62	0.66	0.81	0.84	0.81	0.66
ETC	0.80	0.66	0.82	0.66	0.65	0.76	0.73	0.45	1.00	0.54	0.81	0.73	0.74	0.45	0.51	0.90	0.66	0.64	0.62	0.68	0.87
ETH	0.78	0.94	0.80	0.94	0.79	0.84	0.84	0.67	0.54	1.00	0.73	0.84	0.68	0.67	0.65	0.68	0.66	0.75	0.82	0.87	0.72
LINK	0.68	0.84	0.70	0.84	0.52	0.74	0.65	0.40	0.81	0.73	1.00	0.65	0.62	0.40	0.38	0.78	0.54	0.51	0.55	0.60	0.82
LTC	0.94	0.84	0.96	0.84	0.91	0.95	1.00	0.78	0.73	0.84	0.65	1.00	0.84	0.78	0.82	0.84	0.78	0.89	0.94	0.97	0.86
NEO	0.85	0.69	0.84	0.69	0.79	0.80	0.84	0.62	0.74	0.68	0.62	0.84	1.00	0.62	0.66	0.85	0.96	0.80	0.78	0.81	0.81
QTUM	0.72	0.61	0.74	0.61	0.83	0.77	0.78	1.00	0.45	0.67	0.40	0.78	0.62	1.00	0.96	0.62	0.66	0.81	0.84	0.81	0.66
TRX	0.76	0.59	0.78	0.59	0.87	0.75	0.82	0.96	0.51	0.65	0.38	0.82	0.66	0.96	1.00	0.66	0.70	0.85	0.88	0.84	0.67
WAVE	0.90	0.74	0.88	0.74	0.79	0.84	0.84	0.62	0.90	0.68	0.78	0.84	0.85	0.62	0.66	1.00	0.81	0.80	0.78	0.81	0.96
XEM	0.81	0.63	0.79	0.63	0.83	0.73	0.78	0.66	0.66	0.66	0.54	0.78	0.96	0.66	0.70	0.81	1.00	0.85	0.82	0.79	0.75
XMR	0.90	0.71	0.88	0.71	0.96	0.82	0.89	0.81	0.64	0.75	0.51	0.89	0.80	0.81	0.85	0.80	0.85	1.00	0.93	0.90	0.74
XRP	0.88	0.77	0.89	0.77	0.96	0.87	0.94	0.84	0.62	0.82	0.55	0.94	0.78	0.84	0.88	0.78	0.82	0.93	1.00	0.96	0.79
ZEC	0.91	0.81	0.93	0.81	0.94	0.92	0.97	0.81	0.68	0.87	0.60	0.97	0.81	0.81	0.84	0.81	0.79	0.90	0.96	1.00	0.83
ZRX	0.86	0.78	0.87	0.78	0.76	0.89	0.86	0.66	0.87	0.72	0.82	0.86	0.81	0.66	0.67	0.96	0.75	0.74	0.79	0.83	1.00

Table 4.11 Correlations among cryptocurrencies. Class Downtrend.

	ADA	BCH	BNB	BTC	BTG	DASH	DOGE	EOS	ETC	ETH	LINK	LTC	NEO	QTUM	TRX	WAVE	XEM	XMR	XRP	ZEC	ZRX
ADA	1.00	0.88	0.98	0.95	0.76	0.85	0.76	0.56	0.86	0.97	0.83	0.69	0.64	0.56	0.55	0.76	0.66	0.85	0.87	0.69	0.82
BCH	0.88	1.00	0.86	0.87	0.76	0.73	0.76	0.54	0.74	0.85	0.79	0.54	0.47	0.54	0.54	0.62	0.51	0.73	0.75	0.54	0.82
BNB	0.98	0.86	1.00	0.93	0.74	0.87	0.74	0.53	0.86	0.95	0.82	0.71	0.66	0.53	0.52	0.78	0.68	0.87	0.85	0.71	0.80
BTC	0.95	0.87	0.93	1.00	0.84	0.86	0.84	0.66	0.87	0.98	0.89	0.72	0.68	0.66	0.63	0.80	0.70	0.86	0.88	0.72	0.87
BTG	0.76	0.76	0.74	0.84	1.00	0.90	1.00	0.83	0.91	0.82	0.96	0.83	0.79	0.83	0.68	0.90	0.81	0.90	0.92	0.83	0.97
DASH	0.85	0.73	0.87	0.86	0.90	1.00	0.90	0.69	0.99	0.88	0.95	0.87	0.82	0.69	0.57	0.94	0.84	1.00	0.98	0.87	0.93
DOGE	0.76	0.76	0.74	0.84	1.00	0.90	1.00	0.83	0.91	0.82	0.96	0.83	0.79	0.83	0.68	0.90	0.81	0.90	0.92	0.83	0.97
EOS	0.56	0.54	0.53	0.66	0.83	0.69	0.83	1.00	0.71	0.62	0.78	0.64	0.66	1.00	0.84	0.75	0.65	0.69	0.72	0.64	0.80
ETC	0.86	0.74	0.86	0.87	0.91	0.99	0.91	0.71	1.00	0.89	0.95	0.86	0.81	0.71	0.58	0.93	0.83	0.99	0.99	0.86	0.94
ETH	0.97	0.85	0.95	0.98	0.82	0.88	0.82	0.62	0.89	1.00	0.86	0.75	0.70	0.62	0.59	0.82	0.72	0.88	0.90	0.75	0.85
LINK	0.83	0.79	0.82	0.89	0.96	0.95	0.96	0.78	0.95	0.86	1.00	0.81	0.77	0.78	0.65	0.89	0.79	0.95	0.96	0.81	0.99
LTC	0.69	0.54	0.71	0.72	0.83	0.87	0.83	0.64	0.86	0.75	0.81	1.00	0.95	0.64	0.48	0.92	0.98	0.87	0.85	1.00	0.80
NEO	0.64	0.47	0.66	0.68	0.79	0.82	0.79	0.66	0.81	0.70	0.77	0.95	1.00	0.66	0.50	0.88	0.98	0.82	0.80	0.95	0.76
QTUM	0.56	0.54	0.53	0.66	0.83	0.69	0.83	1.00	0.71	0.62	0.78	0.64	0.66	1.00	0.84	0.75	0.65	0.69	0.72	0.64	0.80
TRX	0.55	0.54	0.52	0.63	0.68	0.57	0.68	0.84	0.58	0.59	0.65	0.48	0.50	0.84	1.00	0.59	0.49	0.57	0.59	0.48	0.67
WAVE	0.76	0.62	0.78	0.80	0.90	0.94	0.90	0.75	0.93	0.82	0.89	0.92	0.88	0.75	0.59	1.00	0.90	0.94	0.92	0.92	0.87
XEM	0.66	0.51	0.68	0.70	0.81	0.84	0.81	0.65	0.83	0.72	0.79	0.98	0.98	0.65	0.49	0.90	1.00	0.84	0.82	0.98	0.78
XMR	0.85	0.73	0.87	0.86	0.90	1.00	0.90	0.69	0.99	0.88	0.95	0.87	0.82	0.69	0.57	0.94	0.84	1.00	0.98	0.87	0.93
XRP	0.87	0.75	0.85	0.88	0.92	0.98	0.92	0.72	0.99	0.90	0.96	0.85	0.80	0.72	0.59	0.92	0.82	0.98	1.00	0.85	0.95
ZEC	0.69	0.54	0.71	0.72	0.83	0.87	0.83	0.64	0.86	0.75	0.81	1.00	0.95	0.64	0.48	0.92	0.98	0.87	0.85	1.00	0.80
ZRX	0.82	0.82	0.80	0.87	0.97	0.93	0.97	0.80	0.94	0.85	0.99	0.80	0.76	0.80	0.67	0.87	0.78	0.93	0.95	0.80	1.00

cryptocurrency clusters characterized by relatively high pairwise similarities. For example, XMR and ZEC exhibit a high level of similarity, likely due to their shared focus on privacy aspects.

