Engineering Applications of Artificial Intelligence 131 (2024) 107713

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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

Hybrid genetic algorithms in agent-based artificial market model for simulating fan tokens trading

David Alaminos [a,*], M. Belén Salas [b,c], Manuel Á. Fernández-Gámez [b,c]

[a] Department of Business, University of Barcelona, Barcelona, Spain
[b] Department of Finance and Accounting, University of Málaga, Málaga, Spain
[c] Cátedra de Economía y Finanzas Sostenibles, University of Málaga, Málaga, Spain

ARTICLE INFO

Keywords:
Cryptographic tokens
Evolutionary computation
Market microstructure
Multi-agent systems
Digital assets

ABSTRACT

In recent years cryptographic tokens have gained popularity as they can be used as a form of emerging alternative financing and as a means of building platforms. The token markets innovate quickly through technology and decentralization, and they are constantly changing, and they have a high risk. Negotiation strategies must therefore be suited to these new circumstances. The genetic algorithm offers a very appropriate approach to resolving these complex issues. However, very little is known about genetic algorithm methods in cryptographic tokens. Accordingly, this paper presents a case study of the simulation of Fan Tokens trading by implementing selected best trading rule sets by a genetic algorithm that simulates a negotiation system through the Monte Carlo method. We have applied Adaptive Boosting and Genetic Algorithms, Deep Learning Neural Network-Genetic Algorithms, Adaptive Genetic Algorithms with Fuzzy Logic, and Quantum Genetic Algorithm techniques. The period selected is from December 1, 2021 to August 25, 2022, and we have used data from the Fan Tokens of Paris Saint-Germain, Manchester City, and Barcelona, leaders in the market. Our results conclude that the Hybrid and Quantum Genetic algorithm display a good execution during the training and testing period. Our study has a major impact on the current decentralized markets and future business opportunities.

1. Introduction

In recent years, crypto assets have garnered remarkable attention from both the industrial and scientific communities, ranging from cryptocurrencies, utility, and security tokens to non-fungible tokens (NFT). Most of these assets are relatively recent innovations and cryptographic tokens may be employed as an alternative source of financing as well as a way of creating platforms (Scharnowski et al., 2021). Some, like utility tokens and collectibles called Fan Tokens, allow the token holder to receive some form of service within their respective blockchain ecosystem, so they are fungible and are therefore quite distinguishable from NFT (Dowling, 2022a,b). Nevertheless, for a fan, having "his" token can offer a non-monetary value comparable to that of possessing a collector's item (Scharnowski et al., 2021).

Fan tokens are blockchain-based cryptographic assets emitted by clubs or organizations, that hold huge fan bases. The greatest market for this type of token so far is those released by soccer clubs. Due to the coronavirus pandemic and the ensuing financial crisis, there was an urgent demand in all sectors of the economy for extra income, especially evident in the entertainment sector. Sports entities, which have lost a considerable proportion of their revenues in the pandemic, are also increasingly expanding their digital business through fan token issuance and NFT collection, not only as a means of generating new funds but as a way of building fan loyalty by giving them the chance to participate in governance decisions making them identify with the club and feel part of it (Солнцев et al., 2022; Zarifis and Cheng, 2022). So, tokens facilitate a further income flow for clubs as well as, ease new ways of engagement with the fan database. Fan tokens diverge from soccer club shares in terms of their ownership features. Fan tokens could be linked to investors' mood swings and responses to match results (Demir et al., 2022). Fan tokens allow for benefits such as rewards and promotions, as well as fans can participate in the decisions of the club by voting, such as the song that plays after a goal is scored, the training camp names, match location, tour bus design, merchandise design, or which new merchandise is promoted. Additional services often provide unique or autographed merchandise, tickets to games, VIP tickets or to meet the players (Ersan et al., 2022; Demir and Aktas, 2022; Scharnowski et al., 2021).

* Corresponding author.
E-mail address: alaminos@ub.edu (D. Alaminos).

https://doi.org/10.1016/j.engappai.2023.107713
Received 27 September 2022; Received in revised form 7 December 2023; Accepted 12 December 2023
Available online 4 January 2024
0952-1976/© 2024 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

The first and currently largest provider of fan tokens is "Socios", a sports fan engagement platform that has been developed by Chiliz, a Malta-based company that states its objective as "pioneering a generational shift in the sports industry through digital assets" (Chiliz, 2021). "Socios" merges supporter engagement with token sales via its exchange and exclusive currency on the platform, i.e., Chiliz. Specifically, using this digital currency, fans may purchase virtual tokens belonging to their favourite sports team, via the Chiliz exchange and the Socios.com website or mobile app, in return for rewards and participation in specific club decisions (Солнцев et al., 2022; Vidal-Tomás, 2022). Other major platforms that offer Fan Tokens and mediate the trading of Fan Tokens are Binance, Bitci, and Paribu. These four platforms first enter into agreements with sports clubs and sign contracts for the tokens and offer them to the public (Demir and Aktas, 2022).

The literature on blockchain-based collectibles is still scarce and mostly pertains to NFTs, and the academic evidence regarding their risks and returns is still scarce. Scharnowski et al. (2021) document that fan token prices are highly volatile and substantially riskier than established cryptocurrencies. Therefore, anomalies such as excess volatility and irrational behaviour are increasingly prevalent in token trading and emerging financial markets in general. Солнцев et al. (2022) determine that the risks and challenges associated with the issuance of fan tokens in different markets and by clubs in several countries and the currency assessment of digital asset effects in the sports industry constitute a further future line of research. Demir and Aktas (2022) conclude that further and different analyses in future studies can assess the relationship between organizations' sporting achievements and the prices of Fan Tokens. These authors state that the effect of a victory, a sporting success, or a championship on the prices of Fan Tokens, for example, could be evaluated using alternative methods as well as through the development of detailed models. They determine that predictive models based on machine learning and artificial intelligence algorithms could be built on Fan Tokens, and thus further academic research can be conducted. For his part, Dowling (2022a,b) concludes that future research may also explore the primary market for fan tokens, specifically drivers that raise the likelihood of success of an initial token offer. Also, the extent to which fan tokens improve fan identification, engagement, and participation remain an area of uncertainty. Hence, there is a need for further research as, on the one hand, fan tokens have experienced dramatic price rises driven, at least in part, by speculative demand, excessive volatility, and irrational behaviour, and, on the other hand, traditional methodologies remain inadequate to solve these emerging problems (Chen et al., 2021).

To fill the gap in this research area, our study aims to simulate Fan Tokens trading through an agent-based model including realistic trading strategies by the theory of genetic algorithms, through the Monte Carlo algorithm, as Tokens appear to be a distinct and exciting new asset class. Our objective is to simulate the trading of the prices of Fan Tokens of football clubs whose output is therefore the price of these Fan Tokens in US dollars (USD), as done in previous works (Boboc and Dinică, 2013; Cocco and Marchesi, 2016; Cocco et al., 2019), so that the algorithms can calculate the optimization of the trading rules that can give the highest profits to the agents. We have employed the following genetic algorithm techniques: Adaptive Boosting and Genetic Algorithm, Deep Learning Neural Network-Genetic Algorithm, Adaptive Genetic Algorithm with Fuzzy Logic, and Quantum Genetic Algorithm. Besides, we have used data from the Fan Tokens of Paris Saint-Germain (PSG), Manchester City, and Barcelona, which are among the market leaders. We targeted soccer clubs since they represent the most relevant market for fan tokens. To date, nine of the ten largest fan tokens in terms of market capitalization are released by football clubs.

We make at least three further contributions to the literature. First, our study generates significant advantages in the financial area as research on blockchain-based collectibles remains limited and is mainly concerned with NFTs. Fan tokens are increasingly spreading in response to the search by sports clubs for alternative revenue models to balance the decline in their revenues during the COVID-19 pandemic period. Available literature mainly addresses the impact of match results on token prices. In addition, to the best of our knowledge, no previous study has performed a simulation of Fan Token trading through the theory of genetic algorithms. Although in the study of Chen et al. (2021) genetic algorithms have been applied in bitcoin trading, the price variation in the case of fan tokens, unlike more traditional cryptocurrencies such as bitcoin, is completely related to the achievements related to their club. In addition, fan tokens hold distinctive characteristics that differentiate them from other asset classes, even other cryptocurrencies, such as bitcoin. Therefore, the observed lack of literature on this relatively new asset class will be filled by our study, enriching the understanding of today's financial markets. Second, our study has focused on one of the most important fan token market segments, such as club and league management, and specifically on soccer clubs. Among these soccer clubs, we have analysed the fan token data of three prestigious football teams.

Third, we have considered GA methods and their hybrids which allow the introduction of time-varying weights and, consequently, can effectively handle numerous highly volatile situations in real markets, as in the case of token trading. The probabilistic and time-varying nature of these models appears to reflect naturally the character of economic human behaviours (Drachal and Pawłowski, 2021). Cocco et al. (2019) simulate the trading of the BTC/USD currency pair, using the theory of GA, and they establish as a future line of research the possibility of applying a more advanced algorithm. Thus, GAs have many promising applications, but could be improved because GAs themselves remain a huge area for further development, as more hybrids could be created, where a GA could still be a useful way to optimize the model parameters (Drachal and Pawłowski, 2021).

Other studies indicate that hybrid genetic algorithms enhance the genetic search performance (Xu et al., 2021; Zhou et al., 2020). These studies validate the effectiveness and advantages of employing hybrid genetic algorithms, demonstrating that integrating a search method into a genetic algorithm can bolster search performance, contingent on their functions working in tandem towards the shared objective of optimization. Additionally, as per the findings of He et al. (2008), genetic algorithms (GA) offer robust and potent adaptive search mechanisms. Their successful application in solving diverse challenges within electric power systems, including economic dispatch, unit commitment, reactive power control, hydrothermal scheduling, and distribution system planning, underscores their versatility. GA conducts simultaneous searches for multiple solutions, distinguishing it from conventional optimal algorithms. The author concludes that GA's primary strength lies in its ability to swiftly uncover nearly optimal solutions, outperforming other random search methods in terms of efficiency.

The rest of the paper is organized as follows. Section 2 provides a literature review of empirical research. Section 3 presents the model we have applied in our research. In section 4, the methodology is described. Section 5 details the sample and data involved in the research. Section 6 points out the results and findings obtained. By last, Section 7 finishes explaining the conclusions reached.

2. Related work

The literature on fan tokens is still rather immature, although they are becoming increasingly popular. Research in related work usually focuses on cryptocurrencies and blockchain. Up to now, just a minority of investigations have been conducted empirically on fan tokens, targeting aspects of the tokens. Most of the available literature mainly addresses the impact of match results on fan token prices. Demir et al. (2022) investigate if movements in the price of fan tokens can be influenced by success in sports or not based on data from fan tokens of 11 football clubs. They report that fan token prices decline after a club loses a match, particularly in UEFA Champions League matches. They find that losses (wins) in the UEFA Champions League competition have

a negative (positive) impact on improper performances, although, in absolute terms, losses are of greater impact. In contrast, Mazur and Vega (2022) observe that matches do not matter because neither the outcome of the match nor the tournament form impacts prices. In addition, they conclude that first-day returns are substantial, but fan token returns are lower in the long run. Vidal-Tomás (2023) demonstrates that both first-day fan token returns and posterior buy-and-hold returns are high. Therefore, this author concludes that in the long term, the return is positive on average, however, supporters should be cautious because some fan tokens have a negative return, due to the high volatility of the tokens. Furthermore, this author conducted his research on the Socios.com platform and its cryptocurrency Chiliz, and showed that fan tokens relate more closely to Chiliz than to the cryptocurrency market index. Scharnowski et al. (2021) conclude that yields are poorer when a club loses a match, especially where that loss happened unexpectedly as betting odds indicate but are greater when there is more investor attention to fan tokens overall. Nevertheless, investor attention toward a particular club on average has no impact on token returns, nor do player transfers. Their results show that fan tokens can supply some non-monetary utility and can often be maintained for sentimental reasons, but as an asset class, they remain extremely speculative with many similarities to cryptocurrencies.

Other studies mention fan tokens as a new financing tool. Солнцев et al. (2022) suggest that fan tokens help football clubs to broaden their earnings and, more relevant for the long term, to build fan loyalty. The research results reveal that the Binance market, in contrast to Socios, appears to be more effective for selling tokens. Such a situation is related to its accessibility and relevance in other spheres beyond fan tokens, as in Binance, a user can buy tokens not only from sports organizations, but also from other digital assets emitted by firms in other sectors. Demir and Aktas (2022) consider that the introduction of Fan Tokens on the market generates great interest in the sports industry, as well as in the financial sector. Sports clubs all over the world are regularly reporting on their success in launching Fan Tokens, the financial benefits they offer, and their upcoming initiatives.

Lopez-Gonzalez and Griffiths (2023) unveil that Fan tokens have garnered acclaim as a fresh avenue of income for sports organizations. This avenue complements their conventional revenue streams, such as broadcasting rights, matchday earnings, and sponsorship agreements. The focus is on attracting younger audiences and infusing a sense of modernity through digitization. Their findings suggest that fan token enterprises have marketed their offerings as 'fan engagement platforms,' fostering closer connections between fans and their teams. Moreover, the adverse economic repercussions of the COVID-19 pandemic on stadium attendance earnings have hastened the widespread adoption of fan token enterprises as a means of diversifying the financial sources of sports teams. Furthermore, in light of the COVID-19 pandemic, Ante et al. (2023) assert that the surge in fan tokens can be attributed to the financial setbacks suffered by major sports clubs amidst the pandemic. This situation prompted renowned teams like FC Barcelona, Atlético Madrid, Paris Saint-Germain, and Juventus F.C. to explore alternative avenues of income. Approaching the matter from a sports management standpoint, fan tokens are being considered an appealing remedy for creating fresh streams of revenue and bolstering the connection between sports clubs and their supporters. Ilia et al. (2022) examine the possibilities stemming from digital assets and their connected risks, highlighting their distinct features within sports organizations. They delve into the allocation plan for fan tokens and subsequently offer pertinent advice to Russian football clubs, which were chosen for this analysis due to the attainability of necessary data and the burgeoning market. The revenue landscape in the sports industry underwent a substantial transformation due to the coronavirus pandemic. Plummeting earnings prompted clubs to explore novel avenues for income, leading them to tackle this predicament through the utilization of digital assets. In the research conducted by Baker et al. (2022), a call is made to sport managers and scholars to actively involve themselves with this emerging technology. They urge for a proactive stance in steering NFT-related advancements to ensure fair results for all stakeholders in the sports industry. This encompasses athletes, sports enthusiasts, teams, leagues, and also strives to foster broader societal benefits.

Studies that have examined the dynamic connection between the fan tokens and their corresponding stocks are that of Ersan et al. (2022). They use daily data from December 11, 2020 to January 31, 2022 for four football clubs, such as Juventus FC, AS Roma, Galatasaray, and Trabzonspor. They are based on the estimation of the TVP-VAR model, and their results indicate that contributions are relatively higher for shares than for tokens, indicating that football clubs' stock returns are rather more influenced by idiosyncratic factors. In the studies of Fındıklı and Saygın (2021), the application in the marketing of Fan Tokens in the concept of "citizen-customer" behaviour is analysed. The assessments were implemented through the use of the stock of the title "Fan Token" on Twitter. In the digital transformation of marketing, there is evidence that tools developed with virtual assets help customer-citizen behaviour. Parham and Breitinger (2022) conclude that a fan token is akin to owning shares in a club, allowing any supporter to possess a modest portion of it. The token's value hinges on various aspects, including the club's worth, earnings, and its popularity. These factors fluctuate over time, influenced by the club's triumphs and setbacks. Zaucha and Agur (2022) examine the concept of "Fandom," which represents the social role fans adopt to indulge in activities associated with their devotion, within the context of sports and markets. Fan communities within the realm of sports exist as part of an "imagined community," but are affected by external shifts such as the widespread use of mass media and the heightened commercialization of fan engagements. Consequently, the act of collecting could be interpreted as a chance for financial investment. Participants assemble collections that not only contribute to potential financial benefits but also facilitate subsequent reinvestments.

Another line of research has been the study of the risk profiles of fan tokens. Scharnowski et al. (2021) report that fan token values remain extremely volatile and considerably more risky than established cryptocurrencies and demonstrated that investor interest in fan tokens, overall, correlates favorably with expected returns on the tokens. Dowling (2022a,b) also analyse the risk characteristics of fan tokens, and their results confirm the findings of Newall and Xiao (2021), who consider fan tokens in the regulatory context of gambling. They suggest that although regulators progressively prohibit marketing and sponsorship activities undertaken by the gambling sector, the same is not true for "gambling-like industries", such as financial trading apps and specific cryptocurrency-based businesses. This topic is also explored by Lopez-Gonzalez and Griffiths (2023), who scrutinize the gambling-like attributes embedded within the design of fan tokens. They examine these attributes from a public health standpoint, drawing comparisons to their implications within gambling contexts and their potential risks for fan token holders. Through an investigation centered on the fan tokens launched by F.C. Barcelona and Socios.com, they highlight the presence of several characteristics akin to those found in gambling products. Their findings suggest that fan tokens are presented in a manner that could lead fans to perceive their purchase as a requisite for demonstrating their devotion. The transformation of fan sentiment into a purchasing choice might be interpreted as exploiting fans, under the implied assumption that those who do not acquire tokens are somehow lesser in their support. The researchers conclude that the issuance of fan tokens could pose a liability for sports organizations if these tokens serve to estrange rather than genuinely engage fans.

Ilia et al. (2022) ascertain that a primary concern regarding fan tokens lies in their intangible nature as digital assets. Another issue involves the environmental impact, given that the mining hardware necessary for launching NFTs often consumes a substantial amount of energy. In response to this issue, new platforms are prioritizing energy efficiency and highlighting their commitment to environmental sustainability. They name the Tezos crypto platform, which has forged a partnership with Manchester United FC. Furthermore, they indicate that

financial and regulatory bodies are also expressing apprehension about potential risks linked to fan tokens. Regulators have expressed worry regarding the ease with which a fan token purchase could be executed using another cryptocurrency. They conclude that there are concerns surrounding the taxation of earnings denominated in cryptocurrencies, particularly in the context of fan tokens.

About the methodology that the fan tokens studies have employed, Ersan et al. (2022) apply a Time-varying parameter Vector Autoregressive methodology. The methodology of the study research of Солнцев et al. (2022) involved the review of yearly reports of football clubs and academic articles and interviews with representatives of the clubs and digital platforms. They examined the commercial performance of 47 foreign clubs, identifying and quantifying the factors that have an impact on their revenues. Demir and Aktas (2022) implement a regression analysis, and they find that fan token markets contain similar risks to crypto asset dominant markets. Fan Token movements exhibit greater similarity to dominant crypto assets when markets are less volatile and relatively calm. In conclusion, previous studies have not applied Genetic Algorithms (GA), which provides a convenient method for finding strategies to reach the greatest returns, as, given the volatility of financial markets, trading strategies require adapting to changing environments (Cocco et al., 2019). GA is a well-known metaheuristic algorithm, inspired by the process of biological development, based on chromosome representation, and this approach is applied to solve numerical optimization problems (Evans et al., 2013). Significant benefits of GA are that they are efficient in handling non-stationary data and that there is no need to adopt a particular distribution of data (Huang et al., 2019; Katoch et al., 2021). There are research studies that use GA from both a theoretical and practical point of view, but the latter is mostly used in engineering sciences, and very little in economics or finance (Drachal and Pawłowski, 2021). Therefore, possible applications of GA could be in finance, in particular bankruptcy prediction, risk management, and forecasting in finance (Sezer et al., 2020; Bustos and Pomares-Quimbaya, 2020).

In the last two decades, there has been an enormous growth in the number of proposed metaheuristic algorithms that are inspired by nature (Agushaka et al., 2022; Abualigah, 2019). However, no guarantee exists that these metaheuristic algorithms provide optimal solutions to many real-world optimization problems. Thus, developing new or hybrid methods is required, as GA, offers solutions to the most complex problems (Agushaka et al., 2022). Metaheuristic algorithms can be classified into four groups according to their source of inspiration: “swarm-based algorithms”, “evolution-based algorithms”, “physics-based algorithms” and” human-based algorithms”. Biological events such as natural evolution inspire evolution-based algorithms. GA stands out as the most popular algorithm in this group (Ezugwu et al., 2022). Meanwhile, Abualigah and Khader (2017) suggest a hybrid of particle swarm optimization algorithms with genetic operators for the feature selection problem. These methods contain population-based optimization algorithms, including harmonic search, genetic algorithm, and particle swarm optimization, and they have received significant research interest. Agushaka et al. (2022) conclude that hybridising algorithms becomes an alternative way of exploiting the advantages of different metaheuristic algorithms to build a more robust optimization algorithm.

Ante et al. (2023) offer a thorough examination of the Socios case study, employing a comprehensive approach drawing inspiration from Yin’s (2018) suggestions. Due to the relatively recent appearance of fan token platforms and systems based on blockchain technology, they opt for a solitary case study methodology in this research venture. This decision arises from the scarcity of comparable or contrasting fan token platforms and systems, rendering a multiple-case study unfeasible or of only partial value. Moreover, a solitary case study facilitates the investigation of real-world occurrences, particularly concerning fan engagement in connection with blockchain-based tokens. Assaf et al. (2023) direct their attention towards the identification of bubbles within fan tokens, employing the Supremum Augmented Dickey-Fuller (SADF) and Generalized Supremum Augmented Dickey-Fuller (GSADF) tests. They utilize the daily closing prices of the leading 20 fan tokens based on their market capitalization, in addition to Bitcoin, Ethereum, and Chiliz. The findings extracted from the GSADF test outcomes reveal that 13 out of the 20 fan tokens, alongside the three primary cryptocurrencies, exhibit periods of rapid price escalation linked to the presence of bubbles.

In a conclusion, there does not seem to be any research on the application of genetic algorithms (and their hybrids) specifically for Fan Token trading. Therefore, this paper fills an important gap in the literature. Table 1 displays a synopsis of this literature.

3. Model and methodology

3.1. Vanilla GA & rule-based GA

Genetic Algorithms (GAs) are commonly used to seek the optimal solution to a problem. The process starts with an initial population of solutions and employs random elements to merge and create new solutions. By iteratively evolving these solutions, the algorithm aims to converge towards the optimal solution.

GAs are considered a straightforward approach for addressing intricate nonlinear problems. This approach relies on the concept of breeding and reproduction within an initial population. It involves several operations such as selection, crossover, and mutation, which aid in generating new and more optimal individuals, as indicated by Liu et al., in 2020. Within the GA algorithm, the population size plays a crucial role, determining the total number of solutions and exerting a significant impact on the problem’s outcomes, as highlighted by Moayedi et al., in 2020. The term “generations” in this context pertains

Table 1
Literature summary.

Fan tokens research line	Author	Year
Impact of match results on fan tokens prices	Demir et al.	2022
	Mazur and Vega	2022
	Scharnowski, Scharnowski and Zimmermann	2021
	Vidal-Tomás	2023
Fan tokens as new financial tool	Aktas	2022
	Ante, Schellinger, and Wazinski	2023
	Baker, Pizzo, and Su	2022
	Солнцев, Алексеева and Сусов	2022
	Ilia, Anastasia, and Yaroslav	2022
	Lopez-Gonzalez and Griffiths	2023
Connection between fan tokens and their stocks	Ersan, Demir and Assaf	2022
	Fındıklı and Saygın	2021
	Parham and Breitinger	2022
	Zaucha and Agur	2022
Risk profile of fan tokens	Dowling	2022
	Ilia, Anastasia, and Yaroslav	2022
	Lopez-Gonzalez and Griffiths	2023
	Scharnowski, Scharnowski and Zimmermann	2021
Methodology used in fan tokens	Abualigah	2019
	Abualigah and Khader	2017
	Agushaka, Ezugwu and Abualigah	2022
	Ante, Schellinger, and Wazinski	2023
	Assaf, Demir, and Ersan	2023
	Bustos and Pomares-Quimbaya	2020
	Cocco, Tonelli and Marchesi	2019
	Солнцев, Алексеева and Сусов	2022
	Demir and Aktas	2022
	Drachal and Pawłowski	2021
	Ersan, Demir and Assaf	2022
	Evans, Pappas and Xhafa	2013
	Ezugwu et al.	2022
	Huang et al.	2019
	Katoch, Chauhan and Kumar	2021
	Sezer et al.	2020
	Yin	2018

to the iterations within the optimization process. In practical applications, the GA method has proven to offer numerous benefits in identifying an ideal set of resources to enhance both cost efficiency and output quality, as described by Liu et al., in 2020.

In this study, we utilize the GA implementation introduced by Boboc and Dinică (2013). This GA is designed to simulate a trading system, where individuals are represented by a set of technical analysis rules. These rules, referred to as chromosomes in GA, consist of two or more parameters known as genes.

To be more specific, the algorithm we utilized follows the following scheme.

1. GA commences by generating the initial population of solutions, referred to as chromosomes, through a random process. In this particular case, the GA creates 100 individuals by forming sets of technical analysis rules and assigning their parameters randomly. In our model, a chromosome represents a collection of these rules.
2. GA evaluates each individual in the population by assessing the profit or loss achieved during the training period. This evaluation process is performed by calculating the financial outcome of executing the trading strategies associated with each individual.
3. Subsequently, the GA sorts the 100 individuals in the population based on their individual profit or loss, which serves as the fitness function used for evaluation by the GA. This sorting process helps determine the relative fitness levels of the individuals within the population.
4. The GA proceeds to generate the new generation by following these steps:
   - The GA integrates the individual with the highest profit into the new generation, adhering to the principle of elitism.
   - Based on their profit and ranking, each of the 100 individuals has a chance of becoming a parent for the new generation. The probability of an individual becoming a parent is determined by their profit, with higher profits resulting in greater probabilities. To facilitate this, we categorized the individuals into ten classes and assigned probabilities to each class based on the best profit performances.
   - Using their assigned probabilities, the GA employs random selection to choose pairs of individuals as parents for generating new offspring. The selection of genes for the new individuals follows the method proposed by Boboc and Dinică (2013). In this approach, there are 24 genes numbered from 1 to 24. The GA randomly selects a number, denoted as “n,” ranging from 1 to 24. The new individual inherits genes from 1 to “n" from one parent and genes from “n + 1″ to 24 from the other parent. This process results in the creation of 80 new individuals.
   - To foster diversity within the new generation, the GA randomly generates the remaining 19 individuals. This random generation process ensures that a variety of individuals with different characteristics are introduced into the population, contributing to a diverse and exploratory search for optimal solutions.
5. Once the GA has generated the complete population of 100 individuals for the new generation, this generation becomes the current one. The GA then proceeds to repeat steps 2, 3, and 4, including the evaluation of individuals, sorting based on profit, and generating the next generation. This iterative process continues as the GA iteratively refines and improves the population towards finding optimal solutions.
6. The GA continues the iterative process by repeating steps 2 to 5 until it reaches 80 iterations. It has been observed that increasing the number of iterations beyond this point does not substantially impact the results. Therefore, the specified number of iterations is considered sufficient for achieving reliable outcomes.

3.2. Comprehensive overview of hybrid genetic algorithms

First, we have employed GA techniques to create the initial sets of market data to perform the simulation. GA represents one of the best-known algorithms, drawing its inspiration from the process of biological evolution. The GA emulates the Darwinian theory of the natural survival of the fittest and was suggested by Holland (1992). The basic elements of GA are chromosome mapping, fitness selection, and biologically instructed operators. Holland (1975) further contributed a new element, inversion, which is commonly applied in GA applications (Katoch et al., 2021). The GA techniques we have applied are Adaptive Boosting and Genetic Algorithm, Deep Learning Neural Network-Genetic Algorithm, Adaptive Genetic Algorithm with Fuzzy Logic, and Quantum Genetic Algorithm. Subsequently, we have implemented the Monte Carlo algorithm, which is used to simulate the market.

In our research, we have integrated several tools with the aim of being able to analyse, compare, and determine the most robust method that achieves the best model optimization. We have endeavoured to shed light on the efficacy and efficiency of hybrid genetic algorithms with diverse techniques. This was accomplished by scrutinising a range of hybrid genetic approaches. This presents an opportunity within hybrid optimization to harness the strengths of each approach. El-Mihoub et al. (2006) demonstrate that hybridisation offers a viable avenue to construct a capable genetic algorithm, capable of swiftly, dependably, and accurately tackling complex challenges, all without necessitating human intervention. Hybrid genetic algorithms can be viewed as supplementary tools that can be amalgamated for comparison, thus determining the most effective approach (Zhou et al., 2020).

