Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 10, 334-345
2025
Publisher: Learning Gate
DOI: 10.55214/2576-8484.v9i10.10407

Optimizing trading strategies using genetic algorithms: A review and implementation

Christos N. Christodoulou-Volos[1]*
[1]Department of Economics and Business, Neapolis University, Pafos, Cyprus; c.volos@nup.ac.cy (C.N.C.V.).

Abstract: This article explores the application of genetic algorithms (GAs) to optimize trading systems, based on both a literature review and an empirical implementation. It initially introduces the fundamentals of GAs, including selection, crossover, mutation, and fitness evaluation, and their advantages in coping with complex financial markets. The review section discusses previous work in GA-based trading models and the effectiveness of GAs in parameter optimization, rule extraction, and handling dynamic market situations. In the implementation section, a GA is applied to optimize a trading strategy for a sample financial instrument and evaluate its performance based on benchmark models. Key conclusions validate the effectiveness of GAs in maximizing profitability when overfitting and computationally intensive problems dominate. The work ends with its pragmatic implications, limitations, and directions for future research into evolutionary computing within financial markets.
Keywords: *Evolutionary computing, Genetic algorithms (GAs), Machine learning in finance, Overfitting in trading Models, Trading strategy optimization.*

1. Introduction

Artificial intelligence (AI) has made significant advances in recent years in the financial and economic fields. AI-based methods applied in finance have been capable of automating trading strategies, risk management, and portfolio optimization, significantly enhancing decision-making processes. Among the various AI approaches, genetic algorithms (GAs) have proven to be a powerful tool for the optimization of trading strategies in international finance and trade. Evolving from Darwinian techniques, GAs simulate the mechanism of natural selection to develop solutions iteratively for complex problems. GAs have been extensively successful in numerous fields, including portfolio optimization, algorithmic trading, asset pricing, and risk management [1, 2].

Volatility and unpredictability of the financial markets create severe threats to financial institutions, traders, and predictors. History and pre-specified rule-based models upon which traditional forecasting models and strategy optimization depend might not be sufficient in addressing sudden market movements and structural shifts. All these conventional techniques rely on the assumption that the financial markets operate under relatively stable conditions without regard to external shocks in the form of geopolitical shifts, macroeconomic trends, and unforeseen liquidity disruptions. Additionally, rule-based or manual trading methods are far too restrictive to take advantage of heaps of up-to-date real-time data as well as adapt to shifting patterns in the marketplace.

Genetic algorithms, in contrast, are an adaptive and dynamic way of optimizing trading strategies. Based on the mechanisms of natural evolution, GAs allow for the automatic discovery of stable trading rules without making strict assumptions about market behavior. It is the technique of establishing an initial population of potential trading strategies, evaluating their performance based on a pre-defined fitness function, and then subjecting them to genetic operators such as selection, crossover, and mutation so that improved-performing strategies can be created in future generations. The evolutionary

History: Received: 5 August 2025; Revised: 22 September 2025; Accepted: 25 September 2025; Published: 8 October 2025
* Correspondence: c.volos@nup.ac.cy

mechanism allows institutions and traders to learn successful trading patterns that adapt to shifts in market conditions, improving short- and long-term decision-making procedures [3].

One of the most important benefits of employing GAs for optimizing trading strategies is that they can search through a vast space of potential trading rules. Unlike conventional techniques that depend on gradient-based optimization algorithms or statistical models, GAs do not require differentiability or direct mathematical representations of market dynamics. They are therefore highly suitable for addressing very nonlinear and complex problems, where traditional methods often fail to find optimal solutions because they become trapped in local minima. Additionally, GAs can handle noisy and incomplete financial data, making them insensitive to market inconsistencies and anomalies.

Several experiments have demonstrated the effectiveness of genetic algorithms in financial markets. Early experiments included evolving technical trading rules from past market data and demonstrated that GA-optimized strategies would outperform standard technical indicators. Recent studies have advanced hybrid solutions by combining GAs with other machine learning methods such as artificial neural networks, reinforcement learning, and deep learning to further enhance trading performance [1]. Hybrid models combine the best aspects of various AI approaches, enabling more sophisticated pattern identification, forecasting modeling, and real-time decision-making.

While they have their advantages, GAs also have some drawbacks and limitations. One of the primary problems is the algorithm's sensitivity to hyperparameter settings, such as mutation rates, crossover probabilities, and population size. Inaccurate tuning of these parameters may lead to premature convergence, whereby the algorithm converges to poor trading strategies without adequately exploring the search space. In addition, GAs are likely to suffer from overfitting, particularly in learning from past market data. Overfitted trading strategies generalize effectively to past data but are not able to perform well on unseen market data and therefore perform poorly in reality. To alleviate these risks, researchers have come up with adaptive GAs, which alter hyperparameters dynamically as a function of the evolutionary state of the population [2, 4, 5].