To further analyze the impact of the time dimension, an additional experiment was conducted to compare the list of feature categories that are more relevant for predicting Uptrend or Downtrend. Three different time periods (P1, P2, P3) were considered to also examine the impact of the time dimension. The results are reported in Table 4.12. The correlations computed for each cryptocurrency in P1, P2, and P3 are presented. In most cases, the correlation value remains stable across the time slots and exceeds 0.7. This indicates that, for the majority of cryptocurrencies, the decision regarding the class label is based on the same feature categories regardless of the predicted label.

Table 4.12 Uptrend/Downtrend correlations

crypto	P1	P2	P3
ADA	0.72	0.62	0.80
BCH	0.82	0.90	0.66
BNB	0.92	0.76	0.80
BTC	0.90	0.83	0.78
BTG	0.96	0.90	0.93
DASH	0.77	0.99	0.97
DOGE	0.87	0.75	0.99
EOS	0.91	1.00	0.93
ETC	0.77	0.65	0.74
ETH	0.69	0.81	0.75
LINK	0.82	0.78	0.71
LTC	0.83	0.94	0.84
NEO	0.94	0.64	0.64
QTUM	0.60	0.76	0.93
TRX	0.76	0.84	0.82
WAVE	0.89	0.75	0.76
XEM	0.59	0.82	0.67
XMR	0.95	0.98	0.79
XRP	0.80	0.82	0.86
ZEC	0.87	0.70	0.87
ZRX	0.59	0.89	0.87

4.3 Findings and discussion

This chapter presented two different approaches to aid investors in better understanding the underlying assets and explaining the motivations of ML-based forecasting models. First, a new approach for generating explainable summaries of financial time series in textual form has been introduced. The key idea involves representing the crucial information about stocks in a unified latent space, which includes price-related time series data and news sentiment scores. By leveraging the vector representation, reliable stock and stock group similarities can be obtained, enabling the automatic estimation of quantifiers and summarizers required to generate the protoforms. Preliminary results indicate that the provided summary examples: (i) achieve satisfactory quality levels according to the metrics defined in [161], (ii) demonstrate coherence with the expected outcomes, and (iii) can be utilized by domain experts to support decision-making.

Second, an Explainable AI tool for forecasting cryptocurrency prices was presented. A visual interface was introduced, enabling domain experts to identify actionable relationships between input data features and Machine Learning predictions. The interactive dashboard includes a SHAP series plot that displays the temporal changes in mean Shapley values associated with the most recent ML predictions. Additionally, there are pop-up summary plots that provide snapshots of the influences of key features at specific time points. The empirical simulation, conducted over an 8-year period, demonstrated the variation in model explanations across 21 cryptocurrencies and 3 reference time periods. This variability was observed in terms of the selected features, as well as feature subcategories and categories.

Chapter 5

Discussion and Conclusions

The field of financial time series analysis has experienced significant advancements in recent years, driven by the integration of machine learning and data science techniques. This thesis aimed at contributing to this evolving field by developing innovative approaches and methodologies to leverage machine learning in financial time series analysis, with the ultimate goal of enhancing decision-making processes and enabling more effective and efficient financial practices. The research conducted in this thesis addressed several fundamental data science problems related to time series analysis and adapted them to the financial domain.

The research questions posed at the beginning of this thesis guided the investigations and provided a framework for exploring various aspects of financial time series analysis. The conclusions drawn from each study shed light on the potential of machine learning techniques in addressing these questions and advancing the understanding of the complexities of financial markets. In this chapter, comprehensive summary of the findings from each study is provided, highlighting their contributions to answering the research questions and offering insights for future research directions.

5.1 Summary of Findings

5.1.1 Research Question 1: Trend is a friend. Ok, but... at which time frequency?

The study presented in Chapter 2 (Section 2.1) addressed this question by investigating the most effective time granularities and model settings for leveraging available data in short-term trading scenarios. The findings indicated that the optimal time frequency for stock trading was not necessarily the most fine-grained or coarse-grained (i.e., in the study performance 120 minutes sampling). The study highlighted the importance of selecting appropriate time frequencies based on the characteristics of the financial data and the specific trading objectives. It should, however, be noted that gathering data at finer-than-daily granularities is not always possible and for this reason most of the other studies presented employ data at daily granularity.

5.1.2 Research Question 2: Can Stock Chart Patterns transcend the myth and prove their utility?

Chapter 2 (Section 2.2) explored the combination of stock chart patterns, a staple of technical analysis-based trading, with data-driven trading signal generation. The study demonstrated the utility of stock chart patterns in filtering machine learning-generated signals, thereby showcasing their complementary nature. This research provided compelling evidence that stock chart patterns, as descriptive features of the time series, play a crucial role in enhancing trading strategies and decision-making processes by effectively gauging the trend.

5.1.3 Research Question 3: Cryptocurrencies vs. stocks. What's new?

Chapter 2 (Section 2.3) focused on comparing cryptocurrencies with equities and exploring the unique behaviors and properties of cryptocurrencies. The study revealed that cryptocurrencies exhibit distinct behaviors that can be leveraged to enhance trading strategies (i.e., the momentum effect). Additionally, Chapter 4 (Section 4.2, studied the application of eXplainable AI techniques to cryptocurrency trading uncovering the usefulness of features derived from the blockchain in forecasting cryptocurrency movements. These research works shed light on the potential advantages and opportunities presented by cryptocurrencies in financial markets.

5.1.4 Research Question 4: *Locally* optimal stock portfolios selection. Possible or not?

Chapter 3 (Section 3.4) addressed the challenge of generating locally optimal stock portfolio selection. The research introduced the integration of an optimization step showcasing its potential in selecting profitable and robust portfolios. The study, nonetheless, emphasized the importance of involving the investor in decision-making aspects to achieve profitable portfolio selection.

5.1.5 Research Question 5: Extracting useful domain related knowledge from financial time series. Really achievable or just wishful thinking?

Chapter 3 (Section 3.3) introduced improvements over a state-of-the-art approach by employing time series clustering to create a data-driven taxonomy. The results demonstrated that a data-driven taxonomy could outperform domain-specific taxonomies in terms of portfolio diversification. Additionally, Chapter 4 (Section 4.1) explored a different method to extract useful domain-related knowledge from financial time series. The research demonstrated the feasibility of generating human-readable summaries through time series embedding, enabling investors to compare the behaviors of different stocks effectively. These findings confirmed that extracting meaningful insights and knowledge from financial time series is indeed achievable and can contribute to more informed decision-making processes.

5.1.6 Research Question 6: Can Machine Learning-based Trading Systems actually be *plug-and-play*? What role should domain experts play?

All the studies conducted in this thesis implicitly or explicitly addressed the role of domain experts and the feasibility of *plug-and-play* machine learning-based trading systems. The findings consistently emphasized the criticality of maintaining investors in the loop. By generating signals, suggestions, and explanations, machine learning-based trading systems can assist traders and investors, but domain experts play a crucial role in interpreting and validating the generated signals. This research highlighted the importance of integrating machine learning approaches with domain expertise to achieve effective and reliable trading systems.

Additionally, the need for maintaining an expert in the loop arises also from the tuning needs of machine learning models. For instance, both Section 2.2 and Section 2.3 report the results of multiple model configurations for the same task, revealing significant variations in performance even under equivalent conditions and over the same assets. These variations demonstrate that model configurations greatly influence their effectiveness and should be tailored to the task at hand. The involvement of experts ensures that the trading systems remain optimized and adapt to evolving market dynamics.