3.2.1. Genetic algorithm methods

GA is regarded as a simple solution to complex nonlinear problems. This method is based on the process of mating, and reproduction in an initial population, combined with various activities like selection, cross over, and mutation, which support the creation of new, more optimal individuals (Liu et al., 2020). In the GA algorithm, the population size is an important factor that captures the total number of solutions and greatly influences the results of the problem (Moayedi et al., 2020), while the named “generations” relate to the iterations of the optimization process. In practice, the GA method has demonstrated many advantages in finding an optimal set of resources to optimize both cost and output (Liu et al., 2020).

3.2.1.1. Adaptive Boosting and genetic algorithm (AdaBoost-GA)

3.2.1.1.1. Introduction. In contrast to the conventional AdaBoost algorithm, which sets the weight of each base classifier without considering adaptivity (Wang et al., 2011), this research incorporates Genetic Algorithm (GA) in the adaptive integration procedures of base classifiers. The number of decision groups corresponds to the number of weak classifiers in AdaBoost, and the weight of each weak classifier serves as the starting population for GA.

Both crossover probability and mutation probability significantly impact the algorithm’s optimization effect (Cheng et al., 2019). To choose the suitable crossover likelihood and mutation likelihood, based on the literature (Drezner and Misevičius, 2013), we describe the crossover likelihood and mutation likelihood as:

$$P_c = \gamma \tag{1}$$

$$P_m = 0.1\ (1 - \gamma) \tag{2}$$

being γ a regulatory factor.

3.2.1.1.2. Integration of AdaBoost and genetic algorithm. The Adaboost function builds base classifiers using training data distribution and assigns weights based on the error rate (Cheng et al., 2019; Qu et al., 2021). The decision group serves as the base classifier in AdaBoost to

enhance ensemble set diversity, and the GA algorithm maximizes the weight of each base classifier by combining all base classifiers.

3.2.1.1.3. Algorithm description. Given $\{\omega_{m,j} | i = 1, 2, \ldots, N; m = 1\ldots, M\}$, which is the weight of every sample in the base classifier. $\omega_{m,i}$ symbols the weight of the ith sample in m_{th} base classifier. Let $y_m(x)$ and Y(x) be a base classifier and strong classifier, respectively (Qu et al., 2021). α_m denotes the weight of m_{th} classifier. ε_m represents the error function of m_{th} base classifier. And M constitutes the number of base classifiers.

The AdaBoost-GA suggested algorithm can be explained as follows:

- **Inputs**: Training sets and testing sets, being $\omega_{1,i}$ the weight of each training sample.
- **Output**: Y (.): The final strong classifier follows the rules:

$$\bullet Y(x) = sign\left(\sum_{j=1}^{M} \alpha_j y_j(x)\right) \tag{3}$$

1. **Initialization**: $\omega_{1,i} = 1/N$
2. **Loop**: For i = 1 to N do:

$$\varepsilon_m = \sum_{i=1}^{N} \omega_{m,i} I(y_m(X_i) \neq y_i) \tag{4}$$

$$\alpha_m = 1n\left\{\frac{1 - \varepsilon_m}{\varepsilon_m}\right\} \tag{5}$$

if $\alpha_m \geq 0$ then α_m increases with the decrement of ε_m.

End if

$$\omega_{m+1,i} = \frac{\omega_{m,i}}{Z_m} exp(-\alpha_m y_i y_m(x_i)) \tag{6}$$

$$Z_m = \sum_{i=1}^{N} \omega_{m,i}\, exp\left(-\alpha_m y_i y_g(x_i)\right) \tag{7}$$

3. Genetic Algorithm Step:

$$\alpha_m = GA(\alpha) \tag{8}$$

4. Final Strong Classifier:

$$Y_x = sign\left(\sum_{j=1}^{M} \alpha_j y_j(x)\right) \tag{9}$$

The structure of the "AdaBoost-GA" consists of three phases (Qu et al., 2021):

- **Pre-processing and Feature Extraction Phase:**
  - o Randomly select two separated training and testing datasets.
  - o Convert symbolic features into numerical ones.
  - o Select the most suitable features.
- **AdaBoost Training and Testing Phase:**
  - o **Training Sub-phase**: Train AdaBoost using the training set, iterating TT rounds, producing weak classifiers htht, and ensemble weights αtαt to form the final strong classifiers.
  - o **Testing Sub-phase:** Measure system performance with the testing set to choose the best hypothesis.
- **Post-Optimization Procedure Phase:**
  - o **Initialization with AdaBoost:** Initialize individuals in the initial population with a proportion of the final classifier produced by AdaBoost.
  - o **Fitness Function:** Measure the goodness of a solution, influencing the next generation based on fitness.

All algorithms are written in standard format according to previous literature (Ford, 2014; Zhang et al., 2019; Rai et al., 2023). The pseudo algorithm of this method is shown below in Table 2.

3.2.1.2. Deep learning neural network- genetic algorithm (DLNN-GA)

3.2.1.2.1. Introduction. The procedure of multiple forward dispersion is defined in the next linear model linking the incident optical modes and the transferred optical modes (Vellekoop and Mosk, 2008).

$$E_m = \sum_{n=1}^{N} t_{mn} E_n = \sum_{n=1}^{N} |t_{mn}| exp(i\varphi_{mn}) |E_n| exp(i\varphi_n) \tag{10}$$

being E_n the nth complex incident mode with amplitude $|E_n|$ and phase φ_n, while E_m represents the mth complex optical mode transferred from the dispersion media. t_{mn} symbols one element in the complex transmission matrix that constitutes light scattering paths. The phase values fulfil $\varphi_n = -\varphi_{mn}$ (Bossy and Gigan, 2016). The light will focus perfectly on the selected location when it is set to this condition.

3.2.1.2.2. Integration of DLNN and genetic algorithm. The procedure of using the GA for wavefront modeling involves five steps: initialization, classification, reproduction, mutation, and iteration. First, a given number G of phase patterns is generated, each phase value being selected from a uniform pseudo-random distribution. Next, these standards are marked using a specially designed fitness function. Following formula (11), the fitness function is described as the intensity of the light at a given location (Conkey et al., 2012)

$$I_m = |E_m|^2 = \frac{1}{N}\left|\sum_{n=1}^{N} t_{mn} A_n\, exp(i\varphi_n)\right|^2 \tag{11}$$

being A_n the amplitude of E_n. Phase patterns are classified according to the results of the fitness function assessment. High scores lead to higher rankings. The second step is breeding. Offspring is produced by offspring = T × ma + (1 − T) × pa, with T being a random template and ma and pa being parental parents. Both parents are sorted under the rule that higher-ranking parents are more likely to be adopted. After reproduction, certain sections of the offspring become mutated and changed by chance. The mutation rate R declines with growing generations n to prevent over mutation, as follows $R = (R_0 - R_{end}) x\, exp(-n/\lambda) + R_{end}$, being R_0, R_{end}, and λ the initial mutation rate, the final mutation rate, and the decay factor, for each of them (Conkey et al., 2012). The progeny shall also be assessed by the fitness function. In each generation, a certain number of offspring will be replicated to substitute already existing patterns with inferior scores (Conkey et al., 2012). Then, the G-stage patterns are all reclassified based on their scores. The

Table 2
Pseudo-code of AdaBoost-GA algorithm.

```
initialize
Input: A set S of m instances: S = {(xm, ym)} where xi ∈ X with labels yi ∈ Y = {-1, +1}.
Parameters: P (population size), G (maximum number of generations), T (initial
  number of weak learners).
Initialization: Generate a randomly generated population of m solutions (consisting of
  a set of T weak learners with their weights produced by AdaBoost).
Evolution Loop:
  For gen = 1 to G:
  Generate a population of bit strings b.
  Evaluate the fitness of the solutions: f(b) = w1 * (1 - L/T) + w2 * (1/E)
  Use vector b to update the set of weak classifiers (T) and their weights.
  Produce a new generation of solutions using genetic operations (selection,
  mutation, and crossover).
  if end condition is met
    Stop the algorithm and return the best solution
  else
    Continue to the next generation.
Output: Final hypothesis with optimized classifiers and their weights
```

Note: In fitness step, b is the evolved best individual, w_1, w_2 are fitness weights, E is the validation error, and L = ∑ bi.

previous reproduction and mutating proceedings will be repeated several times before the final condition is fulfilled. Usually, the iteration ceases when a prespecified number of generations is replayed or the result of the fitness function evaluation meets a certain level of threshold.

The advantages of the GA are important. The GA manages to identify a suitable solution quickly. In addition, GA is robust to noise, as it updates the largest number of pixels rather than adjusting pixels one by one. While GA results are significantly affected by many factors, such as the mutation and reproduction rate, the fitness function, and particularly the number of phase patterns employed in every generation, namely the size of the population, matching the right parameters is not trivial and needs time and experience. Furthermore, several scenarios haphazard introduce start patterns, potentially in the neighbourhood of one or more local minima. GA is susceptible to getting locked into a local minimum, as it is a staged optimization process and descendants reproduce themselves by replicating and mutating existing patterns. This involves the risk that likely better solutions cannot be probed. Consequently, the use of a good initialization is crucial to reach global optima (Bodenhofer, 2003).

3.2.1.2.3. Algorithm description. Deep learning, which is a data-driven process, employs a separate strategy to estimate the phase pattern for focusing the light. The poor proposal and nonlinearity of inverse scattering problems show that direct inversion is impractical, which makes the requirements of iterative algorithms with regulation (Wei and Chen, 2018) to be as below:

$$arg\min_{p}\left\|y - HW_p\right\|_2^2 + \lambda\|p\|_1 \tag{12}$$

being H the forward scattering model, y represents the recorded speckle intensity, W symbols a transformation, and p denotes transformation coefficients so that $\hat{x} = W_p$ is the desired reconstruction.

Nearly every state-of-the-art iterative algorithm for back-scattering problems are cascades of linear convolutions and pointwise nonlinear transactions, (McCann et al., 2017), resembling the structure of convolutional neural networks (CNNs). Examples of a representative implementation are the well-known iterative shrinkage-thresholding algorithms (ISTAs) founded on model blocks as follows:

$$p^{m+1} = A_\theta\left[\frac{1}{L}W * H * y + \left(I - \frac{1}{L}W * H * HW\right)p^m\right] \tag{13}$$

being L the Lipschitz constant. The iterative optimization governed by formula (18) can be handled as a convolutional procedure with kernel $I - \frac{1}{L}W * H * HW$ and bias (1/L) W * H * y, followed by a nonlinear activation function A_θ. So, CNNs can be regarded as inherently suitable for solving the problem of reverse dispersion (Wei and Chen, 2018; Li et al., 2018).

Conventional iterative algorithms are successful and warrant the use of CNNs as a way to approach light via scattering media. Deep CNNs (DCCNs) have been demonstrated powerful in solving inverted problems (Li et al., 2018; Lucas et al., 2018).

Therefore, DCNNs can model the H-1 reverse scattering process through supervised learning, disclosing the connection between the transferred speckles and the incident x optical phase patterns. The inputs of the DCNN are the measured intensity distributions of the transferred speckles captured with a camera, and the outputs are their incident phase patterns modified by a spatial light modulator (SLM). Upon training, the DCNN will accurately map the speckle to the incident phase patterns, and therefore the DCNN can forecast the phase pattern needed to target the light via a particular dispersion media.

Using the DL approach is easier, as the relation of speckles to incident phase patterns is acquired directly via training. However, its output is influenced strongly by the training samples. Accordingly, these samples are typical to represent the complete scattering processes only when the training sample size is large enough, and thus, the DCNN can forecast the best phase pattern for the light focus following the training. However, the supervised learning method experiences the dilemma of achieving the global optimum because it is imprudent to incorporate every conceivable phase pattern and speckle for training, and sample topics are hard to estimate as well. Despite this, the DCNN results can be considered a good initialization for the GA. Concerning the GA, maximum global convergence is obtained under the constraint that the initial figure is close to the global optima (Bodenhofer, 2003).

The suggested method contains two parts. The first part consists of gathering samples to train a DCNN. Following the training, an early focus mote with the phase pattern predicted by the DCNN can be obtained. The second part is the adoption of the GA for optimising the focused process. We present two methods for building early phase patterns, each employing the DCNN results. The first method, called GeneNNv1, adopts the pattern foreseen by the DCNN for one of the starting patterns, whereas other patterns are generated according to a uniform pseudo-random distribution (Conkey et al., 2012). The DCNN pattern will undoubtedly have the strongest classification, thus having the highest chance of being selected for breeding.

In DLNN, every node in a layer becomes assigned a certain weight, denoted as w_{ij}, with every node in the other layers building a fully linked neural system (Zemouri et al., 2020). Except for the input layer, each node is a neuron using a nonlinear activation function. In addition, MLP employs a supervised learning approach named backpropagation for the training process. Due to its multilayer and nonlinear activation functions, DLNN could discriminate separable nonlinear data (Malsburg, 1986).

The development of the DLNN with nonlinear activation functions becomes essential to improve the accuracy of the model and to emulate better the mechanism of operation of biological neurons (Karakış et al., 2013). The use of sigmoidal functions becomes widely accepted in the DLNN network, with two conventional activation functions as follows:

$$y(v_i) = \tanh(v_i)\ and\ y(v_i) = (1 + e^{-v_i})^{-1} \tag{14}$$

The former provides a hyperbolic tangent, with a range from −1 to 1, while the latter is a logistic function with a similar shape but with a range from 0 to 1. In these functions, y (v_i) denotes the output of the node i^{th}, and v_i means the total weight of the input connection (Karakış et al., 2013). In addition, alternative activation functions, like the rectifier, or even more sophisticated ones, such as radial basis functions, are suggested.

Based on the errors of the output versus the target, the linkage weights and biases are adapted, causing the learning process to take place. Hence, this could be regarded as an example of the supervised learning process via the least-squares mean algorithm, generalized as a backpropagation algorithm (Zemouri et al., 2020). Specifically, an error at the output node j at the n^{th} data point comes from:

$$e_j(n) = d_j(n) - y_j(n) \tag{15}$$

being d the target value, y means the value generated by the perceptron system. The next expression is based on error correction to minimize the errors of the expected output to establish the weights of the nodes:

$$\varepsilon\ (n) = \frac{1}{2}\sum_j e_j^2(n) \tag{16}$$

In addition, the expression below employs the gradient descent algorithm to estimate the change, or correction, for each weight:

$$\Delta\omega_{ji}(n) = -\eta\frac{\partial\varepsilon(n)}{\partial v_j(n)}\ y_i(n) \tag{17}$$

being y_i the output of the previous neuron means the learning rate. These parameters are selected to guarantee that the error rapidly becomes convergent and free of oscillations (Conkey et al., 2012). In addition, the derivative is calculated from the local induced field v_j,

expressible as:

$$-\frac{\partial \varepsilon(n)}{\partial v_j(n)} = e_j(n)\,\varphi'\big(v_j(n)\big) \tag{18}$$

being φ' the derivative of the activation function. With the weight change linked to a hidden node, the relevant derivative can be displayed as:

$$-\frac{\partial \varepsilon(n)}{\partial v_j(n)} = \varphi'\big(v_j(n)\big)\sum_k -\frac{\partial \varepsilon(n)}{\partial v_k(n)}\omega_{kj}(n) \tag{19}$$

This function relies on the changes in the weights of the nodes corresponding to k_{th} output layer (Bodenhofer, 2003). This algorithm reflects the inverse backpropagation procedure since the output weights change as a function of the derivative of the activation function, and the weights of the hidden layer consequently change. The pseudo algorithm of this method is shown below in Table 3.

The training process with DLNN means a slow and costly method because of the use of computational resources processing (Dimiduk et al., 2018). Moreover, some features of the dataset could influence the regression results, and unnecessary features could create noise and decrease the prediction precision (Abusamra, 2013). Choosing suitable features needs substantial effort, for example, the sum of combinations C (10,i) for i from 1 to 10 might be produced with a dataset comprising 10.

Table 3
Pseudo-code of DLNN-GA Algorithm.

initialize
Input: Rule 1, Rule 2, Rule 3, Rule 4, Rule 5
Output: Final trained weights W_{ij}
  Set hyper-parameters: $\alpha, \eta, c, \mu_{pre}, \mu_0, \mu_{inc}, E_{pre}, and\ E_{auto}$
  DLNN structure based on nodes in every layer (with 5 layers as inputs)
  Setting type of training database, weights, mini-batch size, and Epoch
    if $\xi_i == 0$
      Training data
      Weight initialization: "He" weight initialization.
      Mini-batch size: $m_i = \mu_{pre}$
      Epoch = E_{pre}
    else if $\xi_i > 0$
      Training data
      Weight initialization
      Mini-batch size: $m_i = \mu_0 + \tau_i.\mu_{inc}$
      Epoch = E_{auto}
    end
  Scan train dataset $\{X_{(k)}, T_{(k)}\}_{k=1}^{k=N_{DB}}$ to get the size of the training dataset N_{DB}
  Obtain scaled train dataset $\{x_{(k)}, t_{(k)}\}_{k=1}^{k=N_{DB}}$
  Calculate the mini-batch learning iteration size: $N_{MB} = floor\left(\frac{N_{DB}}{m_i}\right)$
    for i-epoch = 1: Epoch
      Data-preprocessing: Randomly shuffle and arrange scaled train dataset
      for j = 1: N_{MB}
        Sample scaled training data: $\{x_{(k)}, t_{(k)}\}_{k=1+m_i.(j-1)}^{k=m_i.j}$
        Calculate gradient: $g_W \leftarrow \nabla_W J(W)$
        Gradient clipping: $g_W^{clip} \leftarrow clip(g_W, [-c, c])$
Optimization: $W \leftarrow W + \eta.Adam\,(g_W^{clip}, \beta_1, \beta_2, e\,)$ with default values
      end
      for each chromosome i in Population
        for each gene j
          initialize G_{ij} randomly within a permissible range
        end for
      end for
    Generation k = 1
    for
      for each chromosome i in Population
        Calculate the fitness value of G_i
      end for
      Mating the best chromosomes to produce more agents
      Mutates some agents randomly to attempt to find even better candidates
      Remove the weakest chromosomes, based on fitness value, from the Population k = k + 1
    While maximum generation

Note: α is the negative slope; η is the learning rate; c is the gradient clipping value; μ_{pre} is the mini-batch size for pre-training; μ_0 is initial mini-batch size; μ_{inc} is the increment of mini-batch size; E_{pre} is the pre-training Epoch; E_{auto} is the auto-adaptive training Epoch.

To ease the feature selection procedure, the GA algorithm was used to select the suitable features within the dataset, hoping that a smaller number of input variables could improve the prediction accuracy of DLNN-GA.

3.2.1.3. Adaptive genetic algorithm with fuzzy logic (AGAFL)

3.2.1.3.1. Introduction. A genetic algorithm first generates a group of potential solutions represented as chromosomes. Each chromosome's ability to spread its traits, or fitness, depends on how well it solves the problem - better fits have a greater chance to reproduce. The process starts with making a random population of m chromosomes. It then calculates each chromosome's performance. The algorithm repeatedly selects parents based on fitness, combines their traits in crossover to make new offspring, and occasionally randomly tweaks traits in mutation. New offspring enter the next generation population. This allows the population as a whole to gradually improve as less fit solutions are less likely to pass on their genes over many iterations. This iterative process repeats until a defined stopping point is reached. The technique leverages the newly created sample for additional rounds, ending when the ending requirement is met. Upon reaching this point, the most favorable result from the current sample is provided, and the technique returns to step 2 for additional refinement and improvement.

3.2.1.3.2. Integration of AGAFL with genetic algorithm. The adaptive genetic algorithm (AGA) enhances the basic genetic algorithm by incorporating adaptive mutations to optimize results. Unlike the standard genetic algorithm, AGA uses adaptive mutations on each parent chromosome, where the mutation computation rate is linked to the chromosome's fitness. Adaptive mutation involves creating chromosomes, estimating fitness, crossover, adaptive mutation, and selection.

- **Chromosome Generation (Step 1):** Chromosomes are created from the solution set using fuzzy logic. Each chromosome represents the parameters of fuzzy rules.

$$Ch_k = \left[G_0^k G_1^k \ldots G_{C_L-1}^k\right] 0 \le k \le M-1; 0 \le i \le C_L - 1 \tag{20}$$

being G_i^k the jth gene of the chromosome, M represents the total population and CL denotes the chromosome's length.

- **Calculating Fitness Function (Step 2):** The fitness function is shown in formula (21). The main objective of the fitness function is to maximise the rules as solutions are selected.

$$f_t = \sum_{K=1}^{M} R_s \Big/ M \tag{21}$$

where s represents $\frac{m}{k}$ to be inserted in the sum expression, and some parameters to ameliorate. In this case, R_s is the chosen rule and M is the rule total count.

The fitness value, f_t for every chromosome is calculated according to the adopted rules. Each chromosome is tested with the fitness function (Herrera and Lozano, 2001). Solutions that fulfil the fitness function, using mutation, will be chosen to participate in breeding.

- **Crossover (Step 3):** To create a new chromosome, a cross between two parent chromosomes is made. The new chromosome produced is named the offspring. The crossover is performed according to the targeted genes and the yield of the offspring depends on the crossover rate (CO rate). The formula used to determine the crossover point appears below.

$$CP_{rate} = \frac{CG}{CL} \tag{22}$$

being CP_{rate} the Crossover Rate, CG represents the number of Genes Generated and the CL symbols length of the chromosome.

According to the computed CO rate, the chromosomes of the parents make a crossover that produces a set of new chromosomes called offspring. Through CO, we find the crossover point, and the genes of these points are exchanged from the chromosomes of both parents, resulting in offspring that have features from the chromosomes of both parents (Herrera and Lozano, 2001). Generated chromosomes will have a greater fitness in comparison to the previous generation of chromosomes, so they will be more suitable for processing.

- **Adaptive Mutation (Step 4):** The mutation is done according to a rate of mutation, which is estimated as:

$$MU_r = \frac{P_m}{C_L} \tag{23}$$

being MU_r the Mutation rate, P_m represents the Mutation Point and C_L symbols the length of the chromosome.

Due to the rules determined by fuzzy logic, the fitness value serves as the basis for this method. A mutation rate comparison with the given fitness values is performed based on the threshold and the resulting values are chosen as the final mutation rate. The vector depicting the mutation points follows:

$$MU_r = \left\{mp_{1,} mp_{2,\ldots} mp_{l,}\right\} \tag{24}$$

being l length of the chromosome. The rate of mutation r identification is done according to the fitness f_t

$$MU_r = \begin{cases} 1; if\ f_t \le T \\ 0; else \end{cases} \tag{25}$$

where the calculation of T is performed based on the resulting fuzzy rule. The mutation rate varies for each chromosome over each iteration and relies on the fitness value.

- **Selection (Step 5):** Given the fitness value achieved, the new chromosomes (N_p) are placed in a sorting pool. In the selection pool, the chromosomes with the best fitness value will be kept at the top (Herrera and Lozano, 2001). The highest N_p chromosomes reserved in the selection pool are selected as the next generation among the $2N_p$ chromosomes.

3.2.1.3.3. Algorithm description. The feature mitigation process reduces the computing cost and raises the classification performance. To enhance the results in prediction, rough ensembles for feature reduction are employed in this work; fuzzy logic classifier is utilized to build the rule set. AGA is trained on the solution set to obtain optimal rules for prediction (Buckley and Hayashi, 1994; Herrera and Lozano, 2001). There are several steps in the prediction model: rough set-based attribute reduction, normalisation, and then AGAFL classification. Each step of the described model is detailed below:

Standardization (Step 1): Let us take the dataset comprising the number of attributes and entities. Normalisation is performed on the dataset to decrease the arithmetic complexity of the data changing it into a range of a particular type. The widely used min-max method is used for normalisation. The initial dataset is converted by min-max normalisation to a range via

$$D^n = \frac{D - D_{min}}{D_{max} - D_{min}} X[new_{min} - new_{max}] + new_{min} \tag{26}$$

The range of transform datasets is denoted by new_{min}, new_{max} ; where $new_{min} = 0$ and $new_{max} = 1$.

Reduced Attributes through Rough Sets (Step 2): Reducing the attributes using rough sets is the primary task. Moreover, the number of attributes is minimised and details that are irrelevant, disjoint, noisy, or even redundant are removed (Buckley and Hayashi, 1994; Herrera and Lozano, 2001).

Depiction of the Solution (Step 3): In each bit 1 displays the

selection as 0, meaning the non-selection of the equivalence attribute. To quote a scenario, the data set comprising 10 attributes $(a_1, a_2, a_3, \ldots a_{10})$, then the attributes selected will become $(a_1, a_3, a_5, a_6, a_9)$.

Fitness Function (Step 4): The fitness value of each solution is determined by the fitness function and the best solution is selected according to the fitness value. The set of fuzzy logic classifier rules will constitute the population where the fitness function will be implemented (Buckley and Hayashi, 1994; Herrera and Lozano, 2001). The rule chosen to engage in reproduction to create the following generation will be the superset of S_f. Let $R = \{r_1, r_2, r_3, \ldots r_m\}$ be the set of rules to be considered to spawn the next population. Let $R_f \subseteq R$ *where* R_f is the set of rules holding super sets of S_f. The goodness of each solution is assessed by the fitness function S_f.

Completion Standards (Step 5): Only if the maximum number of iterations is achieved will the algorithm cease its implementation. The solution containing the highest fitness value is chosen using RS, and the AGAFL is employed to rank the datasets (Buckley and Hayashi, 1994; Herrera and Lozano, 2001). The best attributes are provided as input to the fuzzy classifier.

Forecasting with the Fuzzy Logic System (Step 6): Having performed feature reduction on the input dataset, the hybrid ADAFL classifier forecasts the model. The fuzzy logic classifier involves three stages: Fuzzification, Fuzzy inference engine, and De-Fuzzification.

A fuzzy inference method helps to map inputs to the equivalent output employing predicted fuzzy rules. Information support involves if-then rules indicating the relation between the fuzzy input and output groups. The inference system is enforced by a series of actions such as the development of fuzzy rules, fuzzified values of input according to a degree of membership, merging of fuzzified input like fuzzy rules to enhance rules' strength, and at last, the output is destabilised to obtain the output measured as a sharp value (Buckley and Hayashi, 1994; Herrera and Lozano, 2001).

Incoming data is converted to a membership value between 0 and 1 using the membership function (MF). The triangular membership approach is chosen to transform the input data into a fuzzy value (Buckley and Hayashi, 1994; Herrera and Lozano, 2001). The principle used to evaluate the membership values is described next:

$$f_x = \begin{cases} 0 \; if \; x <= i \; and \; if \; x \geq k \\ \dfrac{x-i}{j-i} \; if \; i \leq x \leq j \\ \dfrac{k-x}{k-j} \; if \; j \leq x \leq k \end{cases} \tag{27}$$

The generation of fuzzy rules plays a significant role in helping to record the input in its corresponding output. If $A_1, A_2, \ldots, A_N$ represent the attributes and $C_1, C_2,$ symbol of the class labels therefore a fuzzy rule can be elaborated from the linguistic values such as high, medium, and low. The values N and M represent the number of attributes and the number of classes accordingly. Hence, the fuzzy rule can be formulated in the following way:

- If A_1 is large and A_2 is small and A_3 is central then the class is C_2 ;
- If A_1 is small and A_2 is middle and A_3 is middle then the class is C_1 ;
- If A_1 is large and A_2 is medium and A_3 is small then the class is C_2 ;

Test data containing narrowed attributes are entered into the AGAFL, whereas the test data is transformed into a fuzzy value using the fuzzy membership function. The fuzzy input is then compared with the fuzzy rules proposed in the rule base (Buckley and Hayashi, 1994; Herrera and Lozano, 2001). Again, the rule inference technique is employed to derive the linguistic value, which is then transformed into the fuzzy rating utilizing the weighted average approach. From the derived fuzzy score, the ranking determination is made. In the following Table 4, we explain the algorithm (pseudo-code) of this method.