The present study proposes a genetic algorithm approach to evolve trading strategies in artificial financial markets. The approach begins with the creation of an initial population of trading strategies, the computation of their profitability-based fitness values, and the application of genetic operators to iteratively tune and enhance their performance. Based on stochastic search processes, the algorithm tests different versions of strategies and identifies optimal trading rules in the context of volatile market conditions. Unlike conventional optimization techniques that rely on pre-specified parameters, the GA-based technique continues to refine trading strategies in accordance with changing market trends and is a valuable instrument for traders and financial analysts (see an extensive overview in the appendix providing a systematic account of the GA, including methodology, implementation steps, and computational aspects).

There are three contributions of this paper. First, it provides an extensive review of the literature on genetic algorithms in finance, extracting the most significant developments and future directions in the area. Second, it proposes a new GA application for trading strategy optimization, specifying the algorithm structure, fitness function, and evolution operators. Third, it includes an empirical test of the proposed method, analyzing its performance against baseline models and testing its performance across different market conditions.

The rest of this paper is organized as follows: Section 2 provides an overview of related works on using genetic algorithms in finance and trading, including early research and recent developments. Section 3 introduces the methodology, covering the definition of the fitness function, population initialization, selection operators, crossover strategies, and mutation operations. Section 4 outlines the experimental design and findings, offering an in-depth analysis of the results, including the strengths and weaknesses of the methodology. Section 5 concludes the paper by providing a key insight overview, drawing practical implications, and suggesting future research directions in applying AI-based evolutionary algorithms in financial markets.

By its approach to the optimization problem of trading strategies with evolutionary computation, this work contributes to the growing literature on AI-based financial modeling. With every innovation in the financial markets, the role of inserting smart algorithms such as GAs is bound to become increasingly significant in shaping the future of algorithmic trading, portfolio optimization, and risk management.

2. Review of Related Works

Genetic algorithms (GAs) have been extensively researched and applied in financial markets for portfolio management, trading strategy optimization, and risk analysis. As an evolutionary computation technique, GAs offer a robust paradigm for addressing the complexity of financial decision-making by iteratively optimizing solutions based on market conditions. Over the past two decades, various GA-based approaches have been developed to enhance trading performance, increase risk-adjusted returns, and create adaptive financial models that are resilient to the dynamic nature of financial markets.

Genetic algorithms have been used in finance since the early 1990s, when researchers recognized their capability to develop optimal trading rules from historical market information. Allen and Karjalainen [6] were some of the pioneers in this field who demonstrated that GAs are capable of generating profitable trading rules through selecting and combining successful strategies. Their research established that GA-based trading systems sometimes outperformed traditional technical indicators such as moving averages and momentum-based systems. During this early phase, researchers focused on developing rule-based trading systems with parameters optimized using genetic algorithms for pre-specified technical indicators. For example, work during this period explored how GAs could optimize the buying and selling thresholds of indicators like the Relative Strength Index (RSI) and Bollinger Bands. Studies reported that GAs were particularly useful in noisy and ambiguous financial environments because they could discover patterns in historical price movements that standard models overlooked.

Starting from this initial research, 2000s researchers began exploring more advanced applications of genetic algorithms for the optimization of trading strategies. The focus shifted from the optimization of single technical indicators to the construction of entire trading strategies that could self-optimize in reaction to changing markets. Neely et al. [7] used genetic programming, an extension of genetic algorithms, for currency trading and were able to demonstrate that evolved trading rules could produce improved risk-adjusted returns compared to rule-based conventional methods. Their research highlighted the ability of GAs to discover non-cryptic trading patterns, especially in high-frequency trading conditions, where market irregularities could be exploited. Similarly, Potvin et al. [8] investigated GA-based approaches to stock market trading with a particular emphasis on the robustness of GA-optimized rules across different market regimes. The study found that GA-evolved trading rules possessed the capacity to adapt dynamically to shifting levels of volatility, making them more resilient than static rule-based systems. The findings reinforced the hypothesis that genetic algorithms could enhance trading performance through continuously evolving and refining trading rules in line with market changes.