5.1.7 Research Question 7: Can traders and investors trust machine generated signals?

The results from the various studies provided evidence that traders and investors can indeed trust machine-generated signals, given the appropriate tuning possibilities and motivations behind the signal generation. By incorporating domain knowledge, validating the signals, and understanding the underlying models, traders and investors can gain confidence in the machine-generated signals and leverage them for decision-making processes.

5.1.8 Findings discussion

Overall, this PhD thesis has contributed to the field of financial time series analysis by addressing these research questions and providing insights into the application of machine learning in the financial domain. Based on the findings from the different studies, it can be concluded that machine learning techniques have the potential to improve decision-making processes in the financial domain. By leveraging the strengths of these approaches, such as their ability to identify complex patterns and make accurate predictions, traders and investors can gain valuable insights and generate signals that support informed decision-making. However, it is important to recognize that machine learning-based trading systems should not be viewed as entirely "plug-and-play" solutions. The involvement of domain experts, such as investors and traders, is crucial to interpret the generated signals, provide contextual knowledge, and make informed decisions based on the recommendations. Trust between investors and machine-generated signals can be established through adequate tuning possibilities and the alignment of the motivations behind the signals generated with the objectives of the investors.

5.2 Future Directions

The studies presented in this thesis have successfully addressed the research questions and provided valuable insights into the application of machine learning in financial time series analysis. The findings have shed light on various aspects, including time frequency selection, stock chart patterns usefulness, cryptocurrency analysis, portfolio selection, knowledge extraction from financial time series, and the role of domain experts in machine learning-based trading systems. However, given the rapidly evolving nature of the field, new research questions and opportunities for exploration arise constantly.

Different new research projects are currently ongoing. The main ones focuses on the utilization of contrastive learning-based representations in trading and signal generation. Contrastive learning techniques have shown promising results in various domains (e.g., image recognition, NLP, ...) by learning rich representations that capture the underlying structure of the data. Applying contrastive learning methods to financial time series analysis could potentially enhance feature extraction, pattern recognition, and forecasting capabilities. This research project aims to leverage contrastive learning approaches to improve trading strategies and generate more accurate and reliable signals.

Another active research project centers around the use of NLP techniques for financial entities disambiguation. In the financial domain, the disambiguation of entities, such as company or private persons names, can be challenging due to the existence of multiple entities with similar names or acronyms. NLP techniques can aid in accurately identifying and disambiguating financial entities from textual data. This research project seeks to develop robust NLP-based methods to improve the quality of financial data and enhance the performances of fraud detection systems.

Finally, considering the observed importance of involving experts in the decision-making process, it is crucial not to underestimate the profound impact that Large Language Models (LLMs) are having across various fields. Although initial tests indicate that LLMs may not be proficient in retrieving useful information for this particular field, their immense potential in language understanding and generation can be leveraged in numerous ways. These range from replacing the use of protoforms and enhancing textual summaries after having previously defined the content to serving as a bridge between ML-based signal generators and less technically adept investors.

As the field of financial time series analysis continues to evolve, new research questions will arise, necessitating further investigation. By pursuing these future research directions and continuously exploring new research questions, the field of financial time series analysis can further advance, leading to enhanced decision-making processes, improved trading strategies, and more effective financial practices.

References

[1] Ernie Chan. *Algorithmic Trading: Winning Strategies and Their Rationale*. Wiley Publishing, 1st edition, 2013.

[2] Vincent Van Kervel and Albert J. Menkveld. High-frequency trading around large institutional orders. *The Journal of Finance*, 74(3):1091–1137, 2019.

[3] Mohammed J. Zaki and Wagner Meira, Jr. *Data Mining and Machine Learning: Fundamental Concepts and Algorithms*. Cambridge University Press, 2 edition, 2020.

[4] David Enke and Suraphan Thawornwong. The use of data mining and neural networks for forecasting stock market returns. *Expert Systems with Applications*, 29(4):927 – 940, 2005.

[5] O. Bustos, A. Pomares, and E. Gonzalez. A comparison between svm and multilayer perceptron in predicting an emerging financial market: Colombian stock market. In *2017 Congreso Internacional de Innovacion y Tendencias en Ingenieria (CONIITI)*, pages 1–6, 2017.

[6] Rodolfo Torabio Farias Nazario, Jessica Lima e Silva, Vinicius Amorim Sobreiro, and Herbert Kimura. A literature review of technical analysis on stock markets. *The Quarterly Review of Economics and Finance*, 66:115 – 126, 2017.

[7] J. Wang, W. Shang, Z. Liu, and S. Wang. An enhanced lgsa-svm for s p 500 index forecast. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 4176–4183, 2017.

[8] Chih-Fong Tsai and Zen-Yu Quan. Stock prediction by searching for similarities in candlestick charts. *ACM Trans. Manage. Inf. Syst.*, 5(2):9:1–9:21, July 2014.

[9] Ngo Tung Son, Le Van Thanh, Tran Quy Ban, Duong Xuan Hoa, and Bui Ngoc Anh. An analyze on effectiveness of candlestick reversal patterns for vietnamese stock market. In *Proceedings of the 2018 International Conference on Information Management & Management Science*, pages 89–93. ACM, 2018.

[10] Rudra Kalyan Nayak, Debahuti Mishra, and Amiya Kumar Rath. A naive svm-knn based stock market trend reversal analysis for indian benchmark indices. *Applied Soft Computing*, 35:670 – 680, 2015.

[11] Rajashree Dash and Pradipta Kishore Dash. A hybrid stock trading framework integrating technical analysis with machine learning techniques. *The Journal of Finance and Data Science*, 2(1):42 – 57, 2016.

[12] Hongping Hu, Li Tang, Shuhua Zhang, and Haiyan Wang. Predicting the direction of stock markets using optimized neural networks with google trends. *Neurocomputing*, 285:188 – 195, 2018.

[13] Luca Cagliero, Paolo Garza, Giuseppe Attanasio, and Elena Baralis. Training ensembles of faceted classification models for quantitative stock trading. *Computing*, 102(5):1213–1225, 2020.

[14] Xiang Li and Chao Luo. An intelligent stock trading decision support system based on rough cognitive reasoning. *Expert Systems with Applications*, 160:113763, 2020.

[15] Wei Shen, Xiaopen Guo, Chao Wu, and Desheng Wu. Forecasting stock indices using radial basis function neural networks optimized by artificial fish swarm algorithm. *Knowledge-Based Systems*, 24(3):378 – 385, 2011.

[16] Wen-Chyuan Chiang, David Enke, Tong Wu, and Renzhong Wang. An adaptive stock index trading decision support system. *Expert Systems with Applications*, 59:195 – 207, 2016.

[17] Elena Baralis, Luca Cagliero, and Paolo Garza. Planning stock portfolios by means of weighted frequent itemsets. *Expert Syst. Appl.*, 86:1–17, 2017.

[18] Jacopo Fior, Luca Cagliero, and Paolo Garza. Price series cross-correlation analysis to enhance the diversification of itemset-based stock portfolios. In Douglas Burdick and Jay Pujara, editors, *Proceedings of the Sixth International Workshop on Data Science for Macro-Modeling, DSMM 2020, In conjunction with the ACM SIGMOD/PODS Conference, Portland, OR, USA, June 14, 2020*, pages 1:1–1:6. ACM, 2020.

[19] R. T. Gonzalez, C. A. Padilha, and D. A. C. Barone. Ensemble system based on genetic algorithm for stock market forecasting. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 3102–3108, 2015.

[20] P. Chakraborty, U. S. Pria, M. R. A. H. Rony, and M. A. Majumdar. Predicting stock movement using sentiment analysis of twitter feed. In *2017 6th International Conference on Informatics, Electronics and Vision 2017 7th International Symposium in Computational Medical and Health Technology (ICIEV-ISCMHT)*, pages 1–6, 2017.

[21] Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. Deep learning for event-driven stock prediction. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, page 2327–2333. AAAI Press, 2015.