Table 4
Pseudo-code of AGAFL algorithm.

```
t1 = 0, t2 = 0
Initialize P(t1)
Initialize P(t2) under fuzzy symbols
Evaluate P(t1)
While (not termination-condition) do
  t1 = t1 + 1
  Select P(t1) from P(t1 − 1)
  Assign each fuzzy behaviour in P(t2) to npp pairs of chromosomes in P(t1), at random
  Calculate p for each pair of chromosomes through its fuzzy rules associated
  Perform adaptive crossover (using the p values) and mutation on P(t1)
  Evaluate P(t1)
  Collect performance measures about the behaviour of fuzzy rules in P(t2) on P(t1)
  If then
    Evaluate P(t2)
    t2 = t2 + 1
    Select P(t2) from P(t2 − 1)
    Perform crossover and mutation on P(t2)
```

3.2.1.4. *Quantum genetic algorithm (QGA)*

3.2.1.4.1. Introduction. QGA is an evolutionary algorithm built on the concept of quantum computing. It introduces notions like superposition states in quantum computing and incorporates the single encoding form to obtain improved experimental results in the combinatorial optimization problem (SaiToh et al., 2014). Nevertheless, when it comes to the optimization of multimodal functions using QEA, in the specific, high-dimensional multimodal function optimization problem, it is likely to drop into local optimum and its computing efficiency is poor.

This study aims to improve the global optimization capacity of the genetic algorithm and the local search ability according to the quantum probabilistic model to introduce a new type of quantum evolutionary algorithm, namely the "quantum genetic algorithm", to deal with the above deficiencies of QEA. This algorithm utilises the quantum probabilistic vector encoding mechanism and takes the crossover operator of the genetic algorithm and the updating strategy of quantum computation simultaneously to optimize the global search capacity of the quantum algorithm effectively.

3.2.1.4.2. Algorithm description. This quantum genetic algorithm follows the next steps described below:

- **Step 1: population initialization**

The lowest unit of information in QGA is a quantum bit. The state of a quantum bit can be 0 or 1, expressed as:

$$|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{28}$$

being α, β two complex numbers corresponding to the likelihood of happening of the respective state: $(|\alpha|^2 + |\beta|^2 = 1), |\alpha|^2, |\beta|^2$ symbol the likelihood of the quantum bit in the 0 and 1 state accordingly (Zhang et al., 2006; Nowotniak and Kucharski, 2010).

The most commonly adopted coding techniques in EA include coding, decimal coding, and symbolic coding. In QGA, a new method of coding is introduced using the quantum bit, namely the use of a pair of complex numbers to describe a quantum bit. A system with m quantum bits is expressed as

$$\begin{bmatrix} \alpha_1 & \alpha_2 & \ldots & \alpha_m \\ \beta_1 & \beta_2 & \ldots & \beta_m \end{bmatrix} \tag{29}$$

In the equation, $|\alpha_i|^2 + |\beta_i|^2 = 1$ (i = 1, 2, …, m). This method of display may be applied to describe any linear superposition of states (SaiToh et al., 2014). For instance, a system of three quantum bits having the next probability amplitudes:

$$\left[\begin{array}{c|c|c} \frac{1}{\sqrt{2}} & \frac{\sqrt{3}}{2} & \frac{1}{2} \\ \frac{1}{\sqrt{2}} & \frac{1}{2} & \frac{\sqrt{3}}{2} \end{array}\right] \tag{30}$$

The system state can be defined as

$$\frac{\sqrt{3}}{4\sqrt{2}}|000\rangle+\frac{3}{4\sqrt{2}}|001\rangle+\frac{1}{4\sqrt{2}}|010\rangle+\frac{\sqrt{3}}{4\sqrt{2}}|011\rangle+\frac{\sqrt{3}}{4\sqrt{2}}|100\rangle+\frac{\sqrt{3}}{4\sqrt{2}}|101\rangle+\frac{1}{4\sqrt{2}}|110\rangle+\frac{\sqrt{3}}{4\sqrt{2}}|111\rangle \tag{31}$$

- **Step 2: conduct individual coding and measuring of the population generating units.**

QGA is a probabilistic algorithm analogous to EA. The algorithm is $H(t)=\{Q_1^t, Q_2^t, \ldots Q_h^t, \ldots Q_l^t\}$ h = 1,2, …l) being h the size of the population, $Q_l(t)=\{q_1^t, q_2^t, \ldots q_j^t, \ldots q_n^t\}$ where n represents the number of generator units, t denotes the evolution generation, q_j^t symbols the coding of the generation volume of the jth generator unit (Wang et al., 2013). Its chromosome is shown below:

$$q_j^t=\left[\begin{array}{c|c|c|c} {\alpha_1}^t & {\alpha_2}^t & \ldots & {\alpha_m}^t \\ {\beta_1}^t & {\beta_2}^t & \ldots & {\beta_m}^t \end{array}\right] \tag{32}$$

(j = 1, 2, …, n) (m is the length of the quantum chromosome).

During the "initialization of H(t)," if ${\alpha_1}^t$, ${\beta_1}^t$ (i = 1, 2, …, m) in $Q_l(t)$ and all the q_j^t are initialized, it denotes that all the possible linear superposition states will happen with equal likelihood (Nowotniak and Kucharski, 2010). Over the step of "generating S(t) from H(t)", a common solution set S(t) is created through observation of the state of H(t), where in the t_{th} generation, $S(t)=\{P_1^t, P_2^t, \ldots, P_h^t, \ldots, P_l^t\}, P_l=\{x_1^t, x_2^t, \ldots, x_j^t, \ldots, x_n^t\}$. Every x_j^t (j = 1, 2, …, n) is a series, $(x_1, x_2, \ldots, x_i, \ldots, x_m)$, of length m, which is reached from the amplitude of the quantum bit $|\alpha_i^t|^2$ or $|\beta_i^t|^2$ (i = 1, 2, …, m). The relevant procedure in the scenario is to randomly identify a number [0, 1]. Take "1″ if it is larger than $|\alpha_i^t|^2$; take "0" otherwise.

- **Step 3: Make an individual measure for every item in S(t)**

Employ a fitness assessment function to test each object in S(t) and maintain the best object in the generation (Nowotniak and Kucharski, 2010). If you get a satisfactory solution, the algorithm stops; if not, proceed to the fourth step.

- **Step 4: Apply an appropriate quantum rotation gate U(t) to update S(t)**

The conventional genetic algorithm utilises mating and mutation operations, etc. to keep the population diverse. The quantum genetic algorithm uses a logic gate to the likelihood amplitude of the quantum state to preserve the diversity of the population. Hence, the method of updating by a quantum gate is the essence of the quantum genetic algorithm (Nowotniak and Kucharski, 2010). The system, adaptation values, and the probability amplitude comparison technique are utilized for updating using a quantum gate in the classical genetic algorithm. This approach to updating via a quantum gate is adequate for solving combinatorial optimization problems with an in-principle optimum. Nevertheless, for real optimization problems, especially those optimization problems of multivariable continuous functions, whose best solutions are in principle not available beforehand. Hence, a quantum rotation gate of the quantum logic gate for the new quantum genetic algorithm is assumed here.

$$U=\begin{bmatrix} \cos\theta & \sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{33}$$

being θ the quantum gate rotation angle. Its value is shown as

$$\theta = k.f(\alpha_i, \beta_i) \tag{34}$$

$$k=\pi.\exp\left(-\frac{t}{iter_{max}}\right) \tag{35}$$

We consider k as a variable linked to the evolution generation to adjust the mesh size in a self-adaptive way. Let t be the evolution generation, π is an angle, $iter_{max}$ is a constant that relies on the complexity of the optimization problem (SaiToh et al., 2014). The aim of the function $f(\alpha_i, \beta_i)$ serves to cause the algorithm to seek the best direction. It is based on the idea of gradually bringing the actual search solution closer to the optimal solution and thus setting the direction of the quantum rotation gate.

Thus, the process of implementing the quantum rotation gate to the entire probability amplitude for the individual object in the population, namely by applying the quantum rotation gate U(t) to update S(t), in the quantum genetic algorithm may be written as:

$$S(t+1)=U(t)\times S(t) \tag{36}$$

being t the evolution generation, U(t) represents the t_{th} generation quantum rotation gate, S(t) symbols the t_{th} generation probability amplitude of a certain object, S (t + 1) denotes the $t+1^{th}$ generation probability amplitude of the relevant object.

- **Step 5: perturbation**

Since QGA is inclined to get caught at a better local extreme value, we disturb the population. QGA analysis has shown that if the best individual of the present generation is a local extreme value, the algorithm is very difficult to free (Nowotniak and Kucharski, 2010). Thus, the algorithm is stuck at the local extrema if the best individual remains unchanged in subsequent generations. A perturbation must be implemented to the population at this time to leave the local optimum and begin further research.

Table 5
Pseudo-code of QGA algorithm.

```
begin
t ← 0
initialize Q (0)
make P (0) by observing Q (0)
evaluate P (0)
store the best solution among P (0)
while not termination-criterion do
  t ← t + 1
  make P(t) by observing Q (t-1) population
  evaluate P(t)
  update Q(t) using quantum gates U (θt)
  store the best solution among P(t)
end while
end
```

The full pseudocode of QGA is given in Table 5. At the beginning of the algorithm, the genes from all individuals of the quantum population $\mathscr{C}$ (0) are initiated with linear superposition $\left(\frac{\sqrt{2}}{2}|0\rangle+\frac{\sqrt{2}}{2}|1\rangle\right)$ (Wang et al., 2013). This equates to a sampling uniform distribution of the search space. The evaluation of the fitness of the individual in the algorithm is implemented by observing $\mathscr{C}$ which results in P. In this step, the search space is sampled concerning the probability distributions encoded by quantum individuals Q(t). The genetic operators applied in the algorithm are based on "quantum rotation gates", which rotate state vectors in the quantum gene state space. The quantum gate U(θ) rotates the state of the quantum gene $|\Psi\rangle$ by the angle θ. Thus, states of quantum genes are updated according to the following principle of quantum computing:

$$|\Psi\rangle_{t+1}=\cup(\theta)|\Psi\rangle_t=\begin{bmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{bmatrix}\begin{bmatrix}\alpha\\ \beta\end{bmatrix} \tag{37}$$

where: θ represents the angle of rotation in the qubit state space, and t displays the iteration number (Nowotniak and Kucharski, 2010).

3.2.2. Markov chain Monte Carlo (Metropolis Hasting algorithm)

The Metropolis-Hasting (MH) algorithm is a sophisticated and popular sampling method (Heratha and Herath, 2018). It is based on Markov chains and is related to rejection techniques since it is necessary to propose a test value and it is not necessary to know the normalisation of the distribution function to be sampled. The method is motivated by its analogy with the behaviour of systems near equilibrium in statistical mechanics. The evolution of the system (process) is described by the probability that it goes from state *X* to state *Y*. The condition that the system approaches equilibrium means that on average the system has the same probability of being in *X* and it will happen *Y* what to be in *Y* a to transform to *X*, which is expressed by the detailed balance equation,

$$\mathrm{f(X)P(Y|X)=f(Y)P(X|Y)} \tag{38}$$

where $f(X)P(Y|X)$ is the likelihood $f(X)$ to find the system in the vicinity of *X* by conditional probability $P(Y|X)$ for the system to evolve from *X* to *Y* (Ayekple et al., 2018). In a physical process, $P(Y|X)$ is known, and it is about finding $f(X)$. The MH algorithm claims the opposite: given an $f(X)$ is about finding the transition probability that brings the system to equilibrium. Transitions are proposed from *X* to *Y* following any test distribution $T(Y|X)$. It compares $f(Y)$ with $f(X)$ and it accepts *Y* with likelihood $A(Y|X)$. Therefore,

$$P(Y|X)=A(Y|X)\ T(Y|X) \tag{39}$$

A Markov chain made up of the states is constructed $X_0, X_1, X_2, \ldots X_N$ which is reached in each transition, from an initial X_0 (Heratha and Herath, 2018). Each X_n is a random variable that will satisfy the following condition (Ayekple et al., 2018).

$$\lim_{n\to\infty}\varnothing_n(X)=f(X) \tag{40}$$

At each step of the random path, there is a transition probability $T(Y|X)$ which is normalized,

$$\int dY\ T(Y|X)=1 \tag{41}$$

Assuming it is always possible to go from *Y* to *X* if it is possible to go from *X* to *Y*, and vice versa, we define,

$$q(Y|X)=\frac{T(X|Y)f(Y)}{T(X|Y)f(X)}\geq 0 \tag{42}$$

From this one on $q(Y|X)$ we can take as the probability of acceptance,

$$A(Y|X)=mn\ \{q(Y|X),1\} \tag{43}$$

3.3. Application domain with problem

The model presented in this paper builds upon the research conducted by Cocco and Marchesi (2016). It incorporates realistic trading strategies to simulate the trading of the BTC/USD currency pair and capture the key characteristics observed in the bitcoin price series. The approach used in this study is agent-based, which has been widely recognized in related works (Chakraborti et al., 2011; Chen et al., 2012; Chen et al., 2018) as an effective approach.

Our objective is to simulate the trading of the prices of Fan Tokens of football clubs whose output is therefore the price of these Fan Tokens in US dollars (USD), as done in previous works (Boboc and Dinică, 2013; Cocco and Marchesi, 2016; Cocco et al., 2019), so that the algorithms can calculate the optimization of the trading rules that can give the highest profits to the agents.

This paper introduces an agent-based artificial cryptographic token market based on two agents, random traders, and chartists. Random traders place orders randomly. They operate for various purposes, like diversifying their investment portfolio or supplying a liquidity need, rather than for speculation. They place buy and sell orders with equal probability and in a manner consistent with their readily crypto and fiat cash on hand.

Chartists emit orders using signals derived from specific trading rules. These agents are in the market to speculate and to make large returns through placing buy and sell orders. Given a set of trading rules, consisting of an opening rule and an ongoing rule, a chartist joins the market by opening a position, and thus emitting a buy or sell order, once he has tested his opening rule providing him with buy and sell advice. The chartist, when a position has been opened, expects the best time to execute the position and either take the gain or reduce the losses.

In this paper, traders utilize rules to form expectations regarding prices or gains, drawing inspiration from the works of Chiarella et al. (2009) and Licalzi and Pellizzari (2003), where rules are used to anticipate stock returns. Specifically, Chartists make decisions to open or close positions based on the assessment of technical analysis indicators, such as the Exponential Moving Average, Moving Average Convergence Divergence, Relative Strength Index, or Genetic Filter. Additionally, the Genetic Algorithm (GA), employed to select the optimal set of trading rules used by a subset of Chartists, incorporates a mechanism like the one proposed by Tedeschi et al. (2012). In the study, traders mimic the expectations of the most successful traders, whereas in this paper, the GA selects the trading rules that generate the highest profits.

At the start time t = 0, all traders in the market hold an initial amount $c_i(0)$ of fiat currency (cash, in US dollars) and an initial amount $b_i(0)$ of cryptocurrency (fan tokens), where i represents the trader's index. When the i-th trader enters the market at time $t_i^E > 0$, they only possess an amount c_i (t_i^E) of fiat currency (cash, in dollars), denoted by the superscript E for entering.

Similar to the findings of Cocco, Concas, and Marchesi in their 2017 work, the sum of bitcoins owned by each trader at the initial time represents 60% of the total number of bitcoins in the market. This assumption accounts for the unavailability of 40% of fan tokens for trading. As time progresses, new fan tokens enter the market, mirroring real-life circumstances. We assume that 60% of new fan tokens join the market every 90 days.

To distribute the new fan tokens, we randomly allocated them to a fraction of traders, with the amount received being proportional to their existing fan tokens holdings. This allocation method adheres to the Gibrat principle of preferential attachment, where wealthier individuals have a higher likelihood of receiving more (Gibrat, 1931; Yule, 1925).

The initial distribution of wealth among traders, both in terms of cryptocurrency and fiat cash, follows a Zipf law (as described in Levy and Solomon's 1997 work). This distribution is generated prior to the simulation, using the approach outlined in the works of Cocco et al. (2017) and Cocco and Marchesi (2016).

To determine the number of traders, N_T, in the market at a given time t, we utilized a fitting curve. The curve N_T (t) was fitted using seven data points, following the methodology outlined in the work of Cocco and Marchesi (2016).

The fitting curve for the number of traders, N_T, is characterized by employing a general exponential model:

$$N_T(t) = a * exp^{b*t} \tag{44}$$

The best fitting parameters, a and b, are determined to be 1.744e+04 and 0.002465 respectively. The initial value of t is set to 1824, which corresponds to January 1st, 2014, marking the starting date for our simulations. This selection aligns with the approach proposed in the works of Cocco et al. (2017) and Cocco and Marchesi (2016).

In accordance with the aforementioned approach, when a trader enters the market at a specific time t, they are assigned to either the Chartists' population or the Random Traders' population. The probability of belonging to the Chartists' population, percC, is set to 0.6, while the probability of belonging to the Random Traders' population, percR, is set to 0.4. It is important to note that percC + percR equals 1.

In particular, chartists operate under a precise set of trading rules chosen randomly by themselves upon entering the market, which can be either optimal or random trading rules. The probability, $\boldsymbol{percC_B}$ of employing the best rules equals 0.48, and the probability, $\boldsymbol{percC_R}$ of employing the random rules equals 0.12, being $\boldsymbol{percC_B} + \boldsymbol{percC_R} = \boldsymbol{perC}$. The first _best_ rules are chosen by an GA selecting the parameters of the rules to maximise the profitability of the operators. The second _random_ rules are not optimized, and their parameters are determined randomly. Hereafter, we will refer to the part of chartists embracing the best set of rules by $\boldsymbol{C_{bR}}$, and to the rest by $\boldsymbol{C_{rR}}$. Chartists engage in trading by employing a specific set of trading rules that they personally select. These rules are utilized when they enter the market, and they can either be chosen from a set of well-established trading rules or from a set of randomly generated trading rules (Cocco et al., 2017; Cocco and Marchesi, 2016).

As in the work of Boboc and Dinică (2013), there are 6 rules in our model, specifically four rules for opening a position and two for exiting a position, and 24 parameters. We summarise it in Table 6.

The rules for position opening are:

Rule 1: Filter, based on five parameters:

Table 6
Rules and parameters of the model.

Rules	Parameters
Filter	$filter_{flag}$
	$filter_{periods}$
	$filter_{increaseS}$
	$filter_{decreaseS}$
	$filter_{booleanS}$
Relative Strength Index (RSI)	rsi_{flag}
	rsi_n
	rsi_{os}
	rsi_{ob}
	$rsi_{boolean}$
Exponential Moving Average (EMA)	ema_{flag}
	ema_n
Moving average convergence divergence (MACD)	$macd_{flag}$
	$macd_{periodsS}$
	$macd_{periodsL}$
	$macd_{periodsN}$
	$macd_{booleanS}$
Fixed exit levels (FEL)	fel_{flag}
	fel_{tp}
	fel_{sl}
Trailing exit levels (TEL)	tel_{flag}
	tel_{tp}
	tel_{sl}
	tel_{tl}

1. $filter_{flag}$ turns on or off the rule. The value 0 deactivates the rule, while the value 1 activates it.
2. $filter_{periods}$, noted n. Previous price to be considered.
3. $filter_{increaseS}$, noted p;
4. $filter_{decreaseS}$, noted q;

The third and fourth are variables that display the exchange rate of a currency pair.

5. $filter_{booleanS}$ is a boolean signal that identifies the negotiation signals.

Rule 2: Relative Strength Index (RSI), defined as:

$$RSI_t(n) = \frac{\sum_{i=t-n}^{t} \frac{max_i(P_i - P_{i-1}, 0)}{n}}{\sum_{i=t-n}^{t} \frac{max_i(P_i - P_{i-1}, 0)}{n} + \sum_{i=t-n}^{t} \frac{max_i(P_{i-1} - P_i, 0)}{n}} \tag{45}$$

being P the closing price of the period and n the number of the periods applied to calculate the RSI. RSI provides buy and sell signals according to whether it is higher or lower than the overbought and oversold signals. It is defined based on the next parameters:

6. rsi_{flag} activates or deactivates the rule.
7. rsi_n is the number of terms to be included in the calculation of mean highs and lows.
8. rsi_{os};
9. rsi_{ob};

Parameters eight and nine are the oversold and overbought signals accordingly.

10. $rsi_{boolean}$.is a boolean signal, describing the trading signals, taking values 0 or 1 with the same probability. By acquiring value 0, it provides a buy signal when the RSI value is less than rsi_{os} and a sell signal when the RSI value is greater than rsi_{ob}.

Rule 3: Exponential Moving Average (EMA), which is a weighted moving average and is described as:

$$EMA_t(n, Close) = \frac{2}{n+1} Close_t + EMA_{t-1}\left(1 - \frac{2}{n+1}\right) \tag{46}$$

being *EMA*, *n*, and *Close* the value of this indicator at the prior moment, the number of periods, and the closing price of the period, correspondingly.

This rule has two parameters:

11. ema_{flag} activate and deactivate the rule.
12. ema_n is the period n.

Rule 4: Moving average convergence divergence (MACD), which is described as:

$$MACD_t(p, q, m) = MACD_t(p, q) - Signal_t(m) \tag{47}$$

where:

$$MACDline_t(p, q) = EMA_t(p, close) - EMA_t(q, close) \tag{48}$$

$$SignalLine_t(m) = EMA_t(m, MACD_t(p, q)) \tag{49}$$

being p, q, and m are the number of periods of the short and the long exponential moving average, and of the *MACDline* indicator correspondingly. And Close represents the closing price of the period.

The parameters of this rule are:

13. $macd_{flag}$. The value 0 deactivates the rule, while the value 1 activates it.
14. $macd_{periodsS}$, noted p.
15. $macd_{periodsL}$, noted q and with the restriction q > p
16. $macd_{periodsN}$, noted m, is the difference between the long and short moving mean.
17. $macd_{booleanS}$ takes the values 0 or 1 with the same probability.

The rules for exiting the position are:
Rule 5: Fixed exit levels (FEL), characterized by three parameters:

18. fel_{flag} activate and deactivate the rule.
19. fel_{tp} is a value entering into the assessment of an actual profit-taking situation.
20. fel_{sl} is a trigger value that comes in the analysis of an exit position to stop loss.

Rule 6: Trailing exit levels (TEL), which is characterized by four parameters:

21. tel_{flag} activate and deactivate the rule.
22. tel_{tp} is a trigger value entering into the assessment of an actual profit-taking situation.
23. tel_{sl} is a trigger value that comes in the analysis of an exit position to stop loss.
24. tel_{tl} is a threshold value entering in the assessment of an exit position with $tel_{tl} < tel_{tp}$.

A 24-bit chromosome is used to represent 24 parameters, not all of which are Booleans. Each bit within the 24-bit chromosome corresponds to a specific parameter or feature used in the model, and these parameters encompass a mix of Boolean values and numerical values.

The 24-bit chromosome is used to represent these parameters:

1. **Filter Parameters (Rule 1)**:
   - o $filter_{flag}$ (1 bit): Represents whether the filter rule is activated (Boolean).
   - o $filter_{periods}$ (5 bits): Numerical value representing the number of periods (not Boolean).
   - o $filter_{increaseS}$ (8 bits): Numerical value representing the increase rate for the currency pair (not Boolean).
   - o $filter_{decreaseS}$ (8 bits): Numerical value representing the decrease rate for the currency pair (not Boolean).
   - o $filter_{booleanS}$ (2 bits): Boolean signal for negotiation signals (Boolean).
2. **Relative Strength Index (RSI) Parameters (Rule 2)**:
   - o rsi_{flag} (1 bit): Represents whether the RSI rule is activated (Boolean).
   - o rsi_{n} (5 bits): Numerical value representing the number of terms included in RSI calculation (not Boolean).
   - o rsi_{os} (4 bits): Numerical value representing the oversold signal threshold (not Boolean).
   - o rsi_{ob} (4 bits): Numerical value representing the overbought signal threshold (not Boolean).
   - o $rsi_{boolean}$ (1 bit): Boolean signal for trading signals (Boolean).
3. **Exponential Moving Average (EMA) Parameters (Rule 3)**:
   - o ema_{flag} (1 bit): Represents whether the EMA rule is activated (Boolean).
   - o ema_{n} (5 bits): Numerical value representing the period for EMA calculation (not Boolean).
4. **Moving Average Convergence Divergence (MACD) Parameters (Rule 4)**:
   - o $macd_{flag}$ (1 bit): Represents whether the MACD rule is activated (Boolean).
   - o $macd_{periodsS}$ (5 bits): Numerical value representing the short moving average period (not Boolean).
   - o $macd_{periodsL}$ (5 bits): Numerical value representing the long moving average period (not Boolean).
   - o $macd_{periodsN}$ (5 bits): Numerical value representing the difference between long and short moving averages (not Boolean).
   - o $macd_{booleanS}$ (1 bit): Boolean signal for MACD (Boolean).
5. **Fixed Exit Levels (FEL) Parameters (Rule 5)**:
   - o fel_{flag} (1 bit): Represents whether the FEL rule is activated (Boolean).
   - o fel_{tp} (5 bits): Numerical value representing the trigger value for profit-taking (not Boolean).
   - o fel_{sl} (5 bits): Numerical value representing the trigger value for stop loss (not Boolean).
6. **Trailing Exit Levels (TEL) Parameters (Rule 6)**:
   - o tel_{flag} (1 bit): Represents whether the TEL rule is activated (Boolean).
   - o tel_{tp} (5 bits): Numerical value representing the trigger value for profit-taking (not Boolean).
   - o tel_{sl} (5 bits): Numerical value representing the trigger value for stop loss (not Boolean).
   - o tel_{tl} (5 bits): Numerical value representing the threshold value (not Boolean).

In summary, the 24-bit chromosome encodes a combination of Boolean flags (representing whether specific rules are activated) and numerical values (representing various parameters and thresholds). This encoding allows the Genetic Algorithm to optimize and select the most suitable combination of rules and parameter values for the trading system.

The values of the parameters related to the Genetic Algorithm (GA) used in the study (see Table 7):

Initial Population Size: The initial population size is 100 individuals (chromosomes).
Number of Iterations: The GA continues the iterative process until it reaches 80 iterations.
Crossover Method: Two-point crossover is used for recombination.
Mutation Rate: The mutation probability is set to 0.001, indicating a low likelihood of gene mutation.
Chromosome Encoding: Each chromosome is represented as a 24-bit binary string.
Selection Method: Roulette wheel selection is employed, where selection probabilities are proportional to the fitness of individuals.
Elitism: Elitism is applied by preserving the best chromosome from the previous population.

Table 7
GA parameters values.

Relevant GA parameters	Values
Initial Population Size	100 indiviudals chromosomes
Number of Iterations	80
Crossover Method	Two-point crossover
Mutation Rate	0.001
Chromosome Encoding	24-bit binary string
Selection Method	Roulette wheel selection
Elitism	Preserve the best chromosome from the previous population
Fitness Function	A combination of information gain, F-statistic, and Pearson's correlation coefficient
Crossover Points	Two-point crossover
Replacement Strategy	Update the existing chromosomes with newly generated offspring chromosomes
Genes and Parameters	24 genes representing the trading rules and parameters
GA Termination Criteria	The GA terminates after 80 iterations

Fitness Function: The fitness function is a combination of information gain, F-statistic, and Pearson's correlation coefficient (PCC) to evaluate the relevance and interdependencies among the selected features.
Crossover Points: Two-point crossover is used, implying that two random points on the chromosome are selected for crossing.
Replacement Strategy: Replacement is applied to update the existing chromosomes with newly generated offspring chromosomes. Elitism is also used to preserve the best solution.
Genes and Parameters: There are 24 genes representing the trading rules and parameters. These genes correspond to the features used in the model.
GA Termination Criteria: The GA terminates after 80 iterations, as it has been observed that increasing the number of iterations beyond this point does not substantially impact the results.