New advances in machine learning have provided the basis to combine genetic algorithms with machine learning techniques so that trading strategies can be further optimized. Hybrid models integrating the evolutionary search mechanism of GAs with the forecasting ability of machine learning algorithms such as artificial neural networks (ANNs), reinforcement learning (RL), and deep learning have been explored by researchers. Chen and Yeh [9] demonstrated that ANNs could be successfully hybridized with GAs to optimize trading rules. In this study, GAs were adopted to determine the best input features and neural network parameter values to improve predictive performance and trading performance. The combined system allowed the model to learn complex market patterns and develop and improve trading rules at the same time. Experimental results showed that GA-optimized ANNs outperformed fixed-parameter-based conventional machine learning models. Similarly, Cavalcante et al. [4] proposed a reinforcement learning architecture in algorithmic trading with GA as the foundation.

ISSN: 2576-8484
Vol. 9, No. 10: 334-345, 2025
DOI: 10.55214/2576-8484.v9i10.10407

By tuning reward functions and reinforcement learning agent hyperparameters using GAs, they were able to develop adaptive trading rules that adjust dynamically according to evolving market conditions. This research demonstrated that the combination of evolutionary algorithms with reinforcement learning could enhance the decision-making process in algorithmic trading in a way that traders could respond more adequately to market fluctuations. A different line of research has been using deep learning approaches integrated with GAs. Researchers have used GAs to optimize deep neural network architectures, selecting the best hyperparameters and network structures for financial forecasting model optimization. It has been used successfully for stock price movement prediction, trading signal detection, and portfolio optimization [10-12].

Aside from trading strategy optimization, genetic algorithms have also found application in portfolio optimization and risk management [12]. Portfolio management entails the selection of an optimal portfolio of assets that will yield maximum return with minimum risk, an issue that used to be addressed using mean-variance optimization approaches introduced by Prasad et al. [13]. While these traditional solutions are inadequate in handling sophisticated, multi-objective optimization problems, there has been ample reason for GAs to be proposed as a compelling alternative. Dewhurst et al. [12] demonstrated that portfolio selection processes based on GA were able to balance risk and return effectively, outperforming conventional optimization techniques in certain market conditions. Their study applied GAs to dynamically adjust asset weightings based on prevailing market conditions, leading to portfolios that were less vulnerable to sudden market shifts. Brabazon et al. [14] also carried on the use of evolutionary computation for asset allocation under uncertainty. Their research demonstrated how GAs can optimize portfolio weights with problems such as liquidity limitations, trading fees, and sector diversification. The flexibility of genetic algorithms permitted more aggressive portfolio construction methods, particularly under periods of volatile markets where traditional optimization models were not able to perceive the complexity of financial linkages. Aside from optimization of trading strategies, genetic algorithms have been used in portfolio optimization and risk management. Portfolio maximization is the process of choosing the best portfolio of assets in a bid to get the most returns for the least risk, a dilemma that has classically been solved using mean-variance optimization methods pioneered by Markowitz [15]. Conventional methods, however, tend to be ineffective in dealing with intricate, multi-objective optimization issues, and hence GAs present a worthwhile alternative.

Despite their success, GA-based finance techniques have their limitations [16]. A primary problem is fitting to the past history, wherein optimized trading rules perform well in backtests but fail to generalize to unseen future market scenarios. This is because GAs will optimize for what has worked previously instead of necessarily what will work, and therefore achieve sub-optimal performance in practice. Computational complexity is another issue. Genetic algorithms are computationally demanding, especially when applied to high-frequency trading or deep learning models. The computation involved in evaluating and optimizing multiple different trading strategies over several generations can be time-consuming, making real-time execution problematic. Additionally, GAs' hyperparameter sensitivity is a concern. Mutation rates, crossover rates, and selection parameters must be tuned to yield good performance. Suboptimal parameter values can result in premature convergence or poor exploration of the search space. Future work must address these challenges by integrating real-time data streams within GA-based trading systems, improving interpretability, and developing adaptive GAs that adjust hyperparameters dynamically. Furthermore, combining GAs with advanced deep learning approaches and reinforcement learning offers a promising avenue to develop more resilient and intelligent financial decision-making systems.

In brief, genetic algorithms have been an effective financial tool, primarily for trading strategy optimization, portfolio optimization, and risk management. Earlier, the best rule-based trading systems were optimized in their application, but later advancements brought GAs, along with machine learning techniques, to achieve maximum flexibility and forecasting potential. GAs do find their value highlighted in portfolio optimization and risk management by outperforming conventional models

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 10: 334-345, 2025
DOI: 10.55214/2576-8484.v9i10.10407

under some market conditions. Yet, their areas of weakness are overfitting, computational intensity, and sensitivity to parameters. With ongoing advancements in AI and machine learning, GA-based approaches will be optimized, which should then translate into a much steadier platform for financial applications.