[22] Qili Wang, Wei Xu, and Han Zheng. Combining the wisdom of crowds and technical analysis for financial market prediction using deep random subspace ensembles. *Neurocomputing*, 299:51 – 61, 2018.

[23] O Bustos and A. Pomares-Quimbaya. Stock market movement forecast: A systematic review. *Expert Systems with Applications*, 156:113464, 2020.

[24] Ahmet Murat Ozbayoglu, Mehmet Ugur Gudelek, and Omer Berat Sezer. Deep learning for financial applications : A survey. *Applied Soft Computing*, 93:106384, 2020.

[25] Fan Fang, Carmine Ventre, Michail Basios, Hoiliong Kong, Leslie Kanthan, Lingbo Li, David Martínez-Rego, and Fan Wu. Cryptocurrency trading: A comprehensive survey. *CoRR*, abs/2003.11352, 2020.

[26] Timothy King and Dimitrios Koutmos. Herding and feedback trading in cryptocurrency markets. *Ann. Oper. Res.*, 300(1):79–96, 2021.

[27] Jacopo Fior and Luca Cagliero. Exploring the use of data at multiple granularity levels in machine learning-based stock trading. In *2020 International Conference on Data Mining Workshops (ICDMW)*, pages 333–340, 2020.

[28] J.J. Murphy. *Technical Analysis of the Financial Markets: A Comprehensive Guide to Trading Methods and Applications*. New York Institute of Finance Series. New York Institute of Finance, 1999.

[29] A. O'Sullivan and S.M. Sheffrin. *Economics: Principles in Action*. Prentice Hall Science/Social Studies. Prentice Hall (School Division), 2006.

[30] Rundo, Trenta, di Stallo, and Battiato. Machine learning for quantitative finance applications: A survey. *Applied Sciences*, 9(24):5574, Dec 2019.

[31] Boming Huang, Yuxiang Huan, Li Da Xu, Lirong Zheng, and Zhuo Zou. Automated trading systems statistical and machine learning methods and hardware implementation: a survey. *Enterprise Information Systems*, 13(1):132–144, 2019.

[32] Raul Gomez Martinez, Miguel Prado Roman, and Paola Plaza Casado. Big data algorithmic trading systems based on investors' mood. *Journal of Behavioral Finance*, 20(2):227–238, 2019.

[33] I. Bordino, N. Kourtellis, N. Laptev, and Y. Billawala. Stock trade volume prediction with yahoo finance user browsing behavior. In *2014 IEEE 30th International Conference on Data Engineering*, pages 1168–1173, 2014.

[34] 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. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[35] Luca Cagliero, Jacopo Fior, and Paolo Garza. Shortlisting machine learning-based stock trading recommendations using candlestick pattern recognition. *Expert Systems with Applications*, 216:119493, 2023.

[36] T. Williams and V. Turton. *Trading Economics: A Guide to Economic Statistics for Practitioners and Students*. The Wiley Finance Series. Wiley, 2014.

[37] Elham Ahmadi, Milad Jasemi, Leslie Monplaisir, Mohammad Amin Nabavi, Armin Mahmoodi, and Pegah Amini Jam. New efficient hybrid candlestick technical analysis model for stock market timing on the basis of the support vector machine and heuristic algorithms of imperialist competition and genetic. *Expert Syst. Appl.*, 94:21–31, 2018.

[38] Milad Jasemi, Ali M. Kimiagari, and A. Memariani. A modern neural network model to do stock market timing on the basis of the ancient investment technique of japanese candlestick. *Expert Syst. Appl.*, 38(4):3884–3890, 2011.

[39] A. Marszalek and T. Burczynski. Modeling and forecasting financial time series with ordered fuzzy candlesticks. *Information Sciences*, 273:144 – 155, 2014.

[40] Ian J. Goodfellow, Yoshua Bengio, and Aaron C. Courville. *Deep Learning*. Adaptive computation and machine learning. MIT Press, 2016.

[41] Giuseppe Attanasio, Luca Cagliero, and Elena Baralis. Leveraging the explainability of associative classifiers to support quantitative stock trading. In *Proceedings of the Sixth International Workshop on Data Science for Macro-Modeling*, DSMM ’20, New York, NY, USA, 2020. Association for Computing Machinery.

[42] Alejandro Rodríguez González, Ángel García-Crespo, Ricardo Colomo Palacios, Fernando Guldrís-Iglesias, and Juan Miguel Gómez-Berbís. CAST: using neural networks to improve trading systems based on technical analysis by means of the RSI financial indicator. *Expert Syst. Appl.*, 38(9):11489–11500, 2011.

[43] Lamartine Almeida Teixeira and Adriano Lorena InÃ¡cio de Oliveira. A method for automatic stock trading combining technical analysis and nearest neighbor classification. *Expert Systems with Applications*, 37(10):6885 – 6890, 2010.

[44] R. C. Brasileiro, V. L. F. Souza, B. J. T. Fernandes, and A. L. I. Oliveira. Automatic method for stock trading combining technical analysis and the artificial bee colony algorithm. In *2013 IEEE Congress on Evolutionary Computation*, pages 1810–1817, June 2013.

[45] Yong Hu, Bin Feng, Xiangzhou Zhang, E.W.T. Ngai, and Mei Liu. Stock trading rule discovery with an evolutionary trend following model. *Expert Systems with Applications*, 42(1):212 – 222, 2015.

[46] J. W. Lee, J. Park, J. O, J. Lee, and E. Hong. A multiagent approach to q-learning for daily stock trading. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 37(6):864–877, Nov 2007.

[47] Prodromos E. Tsinaslanidis and Achilleas D. Zapranis. *Technical Analysis for Algorithmic Pattern Recognition*. Springer, 2016.

[48] Rodrigo Naranjo, Javier Arroyo, and Matilde Santos. Fuzzy modeling of stock trading with fuzzy candlesticks. *Expert Systems with Applications*, 93:15 – 27, 2018.

[49] K.H. Lee and G.S. Jo. Expert system for predicting stock market timing using a candlestick chart. *Expert Systems with Applications*, 16(4):357 – 364, 1999.

[50] Roberto CervellÃ³-Royo, Francisco Guijarro, and Karolina Michniuk. Stock market trading rule based on pattern recognition and technical analysis: Forecasting the djia index with intraday data. *Expert Systems with Applications*, 42(14):5963 – 5975, 2015.

[51] William Leigh, Naval Modani, Russell Purvis, and Tom Roberts. Stock market trading rule discovery using technical charting heuristics. *Expert Systems with Applications*, 23(2):155 – 159, 2002.

[52] Jar-Long Wang and Shu-Hui Chan. Stock market trading rule discovery using pattern recognition and technical analysis. *Expert Systems with Applications*, 33(2):304 – 315, 2007.

[53] Ruben Arevalo, Jorge Garcia, Francisco Guijarro, and Alfred Peris. A dynamic trading rule based on filtered flag pattern recognition for stock market price forecasting. *Expert Systems with Applications*, 81:177–192, 2017.

[54] Chih-Fong Tsai and Zen-Yu Quan. Stock prediction by searching for similarities in candlestick charts. *ACM Trans. Manage. Inf. Syst.*, 5(2):9:1–9:21, July 2014.

[55] Prodromos Tsinaslanidis and Francisco Guijarro. What makes trading strategies based on chart pattern recognition profitable? *Expert Systems*, 38(5):e12596, 2021.

[56] Chiung-Hon Leon Lee, Alan Liu, and Wen-Sung Chen. Pattern discovery of fuzzy time series for financial prediction. *IEEE Trans. on Knowl. and Data Eng.*, 18(5):613–625, May 2006.

[57] C. L. Lee. Modeling personalized fuzzy candlestick patterns for investment decision making. In *2009 Asia-Pacific Conference on Information Processing*, volume 2, pages 286–289, July 2009.