3.3.1. Order execution and price formation mechanism

The trading mechanism, in accordance with the studies by Cocco et al. (2017) and Cocco and Marchesi (2016), relies on a realistic order book that effectively organizes buy and sell orders into separate lists and matches them for optimal execution. Buy orders are sorted in descending order based on their limit price, while sell orders are arranged in ascending order. In cases where orders have the same limit price, further sorting is done based on their issuance time. It's important to highlight that the limit price represents the desired price at which traders intend to finalize their transactions, and this value is determined by the current price p(t), and a random variable denoted as $N_i(\mu, \sigma_i)$. Following the approach presented by Cocco et al. (2017) and Cocco and Marchesi (2016), the limit prices associated with buy and sell orders, as well as the price, can be identified as $P^l_{b,i}$ and $P^l_{s,i}$, associated with buy and sell orders respectively, as well as the price, p_T , are identified as in the following:

$$P^l_{b,i}(t) = p(t) * N_i(\mu, \sigma_i) \tag{50}$$

$$P^l_{s,i}(t) = \frac{p(t)}{N_i(\mu, \sigma_i)} \tag{51}$$

being $N_i(\mu, \sigma_i^c)$ a random variable drawn from a Gaussian distribution with average $\mu \backsimeq 1$ and standard deviation σ_i ; p(t) the current fan token price. It is closely connected to the price charged for a transaction, p_T. The mechanism for the formation of the price p_T is the following:

if $P^l_{b,i} > 0$ and $P^l_{s,i} = 0$ then $p_T = \min(P^l_{b,i}, p(t))$;
if $P^l_{s,i} > 0$ and $P^l_{b,i} = 0$ then $p_T = \max(P^l_{s,i}, p(t))$;
if $P^l_{b,i} = 0$ and $P^l_{s,i} = 0$ then $p_T = p(t)$;
if $P^l_{b,i} > 0$ and $P^l_{s,i} > 0$ then $p_T = \frac{P^l_{b,i} + P^l_{s,i}}{2}$.

3.4. Existing difficulties

Simulating financial markets is a formidable task due to their ever-changing and unpredictable nature. These markets are inherently non-linear systems, where minor shifts in one variable can trigger disproportionately significant repercussions in other aspects, rendering the creation of precise predictive models a formidable challenge (Ben Abdelaziz and Mrad, 2023). Moreover, financial markets operate in a realm of randomness and uncertainty. Asset prices and market conduct are subject to an array of unforeseeable events, such as economic data releases, geopolitical occurrences, and unexpected news, making them notably challenging to predict (Maeda et al., 2020). Additionally, the actions and sentiments of individuals play a pivotal role in the functioning of financial markets. Modeling these aspects accurately is intricate since human behavior is frequently irrational and influenced by emotions (Sinha et al., 2019).

The emergence of algorithmic and high-frequency trading has intensified the pace and intricacy of interactions within the market. Achieving precise simulations of these high-frequency transactions necessitates robust computational capabilities. Stock simulators have progressed in sophistication, nearly replicating real-world trading experiences. However, most of these simulators lack a real-time trading environment with live prices. This limitation can significantly affect trading decisions, as they may vary substantially from those made with the advantage of real-time price data, given the common 15–20-min lag in simulated stock trading programs (Zhou, 2021).

Regulatory changes are another constraint in simulating financial markets. These markets are susceptible to alterations and interventions driven by regulations, which can have a substantial impact on market behaviour. Keeping simulation models aligned with evolving regulatory requirements is an ongoing challenge (Hirano et al., 2020).

3.5. Existing solutions with GA or without GA

Over the past two decades, there has been a significant proliferation of metaheuristic algorithms inspired by natural processes (Agushaka et al., 2022; Abualigah, 2019). Nevertheless, there is no assurance that these nature-inspired metaheuristic algorithms can consistently deliver optimal solutions for a wide range of real-world optimization problems. Consequently, there is a need for the development of novel or hybrid methods, such as GAs, which are well-suited for addressing the most intricate problems (Agushaka et al., 2022). Abualigah and Khader (2017) propose a hybrid approach, combining particle swarm optimization algorithms with genetic operators to tackle feature selection problems. These methods encompass various population-based optimization algorithms like harmonic search, GA, and particle swarm optimization, and they have garnered significant research attention. Agushaka et al. (2022) suggest that the integration of algorithms into hybrid forms presents a viable strategy for harnessing the strengths of diverse metaheuristic algorithms, ultimately resulting in more robust optimization algorithms.

GA techniques and their combinations enable the incorporation of dynamically changing weights, making them well-suited for effectively addressing a wide range of highly unpredictable scenarios in real-world markets, such as the volatile nature of token trading. These models, characterized by their probabilistic and time-dependent features, seem to naturally mirror the complex behaviors exhibited by individuals in economic contexts (Drachal and Pawłowski, 2021). In a practical example, Cocco et al. (2019) employ GA theory to simulate the trading of the BTC/USD currency pair. Furthermore, in line with the discoveries made by He et al. (2008), it's evident that GAs provide resilient and powerful adaptive search methods. Their effectiveness in tackling various complex problems within the realm of electric power systems, such as economic dispatch, unit commitment, reactive power control, hydrothermal scheduling, and distribution system planning, highlights their adaptability and versatility. A distinctive characteristic of GAs is their capacity to simultaneously explore multiple solutions, setting them apart from conventional optimization algorithms.

Other research studies have sought solutions without resorting to GA. For example, Ante et al. (2023) have conducted a comprehensive examination of the Socios case study, following an approach inspired by Yin's (2018) recommendations. Given the relatively recent emergence of fan token platforms and blockchain-based systems, they have chosen to adopt a single-case study methodology for their research. This decision stems from the limited availability of comparable or contrasting fan token platforms and systems, making a multiple-case study unfeasible or of limited value. In a similar vein, Assaf et al. (2023) have focused their research on the detection of bubbles within fan tokens. They employ the Supremum Augmented Dickey-Fuller (SADF) and Generalized Supremum Augmented Dickey-Fuller (GSADF) tests. Their analysis encompasses the daily closing prices of the top 20 fan tokens based on their market capitalization, alongside Bitcoin, Ethereum, and Chiliz. This

approach provides an alternative to GA-based methods for addressing the subject at hand.

3.6. Function evaluation

3.6.1. Encoding and fitness function

In the context of utilizing a genetic algorithm, the initial step involves establishing an encoding scheme and defining a fitness function. In this study, the solution provided by the genetic algorithm is represented as a chromosome, utilizing a suitable data structure for encoding. Specifically, a binary bit string was employed for encoding, indicating the inclusion or exclusion of each feature. In simulations, a 24-bit string was utilized as a chromosome for forecasting the behaviour of the Fan Tokens' prices. These specific bit lengths were chosen to accommodate the required parameters of trading rules specified in Table 1 and ensure the appropriate representation of the solution space for the prediction task.

In a genetic algorithm, the assessment of fitness plays a crucial role in evaluating how effectively an encoded chromosome solves a given problem. The fitness value is derived from the implemented fitness function, which varies based on the genetic filter approach. For the genetic filter, fitness is expressed as a numerical value obtained by combining the correlations among the selected features. This fitness function aims to measure the relevance and interdependencies among the chosen features, providing a quantitative assessment of the chromosome's effectiveness.

3.6.2. Selection

Selection is a critical step in each generation of a genetic algorithm, as it determines which parent chromosomes will be chosen to produce offspring chromosomes. In this study, we utilized roulette wheel selection based on fitness as our selection method. Each chromosome was assigned a selection probability proportional to its fitness, and the chromosomes were then randomly chosen based on these probabilities. This approach ensures that chromosomes with higher fitness have a greater likelihood of being selected as parents, while chromosomes with relatively lower fitness have a lower probability of being chosen. By implementing this selection strategy, the genetic algorithm favors the reproduction of individuals with superior fitness, promoting the evolution of the population towards better solutions over successive generations.

3.6.3. Crossover

Crossover is a vital operation that generates offspring for the next generation by combining parental chromosomes selected through the process of selection. In this study, we implemented multi-point crossover as our crossover method. Multi-point crossover extends the concept of one-point crossover by selecting multiple random points on the chromosomes and crossing them at those points. Unlike one-point crossover, which involves a single crossing point, multi-point crossover allows for two or more points to be selected for the crossing process. It is important to note that when an even number of multi-point crossovers is used, it can result in the crossing of chromosomes with a circular shape. This circular shape arises because the first and last genes of the chromosomes are adjacent to each other in this scenario. While multi-point crossover explores a wider solution space compared to one-point crossover, excessive perturbation introduced by multi-point crossover can impede convergence. Additionally, when an odd number of points is used in multi-point crossover, it may not maintain the uniform traits of the selected chromosomes. In this study, we employed circular-shaped chromosomes, representing a list of features without any specific order. To strike a balance between perturbation and convergence, and to facilitate effective crossover within the circular structure, we utilized a 2-point crossover approach. This choice allows for a moderate level of perturbation while still maintaining the desirable traits of the selected chromosomes.

3.6.4. Mutation and replacement

Mutation serves as an essential operator within the genetic algorithm, introducing changes to the genes of a chromosome to prevent premature convergence and promote population diversity. In the mutation process, a random number between 0 and 1 is generated for each gene in the chromosome. If the generated value is below a specified threshold, the corresponding gene is modified arbitrarily. In this study, a mutation probability of 0.001 was set, indicating a low likelihood of gene mutation following previous works (Greenwell et al., 1995; Hassanat et al., 2019; Moon and Kim, 2020).

Replacement, on the other hand, is an operator that replaces the chromosomes of the current population with the offspring chromosomes generated through crossover and mutation. In our approach, we applied replacement to update the existing chromosomes with the newly generated offspring chromosomes. This process ensures that the population evolves and adapts based on the improved solutions obtained from the offspring.

Furthermore, we incorporated elitism in our approach, which involves preserving the best chromosome from the previous population and carrying it over to the next generation (Fig. 1). This elitist strategy ensures that the best solution found so far is retained, preventing the loss of valuable genetic information. By combining replacement with elitism, we maintain a balance between exploration (through the introduction of new offspring) and exploitation (by preserving the best solution) throughout the evolutionary process.

3.6.5. Genetic Filter

Filter-based feature selection techniques, such as those introduced by Moon and Kim (2020), Kim and Yoon (2015), and Cho et al. (2021), offer the advantage of efficiently deriving subsets of features by uncovering correlations among them within a short timeframe. However, they also face challenges in accurately quantifying the relevance and redundancy between the selected features. To address these drawbacks, we propose a novel fitness function in this study that highlights the benefits of filter-based feature selection. The fitness function, denoted as Equation (9), gives priority to feature subsets exhibiting strong correlations with the target variable while minimizing correlations among the selected features themselves. By doing so, this approach aims to strike a balance between maximizing the relevance of the selected features and minimizing redundancy. Using this novel fitness function, we enhance the effectiveness of the feature selection process, enabling the identification of highly relevant feature subsets while reducing redundancy. This approach improves the overall performance of the feature selection algorithm, leading to more accurate and reliable models for subsequent analysis and prediction tasks.

$$fitness = \sum_{i=1}^{n} f_{s_{target},s_i} - \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} f_{s_i,s_j} \tag{52}$$

subject to $f_{s_i,s_j} = IG_{s_i,s_j} + F_{s_i,s_j} + C_{s_i,s_j}$, being the total number of features, s_{target}, the target variable, and IG, F, and C, the information gain, F-statistic, and Pearson correlation coefficient (PCC), accordingly.

Explanation of Equation (52):

Equation (52) represents a fitness function used for filter-based feature selection. It comprises two parts: the first part aims to maximise the correlation between the selected features and the target variable (s_{target}), while the second part aims to minimize the correlations among the selected features themselves. This two-fold approach ensures that the chosen feature subsets are highly relevant to the target variable while avoiding redundancy among the selected features.

The fitness function involves three main components, which are information gain (IG), F-statistic (F), and Pearson correlation coefficient (PCC). IG measures the mutual information between features and the target variable, F-statistic quantifies the relevance of a feature to the target, and PCC assesses linear correlations. By incorporating these metrics, Equation (52) effectively captures diverse correlations within chromosomes.

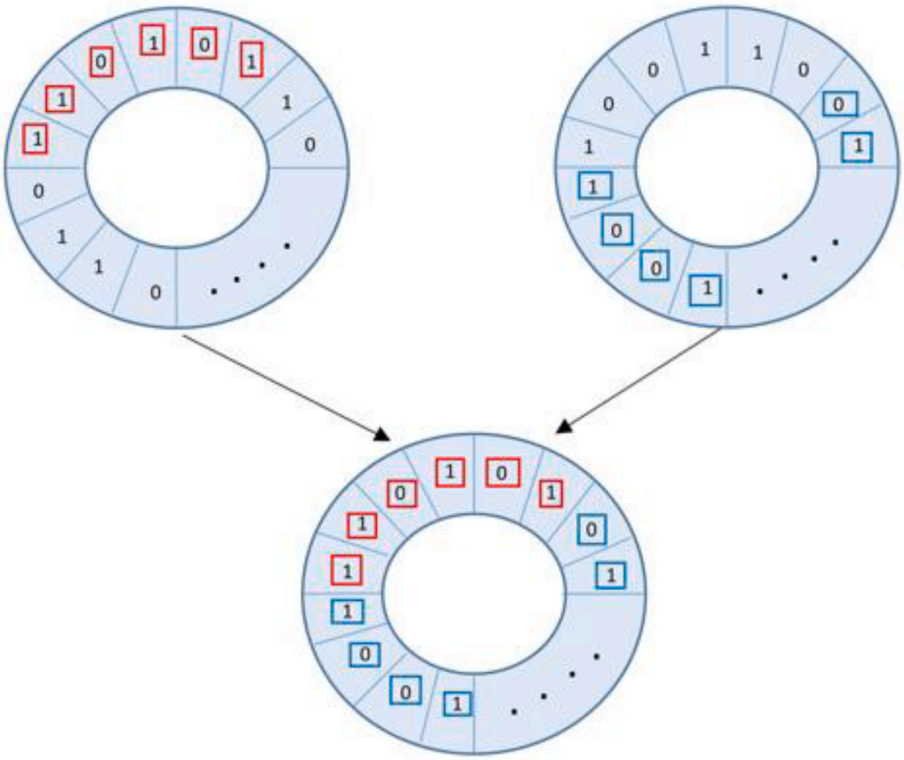


Fig. 1. An example of our two-point crossover.

Importance of Equation (52):

Maximizing Relevance: The first part of Equation (52) emphasizes the importance of selecting features that have a strong correlation with the target variable. This is crucial for improving the predictive power of models. Features that are highly correlated with the target variable are more likely to provide valuable information for the analysis and prediction tasks at hand.
Minimizing Redundancy: The second part of Equation (52) focuses on reducing redundancy among the selected features. Redundant features can lead to overfitting and increased computational costs. Minimizing correlations among selected features helps ensure that the chosen feature subsets are not duplicating information, resulting in more efficient and reliable models.
Striking a Balance: Equation (52) strikes a balance between maximizing relevance and minimizing redundancy. This balance is essential for an effective feature selection process. Selecting features solely based on their correlation with the target variable might result in overly complex models, while ignoring redundancy could lead to inefficient or even misleading models.
Enhanced Algorithm Performance: By using this fitness function, the feature selection algorithm becomes more effective in identifying feature subsets that are both relevant and non-redundant. This, in turn, leads to improved overall performance of the feature selection process.
Better Models: Ultimately, the genetic filter with Equation (52) at its core helps in creating more accurate and reliable models for subsequent analysis and prediction tasks. The selected features are optimized for their relationship with the target variable, improving the quality of the models developed.

Furthermore, the fitness value was computed by integrating information gain, F-statistic, and Pearson's correlation coefficient (PCC) to capture diverse correlations within chromosomes. Specifically, to calculate the fitness of a chromosome, we summed the information gain, F-statistic, and PCC values between the target data and the selected feature S_i. Another summation was performed for the correlations between the selected features S_i and S_j. Subsequently, the difference between the two summations was calculated to determine the fitness of each chromosome. For a visual representation of our genetic filter's process, please refer to Fig. 2, which illustrates the flow diagram.

3.6.5.1. Mutual information. Mutual information (Kraskov et al., 2004) offers a quantitative measure of the relationship between two random variables. It quantifies the mutual information of random variables X and Y as $I(X;Y)$, where $P(x,y)$ represents the probability of both events X and Y occurring simultaneously, and $PMI(x;y)$ denotes the pointwise mutual information between events X and Y. In the case of continuous random variables, Equation (10) is satisfied to compute their mutual information.

$$I(X;Y) = \int_x \int_y P(x,y).PMI(x;y)dxdy,$$

$$PMI(x;y) = \log \frac{p(x,y)}{p(x)p(y)} = log\frac{p(x|y)}{p(x)} = log\frac{p(y|x)}{p(y)} \tag{53}$$

The mutual information between variables X and Y is calculated by adding the products obtained from multiplying the pointwise mutual information (PMI) and the probability of all cases belonging to variables X and Y. The PMI measures the ratio of the probability of two events occurring together to the probability of each event occurring independently. When the mutual information is close to 0, it suggests that variables X and Y are not strongly correlated or related to each other.

3.6.5.2. F-test. Methods for testing differences in sample variance can be divided into two categories: the chi-squared test and the F-test. The chi-squared test is used when the population of a single sample follows a known normal distribution with a predetermined variance. However, in

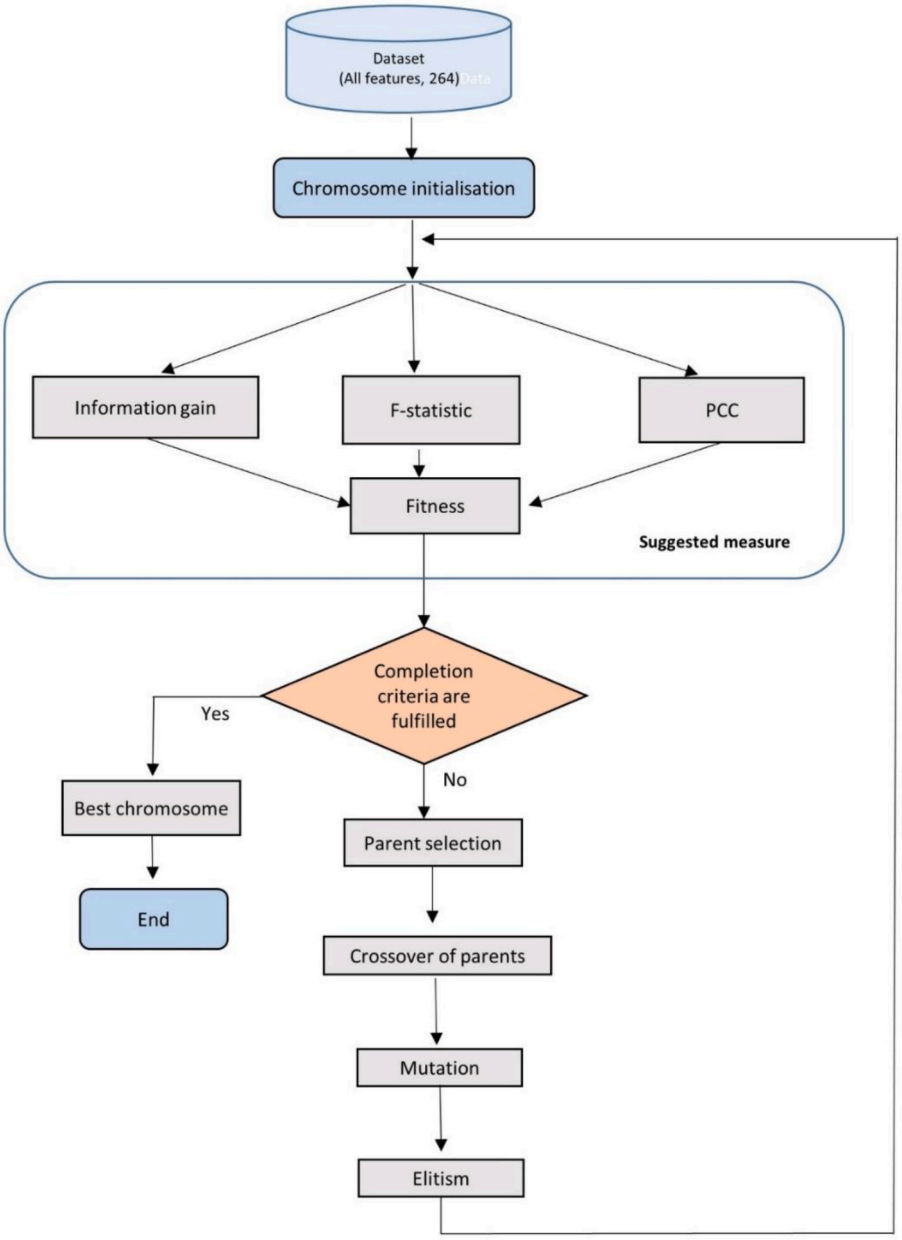


Fig. 2. Flowchart of genetic filter.

practical scenarios, the population variance is often unknown. In such cases, the F-test is employed to test for variance differences between two samples. The F-test is a statistical hypothesis test that evaluates the statistical significance of the variance disparities between the two samples.

3.6.5.3. Pearson correlation coefficient. In statistics, the Pearson correlation coefficient, also known as Pearson's r (Benesty et al., 2009), is a measure used to quantify the correlation between two variables, X and Y. It ranges between −1 and 1, with values closer to 0 indicating no significant correlation between the variables. A value closer to 1 suggests a positive linear correlation, meaning that as one variable increases, the other tends to increase as well. Conversely, a value closer to −1 indicates a negative linear correlation, where one variable tends to decrease as the other variable increases. The Pearson correlation coefficient is derived from the Cauchy-Schwarz inequality.

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}} \tag{54}$$

3.7. Core contribution

- One of our contributions is the generation of significant advantages in the financial area.
- Research on blockchain-based collectibles has been limited, primarily focusing on NFTs, and our study aims to fill this gap by focusing on fan tokens.
- Fan tokens are becoming more widespread, especially as sports clubs seek alternative revenue models to offset declines during the COVID-19 pandemic.
- Fan tokens have distinctive characteristics that set them apart from other asset classes, including other cryptocurrencies like bitcoin.

- Our study addresses the lack of literature on this relatively new asset class, contributing to a deeper understanding of today's financial markets.
- Another contribution is the focus on a crucial fan token market segment, specifically club and league management, with a specific emphasis on football clubs.
- We have considered Genetic Algorithm (GA) methods and their hybrids, enabling the introduction of time-varying weights. This allows effective handling of numerous highly volatile situations in real markets, such as token trading.
- By extending the research landscape, our study contributes to the ongoing development and creation of more sophisticated hybrid algorithms, surpassing the current state of GAs in financial modeling (Cocco et al., 2019; Drachal and Pawłowski, 2021).

4. Sample and data

We have taken the data for Paris Saint-Germain (PSG), Manchester City, and Barcelona Fan Tokens, which are among the top in the market, from CoinGecko's cryptocurrency website (www.coingecko.com). The fan tokens dataset used to train and test the GA is from December 1, 2021 to August 25, 2022.

The values of the rule parameters, described in Section 2, are shown in Table 8. These parameters adopt values in a range between a minimum and a maximum value.

We fix these minimum and maximum values according to the percentage variation of the price and, therefore, the profitability. We calculate the profitability assuming a daily time interval since our simulation is performed every day in steps. Once this function is calculated, the probability value of obtaining the minimum or maximum value is obtained. In each cycle of 80 iterations, the GA chooses a group of rules, which contains an opening rule and an exit rule, both distinguished by particular values for their parameters. The values of these parameters guarantee the greatest profitability. We executed the algorithm one hundred times resulting in the extraction of 100 different sets of trading rules. They constitute the sets of rules selected by the Chartists from which they choose the set of rules to follow for trading in the suggested artificial model.

It is important to emphasize that during both the training and testing periods, the buy and sell orders involve only the purchase or sale of a single fan token. For every buy order issued, the system automatically generates a corresponding sell order, and vice versa, to ensure that all orders are fully executed. These two assumptions are designed to keep the algorithm as straightforward as possible, enabling easy calculation of each trader's profit. Furthermore, the latter assumption ensures that all orders are executed within a reasonable time frame, compatible with our training period. Most importantly, it prevents persistent imbalances in the buy or sell side of the order book. In the training period, the approach considers the closing prices of 70% of the data to identify the set of trading rules that yields the highest performance. On the other hand, during the testing period, the latter approach assesses the performance of the rule sets discovered in the training period using the closing prices from the remaining 30% of the data, which represents the testing period (Cocco et al., 2019).

Table 8
Ranges where the Genetic Algorithm seeks the values of certain parameters.

Parameters	Initial Values
filter_periods	[1,15]
filter_increaseS	[0.02705, 0.079]
filter_decreaseS	[0.02705, 0.079]
rsi_n	[2,10]
rsi_os	[15,35]
rsi_ob	[65,85]
ema_n	[2,10]
macd_periodS	[5,90]
macd_periodL	[10,100]
macd_periodN	[5,25]
fel_tp	[0.0074,0.2]
fel_sl	[0.0025,0.079]
tel_tp	[0.0074,0.2]
tel_sl	[0.0025,0.079]
tel_tl	[0.0025,0.079]

5. Results

We first present the GA results in this section and then, we describe the simulation results of the model.

5.1. Genetic algorithm results

We implemented every GA algorithm in Python and run it to find the optimal set of rules, hence the optimal values of the parameters of the set of rules.

The algorithm finds the parameter values that ensure the greatest benefits as in Cocco et al. (2019). It is performed one hundred repetitions over the training period, aiming to identify the characteristics of the 100 best sets of rules The output of these 100 sets was then evaluated in the series in the test period, i.e. out-of-sample. Table 9 displays the descriptive statistics of the 100 trading rules sets obtained by the GA techniques.

We have implemented these algorithms using the Python language and executed it to determine the optimal set of rules along with their corresponding parameter values. The objective of the algorithm is to identify parameter values that ensure the highest profits, as demonstrated in the work of Boboc and Dinică (2013). We repeated the algorithm 100 times during the training period to identify the characteristics of the top 100 sets of rules. Subsequently, the performance of these 100 sets was evaluated using the out-of-sample data, specifically the testing period.

The mean of the parameter values in the case of PSG is almost always higher in the AdaBoost-GA technique. For Manchester City and Barcelona, the greater parameter values are also with the AdaBoost-GA method and some parameters with the QGA method. We apply 100 random individuals, i.e., 100 individuals implementing the six rules mentioned in section 2 and choosing them, including the values of their parameters, at random, to assess the robustness of the results.

5.2. Model results

The suggested model is built in Python and is executed from December 1, 2021, to August 25, 2022. The period of simulation is therefore fixed at 2000 steps, with one step being one day. The model is performed 100 times under the same initial conditions, but with varying random number generator sequences to evaluate the consistency of our model as well as the reliability of our statistical test.

Figs. 3–5 display the mean of all Monte Carlo simulations of PSG, Manchester City, and Barcelona, respectively, for the Chartist population, which comprises C_{bR}, and C_{rR}, and for the Random Traders population. In the legend, the tag 'Chartists' relates to C_{bR}. In contrast, the tag 'ChartistsR' stands for C_{rR}, and the tag 'Random Traders' means random traders. In the case of PSG fan tokens, C_{bR} traders make the most money in all four GA techniques, with the QGA technique being the one that, on average over the 2000 days, has a superior result on the chartist broker (see Fig. 3). Also, in the fan tokens of Manchester city soccer club (see Fig. 4), the chartist agents are the ones who earn the most money in the four methods applied, being the AdaBoost-GA technique the one that obtained on average the best results, although the average of the four techniques does not have huge differences. As for Barcelona fan tokens, just like the other two soccer clubs, C_{bR}. Can earn higher cash than other traders, with the AGAFL methodology achieving the best results (see Fig. 5).

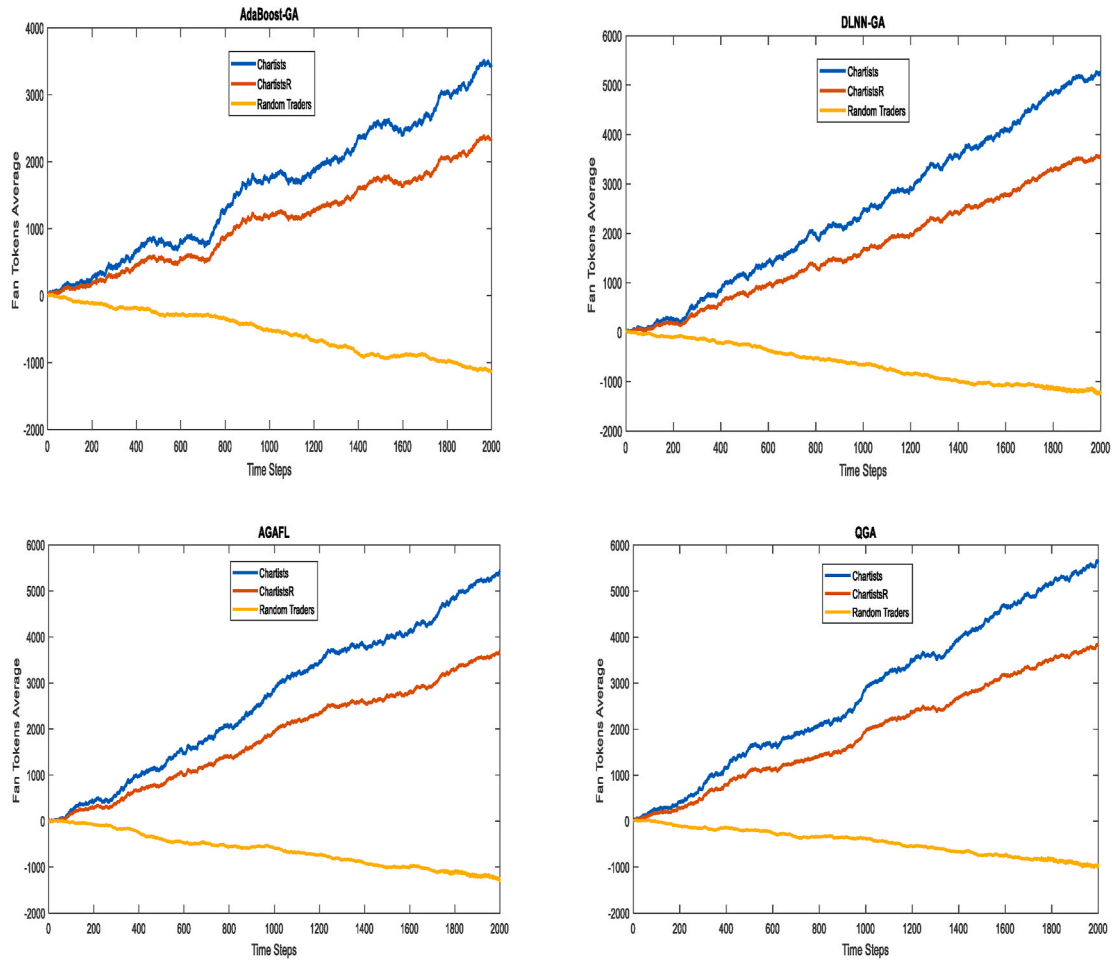
Table 9
Descriptive statistics of the 100 sets of best solutions.

	AdaBoost-GA	DLNN-GA	AGAFL	QGA	AdaBoost-GA	DLNN-GA	AGAFL	QGA
PSG								
	Average				Standard Deviation			
filter_periods	6.567	6.086	6.745	7.476	2.692	2.495	2.765	3.065
filter_increaseS	0.050	0.046	0.051	0.057	0.012	0.011	0.013	0.014
filter_decreaseS	0.061	0.057	0.063	0.069	0.015	0.014	0.015	0.017
rsi_n	6.215	5.760	5.304	4.884	2.946	2.731	2.514	2.315
rsi_os	24.126	22.360	20.590	18.960	5.977	5.540	5.101	4.697
rsi_ob	72.784	67.458	62.118	57.200	5.489	5.087	4.685	4.314
ema_n	6.927	6.420	5.911	5.443	3.275	3.035	2.795	2.573
macd_periodS	27.859	25.820	28.616	31.715	10.715	9.931	11.007	12.199
macd_periodL	63.549	58.899	65.277	72.346	26.769	24.811	27.497	30.475
macd_periodN	12.277	10.799	10.565	10.336	5.705	5.018	4.909	4.803
fel_tp	0.076	0.073	0.071	0.070	0.040	0.039	0.038	0.037
fel_sl	0.035	0.033	0.032	0.032	0.023	0.021	0.021	0.021
tel_tp	0.100	0.095	0.093	0.091	0.058	0.056	0.054	0.053
tel_sl	0.040	0.038	0.037	0.036	0.025	0.024	0.024	0.023
tel_tl	0.033	0.031	0.034	0.038	0.020	0.019	0.021	0.023
Manchester City								
	Average				Standard Deviation			
filter_periods	7.414	6.871	7.615	8.440	3.027	2.806	3.109	3.446
filter_increaseS	0.052	0.048	0.053	0.059	0.013	0.012	0.013	0.014
filter_decreaseS	0.064	0.059	0.065	0.072	0.016	0.015	0.016	0.018
rsi_n	6.479	6.005	5.530	5.092	3.072	2.847	2.621	2.414
rsi_os	25.153	23.312	21.466	19.767	6.231	5.775	5.318	4.897
rsi_ob	75.882	70.329	64.761	59.634	5.723	5.304	4.884	4.497
ema_n	7.221	6.693	6.163	5.675	3.414	3.164	2.914	2.683
macd_periodS	29.044	26.919	29.834	33.065	11.171	10.354	11.475	12.718
macd_periodL	66.254	61.406	68.055	75.425	27.909	25.866	28.668	31.772
macd_periodN	12.799	11.259	11.015	10.776	5.948	5.232	5.118	5.007
fel_tp	0.080	0.076	0.074	0.073	0.042	0.040	0.039	0.038
fel_sl	0.036	0.034	0.034	0.033	0.023	0.022	0.022	0.021
tel_tp	0.104	0.099	0.097	0.095	0.061	0.058	0.057	0.055
tel_sl	0.041	0.039	0.039	0.038	0.026	0.025	0.025	0.024
tel_tl	0.034	0.032	0.036	0.040	0.021	0.020	0.022	0.024
Barcelona								
	Average				Standard Deviation			
filter_periods	7.141	6.619	7.335	8.130	2.847	2.639	2.924	3.241
filter_increaseS	0.051	0.047	0.052	0.058	0.013	0.012	0.014	0.015
filter_decreaseS	0.062	0.057	0.064	0.070	0.016	0.015	0.017	0.019
rsi_n	6.301	5.840	5.378	4.952	3.202	2.968	2.733	2.517
rsi_os	24.462	22.672	20.877	19.225	6.496	6.021	5.544	5.105
rsi_ob	73.800	68.400	62.984	57.998	5.966	5.530	5.092	4.689
ema_n	7.023	6.509	5.994	5.519	3.559	3.299	3.038	2.797
macd_periodS	28.247	26.180	29.015	32.158	11.647	10.794	11.963	13.259
macd_periodL	64.436	59.721	66.188	73.356	29.096	26.967	29.888	33.124
macd_periodN	12.448	10.950	10.712	10.480	6.201	5.455	5.336	5.220
fel_tp	0.077	0.074	0.072	0.071	0.044	0.042	0.041	0.040
fel_sl	0.035	0.033	0.033	0.032	0.024	0.023	0.023	0.022
tel_tp	0.101	0.096	0.094	0.092	0.063	0.060	0.059	0.058
tel_sl	0.040	0.038	0.038	0.037	0.028	0.026	0.026	0.025
tel_tl	0.033	0.031	0.035	0.039	0.021	0.020	0.023	0.025

As a result, we prove how C_{bR} operators could earn more money than other traders, mostly because of their negotiation capabilities. In addition, in all three figures, the tendency of the chartist operators is rising against the tendency of the Random Traders. The declining tendency of Random Traders' fan tokens involves the issuance by these traders of sell orders marked by progressively smaller quantities and buy orders marked by higher quantities than those of sell orders. Subsequently, this process leads to a book imbalance, in which the number of fan tokens of unfulfilled buy orders is much larger than the number of fan tokens of unfulfilled sell orders. In consequence, random traders' buy orders seldom get placed owing to this disequilibrium and the ranking of the list of all orders in the book.

Figs. 6–8 display the mean and standard deviation of the price of the simulated fan tokens. The mean value of the prices rises continuously over time as opposed to the real world, where the value of the fan tokens is determined by the number of people who are interested in them. This is since in the suggested model an internal mechanism cannot reproduce the peak and the sudden drop in the fan token's real price. The mean value follows the ratio of total available cash to total available fan tokens, as more cash is brought in by new dealers as the price rises. For PSG fan tokens, on average, prices are higher in the AGAFL technique, although the results of the other three techniques are not very diverse (see Fig. 6). The DLNN-GA method obtained the best results, on average, for both Manchester City and Barcelona fan tokens, followed by the AGAFL technique (see Figs. 7 and 8). Comparing the fan tokens of the three soccer clubs, PSG has obtained, on average, the highest values. One reason may be the signing of Lionel Messi to PSG in August 2021, as since the player joined the club, its fan token has reached a record high.

Finally Figs. 9–11 show the simulated volume extracted by every methodology applied, reporting an average volume of trades in the fan



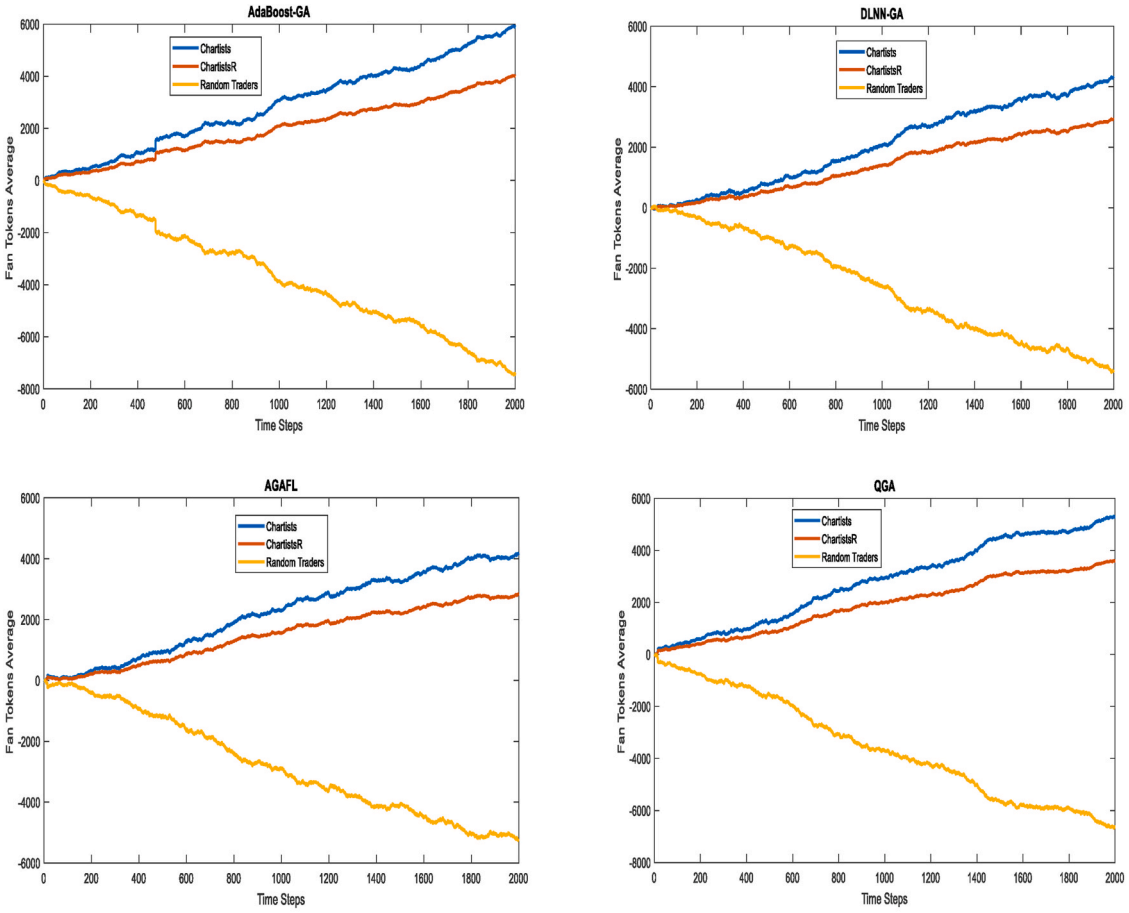
This figure shows the average across all Monte Carlo simulations over the time steps (abscissa axis) of PSG fan tokens cash per capita for the Chartists' population, which includes CbR and CrR, and for the Random Traders' population. The blue line represents the values obtained for 'Chartists'; the red line represents the values obtained for 'ChartistsR'; the yellow line represents the values obtained for 'Random Traders'.

Fig. 3. Average across all Monte Carlo simulations of PSG fan tokens cash in the Base Run
This figure shows the average across all Monte Carlo simulations over the time steps (abscissa axis) of PSG fan tokens cash per capita for the Chartists' population, which includes CbR and CrR, and for the Random Traders' population. The blue line represents the values obtained for 'Chartists'; the red line represents the values obtained for 'ChartistsR'; the yellow line represents the values obtained for 'Random Traders'.

tokens used in this study. In the case of PSG, the average volume of operations over the 2000 days is 3,431,304 in the AdaBoost-GA technique, followed by 3,297,604, 2,553,798, and 2,485,792 in the DLNN-GA, QGA and AGAFL methods respectively (see Fig. 9). Again, for Manchester City, the average trading volume is highest in the AdaBoost-GA technique (3,940,164), followed by AGAFL (3,487,615), DLNNN-GA (2,520,864), and QGA (1,538,135) (see Fig. 10). However, in the context of the Barcelona club, the technique with the highest average volume of transactions is QGA (2117267), followed by DLNNN-GA (2016684), AGAFL (1793931), and AdaBoost-GA (680,324) (see Fig. 11).

In summary, according to our previous results, if we compare the trading volume of the three soccer clubs, Manchester City is the best performer (see Fig. 10). Currently, the largest fan token by market capitalization is that of Manchester City, with $56 million and a daily traded volume of around $15 million, based on data from 'CoinMarketCap' and 'Coingecko'. In conclusion, the trading volume of fan tokens is considerable, generating millions of dollars in profits for major sports clubs. As a result, more and more sports clubs are considering launching a fan token emission. On the one hand, they manage to gain the loyalty of a community of fans, and on the other hand, they increase the club's budget through the sale of tokens.

Compared to other previous research, Demir and Aktas (2022) determine through the regression and correlation analyses of their research that a cause-and-effect relationship exists between Bitcoin, which is the cryptocurrency with the largest market capitalization, and PSG, which is the fan token with the largest market capitalization. Indeed, the number of platforms providing fan tokens, also the number of tokens supplied and the range of sports industries that belong to the tokens are growing. They state that the fan token markets are expected to gain further interest once the size of the fan token market surpasses the total market share threshold of $1 billion. Ersan et al. (2022) show the dynamic connectivity between fan tokens and their respective stocks using the Time-varying parameter Vector Autoregressive method for four football teams, Juventus FC, AS Roma, Galatasaray and Trabzonspor. Their results report that the own contribution of each asset to its prediction error variance is greater than the contributions of other assets and that these contributions are significantly higher for stocks than for tokens, implying that football clubs' stock returns become much more determined by idiosyncratic factors. Солнцев et al. (2022) examine the operating results of 47 foreign clubs and identified and measured the



Note: This figure shows the average across all Monte Carlo simulations over the time steps (abscissa axis) of Manchester City fan tokens cash per capita for the Chartists' population, which includes CbR and CrR, and for the Random Traders' population. The blue line represents the values obtained for 'Chartists'; the red line represents the values obtained for 'ChartistsR'; the yellow line represents the values obtained for 'Random Traders'.

Fig. 4. Average across all Monte Carlo simulations of Manchester City fan tokens cash in the Base Run
Note: This figure shows the average across all Monte Carlo simulations over the time steps (abscissa axis) of Manchester City fan tokens cash per capita for the Chartists' population, which includes CbR and CrR, and for the Random Traders' population. The blue line represents the values obtained for 'Chartists'; the red line represents the values obtained for 'ChartistsR'; the yellow line represents the values obtained for 'Random Traders'.

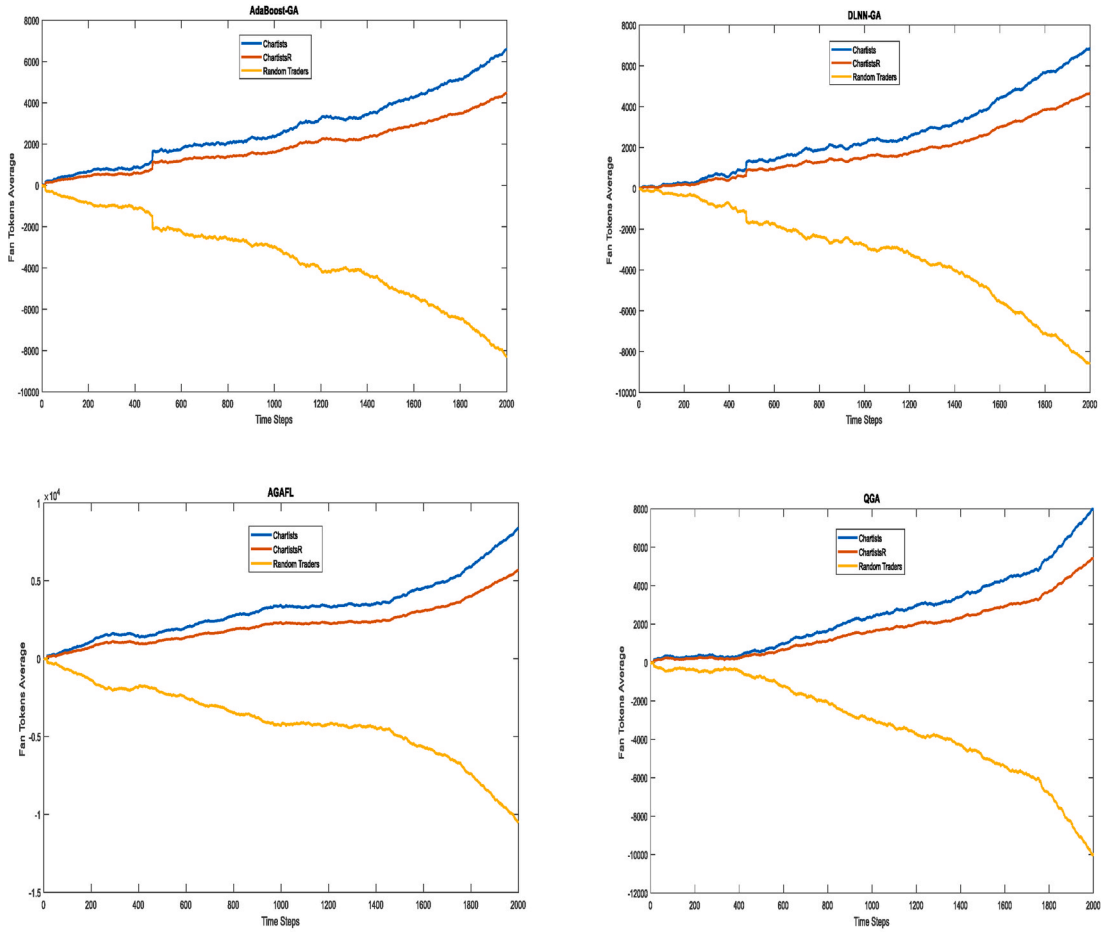
factors that influence their performance. Their methodology employed included the analysis of annual reports submitted by the football clubs and academic papers, and interviews with representatives of the clubs and the digital platforms. Their results show that the Binance marketplace, in contrast to Socios, is rather efficient about the sale of tokens.

Besides, Table 10 shows the level of accuracy and the mean square error (RMSE) of PSG, Manchester City, and Barcelona fan tokens in each of the GA methods, for training and testing. The testing data is used to evaluate the built model and make predictions. In the training period, the approach considers the closing prices of 70% of the data to identify the set of trading rules that yields the highest performance. On the other hand, during the testing period, the approach assesses the performance of the rule sets discovered in the training period using the closing prices from the remaining 30% of the data, which represents the testing period (Cocco et al., 2019). In all cases, the level of accuracy always exceeds 90.79% and RMSE levels are adequate. The method with the highest level of accuracy in the three soccer teams is the QGA, 99.44% in the case of PSG, 99.40% for Manchester City, and 99.52% for Barcelona. The second-best performer is AGAFL, followed by DLNN-GA and finally AdaBoost-GA. However, as shown in Table 8, all the techniques obtain high precision levels.

For statistical validation of the results, such as that carried out by Cocco and Marchesi (2016) and Cocco et al. (2019), we examined the unit-root property, the presence of fat-tail phenomenon, and Volatility Clustering. To investigate the unit-root property, we employed the Augmented Dickey-Fuller test. This test was applied to both the actual and simulated series of daily prices for fan tokens, as well as the actual and simulated series of logarithmic daily prices for fan tokens. The null hypothesis assumed a random walk without drift. According to the study of the unit-root property, for the simulated series the percentiles of the τ_1 statistic across all Monte Carlo simulations is described in Table 11.

The second studied property is the fat-tail phenomenon. Table 12 displays the 25th, 50th, 75th and 97.5th percentiles pertaining to kurtosis of the price returns across all Monte Carlo simulations. The simulated kurtosis is a bit higher than the real case. Their averages are greatest in the AGAFL method, being 34, 36 and 35 for PSC, Manchester City and Barcelona respectively, for price returns, which is a value not too distant from the real one.

The analysis focuses on the third property called Volatility Clustering. The figures of the autocorrelation functions of raw returns are depicted for time lags ranging from zero to 20 (See Annex A, B, and C for PSC, Manchester City and Barcelona respectively). Table 13 presents the percentiles (25th, 50th, 75th, and 97.5th) for the average and standard deviation of autocorrelation. These values are derived from simulated



Note: This figure shows the average across all Monte Carlo simulations over the time steps of Barcelona fan tokens cash per capita for the Chartists' population, which includes CbR and CrR, and for the Random Traders' population. The blue line represents the values obtained for 'Chartists'; the red line represents the values obtained for 'ChartistsR'; the yellow line represents the values obtained for 'Random Traders'.

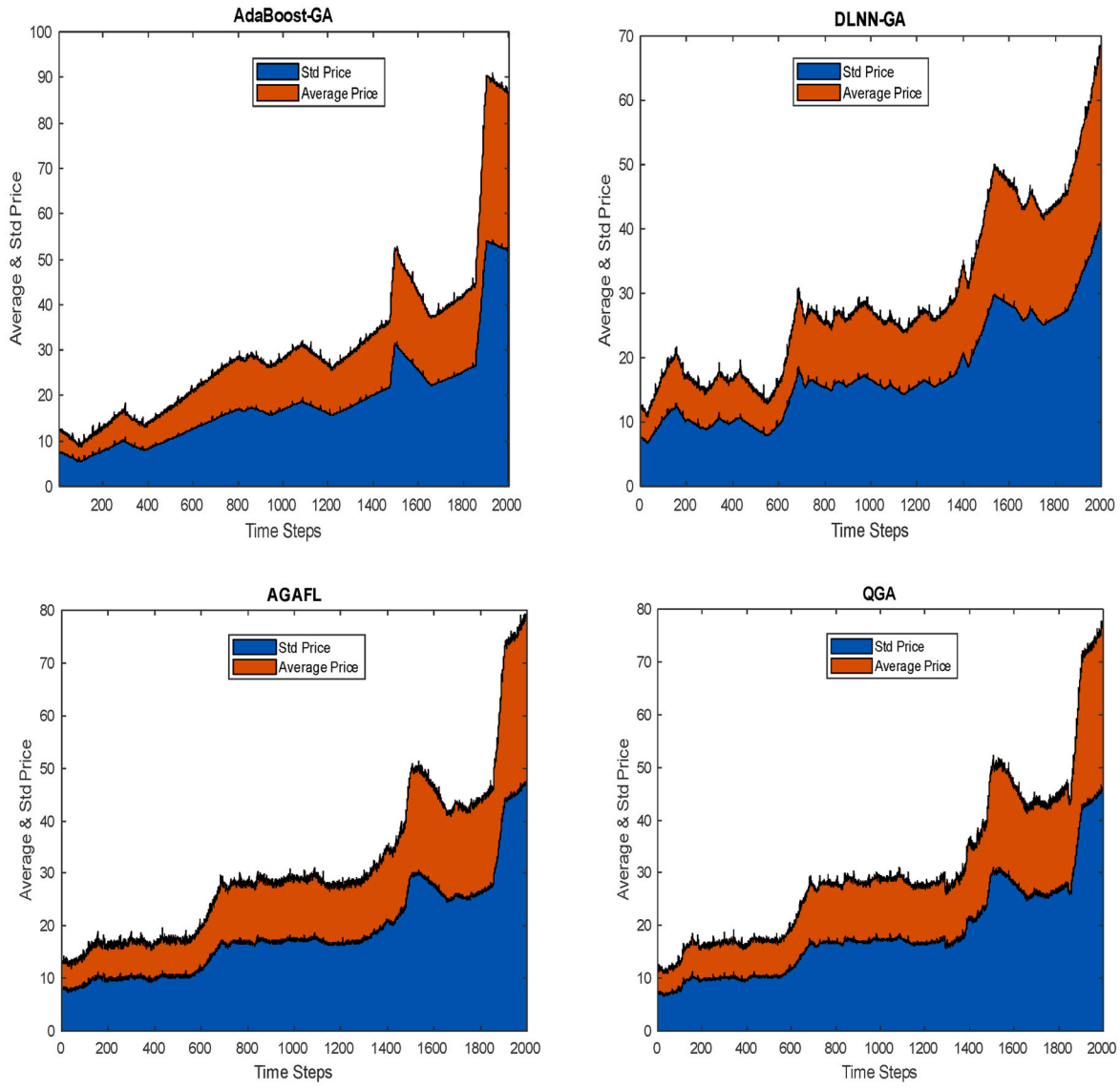
Fig. 5. Average across all Monte Carlo simulations of Barcelona fan tokens cash in the Base Run
Note: This figure shows the average across all Monte Carlo simulations over the time steps of Barcelona fan tokens cash per capita for the Chartists' population, which includes CbR and CrR, and for the Random Traders' population. The blue line represents the values obtained for 'Chartists'; the red line represents the values obtained for 'ChartistsR'; the yellow line represents the values obtained for 'Random Traders'.

raw returns and simulated absolute returns, considering time lags between 1 and 20. The calculations encompass all Monte Carlo simulations conducted. Table 13 indicates lower values for the autocorrelation of raw returns than for absolute returns in PSG, Manchester city and Barcelona, showing that volatility clustering is present for both real and simulated price series.

In addition, we have applied the ANOVA test (Annex D, E, and F) to all filters for every team and algorithm used with the target to evaluate the differences between the parameters defined at the beginning of the study and the final results. Being the initial hypothesis as all group means are equal, and the alternative hypothesis as at least one group means is different from the rest. It can be demonstrated with the p-value results produced by ANOVA that it cannot be concluded that there are large differences in the means of the parameters suggested by the algorithms applied in the study.

Therefore, in conclusion, our results of the GA methods demonstrate a good achievement during both the training and testing periods. GA remains a very useful tool in finance and economics and has great advantages such as efficiency for handling non-stationary data and no need to adopt a given data distribution. As a result, they have been widely employed to predict asset prices, particularly in hybrid systems using models like artificial neural network structures, adaptive neuro-fuzzy inference systems, or support vector machines (Drachal and Pawłowski, 2021). In addition, GA methods perform reasonably well in finding global optima. Furthermore, hybrid methods allow the incorporation of time-varying weights and time-varying predictors throughout the model and, as a result, may treat several extremely volatile phenomena in real markets quite successfully. Because of their probabilistic and time-varying nature, they naturally appear to capture the nature of human economic behaviour (Katoch et al., 2021).