[58] Takenori Kamo and Cihan H. Dagli. Hybrid approach to the japanese candlestick method for financial forecasting. *Expert Syst. Appl.*, 36(3):5023–5030, 2009.

[59] Rodolfo C. Cavalcante, Rodrigo C. Brasileiro, Victor L.F. Souza, Jarley P. Nobrega, and Adriano L.I. Oliveira. Computational intelligence and financial markets: A survey and future directions. *Expert Systems with Applications*, 55:194 – 211, 2016.

[60] Tsung-Hsun Lu, Yung-Ming Shiu, and Tsung-Chi Liu. Profitable candlestick trading strategies: The evidence from a new perspective. *Review of Financial Economics*, 21(2):63–68, 2012.

[61] Weiwei Jiang. Applications of deep learning in stock market prediction: Recent progress. *Expert Systems with Applications*, 184:115537, 2021.

[62] Antonio Gulli and Sujit Pal. *Deep learning with Keras*. Packt Publishing Ltd, 2017.

[63] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, December 2006.

[64] Winter Sinkala. On the generation of arbitrage-free stock price models using lie symmetry analysis. *Comput. Math. Appl.*, 72(5):1386–1393, 2016.

[65] Guglielmo Maria Caporale and Alex Plastun. Momentum effects in the cryptocurrency market after one-day abnormal returns. *Financial Markets and Portfolio Management*, 34:251–266, 2020.

[66] Alex Plastun, Elie Bouri, Rangan Gupta, and Qiang Ji. Price effects after one-day abnormal returns in developed and emerging markets: Esg versus traditional indices. *The North American Journal of Economics and Finance*, 59:101572, 10 2021.

[67] Giuseppe Attanasio, Luca Cagliero, Paolo Garza, and Elena Baralis. Quantitative cryptocurrency trading: exploring the use of machine learning techniques. In *Proceedings of the 5th Workshop on Data Science for Macro-modeling with Financial and Economic Datasets, DSMM@SIGMOD 2019, Amsterdam, The Netherlands, June 30, 2019*, pages 1:1–1:6. ACM, 2019.

[68] Salim Lahmiri and Stelios D. Bekiros. Deep learning forecasting in cryptocurrency high-frequency trading. *Cogn. Comput.*, 13(2):485–487, 2021.

[69] Ioannis E. Livieris, Emmanuel G. Pintelas, Stavros Stavroyiannis, and Panayiotis E. Pintelas. Ensemble deep learning models for forecasting cryptocurrency time-series. *Algorithms*, 13(5):121, 2020.

[70] Olivier Kraaijeveld and Johannes De Smedt. The predictive power of public twitter sentiment for forecasting cryptocurrency prices. *Journal of International Financial Markets, Institutions and Money*, 65:101188, 2020.

[71] Maria Glenski, Tim Weninger, and Svitlana Volkova. Improved forecasting of cryptocurrency price using social signals, 2019.

[72] V Derbentsev, V Babenko, KIRILL Khrustalev, Hanna Obruch, and SOFIIA Khrustalova. Comparative performance of machine learning ensemble algorithms for forecasting cryptocurrency prices. *International Journal of Engineering*, 34(1):140–148, 2021.

[73] Ioannis E. Livieris, Emmanuel Pintelas, Stavros Stavroyiannis, and Panagiotis Pintelas. Ensemble deep learning models for forecasting cryptocurrency time-series. *Algorithms*, 13(5), 2020.

[74] Xiaolei Sun, Mingxi Liu, and Zeqian Sima. A novel cryptocurrency price trend forecasting model based on lightgbm. *Finance Research Letters*, 32:101084, 2020.

[75] Zhuorui Zhang, Hong-Ning Dai, Junhao Zhou, Subrota Kumar Mondal, Miguel Martínez García, and Hao Wang. Forecasting cryptocurrency price using convolutional neural networks with weighted and attentive memory channels. *Expert Systems with Applications*, 183:115378, 2021.

[76] Jakub Bartos. Does bitcoin follow the hypothesis of efficient market? *International Journal of Economic Sciences*, IV:10–23, 01 2015.

[77] Andrew Urquhart. The inefficiency of bitcoin. *Economics Letters*, 148, 09 2016.

[78] Aurelio F. Bariviera. The inefficiency of bitcoin revisited: A dynamic approach. *Economics Letters*, 161:1–4, 2017.

[79] Shaen Corbet, Brian Lucey, and Larisa Yarovaya. Datestamping the bitcoin and ethereum bubbles. *Finance Research Letters*, 26(C):81–88, 2018.

[80] Neil Gandal and Hanna Halaburda. Competition in the cryptocurrency market. Working Papers 14-17, NET Institute, 2014.

[81] Yutaka Kurihara and Akio Fukushima. The market efficiency of bitcoin: A weekly anomaly perspective. *Journal of Applied Finance & Banking*, 7(3), 2017.

[82] Thanaset Chevapatrakul and Danilo V. Mascia. Detecting overreaction in the bitcoin market: A quantile autoregression approach. *Finance Research Letters*, 30:371–377, 09 2019.

[83] Giuseppe Attanasio, Luca Cagliero, and Elena Baralis. Leveraging the explainability of associative classifiers to support quantitative stock trading. In *Proceedings of the Sixth International Workshop on Data Science for Macro-Modeling*, pages 1–6, 2020.

[84] Pang-Ning Tan, Michael S. Steinbach, Anuj Karpatne, and Vipin Kumar. *Introduction to Data Mining (Second Edition)*. Pearson, 2019.

[85] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gael Varoquaux. Api design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

[86] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[87] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[88] Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, pages 1322–1328. IEEE, 2008.

[89] Gian Pietro Bellocca, Giuseppe Attanasio, Luca Cagliero, and Jacopo Fior. Leveraging the momentum effect in machine learning-based cryptocurrency trading. *Machine Learning with Applications*, 8:100310, 2022.

[90] Gen-Huey Chen, Ming-Yang Kao, Yuh-Dauh Lyuu, and Hsing-Kuo Wong. Optimal buy-and-hold strategies for financial markets with bounded daily returns. In *Proceedings of the Thirty-first Annual ACM Symposium on Theory of Computing*, STOC '99, pages 119–128, New York, NY, USA, 1999. ACM.

[91] Harry M. Markowitz. *Portfolio Selection: Efficient Diversification of Investments*. Wiley, 2 edition, September 1991.

[92] Ming-Feng Tsai, Chuan-Ju Wang, and Po-Chuan Chien. Discovering finance keywords via continuous-space language models. *ACM Trans. Manage. Inf. Syst.*, 7(3), August 2016.

[93] Wijnand Nuij, Viorel Milea, Frederik Hogenboom, Flavius Frasincar, and Uzay Kaymak. An automated framework for incorporating news into stock trading strategies. *IEEE Trans. on Knowl. and Data Eng.*, 26(4):823–835, April 2014.

[94] Binoy B. Nair, P.K. Saravana Kumar, N.R. Sakthivel, and U. Vipin. Clustering stock price time series data to generate stock trading recommendations: An empirical study. *Expert Systems with Applications*, 70:20 – 36, 2017.

[95] Mehmed Kantardzic, Pedram Sadeghian, and Chun Shen. The time diversification monitoring of a stock portfolio: An approach based on the fractal dimension. In *Proceedings of the 2004 ACM Symposium on Applied Computing*, SAC '04, pages 637–641, New York, NY, USA, 2004. ACM.

[96] G.Y. Wang. Portfolio diversification and risk reduction- evidence from taiwan stock mutual funds. In *Management and Service Science (MASS), 2010 International Conference on*, pages 1–4, Aug 2010.

[97] R. de Frein, K. Drakakis, and Scott Rickard. Portfolio diversification using subspace factorizations. In *Information Sciences and Systems, 2008. CISS 2008. 42nd Annual Conference on*, pages 1075–1080, March 2008.

[98] V. Parque, S. Mabu, and K. Hirasawa. Global portfolio diversification by genetic relation algorithm. In *ICCAS-SICE, 2009*, pages 2567–2572, Aug 2009.

[99] Yan Chen, S. Mabu, K. Hirasawa, and Jinglu Hu. Genetic network programming with sarsa learning and its application to creating stock trading rules. In *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, pages 220–227, Sept 2007.

[100] S.R. Nanda, B. Mahanty, and M.K. Tiwari. Clustering indian stock market data for portfolio management. *Expert Systems with Applications*, 37(12):8793 – 8798, 2010.

[101] Long Nguyen Cong, N. Wisitpongphan, Phayung Meesad, and Herwig Unger. Clustering stock data for multi-objective portfolio optimization. *International Journal of Computational Intelligence and Applications*, 13, 06 2014.

[102] Omar Alqaryouti, Tarek Farouk, and Nur Siyam. Clustering stock markets for balanced portfolio construction. In Aboul Ella Hassanien, Mohamed F. Tolba, Khaled Shaalan, and Ahmad Taher Azar, editors, *Proceedings of the International Conference on Advanced Intelligent Systems and Informatics, AISI 2018, Cairo, Egypt, September 3-5, 2018*, volume 845 of *Advances in Intelligent Systems and Computing*, pages 577–587. Springer, 2018.

[103] Robert C. Merton. Lifetime portfolio selection under uncertainty: The continuous-time case. *The Review of Economics and Statistics*, 51(3):247–257, 1969.

[104] Harry Markowitz. Portfolio selection. *The Journal of Finance*, 7(1):77–91, 1952.

[105] Renata Mansini, Wlodzimierz Ogryczak, and M. Grazia Speranza. Twenty years of linear programming based portfolio optimization. *European Journal of Operational Research*, 234(2):518–535, 2014.

[106] Eugene F. Fama and Kenneth R. French. The capital asset pricing model: Theory and evidence. *Journal of Economic Perspectives*, 18(3):25–46, September 2004.

[107] Constantin Zopounidis, Emilios Galariotis, Michael Doumpos, Stavroula Sarri, and Kostas Andriosopoulos. Multiple criteria decision aiding for finance: An updated bibliographic survey. *European Journal of Operational Research*, 247(2):339–348, 2015.

[108] Constantin Zopounidis, Michalis Doumpos, and Dimitrios Niklis. Financial decision support: an overview of developments and recent trends. *EURO Journal on Decision Processes*, 6(1):63–76, June 2018.

[109] Xidonas, Panos, Doukas, Haris, and Sarmas, Elissaios. A python-based multicriteria portfolio selection dss. *RAIRO-Oper. Res.*, 55:S3009–S3034, 2021.

[110] Enrico Angelelli, Renata Mansini, and M. Grazia Speranza. A comparison of mad and cvar models with real features. *Journal of Banking & Finance*, 32(7):1188–1197, 2008.

[111] Renata Mansini and Maria Grazia Speranza. Heuristic algorithms for the portfolio selection problem with minimum transaction lots. *European Journal of Operational Research*, 114(2):219–233, 1999.

[112] Luca Chiodi, Renata Mansini, and Maria Speranza. Semi-Absolute Deviation Rule for Mutual Funds Portfolio Selection. *Annals of Operations Research*, 124(1):245–265, November 2003.

[113] Enrico Angelelli, Renata Mansini, and Maria Grazia Speranza. Kernel search: a new heuristic framework for portfolio selection. *Comput. Optim. Appl.*, 51(1):345–361, 2012.

[114] Felipe Dias Paiva, Rodrigo TomÃ¡s Nogueira Cardoso, Gustavo Peixoto Hanaoka, and Wendel Moreira Duarte. Decision-making for financial trading: A fusion approach of machine learning and portfolio selection. *Expert Systems with Applications*, 115:635–655, 2019.

[115] Bilian Chen, Jingdong Zhong, and Yuanyuan Chen. A hybrid approach for portfolio selection with higher-order moments: Empirical evidence from shanghai stock exchange. *Expert Systems with Applications*, 145:113104, 2020.

[116] Kin Keung Lai, Lean Yu, Shouyang Wang, and Chengxiong Zhou. A double-stage genetic optimization algorithm for portfolio selection. In Irwin King, Jun Wang, Lai-Wan Chan, and DeLiang Wang, editors, *Neural Information Processing*, pages 928–937, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[117] Fengmei Yang, Zhiwen Chen, Jingjing Li, and Ling Tang. A novel hybrid stock selection method with stock prediction. *Applied Soft Computing*, 80:820–831, 2019.

[118] V. Kedia, Z. Khalid, S. Goswami, N. Sharma, and K. Suryawanshi. Portfolio generation for indian stock markets using unsupervised machine learning. In *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, pages 1–5, 2018.

[119] Chun-Hao Chen, Cheng-Yu Lu, and Cheng-Bon Lin. An intelligence approach for group stock portfolio optimization with a trading mechanism. *Knowl. Inf. Syst.*, 62(1):287–316, 2020.

[120] Okkes Ertenlice and Can B. Kalayci. A survey of swarm intelligence for portfolio optimization: Algorithms and applications. *Swarm and Evolutionary Computation*, 39:36 – 52, 2018.

[121] Yao-Hsin Chou, Shu-Yu Kuo, and Yu-Chi Jiang. A novel portfolio optimization model based on trend ratio and evolutionary computation. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 3(4):337–350, 2019.

[122] Yao-Hsin Chou, Yu-Chi Jiang, Yi-Rui Hsu, Shu-Yu Kuo, and Sy-Yen Kuo. A weighted portfolio optimization model based on the trend ratio, emotion index, and angqts. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 6(4):867–882, 2022.

[123] Elena Baralis, Luca Cagliero, and Paolo Garza. Planning stock portfolios by means of weighted frequent itemsets. *Expert Systems with Applications*, 86:1 – 17, 2017.

[124] Rakesh Agrawal, Tomasz Imielinski, and Arun N. Swami. Mining association rules between sets of items in large databases. In Peter Buneman and Sushil Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, DC, USA, May 26-28, 1993*, pages 207–216. ACM Press, 1993.

[125] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, page 487–499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc.

[126] Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, SIGMOD '00, page 1–12, New York, NY, USA, 2000. Association for Computing Machinery.

[127] Feng Tao, Fionn Murtagh, and Mohsen Farid. Weighted association rule mining using weighted support and significance framework. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, page 661–666, New York, NY, USA, 2003. Association for Computing Machinery.

[128] L. Cagliero and P. Garza. Infrequent weighted itemset mining using frequent pattern growth. *IEEE Transactions on Knowledge & Data Engineering*, 26(04):903–915, apr 2014.

[129] Haoyuan Li, Yi Wang, Dong Zhang, Ming Zhang, and Edward Y. Chang. Pfp: Parallel fp-growth for query recommendation. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, RecSys '08, page 107–114, New York, NY, USA, 2008. Association for Computing Machinery.

[130] Jacopo Fior, Luca Cagliero, and Paolo Garza. Price series cross-correlation analysis to enhance the diversification of itemset-based stock portfolios. In *Proceedings of the Sixth International Workshop on Data Science for Macro-Modeling*, DSMM '20, New York, NY, USA, 2020. Association for Computing Machinery.

[131] John Paparrizos and Luis Gravano. K-shape: Efficient and accurate clustering of time series. *SIGMOD Rec.*, 45(1):69–76, June 2016.

[132] Boris Podobnik and H. Eugene Stanley. Detrended cross-correlation analysis: A new method for analyzing two nonstationary time series. *Phys. Rev. Lett.*, 100:084102, Feb 2008.