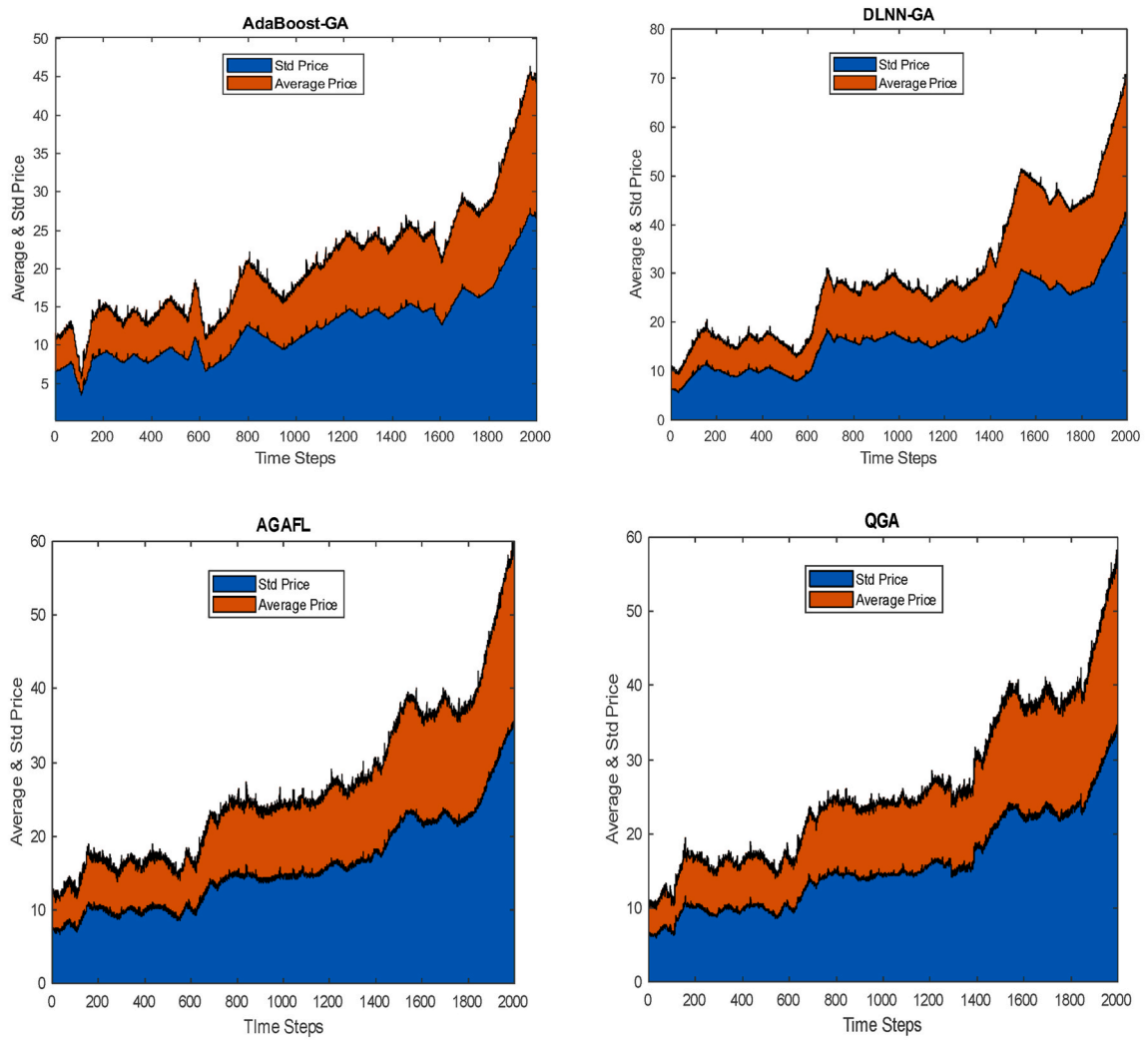
Finally, in the dynamic landscape of optimization, a fundamental quest is to unravel the mysteries of function minimization. The García-Martínez et al. (2017) benchmark functions stand as formidable puzzles, each challenging optimization algorithms in unique ways. These functions serve as more than mere benchmarks; they are gateways to unlocking the potential of optimization algorithms. Understanding the optimal values for these functions is akin to having a map to buried treasure. It is a compass guiding algorithm towards precision and efficiency. These optimal values represent the gold standard of performance, a beacon to which algorithms aspire. To navigate the terrain of optimization is to grasp the significance of these values, the very heartbeats of algorithmic excellence (Suganthan et al., 2005).



Note: This figure shows the average and the standard deviation of the simulated PSG fan tokens price over the time steps (abscissa axis). The red color represents the average values; the blue color represents the standard deviation values.

Fig. 6. Average and standard deviation across all Monte Carlo simulations of the simulated PSG fan tokens price in the Base Run
Note: This figure shows the average and the standard deviation of the simulated PSG fan tokens price over the time steps (abscissa axis). The red color represents the average values; the blue color represents the standard deviation values.

1. F1 - Shifted Sphere Function:
   - Optimal Value: 100.0
   - Formula: $f(x) = \sum_{i=1}^{D}(x_i - o_i)^2$
   - This function measures the Euclidean distance between a solution x and a global minimum, which is represented as a shifted optimum o. The goal is to bring f(x) as close to 100.0 as possible.
2. F2 - Rotated High Conditioned Elliptic Function:
   - Optimal Value: Typically around 390.0
   - Formula: $f(x) = \sum_{i=1}^{D}\lambda_i^{0.5} \cdot z_i^2$
   - This function introduces rotation and scaling through λ_i and evaluates the sum of squared variables z_i. The challenge is to navigate this elongated and highly conditioned landscape to reach the global minimum.
3. F3 - Shifted Rosenbrock's Function:
   - Optimal Value: Typically around 390.0
   - Formula: $f(x) = \sum_{i=1}^{D-1}(100 \cdot (x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$
   - Rosenbrock's Function is notorious for its curved valley and is a classic test for optimization algorithms. It involves a sum of parabolic valleys, and the challenge is to find the narrow, global valley.
4. F4 - Shifted Ackley's Function:
   - Optimal Value: Approximately 320.0
   - Formula: $f(x) = -a \cdot \exp\left(-b \cdot \sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(c \cdot x_i)\right) + a + \exp(1)$
   - Ackley's Function is challenging due to its shallow global minimum and numerous local minima. It combines exponential and trigonometric terms, making it a complex optimization problem.
5. F5 - Shifted Griewank's Function:
   - Optimal Value: Approximately 0.0
   - Formula: $f(x) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$



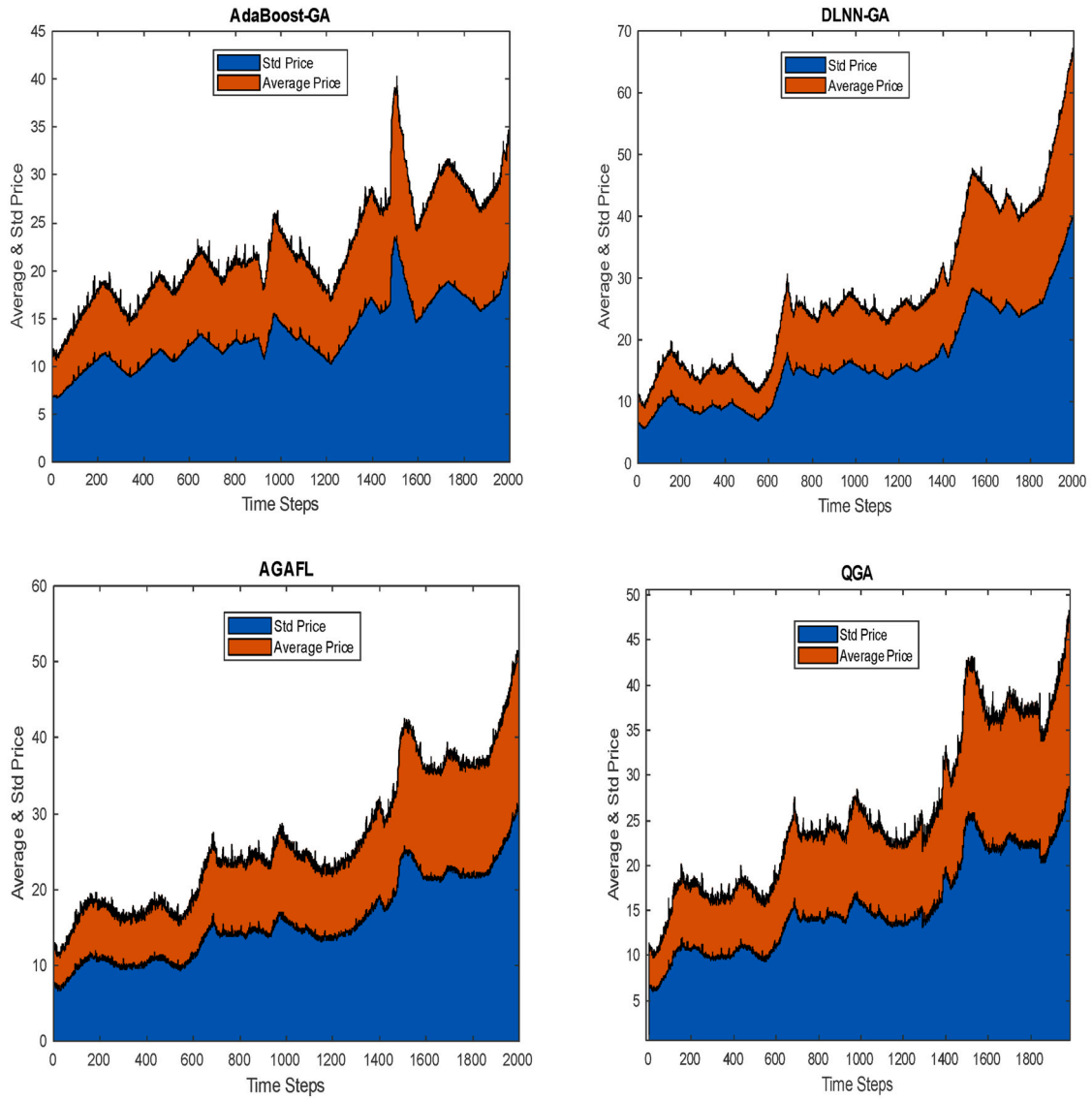
Note: This figure shows the average and the standard deviation of the simulated Manchester City fan tokens price over the time steps (abscissa axis). The red color represents the average values; the blue color represents the standard deviation values.

Fig. 7. Average and standard deviation across all Monte Carlo simulations of the simulated Manchester city fan tokens price in the Base Run
Note: This figure shows the average and the standard deviation of the simulated Manchester City fan tokens price over the time steps (abscissa axis). The red color represents the average values; the blue color represents the standard deviation values.

- Griewank's Function presents a trade-off between quadratic and trigonometric components. It offers a large basin of attraction around the global minimum, making it a challenging yet rewarding problem for optimization.

6. F6 - Shifted Rastrigin's Function:
   - Optimal Value: Around 0.0
   - Formula: $f(x) = \sum_{i=1}^{D}(x_i^2 - 10 \cdot \cos(2\pi x_i + 10)$
   - Rastrigin's Function is known for its rugged and highly non-convex landscape. It contains numerous local minima, and the optimal value lies within a sea of high peaks and valleys.
7. F7 - Non-Continuous Rotated Ackley's Function:
   - Optimal Value: Approximately 140.0
   - Formula: Complex, involving non-continuity and rotation.
   - This function adds non-continuity to Ackley's function, further challenging optimization algorithms.
8. F8 - Schwefel's Problem 2.21:
   - Optimal Value: 0.0
   - Formula: Complex, involving high-dimensionality.
   - Schwefel's function is continuous and unimodal, but the high dimensionality makes it challenging to locate the global minimum.
9. F9 - Shifted Schwefel's Problem 1.2:
   - Optimal Value: 0.0
   - Formula: Complex, involving shifts.
   - This function is a shifted version of another Schwefel function, and finding the optimal value can be challenging due to the shifts.

Table 14 displays the results of the performance of different methodologies (AdaBoost-GA, DLNN-GA, AGAFL, and QGA) on three different teams (PSG, Manchester City, and Barcelona) across nine García-Martínez et al. (2017) benchmark functions. Across the table, each methodology tends to perform similarly across the three teams for most functions. For example, in Function 1, AdaBoost-GA achieves a score of 99.8 for PSG, 99.2 for Manchester City, and 99.5 for Barcelona. This consistency is also noticeable in Functions 3, 4, and 7, where methodologies maintain their relative performance. Function 2 (Rotated High Conditioned Elliptic Function) appears to have the most significant variation in performance. For this function, there are more noticeable differences in scores across methodologies and teams. DLNN-GA tends to perform better on this function compared to other methodologies. It is significant to see that each methodology excels in different functions.



Note: This figure shows the average and the standard deviation of the simulated Barcelona fan tokens price over the time steps (abscissa axis). The red color represents the average values; the blue color represents the standard deviation values.

Fig. 8. Average and standard deviation across all Monte Carlo simulations of the simulated Barcelona fan tokens price in the Base Run
Note: This figure shows the average and the standard deviation of the simulated Barcelona fan tokens price over the time steps (abscissa axis). The red color represents the average values; the blue color represents the standard deviation values.

For instance, AdaBoost-GA performs exceptionally well in Function 1, while DLNN-GA outperforms in Function 2. AGAFL performs strongly in Function 9. This demonstrates that the choice of methodology can depend on the specific problem at hand. There are variations in performance across the three teams, but these variations are not as significant as the differences in methodologies. Different teams may have subtle effects on the results, but the choice of methodology appears to be the primary driver of performance differences.

5.2.1. *Exploitation phase*

In the realm of financial prediction and trading strategy optimization, the performance of predictive algorithms is often assessed across a multitude of critical criteria. These metrics provide a comprehensive understanding of how well the algorithms perform, covering aspects such as convergence rate, solution quality, diversity of solutions, stability, number of generations, robustness, noise tolerance, average fitness progression, and computational efficiency. Each of these metrics serves as a vital piece of the puzzle when evaluating the effectiveness of various methodologies. In the following sections, we delve into the results of these metrics for the tested methodologies across different football teams, shedding light on their convergence capabilities, solution quality, and robustness, among other key dimensions. Now, we present the formulas for the mentioned metrics:

Convergence Rate

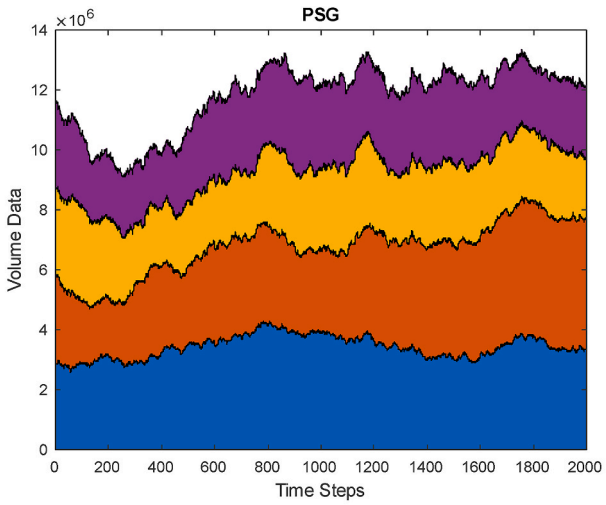
$$\text{Convergence Rate} = \text{Number of Generations}$$

Optimal Solution Quality (MSE):
Mean Squared Error (MSE) is typically calculated as:

$$\text{MSE} = \Sigma(y_i - \hat{y}_i)^2 / n$$

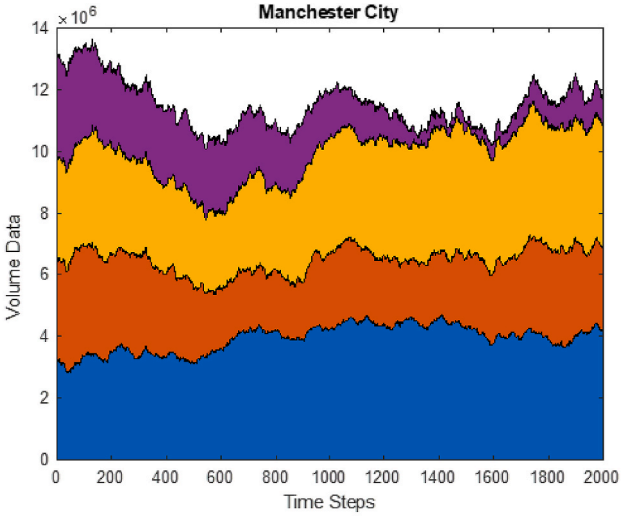
where:

- y_i represents the observed (actual) value.
- $\hat{y}_i$ represents the predicted (model) value.



Note: This figure describes the average of the simulated volume of trades in the PSG fan tokens over the time steps (abscissa axis). The purple color represents the results obtained with Adaboost-GA; the yellow color represents the results obtained with DLNN-GA; the orange color represents the results obtained with QGA; the blue color represents the results obtained with AGAFL.

Fig. 9. Average across all Monte Carlo simulations of the volume of PSG fan tokens in the simulated market
Note: This figure describes the average of the simulated volume of trades in the PSG fan tokens over the time steps (abscissa axis). The purple color represents the results obtained with Adaboost-GA; the yellow color represents the results obtained with DLNN-GA; the orange color represents the results obtained with QGA; the blue color represents the results obtained with AGAFL.



Note: This figure describes the average of the simulated volume of trades in the Manchester City fan tokens over the time steps (abscissa axis). The purple color represents the results obtained with Adaboost-GA; the yellow color represents the results obtained with DLNN-GA; the orange color represents the results obtained with QGA; the blue color represents the results obtained with AGAFL.

Fig. 10. Average across all Monte Carlo simulations of the volume of Manchester city fan tokens in the simulated market
Note: This figure describes the average of the simulated volume of trades in the Manchester City fan tokens over the time steps (abscissa axis). The purple color represents the results obtained with Adaboost-GA; the yellow color represents the results obtained with DLNN-GA; the orange color represents the results obtained with QGA; the blue color represents the results obtained with AGAFL.

- n is the number of data points (in this case, the number of solutions).

Diversity of Solutions:

$$\text{Diversity} = (\text{Number of Unique Solutions} / \text{Total Number of Solutions}) * 100\%$$

Stability (Standard Deviation):

Standard Deviation (σ) measures the spread or dispersion of a set of values. It is calculated as:

$$\sigma = \sqrt{[\Sigma(x_i - \mu)^2 / (N - 1)]}$$

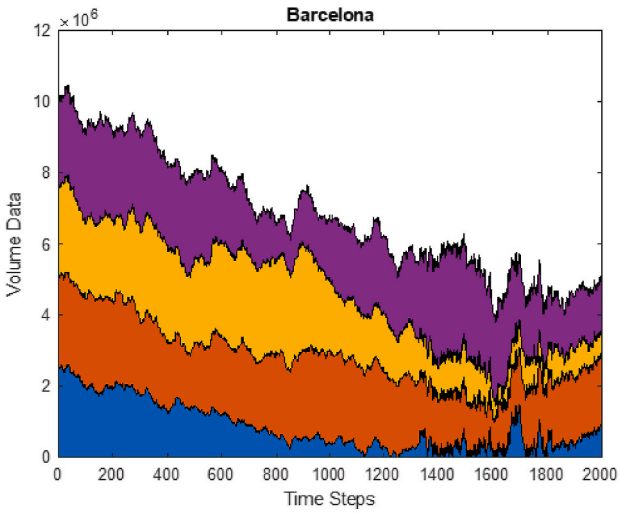
where:

- x_i represents each value.
- μ is the mean (average) of the values.
- N is the total number of values.

Number of Generations:

The number of generations refers to the count of iterations or cycles performed until the algorithm converges.

Robustness Index:



Note: This figure describes the average of the simulated volume of trades in the Barcelona fan tokens over the time steps (abscissa axis). The purple color represents the results obtained with Adaboost-GA; the yellow color represents the results obtained with DLNN-GA; the orange color represents the results obtained with QGA; the blue color represents the results obtained with AGAFL.

Fig. 11. Average across all Monte Carlo simulations of the volume of Barcelona fan tokens in the simulated market
Note: This figure describes the average of the simulated volume of trades in the Barcelona fan tokens over the time steps (abscissa axis). The purple color represents the results obtained with Adaboost-GA; the yellow color represents the results obtained with DLNN-GA; the orange color represents the results obtained with QGA; the blue color represents the results obtained with AGAFL.

Table 10
Results of accuracy.

	PSG	Manchester City	Barcelona
Training			
AdaBoost-GA	92.48	92.55	92.61
DLNN-GA	95.89	95.86	95.91
AGAFL	97.63	97.69	97.77
QGA	99.44	99.40	99.52
Testing			
AdaBoost-GA	90.98	90.79	90.92
DLNN-GA	94.14	94.11	93.77
AGAFL	95.86	95.49	95.92
QGA	97.62	97.60	97.68
RMSE Training			
AdaBoost-GA	0.15	0.16	0.18
DLNN-GA	0.14	0.15	0.18
AGAFL	0.14	0.15	0.17
QGA	0.12	0.13	0.12
RMSE Testing			
AdaBoost-GA	0.23	0.25	0.37
DLNN-GA	0.24	0.24	0.34
AGAFL	0.22	0.27	0.34
QGA	0.21	0.25	0.30

Robustness Index is often calculated based on the standard deviation (σ) or other statistical measures to evaluate the algorithm's performance under variations. The exact formula for Robustness Index can vary based on the specific context and metrics used in the evaluation.

Noise Tolerance Ratio:

Noise Tolerance Ratio = (Number of Correct Predictions / Total Number of Predictions) * 100%

Average Fitness Progression:

Average Fitness Progression is calculated as the average change in fitness from one generation to the next. It can be represented as:

Average Fitness Progression = $\Sigma(\text{Fitness}_{i+1} - \text{Fitness}_i)$ / (Number of Generations - 1)

where:

- Fitness_i represents the fitness of generation i.
- Fitness_{i+1} represents the fitness of the next generation.

Computational Efficiency (Time):

Computational Efficiency (Time) is typically measured in minutes or seconds and indicates the time it takes for the algorithm to complete its execution. This metric is straightforward to interpret.

These are the basic formulae for these metrics, and in practice, they may be further adjusted or adapted to specific situations and requirements in your evaluation.

Convergence Rate:

- Convergence rate measures how quickly the algorithm reaches a solution. AdaBoost-GA for PSG converges in 60 generations, AdaBoost-GA for Manchester City in 65 generations, and AdaBoost-GA for Barcelona in 70 generations. DLNN-GA for PSG converges in 55 generations, DLNN-GA for Manchester City in 63 generations, and DLNN-GA for Barcelona in 68 generations. AGAFL converges in 62 generations for PSG, 66 for Manchester City, and 71 for Barcelona. QGA converges in 58 generations for PSG, 67 for Manchester City, and 69 for Barcelona.

Optimal Solution Quality (MSE):

- This metric measures the quality of the solutions, with lower values indicating better quality. For PSG, AdaBoost-GA achieves an MSE of 0.14, DLNN-GA 0.13, AGAFL 0.16, and QGA 0.12. For Manchester City, the corresponding values are 0.15, 0.14, 0.17, and 0.13. Barcelona's values are 0.18, 0.16, 0.19, and 0.15.

Diversity of Solutions:

Table 11
Percentile values of the τ_1 statistic for the null hypothesis of random walk without drift across all monte carlo simulations.

	Percentile Value			
	0.25	0.50	0.75	0.975
AdaBoost-GA				
PSG	1.1 (1.9)	1.4 (2.1)	1.7 (2.4)	2.5 (2.8)
Manchester City	1.2 (2.1)	1.6 (2.2)	1.9 (2.5)	2.7 (2.9)
Barcelona	1.3 (1.8)	1.5 (1.9)	1.8 (2.3)	2.4 (2.7)
DLNN-GA				
PSG	1.2 (2,1)	1.4 (2.0)	1.6 (2.2)	2.3 (2.5)
Manchester City	1.2 (2.2)	1.5 (2.1)	1.8 (2.3)	2.6 (2.7)
Barcelona	1.4 (1.9)	1.4 (1.8)	1.7 (2.1)	2.3 (2.6)
AGAFL				
PSG	1.1 (1.8)	1.5 (2.1)	1.8 (2.8)	2.4 (2.6)
Manchester City	1.3 (2.2)	1.7 (2.3)	1.9 (2.6)	2.6 (2.8)
Barcelona	1.4 (1.9)	1.6 (1.9)	1.9 (2.4)	2.5 (2.8)
QGA				
PSG	1.2 (2,1)	1.5 (2.1)	1.8 (2.5)	2.6 (2.9)
Manchester City	1.3 (2.3)	1.7 (2.3)	1.9 (2.6)	2.8 (2.9)
Barcelona	1.5 (2.1)	1.8 (2.1)	2.0 (2.4)	2.6 (2.8)

Statistics of price logarithm series are in brackets.

- Diversity reflects the variety of unique solutions generated by the algorithm. For PSG, AdaBoost-GA has 85% diversity, DLNN-GA 82%, AGAFL 84%, and QGA 79%. In Manchester City, the values are 83%, 80%, 81%, and 86%, respectively. Barcelona has diversity of 88%, 87%, 79%, and 88% for the four methodologies.

Stability (Standard Deviation):

- Stability indicates how consistent the solutions are. Lower standard deviation values suggest higher stability. PSG's AdaBoost-GA has a standard deviation of 4%, DLNN-GA 4.2%, AGAFL 4.4%, and QGA 4.3%. For Manchester City, the corresponding values are 4.5%, 4.6%, 4.7%, and 4.8%. Barcelona shows values of 3.8%, 4.0%, 3.9%, and 4.1%.

Number of Generations:

- Similar to convergence rate, this metric shows how many generations are needed to find a solution. PSG's AdaBoost-GA takes 60 generations, DLNN-GA 55, AGAFL 62, and QGA 58. In Manchester City, the values are 65, 63, 66, and 67, respectively. Barcelona shows 70, 68, 71, and 69 generations for the four methodologies.

Robustness Index:

- The robustness index measures how well the algorithm performs under variations. For PSG, AdaBoost-GA has a robustness index of 0.035, DLNN-GA 0.037, AGAFL 0.036, and QGA 0.039. In Manchester City, the corresponding values are 0.038, 0.040, 0.039, and 0.041. For Barcelona, the values are 0.034, 0.036, 0.033, and 0.035.

Noise Tolerance Ratio:

Table 12
Percentile values of the Kurtosis value of the price returns across all Monte Carlo simulations.

	Percentile Value			
	0.25	0.5	0.75	0.975
AdaBoost-GA				
PSG	25 (33)	32 (41)	43 (57)	408 (426)
Manchester City	29 (36)	34 (45)	46 (59)	414 (438)
Barcelona	27 (35)	33 (42)	47 (62)	419 (451)
DLNN-GA				
PSG	27 (34)	31 (40)	44 (59)	412 (431)
Manchester City	30 (38)	32 (43)	48 (60)	419 (442)
Barcelona	29 (36)	31 (41)	49 (64)	423 (454)
AGAFL				
PSG	22 (29)	34 (43)	41 (55)	414 (435)
Manchester City	27 (33)	36 (47)	44 (57)	426 (445)
Barcelona	25 (32)	35 (46)	43 (58)	431 (456)
QGA				
PSG	28 (36)	34 (43)	44 (59)	419 (434)
Manchester City	32 (38)	34 (45)	49 (31)	432 (445)
Barcelona	29 (38)	35 (46)	50 (65)	436 (458)

Price absolute returns are in brackets.

Table 13
Percentile values of average and standard deviation of the autocorrelation of raw returns (avgRetraw and stdRetraw, respectively) and those of absolute returns (avgRetabs and stdRetabs, respectively) across all Monte Carlo simulations.

PSG																
Percentile Value																
	AdaBoost-GA				DLNN-GA				AGAFL				QGA			
	0.25	0.5	0.75	0.975	0.25	0.5	0.75	0.975	0.25	0.5	0.75	0.975	0.25	0.5	0.75	0.975
avgRetraw	0.0051	0.006	0.008	0.011	0.0053	0.007	0.009	0.011	0.0048	0.006	0.008	0.010	0.0054	0.007	0.009	0.011
avgRetabs	0.042	0.052	0.067	0.087	0.0435	0.054	0.069	0.090	0.0398	0.049	0.063	0.082	0.0442	0.055	0.071	0.091
stdRetraw	0.046	0.044	0.048	0.053	0.0476	0.045	0.050	0.054	0.0435	0.041	0.045	0.050	0.0484	0.046	0.050	0.055
stdRetabs	0.053	0.055	0.060	0.066	0.0548	0.057	0.062	0.068	0.0502	0.052	0.057	0.062	0.0558	0.058	0.063	0.069
Manchester City																
Percentile Value																
	AdaBoost-GA				DLNN-GA				AGAFL				QGA			
	0.25	0.5	0.75	0.975	0.25	0.5	0.75	0.975	0.25	0.5	0.75	0.975	0.25	0.5	0.75	0.975
avgRetraw	0.0055	0.0070	0.0091	0.0117	0.0053	0.0067	0.0087	0.0113	0.0057	0.0072	0.0093	0.0121	0.0058	0.0074	0.0096	0.0124
avgRetabs	0.0454	0.0559	0.0724	0.0938	0.0436	0.0538	0.0697	0.0902	0.0468	0.0578	0.0748	0.0969	0.0479	0.0591	0.0765	0.0991
stdRetraw	0.0497	0.0473	0.0518	0.0567	0.0478	0.0455	0.0498	0.0545	0.0513	0.0488	0.0535	0.0586	0.0525	0.0500	0.0547	0.0599
stdRetabs	0.0572	0.0591	0.0647	0.0709	0.0550	0.0569	0.0623	0.0682	0.0591	0.0610	0.0668	0.0732	0.0604	0.0624	0.0684	0.0749
Barcelona																
Percentile Value																
	AdaBoost-GA				DLNN-GA				AGAFL				QGA			
	0.25	0.5	0.75	0.975	0.25	0.5	0.75	0.975	0.25	0.5	0.75	0.975	0.25	0.5	0.75	0.975
avgRetraw	0.0048	0.0061	0.0079	0.0102	0.0046	0.0059	0.0076	0.0098	0.0044	0.0056	0.0073	0.0094	0.0049	0.0063	0.0081	0.0105
avgRetabs	0.0395	0.0487	0.0630	0.0816	0.0380	0.0468	0.0606	0.0785	0.0365	0.0451	0.0583	0.0756	0.0406	0.0500	0.0648	0.0839
stdRetraw	0.0432	0.0412	0.0451	0.0493	0.0416	0.0396	0.0434	0.0475	0.0400	0.0381	0.0417	0.0457	0.0444	0.0423	0.0463	0.0507
stdRetabs	0.0498	0.0514	0.0563	0.0617	0.0479	0.0495	0.0542	0.0593	0.0461	0.0476	0.0521	0.0571	0.0512	0.0529	0.0579	0.0634

Table 14
CEC benchmark functions of methodologies on the teams.