[133] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison-Wesley, 2006.

[134] H.M. Markowitz. *Portfolio Selection: Efficient Diversification of Investments*. Monograph / Cowles Foundation for Research in Economics at Yale University. Wiley, 1991.

[135] Paolo Brandimarte. *An Introduction to Financial Markets: A Quantitative Approach*. John Wiley & Sons, 2017.

[136] Victor DeMiguel, Lorenzo Garlappi, and Raman Uppal. Optimal Versus Naive Diversification: How Inefficient is the 1/N Portfolio Strategy? *The Review of Financial Studies*, 22(5):1915–1953, 12 2007.

[137] A. Ponsich, A. L. Jaimes, and C. A. C. Coello. A survey on multiobjective evolutionary algorithms for the solution of the portfolio optimization problem and other finance and economics applications. *IEEE Transactions on Evolutionary Computation*, 17(3):321–344, 2013.

[138] Pang-Ning Tan, Michael S. Steinbach, and Vipin Kumar. *Introduction to Data Mining*. Addison-Wesley, 2005.

[139] Messod D. Beneish, Charles M. C. Lee, and Robin L. Tarpley. Contextual fundamental analysis through the prediction of extreme returns. *Review of Accounting Studies*, 6(2):165–189, 2001.

[140] Messod D. Beneish. The detection of earnings manipulation. *Financial Analysts Journal*, 55(5):24–36, 1999.

[141] Jeffery S. Abarbanell and Brian J. Bushee. Abnormal returns to a fundamental analysis strategy. *The Accounting Review*, 73(1):19–45, January 1998.

[142] Victor L. Bernard and Jacob K. Thomas. Post-earnings-announcement drift: Delayed price response or risk premium? *Journal of Accounting Research*, 27:1–36, 1989.

[143] Richard G. Sloan. Do stock prices fully reflect information in accruals and cash flows about future earnings? *The Accounting Review*, 71(3):289–315, 1996.

[144] Partha S. Mohanram. Separating winners from losers among lowbook-to-market stocks using financial statement analysis. *Review of Accounting Studies*, 10(2):133–170, 2005.

[145] Marcos Escobar-Anel. Multivariate risk aversion utility, application to esg investments. *The North American Journal of Economics and Finance*, 63:101790, 2022.

[146] David Bailey and Marcos Lopez de Prado. The sharpe ratio efficient frontier. *The Journal of Risk*, 15:3–44, 12 2012.

[147] William F. Sharpe. The sharpe ratio. *The Journal of Portfolio Management*, 21(1):49–58, 1994.

[148] Xiao-Yang Liu, Hongyang Yang, Qian Chen, Runjia Zhang, Liuqing Yang, Bowen Xiao, and Christina Dan Wang. Finrl: A deep reinforcement learning library for automated stock trading in quantitative finance. *CoRR*, abs/2011.09607, 2020.

[149] Guizhen Yang. The complexity of mining maximal frequent itemsets and maximal frequent patterns. KDD '04, page 344–353, New York, NY, USA, 2004. Association for Computing Machinery.

[150] Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. *SIGMOD Rec.*, 29(2):1–12, May 2000.

[151] Xiangrui Meng, Joseph Bradley, Burak Yavuz, Evan Sparks, Shivaram Venkataraman, Davies Liu, Jeremy Freeman, DB Tsai, Manish Amde, Sean Owen, Doris Xin, Reynold Xin, Michael J. Franklin, Reza Zadeh, Matei Zaharia, and Ameet Talwalkar. Mllib: Machine learning in apache spark. *J. Mach. Learn. Res.*, 17(1):1235–1241, January 2016.

[152] Jacopo Fior, Luca Cagliero, and Tommaso Calò. Generating comparative explanations of financial time series. In Silvia Chiusano, Tania Cerquitelli, and Robert Wrembel, editors, *Advances in Databases and Information Systems*, pages 121–132, Cham, 2022. Springer International Publishing.

[153] Jacopo Fior, Luca Cagliero, and Paolo Garza. Leveraging explainable ai to support cryptocurrency investors. *Future Internet*, 14(9), 2022.

[154] Joost van der Burgt. Explainable ai in banking. *Journal of Digital Banking*, 4(4):344–350, 2020.

[155] Patrick Weber, K Valerie Carl, and Oliver Hinz. Applications of explainable artificial intelligence in finance—a systematic review of finance, information systems, and computer science literature. *Management Review Quarterly*, pages 1–41, 2023.

[156] Niklas Bussmann, Paolo Giudici, Dimitri Marinelli, and Jochen Papenbrock. Explainable ai in fintech risk management. *Frontiers in Artificial Intelligence*, 3, 2020.

[157] Kirti Goyal and Satish Kumar. Financial literacy: A systematic review and bibliometric analysis. *International Journal of Consumer Studies*, 45(1):80–105, 2021.

[158] Charalampos M. Liapis, Aikaterini Karanikola, and Sotiris Kotsiantis. A multi-method survey on the use of sentiment analysis in multivariate financial time series forecasting. *Entropy*, 23(12), 2021.

[159] Janusz Kacprzyk and Sławomir Zadrony. Linguistic database summaries and their protoforms: Towards natural language based knowledge discovery tools. *Inf. Sci.*, 173(4):281–304, jun 2005.

[160] Rita M Catillo-Ortega, Nicolás Marín, and Daniel Sánchez. A fuzzy approach to the linguistic summarization of time series. *Journal of Multiple-Valued Logic & Soft Computing*, 17, 2011.

[161] Jonathan J. Harris, Ching-Hua Chen, and Mohammed J. Zaki. A framework for generating summaries from temporal personal health data. *ACM Trans. Comput. Healthcare*, 2(3), jul 2021.

[162] Chris van der Lee, Emiel Krahmer, and Sander Wubben. Automated learning of templates for data-to-text generation: comparing rule-based, statistical and neural methods. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 35–45, Tilburg University, The Netherlands, November 2018. Association for Computational Linguistics.

[163] Albert Gatt, François Portet, Ehud Reiter, Jim Hunter, Saad Mahamood, Wendy Moncur, and Somayajulu Sripada. From data to text in the neonatal intensive care unit: Using nlg technology for decision support and information management. *AI Commun.*, 22(3):153–186, aug 2009.

[164] Janusz Kacprzyk, Anna Wilbik, and Slawomir Zadrozny. An approach to the linguistic summarization of time series using a fuzzy quantifier driven aggregation. *Int. J. Intell. Syst.*, 25(5):411–439, 2010.

[165] Christoforos Nalmpantis and Dimitris Vrakas. Signal2vec: Time series embedding representation. In John MacIntyre, Lazaros S. Iliadis, Ilias Maglogiannis, and Chrisina Jayne, editors, *Engineering Applications of Neural Networks -*

20th International Conference, EANN 2019, Hersonissos, Crete, Greece, May 24-26, 2019, Proceedings, volume 1000 of *Communications in Computer and Information Science*, pages 80–90. Springer, 2019.

[166] C. Hutto and Eric Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. *Proceedings of the International AAAI Conference on Web and Social Media*, 8(1):216–225, May 2014.

[167] Yuxuan Huang, Luiz Fernando Capretz, and Danny Ho. Machine learning for stock prediction based on fundamental analysis. In *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 01–10, 2021.

[168] Jacopo Fior and Luca Cagliero. Estimating the incidence of adverse weather effects on road traffic safety using time series embeddings. In *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 402–407, 2021.

[169] Jacopo Fior and Luca Cagliero. Correlating extreme weather conditions with road traffic safety: A unified latent space model. *IEEE Access*, 10:73005–73018, 2022.

[170] Eamonn Keogh, Jessica Lin, and Ada Fu. Hot sax: Efficiently finding the most unusual time series subsequence. In *Proceedings of the Fifth IEEE International Conference on Data Mining*, ICDM ’05, page 226–233, USA, 2005. IEEE Computer Society.