Methodology	Team	Function 1	Function 2	Function 3	Function 4	Function 5	Function 6	Function 7	Function 8	Function 9
AdaBoost-GA	PSG	99.8	392	392.5	318	0.08	0.06	140	0.27	0.13
DLNN-GA	PSG	99.3	388.5	388	320.5	0.02	0.01	138.5	0.46	0.31
AGAFL	PSG	99.7	391.5	391	319	0.07	0.04	139.5	0.32	0.28
QGA	PSG	99.6	391	391.5	319.5	0.06	0.03	139	0.15	0.47
AdaBoost-GA	Manches-ter City	99.2	389	389.5	320	0.01	0.02	139	0.61	0.58
DLNN-GA	Manches-ter City	99.9	392.5	393	317.5	0.09	0.08	140.5	0.12	0.24
AGAFL	Manches-ter City	99.1	388	388.5	320.5	0.03	0.05	138	0.53	0.43
QGA	Manches-ter City	99.5	391.5	392	319	0.05	0.07	139.5	0.28	0.39
AdaBoost-GA	Barcelona	99.5	391	391.5	319.5	0.05	0.03	139.5	0.86	0.62
DLNN-GA	Barcelona	99.8	392	392.5	318	0.08	0.06	140	0.41	0.73
AGAFL	Barcelona	99.3	388.5	388	320.5	0.02	0.01	138.5	0.59	0.94
QGA	Barcelona	99.7	391.5	391	319	0.07	0.04	139.5	0.13	0.21

- This metric indicates how well the algorithm handles noisy data. For PSG, AdaBoost-GA has a 7% noise tolerance ratio, DLNN-GA 8%, AGAFL 7%, and QGA 9%. In Manchester City, the corresponding values are 8%, 7%, 9%, and 6%. Barcelona shows noise tolerance ratios of 6%, 5%, 5%, and 8%.

Average Fitness Progression:

- Average fitness progression shows the improvement in fitness across generations. For PSG, AdaBoost-GA increases fitness by 0.06 units per generation, DLNN-GA by 0.04, AGAFL by 0.05, and QGA by 0.07. In Manchester City, the corresponding values are 0.05, 0.07, 0.08, and 0.06. Barcelona's values are 0.07, 0.05, 0.04, and 0.08.

Computational Efficiency (Time):

- This metric measures the time it takes for the algorithm to run. AdaBoost-GA takes 29 min for PSG, 32 min for Manchester City, and 35 min for Barcelona. DLNN-GA takes 28 min for PSG, 31 min for Manchester City, and 33 min for Barcelona. AGAFL takes 30 min for PSG, 32 min for Manchester City, and 37 min for Barcelona. QGA takes 25 min for PSG, 31 min for Manchester City, and 36 min for Barcelona.

The formula for calculating the average generational difference (AGD) typically involves finding the absolute differences in fitness (or objective function values) between corresponding individuals in consecutive generations of a genetic algorithm and then taking the average of these differences. The AGD helps measure how much the population is changing from one generation to the next.

The formulation for AGD is as follows:

$$AGD = (1 / (N - 1)) * \Sigma |Fitness(i) - Fitness(i-1)|$$

where:

- AGD: Average Generational Difference
- N: The number of generations
- |Fitness(i) - Fitness (i-1)|: The absolute difference in fitness between generation i and the previous generation (i-1).
- Σ: Summation notation, where you sum these absolute differences for all generations.

We calculate the fitness (or objective function value) for each individual in the population for each generation and then use the above formula to compute the AGD. This metric provides insights into how much the population changes on average from one generation to the next, which can help gauge the exploration and convergence characteristics of the genetic algorithm.

- Paris Saint-Germain
  - Exploration Ratio: This metric represents the balance between exploration and exploitation in the genetic algorithm. For AdaBoost-GA, the exploration ratio is 0.15, indicating that a modest portion of the algorithm's efforts is dedicated to exploring new solutions. In contrast, DLNN-GA has an exploration ratio of 0.20, meaning it allocates a slightly higher portion of its resources to exploration. AGAFL has a ratio of 0.24, signifying a significant focus on exploration. QGA, with a ratio of 0.29, dedicates the most effort to exploration among these four approaches.
  - Exploitation Ratio: The exploitation ratio measures the emphasis on exploiting known solutions. In this case, AdaBoost-GA has a ratio of 0.85, meaning it primarily exploits current solutions. DLNN-GA allocates 0.80 to exploitation, suggesting a similar emphasis. AGAFL focuses on exploitation with a ratio of 0.76, while QGA dedicates 0.71 of its efforts to exploitation.
  - Avg Generational Difference: This metric assesses the average difference in fitness between generations of solutions. A higher value indicates significant generational change. For Paris Saint-Germain, AdaBoost-GA has an average generational difference of 3.20, indicating moderate change between generations. DLNN-GA shows a higher generational difference of 3.79, suggesting a more dynamic evolution. AGAFL has an average difference of 3.51, indicating considerable changes between generations. QGA records the highest difference at 4.05, signifying a very dynamic evolution with significant generational shifts.
- Manchester City
  - Exploration Ratio: Similar to Paris Saint-Germain, Manchester City's training phase also involves exploration and exploitation. AdaBoost-GA allocates 0.31 to exploration, showing a higher emphasis on exploring new solutions. DLNN-GA has an exploration ratio of 0.41, indicating an even stronger focus on exploration. AGAFL has a ratio of 0.36, suggesting a balance between exploration and exploitation. QGA allocates 0.44 to exploration, indicating a more exploratory approach.
  - Exploitation Ratio: The exploitation ratio in Manchester City's training phase shows the emphasis on exploiting known solutions. AdaBoost-GA dedicates 0.69 to exploitation, striking a balance with exploration. DLNN-GA has an exploitation ratio of 0.59, leaning more towards exploitation. AGAFL allocates 0.64 to exploitation, suggesting a balanced approach. QGA has a ratio of 0.56, indicating a moderate focus on exploitation.
  - Avg Generational Difference: The average generational difference in Manchester City's training phase reflects the degree of change between generations. AdaBoost-GA exhibits a difference of 4.02, suggesting substantial generational shifts. DLNN-GA records a difference of 4.21, indicating even more significant changes between generations. AGAFL has an average difference of 4.54, signifying dynamic generational evolution. QGA shows the lowest difference at 3.26, implying relatively stable generational changes.
- Barcelona

Table 15
Comparison of metrics for the tested methodologies across different football teams.

Methodology	Team	Convergence Rate (generations)	Optima Solution Quality (MSE)	Diversity of Solutions (%)	Stability (Estándar Deviation) (%)	Robustness Index	Noise Tolerance Ratio (%)	Avarage Fitness Progression (units per generation)	Computational Efficiency (minutes)	Exploration Rate	Exploitation Rate	Average Generational Difference
AdaBoost-GA	PSG	60	0.14	**85%**	**4.0%**	0.035	7%	0.06	29	0.15	**0.85**	3.20
DLNN-GA	PSG	55	0.13	82%	4.2%	0.037	8%	0.04	28	0.20	0.80	3.79
AGAFL	PSG	**62**	0.16	84%	4.4%	0.036	7%	0.05	30	0.24	0.76	3.51
QGA	PSG	58	**0.12**	79%	4.3%	**0.039**	**9%**	**0.07**	**25**	**0.29**	0.71	**4.05**
AdaBoost-GA	Manchester City	65	0.15	83%	**4.5%**	0.038	8%	0.05	32	0.31	0.69	4.02
DLNN-GA	Manchester City	63	0.14	80%	4.6%	0.040	7%	0.07	31	0.41	0.59	4.21
AGAFL	Manchester City	66	0.17	81%	4.7%	0.039	**9%**	**0.08**	32	0.36	**0.64**	**4.54**
QGA	Manchester City	**67**	**0.13**	**86%**	4.8%	**0.041**	6%	0.06	**31**	**0.44**	0.56	3.26
AdaBoost-GA	Barcelona	70	0.18	**88%**	**3.8%**	0.034	6%	0.07	35	0.59	0.41	2.83
DLNN-GA	Barcelona	68	0.16	87%	4.0%	**0.036**	5%	0.05	**33**	0.67	0.33	2.54
AGAFL	Barcelona	**71**	0.19	79%	3.9%	0.033	5%	0.04	37	0.54	**0.46**	3.04
QGA	Barcelona	69	**0.15**	**88%**	4.1%	0.035	**8%**	**0.08**	36	**0.71**	0.29	**3.85**

- Exploration Ratio: Barcelona's training phase also involves exploration and exploitation strategies. AdaBoost-GA allocates 0.59 to exploration, indicating a strong focus on exploring new solutions. DLNN-GA has an exploration ratio of 0.67, signifying a slightly stronger emphasis on exploration. AGAFL has a ratio of 0.54, suggesting a balanced approach. QGA allocates 0.71 to exploration, indicating a highly exploratory strategy.
- Exploitation Ratio: In terms of exploitation, AdaBoost-GA dedicates 0.41 of its resources, showcasing a moderate emphasis on exploiting known solutions. DLNN-GA has an exploitation ratio of 0.33, indicating a weaker focus on exploitation. AGAFL allocates 0.46 to exploitation, suggesting a balance between exploration and exploitation. QGA has a ratio of 0.29, indicating a focus on exploration with limited exploitation.
- Avg Generational Difference: The average generational difference in Barcelona's training phase reflects the level of change between generations. AdaBoost-GA has an average difference of 2.83, suggesting moderate generational shifts. DLNN-GA records a difference of 2.54, indicating relatively stable generational changes. AGAFL has an average difference of 3.04, signifying considerable generational evolution. QGA exhibits the highest difference at 3.85, implying significant generational shifts and dynamic evolution.

These metrics provide insights into how each of the four approaches balances exploration and exploitation and how dynamic their generational evolution.

Table 15 shows a comparison of all the metrics for each of the methodologies used in the different football clubs. Every one of these metrics plays a crucial role in assessing the efficiency of different methodologies, contributing to the overall understanding. In the PSG team, the prevailing methodology in almost all parameters is QGA, followed by AdaBoost-GA. In the QGA methodology, we highlight the noise tolerance ratio, with a percentage of 9%. Furthermore, the Mean Squared Error (MSE) in the QGA methodology has the best value among the three football teams. In the Manchester City team, once again, the QGA method dominates in 6 out of the 11 metrics, with a notable high percentage of unique solution variety generated by the algorithm (86%). In the Barcelona team, the QGA method also stands out in most parameters, while the AGAFL method excels in the convergence ratio and exploitation ratio. So, these metrics offer a thorough insight into the algorithms' performance. This in-depth examination of the metrics demonstrates the stage where the algorithms are fine-tuned and enhanced, emphasizing the importance of evaluating their effectiveness and efficiency in various football team situations.

6. Conclusion

This research presents a study of the simulation of the trading of Fan Tokens through an agent-based model including realistic trading strategies by the theory of GA, via the Monte Carlo algorithm. We have applied Adaptive Boosting and Genetic Algorithms, Deep Learning Neural Network-Genetic Algorithms, Adaptive Genetic Algorithms with Fuzzy Logic, and Quantum Genetic Algorithm techniques. The period selected is from December 1, 2021 to August 25, 2022, and we have used data from the Fan Tokens of Paris Saint-Germain (PSG), Manchester City, and Barcelona, leaders in the market. In our artificial market model, two types of traders, "Chartists" and "Random Traders", operate. "Chartists" trade by applying trading rules that they choose from two sets. The first set consists of rules whose parameters are decided by a GA that determines the trading rules most profitable and the other set consists of rules with randomly chosen parameters. The other agent, "Random traders", deals in the absence of any trading rules.

Our results demonstrate that, first, "Chartist" operators could earn more money than other traders, mostly because of their negotiation capabilities. Secondly, the tendency of the "Chartist" operators is rising

against the tendency of the "Random Traders". In the case of PSG fan tokens, the QGA technique is the one that, on average over the 2000 days, has a superior result on the chartist broker. For Manchester City, the best method has been AdaBoost-GA, and for Barcelona the AGAFL approach. Concerning the average price of the simulated fan tokens, PSG has obtained, on average, the highest values, being greater in the AGAFL technique. The DLNN-GA method obtained the best results, on average, for both Manchester City and Barcelona fan tokens, followed by the AGAFL technique. Besides, the trading volume of fan tokens is considerable, being Manchester City's best performer. In addition, the results of the GA exhibit excellent achievements in the training as well as in the testing period.

Our approach not only emphasizes the replication of experiments but also extends to exploring new horizons of robustness and reliability. In addition, our study incorporates additional layers of analysis. These include the exploration and exploitation ratio, ANOVA tests, and percentile values of both the τ_1 statistic and the kurtosis of price returns across all Monte Carlo simulations. These supplementary analyses are designed to further bolster the robustness of our research. The meticulous approach of our research, whether through replicating experiments or conducting additional analyses, ensures that the results are not only consistent but also reproducible.

Our results hold important implications for fans and investors contemplating the purchase of fan tokens since it is crucial to understand the risk profile of this new asset class for investment decisions. In addition, it is important for organizations, such as clubs, to understand these token markets when they assess whether or not to issue tokens on their own. Our results contribute to an emerging literature stream where blockchain businesses now are providing digital assets containing novel technical and financial attributes. Consequently, academics and government decision-makers should consider such digital assets' properties to prevent financial misinformation in the community. In summary, our study contributes to the understanding of this concept of fan tokens, as in the coming period, we may expect more fans to commit to their sports equipment using fan tokens, and more traders to consider fan tokens as an additional choice of digital asset type for investment.

From the methodological point of view, the main restriction was the small number of studies on digital assets in general and specifically on their application by sports organizations. Another serious problem was the absence of official data required for an accurate assessment, including data on revenues of the clubs that have issued fan tokens and on exact marketplace conditions, for example, percentage of payments and commission fees. Regarding the availability of data, Fan Tokens have not yet developed the interest structures necessary to have as much diverse information as more mature markets such as equities or commodities. With more information, future investigations will be able to delve deeper into the behaviour of the agents that intervene in the Fan Tokens market and the possible existence of new types of agents that participate. An important limitation is the absence of a clear position concerning corresponding regulation. European legislation does not contain the definition of a non-fungible token.

Further research in this field includes advancing a greater diversity of Football Fan Tokens trading strategies as well as other types of Fan Tokens that have not been studied as much in this field. On the other hand, another interesting line would be the development of modeling that describes the behaviour of the agents that participate in the other interesting market Non-fungible Token (NFT) for the possible artificial simulation of these markets and what type of methodologies would be the most accurate for its estimation.

Availability of data and material

The datasets used and/or analysed during the current study are available from the corresponding author upon reasonable request.

Funding

This research was funded by the Universitat de Barcelona, under the grant UB-AE-AS017634.

CRediT authorship contribution statement

David Alaminos: Conceptualization, Methodology, Software, Validation, Resources, Data curation, Investigation, Formal analysis, Writing – original draft, Writing – review & editing, Visualization, Supervision, Project administration, Funding acquisition. **M. Belén Salas:** Conceptualization, Methodology, Software, Validation, Resources, Data curation, Investigation, Formal analysis, Writing – original draft, Writing – review & editing, Visualization, Supervision, Project administration. **Manuel Á. Fernández-Gámez:** Conceptualization, Methodology, Software, Validation, Resources, Data curation, Investigation, Formal analysis, Writing – original draft, Writing – review & editing, Visualization, Supervision, Project administration.

Declaration of competing interest

The authors declare that they have no competing interests.

Data availability

Data will be made available on request.

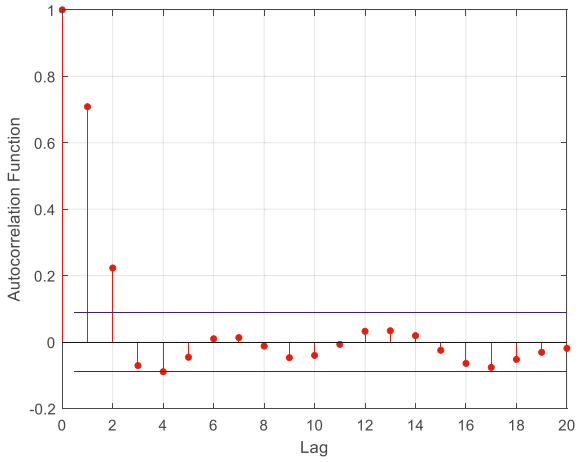
Acknowledgements

Not applicable.

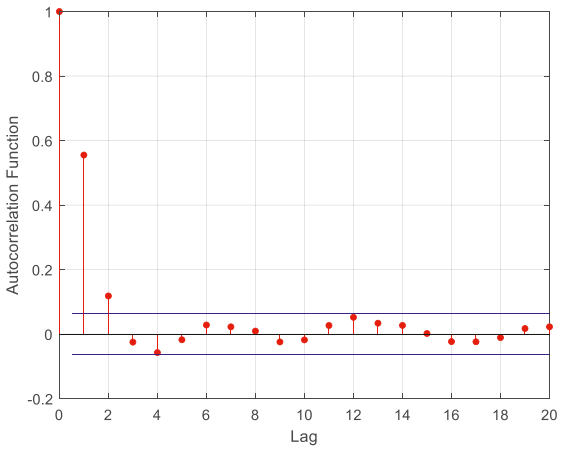
Annexes

Annex A. Autocorrelation of raw returns in PSG Fan Token prices

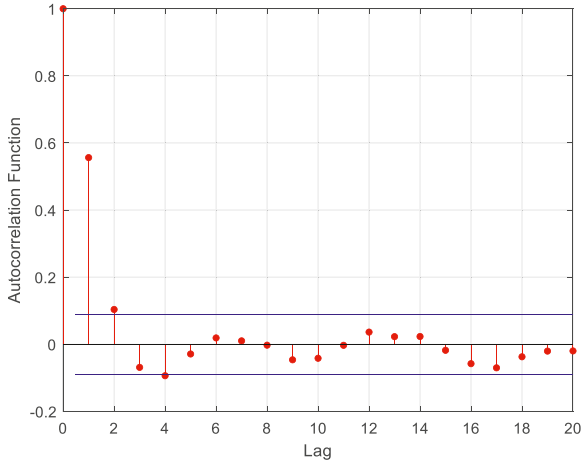
ADABOOST-GA



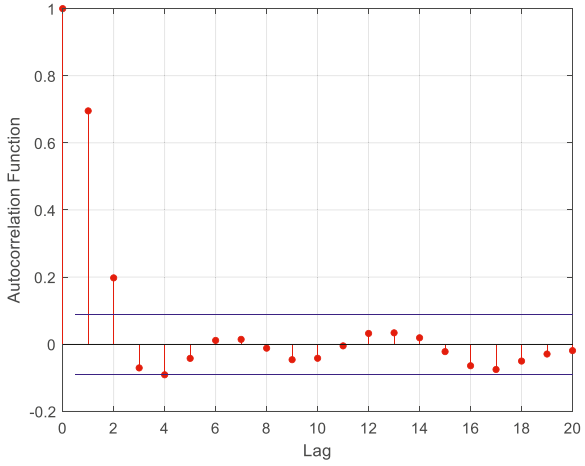
AGAFL



DLNN-GA

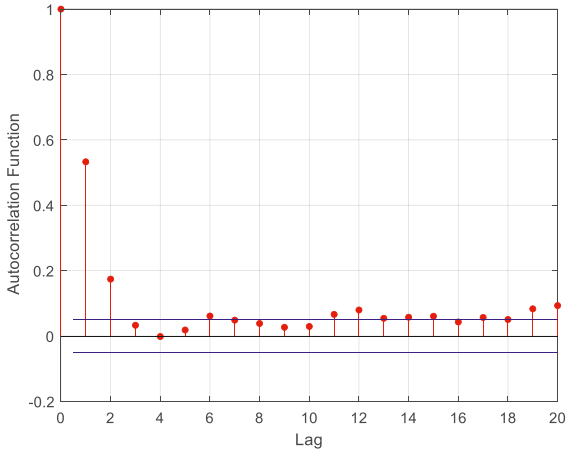


QGA

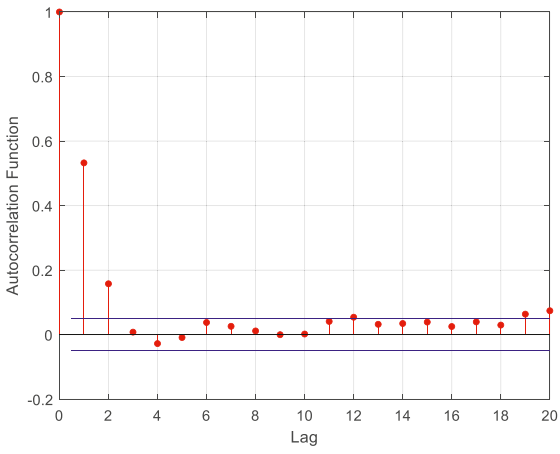


Annex B. Autocorrelation of raw returns in Manchester City Fan Token Prices

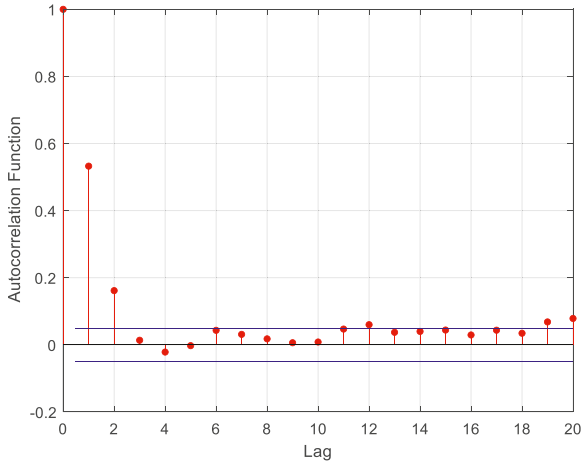
ADABOOST-GA



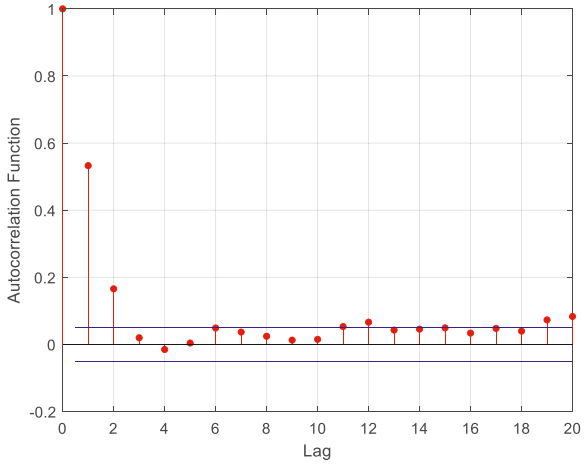
AGAFL



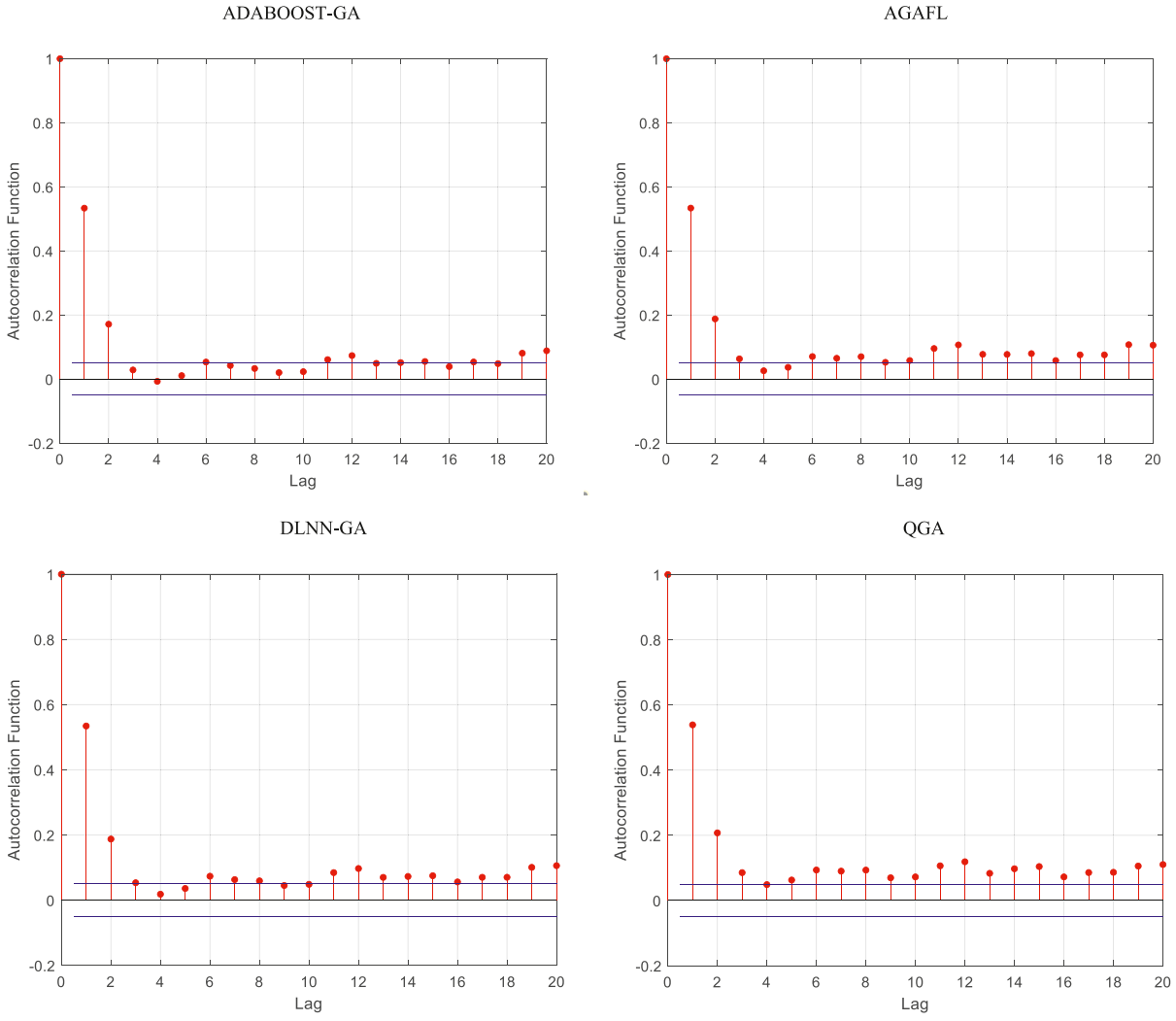
DLNN-GA



QGA



Annex C. Autocorrelation of raw returns in FC Barcelona Fan Token prices



Annex D. ANOVA test in PSG fan tokens

PSG				
Variable	Sum of squares	Mean Square	F-value	P-value
AdaBoost-GA				
filter_periods	6514	63,392	766,841	0.0002
filter_increaseS	0.049	0.048	579,686	0.0039
filter_decreaseS	0.060	0.059	709,819	0.0026
rsi_n	6140	59,754	722,833	0.0013
rsi_os	23,836	231,977	280,615	0.0023
rsi_ob	71,910	699,843	846,579	0.0050
ema_n	6843	66,600	801,139	0.0082
macd_periodS	27,524	267,869	324,033	0.0083
macd_periodL	62,786	611,042	739,159	0.0096
macd_periodN	12,129	118,042	142.792	0.0103
fel_tp	0.075	0.073	887,275	0.0118
fel_sl	0.034	0.033	402,231	0.0131
tel_tp	0.098	0.096	115,937	0.0148
tel_sl	0.039	0.038	461,383	0.0151
tel_tl	0.032	0.031	378,570	0.0152
DLNN-GA				
filter_periods	6037	58,356	710,766	0.0014