[171] Giancarlo Giudici, Alistair Milne, and Dmitri Vinogradov. Cryptocurrencies: market analysis and perspectives. *Journal of Industrial and Business Economics*, 47(1):1–18, September 2019.

[172] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review*, page 21260, 2008.

[173] Laura Alessandretti, Abeer ElBahrawy, Luca Maria Aiello, and Andrea Baronchelli. Anticipating cryptocurrency prices using machine learning. *Complex.*, 2018:8983590:1–8983590:16, 2018.

[174] Jifeng Sun, Yi Zhou, and Jianwu Lin. Using machine learning for cryptocurrency trading. In *IEEE International Conference on Industrial Cyber Physical Systems, ICPS 2019, Taipei, Taiwan, May 6-9, 2019*, pages 647–652. IEEE, 2019.

[175] Thomas E. Koker and Dimitrios Koutmos. Cryptocurrency trading using machine learning. *Journal of Risk and Financial Management*, 13(8), 2020.

[176] Leopoldo Catania, Stefano Grassi, and Francesco Ravazzolo. Forecasting cryptocurrencies under model and parameter instability. *International Journal of Forecasting*, 35(2):485–501, 2019.

[177] Zeinab Shahbazi and Yung-Cheol Byun. Machine learning-based analysis of cryptocurrency market financial risk management. *IEEE Access*, 10:37848–37856, 2022.

[178] Marco Ortu, Nicola Uras, Claudio Conversano, Silvia Bartolucci, and Giuseppe Destefanis. On technical trading and social media indicators for cryptocurrency price classification through deep learning. *Expert Systems with Applications*, 198:116804, 2022.

[179] Sashank Sridhar and Sowmya Sanagavarapu. Multi-head self-attention transformer for dogecoin price prediction. In *2021 14th International Conference on Human System Interaction (HSI)*, pages 1–6, 2021.

[180] Valeria Dâ€™Amato, Susanna Levantesi, and Gabriella Piscopo. Deep learning in predicting cryptocurrency volatility. *Physica A: Statistical Mechanics and its Applications*, 596:127158, 2022.

[181] Fan Fang, Carmine Ventre, Michail Basios, Hoiliong Kong, Leslie Kanthan, Lingbo Li, David Martínez-Rego, and Fan Wu. Cryptocurrency trading: A comprehensive survey. *CoRR*, abs/2003.11352, 2020.

[182] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[183] L. S. Shapley. *A Value for n-Person Games*, pages 307–318. Princeton University Press, 2016.

[184] William Webber, Alistair Moffat, and Justin Zobel. A similarity measure for indefinite rankings. *ACM Trans. Inf. Syst.*, 28(4), nov 2010.

[185] Jun-Hao Chen, Cheng-Han Wu, Yun-Chneg Tsai, and Samuel Yen-Chi Chen. Explainable digital currency candlestick pattern ai learner. In *2022 14th International Conference on Knowledge and Smart Technology (KST)*, pages 91–96, 2022.

[186] Salvatore M. Carta, Sergio Consoli, Luca Piras, Alessandro Sebastian Podda, and Diego Reforgiato Recupero. Explainable machine learning exploiting news and domain-specific lexicon for stock market forecasting. *IEEE Access*, 9:30193–30205, 2021.

[187] Ana Todorovska, Eva Spirovska, Gorast Angelovski, Hristijan Peshov, Ivan Rusevski, Jovana Marojevikj, Irena Vodenska, Lubomir T. Chitkushev, and Dimitar Trajanov. *Analysis of Cryptocurrency Interdependencies.*

[188] Yulin Liu and Luyao Zhang. Cryptocurrency valuation and machine learning. *SSRN Electronic Journal*, 2020.

[189] Golnoosh Babaei, Paolo Giudici, and Emanuela Raffinetti. Explainable artificial intelligence for crypto asset allocation. *Finance Research Letters*, 47:102941, 2022.

[190] Warren Freeborough and Terence van Zyl. Investigating explainability methods in recurrent neural network architectures for financial time series data. *Applied Sciences*, 12(3), 2022.

[191] Richard Meyes, Melanie Lu, Constantin Waubert de Puiseau, and Tobias Meisen. Ablation studies in artificial neural networks, 2019.

[192] Tao Yin, Xingbo Du, Weipeng Zhang, Yunan Zhao, Bing Han, and Junchi Yan. Real-trading-oriented price prediction with explainable multiobjective optimization in quantitative trading. *IEEE Access*, 10:57685–57695, 2022.

[193] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 3149–3157, Red Hook, NY, USA, 2017. Curran Associates Inc.

[194] Abdul Jabbar, Xi Li, and Bourahla Omar. A survey on generative adversarial networks: Variants, applications, and training. *ACM Comput. Surv.*, 54(8):157:1–157:49, 2022.

[195] Artur Sokolovsky, Luca Arnaboldi, Jaume Bacardit, and Thomas Gross. Interpretable ml-driven strategy for automated trading pattern extraction. *CoRR*, abs/2103.12419, 2021.

[196] Ana Todorovska, Eva Spirovska, Gorast Angelovski, Hristijan Peshov, Ivan Rusevski, Jovana Marojevikj, Irena Vodenska, Lubomir T. Chitkushev, and Dimitar Trajanov. *Analysis of Cryptocurrency Interdependencies*.

[197] Weilong Hu, Yain-Whar Si, Simon Fong, and Raymond Yiu Keung Lau. A formal approach to candlestick pattern classification in financial time series. *Applied Soft Computing*, 84:105700, 2019.

[198] Johann Gamper and Anton Dignös. Processing temporal and time series data: Present state and future challenges. In Jérôme Darmont, Boris Novikov, and Robert Wrembel, editors, *Advances in Databases and Information Systems*, pages 8–14, Cham, 2020. Springer International Publishing.

[199] Salvatore Carta, Alessandro Sebastian Podda, Diego Reforgiato Recupero, and Maria Madalina Stanciu. Explainable AI for financial forecasting. In Giuseppe Nicosia, Varun Ojha, Emanuele La Malfa, Gabriele La Malfa, Giorgio Jansen, Panos M. Pardalos, Giovanni Giuffrida, and Renato Umeton, editors, *Machine Learning, Optimization, and Data Science - 7th International Conference, LOD 2021, Grasmere, UK, October 4-8, 2021, Revised Selected Papers, Part II*, volume 13164 of *Lecture Notes in Computer Science*, pages 51–69. Springer, 2021.

[200] Christoph Molnar, Giuseppe Casalicchio, and Bernd Bischl. Interpretable machine learning - A brief history, state-of-the-art and challenges. In Irena Koprinska, Michael Kamp, Annalisa Appice, Corrado Loglisci, Luiza Antonie, Albrecht Zimmermann, Riccardo Guidotti, Özlem Özgöbek, Rita P. Ribeiro, Ricard Gavaldà, João Gama, Linara Adilova, Yamuna Krishnamurthy, Pedro M. Ferreira, Donato Malerba, Ibéria Medeiros, Michelangelo Ceci, Giuseppe Manco, Elio Masciari, Zbigniew W. Ras, Peter Christen, Eirini Ntoutsi, Erich Schubert, Arthur Zimek, Anna Monreale, Przemyslaw Biecek, Salvatore Rinzivillo, Benjamin Kille, Andreas Lommatzsch, and Jon Atle Gulla, editors, *ECML PKDD 2020 Workshops - Workshops of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2020): SoGood 2020, PDFL 2020, MLCS 2020, NFMCP 2020, DINA 2020, EDML 2020, XKDD 2020 and INRA 2020, Ghent, Belgium, September 14-18, 2020, Proceedings*, volume 1323 of *Communications in Computer and Information Science*, pages 417–431. Springer, 2020.