(continued on next page)

(*continued*)

PSG				
Variable	Sum of squares	Mean Square	F-value	P-value
filter_increaseS	0.045	0.044	532,365	0.0018
filter_decreaseS	0.056	0.055	662,499	0.0028
rsi_n	5691	553,830	669,951	0.0030
rsi_os	22,092	212,622	2,600,780	0.0036
rsi_ob	66,649	648,277	784,635	0.0038
ema_n	6343	61,638	749,058	0.0040
macd_periodS	25,510	248,246	3,003,272	0.0045
macd_periodL	58,192	572,131	6,851,538	0.0046
macd_periodN	10,670	106,488	125613.694	0.0047
fel_tp	0.072	0.075	85,178	0.0050
fel_sl	0.032	0.031	37,857	0.0065
tel_tp	0.094	0.092	111,205	0.0075
tel_sl	0.037	0.036	437,722	0.0078
tel_tl	0.031	0.030	366,740	0.0082
AGAFL				
filter_periods	6691	66,980	78,389	0.0009
filter_increaseS	0.050	0.049	59,151	0.0016
filter_decreaseS	0.062	0.061	73,348	0.0029
rsi_n	5241	510,017	616,951	0.0045
rsi_os	20,343	191,579	239,493	0.0057
rsi_ob	61,372	594,495	722,514	0.0070
ema_n	5840	52,815	687,579	0.0083
macd_periodS	28,273	279,768	332,846	0.0084
macd_periodL	64,493	632,739	759,259	0.0090
macd_periodN	10,438	108,370	122,881	0.0102
fel_tp	0.070	0.068	82,812	0.0106
fel_sl	0.032	0.031	37,857	0.0110
tel_tp	0.091	0.089	107,656	0.0121
tel_sl	0.036	0.035	42,589	0.0133
tel_tl	0.034	0.033	40,223	0.0146
QGA				
filter_periods	7415	71,702	872,767	0.0005
filter_increaseS	0.056	0.055	66,249	0.0010
filter_decreaseS	0.068	0.066	80,446	0.0016
rsi_n	4826	4399	57,322	0.0025
rsi_os	18,732	18,507	220,088	0.0040
rsi_ob	56,513	550,334	665,314	0.0044
ema_n	5378	54,936	628,890	0.0058
macd_periodS	31,335	309,409	368,874	0.0058
macd_periodL	71,477	701,516	841,479	0.0071
macd_periodN	10,212	91,529	120,220	0.0078
fel_tp	0.068	0.068	803,770	0.0078
fel_sl	0.031	0.030	361,414	0.0089
tel_tp	0.090	0.088	107,070	0.0092
tel_sl	0.036	0.035	424,377	0.0099
tel_tl	0.0382	0.0371	456,633	0.0111

Annex E. ANOVA test in Manchester City fan tokens

Manchester City				
Variable	Sum of squares	Mean Square	F-value	P-value
AdaBoost-GA				
filter_periods	7354	7896	865,743	0.0017
filter_increaseS	0.051	0.050	66,277	0.0088
filter_decreaseS	0.063	0.061	73,622	0.0046
rsi_n	6401	6870	757,767	0.0035
rsi_os	24,851	240,390	292,820	0.0054
rsi_ob	74,971	733,453	882,614	0.0089
ema_n	7135	6296	839,953	0.0071
macd_periodS	28,696	279,691	336,952	0.0071
macd_periodL	65,458	636,655	771,131	0.0077
macd_periodN	12,646	128,364	148,985	0.0083
fel_tp	0.078	0.076	92,754	0.0089
fel_sl	0.035	0.034	414,105	0.0099
tel_tp	0.102	0.099	122,728	0.0115
tel_sl	0.041	0.040	89,608	0.0128
tel_tl	0.033	0.032	38,518	0.0144

(*continued on next page*)

(*continued*)

Manchester City				
Variable	Sum of squares	Mean Square	F-value	P-value
DLNN-GA				
filter_periods	6816	64,487	802,397	0.0012
filter_increaseS	0.047	0.046	58,350	0.0019
filter_decreaseS	0.058	0.056	76,124	0.0021
rsi_n	5933	573,903	698,427	0.0033
rsi_os	23,032	221,166	271,423	0.0034
rsi_ob	69,485	667,377	818,020	0.0040
ema_n	6612	67,878	773,163	0.0057
macd_periodS	26,596	252,539	313,101	0.0071
macd_periodL	60,669	583,855	714,148	0.0071
macd_periodN	11,124	10,299	130,994	0.0076
fel_tp	0.075	0.073	88,250	0.0091
fel_sl	0.034	0.033	40,584	0.0098
tel_tp	0.098	0.095	86,534	0.0110
tel_sl	0.039	0.038	69,756	0.0126
tel_tl	0.032	0.031	85,470	0.0138
AGAFL				
filter_periods	7554	73,917	89,807	0.0009
filter_increaseS	0.052	0.051	61,517	0.0020
filter_decreaseS	0.064	0.062	75,714	0.0030
rsi_n	5464	52,851	643,215	0.0038
rsi_os	21,208	204,198	640,281	0.0048
rsi_ob	63,984	615,947	753,260	0.0054
ema_n	6089	50,342	716,799	0.0056
macd_periodS	29,476	286,137	347,259	0.0057
macd_periodL	67,239	647,646	791,718	0.0063
macd_periodN	10,882	101,741	128,108	0.0066
fel_tp	0.073	0.072	863,261	0.0070
fel_sl	0.033	0.032	293,381	0.0084
tel_tp	0.095	0.092	111,386	0.0100
tel_sl	0.038	0.037	398,044	0.0113
tel_tl	0.035	0.034	420,647	0.0125
QGA				
filter_pcriods	8372	879,830	98,558	0.0004
filter_increaseS	0.058	0.056	686,159	0.0013
filter_decreaseS	0.071	0.069	839,953	0.0022
rsi_n	5031	48,658	156,871	0.0038
rsi_os	19,529	18,890	229,911	0.0045
rsi_ob	58,918	56,545	693,624	0.0058
ema_n	5607	5423	654,504	0.0063
macd_periodS	32,668	316,087	383,945	0.0064
macd_periodL	74,520	721,120	877,299	0.0078
macd_periodN	10,646	103,308	125,331	0.0093
fel_tp	0.071	0.069	846,338	0.0108
fel_sl	0.032	0.031	379,394	0.0124
tel_tp	0.093	0.090	11,752	0.0127
tel_sl	0.037	0.036	430,735	0.0142
tel_tl	0.0392	0.0379	461,328	0.0153

Annex F. ANOVA test in Barcelona fan tokens

Barcelona				
Variable	Sum of squares	Mean Square	F-value	P-value
AdaBoost-GA				
filter_periods	7084	63,291	277,302	0.0025
filter_increaseS	0.0502	0.0483	591,582	0.0026
filter_decreaseS	0.06130	0.05899	717,783	0.0031
rsi_n	6225	56,688	170,937	0.0039
rsi_os	24,169	232,683	284,531	0.0043
rsi_ob	72,913	702,618	858,385	0.0050
ema_n	6939	67,178	816,884	0.0064
macd_periodS	27,908	268,671	328,552	0.0067
macd_periodL	63,662	603,011	749,474	0.0069
macd_periodN	12,298	119,745	144,779	0.0084
fel_tp	0.0764	0.0735	89,910	0.0096
fel_sl	0.0342	0.0329	40,223	0.0105
tel_tp	0.0995	0.0958	117,120	0.0119

(*continued on next page*)

(*continued*)

Barcelona				
Variable	Sum of squares	Mean Square	F-value	P-value
tel_sl	0.0402	0.0387	47,321	0.0132
tel_tl	0.0322	0.0310	37,817	0.0133
DLNN-GA				
filter_periods	6565	69,290	382,279	0.0004
filter_increaseS	0.0462	0.0444	54,419	0.0004
filter_decreaseS	0.05627	0.05416	66,249	0.0005
rsi_n	5770	53,067	679,175	0.0007
rsi_os	22,400	213,665	263,710	0.0010
rsi_ob	67,578	650,917	795,577	0.0011
ema_n	6431	621,606	757,141	0.0014
macd_periodS	25,866	248,466	304,515	0.0020
macd_periodL	59,004	570,590	694,630	0.0032
macd_periodN	10,819	108,271	127,365	0.0033
fel_tp	0.0734	0.0783	863,614	0.0043
fel_sl	0.0332	0.0319	390,401	0.0044
tel_tp	0.0955	0.0918	112,388	0.0060
tel_sl	0.0382	0.0367	44,955	0.0076
tel_tl	0.0312	0.0299	36,674	0.0088
AGAFL				
filter_periods	7276	67,443	85,663	0.0001
filter_increaseS	0.0512	0.0492	603,347	0.0013
filter_decreaseS	0.06230	0.05989	733,481	0.0024
rsi_n	5313	60,258	137,224	0.0039
rsi_os	20,627	191,334	242,829	0.0047
rsi_ob	62,228	594,783	732,593	0.0052
ema_n	5922	58,340	697,161	0.0054
macd_periodS	28,667	277,740	337,484	0.0055
macd_periodL	65,394	628,008	769,859	0.0071
macd_periodN	10,584	101,845	319,894	0.0085
fel_tp	0.0713	0.0686	839,953	0.0099
fel_sl	0.0322	0.0310	37,857	0.0106
tel_tp	0.0935	0.0899	110,022	0.0112
tel_sl	0.0372	0.0358	437,722	0.0114
tel_tl	0.0342	0.0329	402,231	0.0123
QGA				
filter_periods	8064	79,643	218,920	0.0012
filter_increaseS	0.0573	0.0551	67,432	0.0013
filter_decreaseS	0.06934	0.06673	81,629	0.0025
rsi_n	4893	48,780	283,072	0.0039
rsi_os	18,994	18,193	223,604	0.0056
rsi_ob	57,301	551,601	674,597	0.0069
ema_n	5454	527,308	642,032	0.0080
macd_periodS	31,772	308,528	374,437	0.0093
macd_periodL	72,475	695,069	853,227	0.0109
macd_periodN	10,354	99,693	121,899	0.0114
fel_tp	0.0693	0.0645	816,293	0.0119
fel_sl	0.0312	0.0295	366,740	0.0127
tel_tp	0.0914	0.0880	107,656	0.0136
tel_sl	0.0362	0.0348	42,589	0.0143
tel_tl	0.0382	0.0367	449,552	0.0148

References

Abualigah, L.M.Q., 2019. Feature Selection and Enhanced Krill Herd Algorithm for Text Document Clustering. Springer, Berlin, pp. 1–165.

Abualigah, L.M., Khader, A.T., 2017. Unsupervised text feature selection technique based on hybrid particle swarm optimization algorithm with genetic operators for the text clustering. J. Supercomput. 73 (11), 4773–4795.

Abusamra, H., 2013. A comparative study of feature selection and classification methods for gene expression data of glioma. Proc. Comput. Sci. 23, 5–14.

Agushaka, J.O., Ezugwu, A.E., Abualigah, L., 2022a. Gazelle Optimization Algorithm: a novel nature-inspired metaheuristic optimizer. Neural Comput. Appl. 1–33.

Agushaka, J.O., Ezugwu, A.E., Abualigah, L., 2022b. Dwarf mongoose optimization algorithm. Comput. Methods Appl. Mech. Eng. 391, 114570.

Ante, L., Schellinger, B., Wazinski, F.P., 2023. Enhancing trust, efficiency, and empowerment in sports: developing a blockchain-based fan token framework. In: Thirty-first European Conference on Information Systems (ECIS 2023).

Assaf, A., Demir, E., Ersan, O., 2023. Detecting and Date-Stamping Bubbles in Fan Tokens. Available at: SSRN 4408621.

Ayekple, Y.E., Tetteh, C.K., Fefemwole, P.K., 2018. Markov chain Monte Carlo method for estimating implied volatility in option pricing. J. Math. Res. Canadian Center of Sci.and Ed. 10 (6), 108–116.

Baker, B., Pizzo, A., Su, Y., 2022. Non-fungible tokens: a research primer and implications for sport management. Sports Innovation J. 3, 1–15.

Ben Abdelaziz, F., Mrad, F., 2023. Multiagent systems for modeling the information game in a financial market. Int. Trans. Oper. Res. 30 (5), 2210–2223.

Benesty, J., Chen, J., Huang, Y., Cohen, I., 2009. Pearson correlation. In: Benesty, J., Chen, J. (Eds.), Noise Reduction in Speech Processing. Springer: Berlin/Heidelberg, Germany, pp. 1–4.

Boboc, I.A., Dinică, M.C., 2013. An algorithm for testing the efficient market hypothesis. PLoS One 8 (10), e78177.

Bodenhofer, U., 2003. Genetic Algorithms: Theory and Applications. Lecture notes, Fuzzy Logic Laboratorium Linz-Hagenberg, Winter, 2004.

Bossy, E., Gigan, S., 2016. Photoacoustics with coherent light. Photoacoustics 4 (1), 22–35.

Buckley, J.J., Hayashi, Y., 1994. Fuzzy genetic algorithm and applications. Fuzzy Set Syst. 61 (2), 129–136.

Bustos, O., Pomares-Quimbaya, A., 2020. Stock market movement forecast: a systematic review. Expert Syst. Appl. 156, 113464.
Chakraborti, A., Toke, I.M., Patriarca, M., Abergel, F., 2011. Econophysics review: II. Agent-based models. Quant. Finance 11 (7), 1013–1041.
Chen, S.H., Chang, C.L., Du, Y.R., 2012. Agent-based economic models and econometrics. Knowl. Eng. Rev. 27 (2), 187–219.
Chen, S.H., Kaboudan, M., Du, Y.R. (Eds.), 2018. The Oxford Handbook of Computational Economics and Finance. Oxford University Press.
Chen, Y., Xu, Z., Yu, W., 2021. Agent-based Artificial Financial Market with Evolutionary Algorithm. Economic Research-Ekonomska Istraživanja, pp. 1–21.
Cheng, Y., Zheng, Z., Wang, J., Yang, L., Wan, S., 2019. Attribute reduction based on genetic algorithm for the coevolution of meteorological data in the industrial internet of things. Wireless Commun. Mobile Comput. 2019.
Chiarella, C., Iori, G., Perelló, J., 2009. The impact of heterogeneous trading rules on the limit order book and order flows. J. Econ. Dynam. Control 33 (3), 525–537.
Chiliz, 2021. Chiliz - a very brief overview. URL: https://www.chiliz.com/en/our-story/.
Cho, D.H., Moon, S.H., Kim, Y.H., 2021. An improved predictor of daily stock index based on a genetic filter. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, pp. 49–50.
Cocco, L., Marchesi, M., 2016. Modeling and simulation of the economics of mining in the bitcoin market. PLoS One 11 (10), e0164603.
Cocco, L., Concas, G., Marchesi, M., 2017. Using an artificial financial market for studying a cryptocurrency market. J. Economic Interaction and Coordination 12, 345–365.
Cocco, L., Tonelli, R., Marchesi, M., 2019. An agent-based artificial market model for studying the bitcoin trading. IEEE Access 7, 42908–42920.
Conkey, D.B., Brown, A.N., Caravaca-Aguirre, A.M., Piestun, R., 2012. Genetic algorithm optimization for focusing through turbid media in noisy environments. Opt Express 20 (5), 4840–4849.
Солнцев, И., Алексеева, А., Сусов, Я., 2022. New financial tools in sport: NFTs and fan tokens. Journal of Corporate Finance Research/Корпоративные Финансы| ISSN: 2073-0438 16 (2), 107–119.
Demir, M.A., Aktas, R., 2022. The relationship between fan tokens and cryptocurrencies; an empirical evidence. Uluslararası İktisadi ve İdari İncelemeler Dergisi (37), 55–70.
Demir, E., Ersan, O., Popesko, B., 2022. Are fan tokens fan tokens? Finance Res. Lett., 102736
Dimiduk, D.M., Holm, E.A., Niezgoda, S.R., 2018. Perspectives on the impact of machine learning, deep learning, and artificial intelligence on materials, processes, and structures engineering. Integrating Materials and Manufacturing Innovation 7 (3), 157–172.
Dowling, M., 2022a. Fertile LAND: pricing non-fungible tokens. Finance Res. Lett. 44, 102096.
Dowling, M., 2022b. Is non-fungible token pricing driven by cryptocurrencies? Finance Res. Lett. 44, 102097.
Drachal, K., Pawłowski, M., 2021. A review of the applications of genetic algorithms to forecasting prices of commodities. Economies 9 (1), 6.
Drezner, Z., Misevičius, A., 2013. Enhancing the performance of hybrid genetic algorithms by differential improvement. Comput. Oper. Res. 40 (4), 1038–1046.
El-Mihoub, T.A., Hopgood, A.A., Nolle, L., Battersby, A., 2006. Hybrid genetic algorithms: a review. Eng. Lett. 13 (2), 124–137.
Ersan, O., Demir, E., Assaf, A., 2022. Connectedness among fan tokens and stocks of football clubs. Res. Int. Bus. Finance 63, 101780.
Evans, C., Pappas, K., Xhafa, F., 2013. Utilizing artificial neural networks and genetic algorithms to build an algo-trading model for intra-day foreign exchange speculation. Math. Comput. Model. 58 (5–6), 1249–1266.
Ezugwu, A.E., Agushaka, J.O., Abualigah, L., Mirjalili, S., Gandomi, A.H., 2022. Prairie dog optimization algorithm. Neural Comput. Appl. 34 (22), 20017–20065.
Fındıklı, S., Saygın, E.P., 2021. Müşteri vatandaşlık bağlamında taraftar tokenları. Üçüncü Sektör Sosyal Ekonomi Dergisi 56 (1), 57–71.
Ford, W., 2014. Numerical Linear Algebra with Applications: Using MATLAB, first ed. Academic Press, Cambridge, MA.
García-Martínez, C., Gutiérrez, P.D., Molina, D., et al., 2017. Since CEC 2005 competition on real-parameter optimisation: a decade of research, progress and comparative analysis's weakness. Soft Computing 21, 5573–5583. https://doi.org/10.1007/s00500-016-2471-9.
Gibrat, R., 1931. Les Inégalités Économiques. Librairie du Recueil Sirey, Paris, France.
Greenwell, R.N., Angus, J.E., Finck, M., 1995. Optimal mutation probability for genetic algorithms. Math. Comput. Model. 21 (8), 1–11.
Hassanat, A., Almohammadi, K., Alkafaween, E.A., Abunawas, E., Hammouri, A., Prasath, V.S., 2019. Choosing mutation and crossover ratios for genetic algorithms—a review with a new dynamic approach. Information 10 (12), 390.
He, D., Wang, F., Mao, Z., 2008. A hybrid genetic algorithm approach based on differential evolution for economic dispatch with valve-point effect. Int. J. Electr. Power Energy Syst. 30 (1), 31–38.
Heratha, S.B., Herath, T.C., 2018. Post-audits for managing cyber security investments: bayesian post-audit using Markov chain Monte Carlo (MCMC) simulation. J. Account. Publ. Pol. 37 (6), 545–563.
Herrera, F., Lozano, M., 2001. Adaptive genetic operators based on coevolution with fuzzy behaviors. IEEE Trans. Evol. Comput. 5 (2), 149–165.
Hirano, M., Izumi, K., Shimada, T., Matsushima, H., Sakaji, H., 2020. Impact analysis of financial regulation on multi-asset markets using artificial market simulations. J. Risk Financ. Manag. 13 (4), 75.
Holland, J.H., 1975. Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor.
Holland, J.H., 1992. Adaptation in Natural and Artificial Systems. MIT Press, Cambridge. MA.
Huang, B., Huan, Y., Xu, L.D., Zheng, L., Zou, Z., 2019. Automated trading systems statistical and machine learning methods and hardware implementation: a survey. Enterprise Inf. Syst. 13 (1), 132–144.
Ilia, S., Anastasia, A., Yaroslav, S., 2022. New financial tools in sport: NFTs and fan tokens. Корпоративные финансы 16 (2), 107–119.
Karakış, R., Tez, M., Kılıç, T.A., Kuru, Y., Güler, I., 2013. A genetic algorithm model based on artificial neural network for prediction of the axillary lymph node status in breastcancer. Eng. Appl. Artif. Intell. 26 (3), 945–950.
Katoch, S., Chauhan, S.S., Kumar, V., 2021. A review on genetic algorithm: past, present, and future. Multimed. Tool. Appl. 80 (5), 8091–8126.
Kim, Y.H., Yoon, Y., 2015. A genetic filter for cancer classification on gene expression data. Bio Med. Mater. Eng. 26 (s1), S1993–S2002.
Kraskov, A., Stögbauer, H., Grassberger, P., 2004. Estimating mutual information. Phys. Rev. 69 (6), 066138.
Levy, M., Solomon, S., 1997. New evidence for the power-law distribution of wealth. Phys. Stat. Mech. Appl. 242 (1–2), 90–94.
Li, S., Deng, M., Lee, J., Sinha, A., Barbastathis, G., 2018. Imaging through glass diffusers using densely connected convolutional networks. Optica 5 (7), 803–813.
LiCalzi, M., Pellizzari, P., 2003. Fundamentalists clashing over the book: a study of order-driven stock markets. Quant. Finance 3 (6), 470.
Liu, L., Moayedi, H., Rashid, A.S.A., Rahman, S.S.A., Nguyen, H., 2020. Optimizing an ANN model with genetic algorithm (GA) predicting load-settlement behaviours of eco-friendly raft-pile foundation (ERP) system. Eng. Comput. 36 (1), 421–433.
Lopez-Gonzalez, H., Griffiths, M.D., 2023. Gambling-like features in fan tokens. J. Gambl. Stud. 1–18.
Lucas, A., Iliadis, M., Molina, R., Katsaggelos, A.K., 2018. Using deep neural networks for inverse problems in imaging: beyond analytical methods. IEEE Signal Process. Mag. 35 (1), 20–36.
Maeda, I., deGraw, D., Kitano, M., Matsushima, H., Sakaji, H., Izumi, K., Kato, A., 2020. Deep reinforcement learning in agent based financial market simulation. J. Risk Financ. Manag. 13 (4), 71.
Malsburg, C.V.D., 1986. Frank rosenblatt: principles of neurodynamics: perceptrons and the theory of brain mechanisms. In: Brain Theory. Springer, Berlin, Heidelberg, pp. 245–248.
Mazur, M., Vega, M., 2022. Football and Cryptocurrencies. Available at: SSRN 4035558.
McCann, M.T., Jin, K.H., Unser, M., 2017. Convolutional neural networks for inverse problems in imaging: a review. IEEE Signal Process. Mag. 34 (6), 85–95.
Moayedi, H., Raftari, M., Sharifi, A., Jusoh, W.A.W., Rashid, A.S.A., 2020. Optimization of ANFIS with GA and PSO estimating α ratio in driven piles. Eng. Comput. 36 (1), 227–238.
Moon, S.H., Kim, Y.H., 2020. An improved forecast of precipitation type using correlation-based feature selection and multinomial logistic regression. Atmos. Res. 240, 104928.
Newall, P.W., Xiao, L.Y., 2021. Gambling marketing bans in professional sports neglect the risks posed by financial trading apps and cryptocurrencies. Gaming Law Rev. 25 (9), 376–378.
Nowotniak, R., Kucharski, J., 2010. Building Blocks Propagation in Quantum-Inspired Genetic Algorithm. ArXiv Paper, arXiv:1007.4221.
Parham, A., Breitinger, C., 2022. Non-fungible Tokens: Promise or Peril?. arXiv Preprint arXiv:2202.06354.
Qu, Z., Liu, H., Wang, Z., Xu, J., Zhang, P., Zeng, H., 2021. A combined genetic optimization with AdaBoost ensemble model for anomaly detection in buildings electricity consumption. Energy Build. 248, 111193.
Rai, S., Belwal, R.C., Gupta, A., 2023. Is the corpus ready for machine translation? A case study with Python to pseudo-code corpus. Arabian J. Sci. Eng. 48, 1845–1858. https://doi.org/10.1007/s13369-022-07049-0.
SaiToh, A., Rahimi, R., Nakahara, M., 2014. A quantum genetic algorithm with quantum crossover and mutation operations. Quant. Inf. Process. 13, 737–755.
Scharnowski, M., Scharnowski, S., Zimmermann, L., 2021. Fan Tokens: Sports and Speculation on the Blockchain. Available at: SSRN 3992430.
Sezer, O.B., Gudelek, M.U., Ozbayoglu, A.M., 2020. Financial time series forecasting with deep learning: a systematic literature review: 2005–2019. Appl. Soft Comput. 90, 106181.
Sinha, A., Horvath, P.A., Beason, T., Roos, K.R., 2019. Simulation of a financial market: the possibility of catastrophic disequilibrium. Chaos, Solit. Fractals 125, 13–16.
Suganthan, P.N., Hansen, N., Liang, J.J., Deb, K., Chen, Y.P., Auger, A., Tiwari, S., 2005. Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. KanGAL report, 2005005(2005), 2005.
Tedeschi, G., Iori, G., Gallegati, M., 2012. Herding effects in order driven markets: the rise and fall of gurus. J. Econ. Behav. Organ. 81 (1), 82–96.
Vellekoop, I.M., Mosk, A.P., 2008. Phase control algorithms for focusing light through turbid media. Opt Commun. 281 (11), 3071–3080.
Vidal-Tomás, D., 2022. The new crypto niche: NFTs, play-to-earn, and metaverse tokens. Finance Research Letters 47 (PB), 102742.
Vidal-Tomás, D., 2023. Blockchain, sport and fan tokens. J. Econ. Stud.
Wang, J., Gao, L., Zhang, H., Xu, J., 2011. Adaboost with SVM-based classifier for the classification of brain motor imagery tasks. In: International Conference on Universal Access in Human-Computer Interaction. Springer, Berlin, Heidelberg, pp. 629–634.
Wang, H., Liu, J., Zhi, J., Fu, C., 2013. The improvement of quantum genetic algorithm and its application on function optimization. Math. Probl Eng. 2013. Article ID 730749.
Wei, Z., Chen, X., 2018. Deep-learning schemes for full-wave nonlinear inverse scattering problems. IEEE Trans. Geosci. Rem. Sens. 57 (4), 1849–1860.
Xu, W., Hu, Y., Luo, W., Wang, L., Wu, R., 2021. A multi-objective scheduling method for distributed and flexible job shop based on hybrid genetic algorithm and tabu search

considering operation outsourcing and carbon emission. Comput. Ind. Eng. 157, 107318.
Yin, R.K., 2018. Case Study Research and Applications: Design and Methods. SAGE, Los Angeles.
Yule, G., 1925. A mathematical theory of evolution based on the conclusions of Dr. J. C. Willis, F.R.S.,''. Phil. Trans. B 213, 21–87.
Zarifis, A., Cheng, X., 2022. The business models of NFTs and fan tokens and how they build trust. Journal of Electronic Business & Digital Economics (ahead-of-print).
Zaucha, T., Agur, C., 2022. Newly Minted: Non-fungible Tokens and the Commodification of Fandom, vol. 14614448221080481. New Media & Society.
Zemouri, R., Omri, N., Fnaiech, F., Zerhouni, N., Fnaiech, N., 2020. A new growing pruning deep learning neural network algorithm (GP-DLNN). Neural Comput. Appl. 32 (24), 18143–18159.
Zhang, Gx, Li, N., Jin, Wd, Hu, L., 2006. Novel quantum genetic algorithm and its applications. Front. Electr. Electron. Eng. China 1, 31–36.
Zhang, C., Qi, D., Cai, Z., et al., 2019. MII: a novel content defined chunking algorithm for finding incremental data in data synchronization. IEEE Access 7, 86932–86945.
Zhou, D., 2021. Financial market prediction and simulation based on the FEPA model. J. Math. 2021, 1–11.
Zhou, Y., Wang, Y., Wang, K., Kang, L., Peng, F., Wang, L., Pang, J., 2020. Hybrid genetic algorithm method for efficient and robust evaluation of remaining useful life of supercapacitors. Appl. Energy 260, 114169.