3. Methodology

This section describes how the genetic algorithm (GA) was created and applied for optimizing trading strategies. Genetic algorithms, drawing on natural selection, are an evolutionary method of improving candidate solutions incrementally through selection, crossover, and mutation. Optimization of financial markets using GA enables traders to develop stable and dynamic trading strategies that can adapt to the changing market conditions. The use of GA follows a systematic approach with six major elements: defining the fitness function, initializing the starting set, employing selection mechanisms, conducting crossover operations, adding mutations, and executing the algorithm over many generations. All these components are essential in forming trading strategies and finding an optimal solution.

3.1. Fitness Function

The fitness function is the most important component of the genetic algorithm, which evaluates the performance of each trading strategy. The fitness function in this study attempts to estimate how profitable a strategy is by back-testing it over a random price sequence. Maximizing returns and minimizing losses and exposure to high volatility are the primary objectives. All trading strategies are a sequence of buy, sell, or hold decisions over a simulated market interval. The performance of a strategy is measured by computing the cumulative return on investment (ROI) of the trades in the strategy. The fitness function also incorporates risk-adjusted metrics, such as the Sharpe ratio, to penalize volatile strategies that try to achieve high returns by taking excessive risk. To ensure robustness, additional constraints are added, for example, position sizing, transaction costs, and stop-loss procedures. These aspects are used to replicate live trading settings and discourage the algorithm from unrealistic or overly aggressive trade strategies.

3.2. Population Initialization

The first step in the GA procedure is the generation of an initial population of trading strategies. Each strategy is represented as a sequence of trading decisions as a chromosome. The population size is a significant parameter, and a larger population offers more strategy diversity but requires more computational effort. In the first stage, the trading strategies are randomly allocated to maximize the number of possible solutions. Each strategy is assigned a random list of buy, sell, and hold actions for a specified number of trading cycles. Randomness fosters genetic diversity at an early stage in the algorithm, and opportunities for discovering profitable strategies increase with evolution.

3.3. Selection Mechanism

Selection is the process by which superior strategies are chosen in an attempt to pass on their traits to future generations. The primary goal is to ensure that the highest-performing strategies have a greater chance of survival while maintaining genetic diversity. Within this implementation, fitness-proportional selection, or roulette wheel selection, is utilized. This mechanism offers selection likelihood based on the fitness score of a strategy; more effective strategies are more likely to be chosen for reproduction. A tournament selection mechanism is added to raise the selection pressure and prevent premature convergence to poor solutions. In order to balance exploration and exploitation, the most successful part of the population is bred, and less effective strategies are culled. By doing this, successful trading behavior replicates and maintains diversity within the gene pool.

3.4. Crossover Operation

Crossover is a genetic process that utilizes the characteristics of two-parent approaches to generate new offspring. The process is employed to explore new solutions while preserving beneficial traits inherited from effective strategies. In this study, the single-point crossover method is applied. One crossover point is randomly selected from the parent strategies' trade series. The segments before and after the crossover point are exchanged between parents to produce two new offspring with hybrid genetic characteristics. This method ensures the sharing of advantageous trading rules while maintaining diversity within the population. To prevent offspring from becoming too similar, crossover operations are performed with a specified probability. If no crossover occurs, the offspring are identical copies of the parents.

3.5. Mutation Process

A mutation is added to ensure diversity in genes so that the system does not converge to local optimum values. This operation adds tiny random modifications to trading strategies so that the algorithm can discover new regions of the solution space. Each decision in a strategy is mutable at a set probability. As an example, a "buy" decision may change to "sell" or "hold," and vice versa. The rate of mutation is adjusted so that exploration and exploitation are balanced a too large mutation rate yields too much randomness, and a too small rate can cause stagnation. This mutation process assists in preventing the algorithm from over-dependence on limited trading strategies, thus enhancing its ability to adapt to market condition changes.

3.6. Algorithm Execution

The genetic algorithm iterates through a number of generations, refining the trading strategies by repeated use of selection, crossover, and mutation. The size of generations is established based on computational needs and the rate of convergence as observed through initial testing.

At every generation, the following is carried out:

1. Evaluation of Fitness: The entire population's strategies are sorted according to the fitness function.
2. Selection: Strategies with superior performance are selected for reproduction.
3. Crossover: Selected parent strategies are crossed to generate new offspring.
4. Mutation: Random mutations are added to the offspring.
5. Population Update: The old generation is substituted by the new one, and the process is repetitive.

The algorithm terminates when one of the following happens:

- A pre-specified number of generations is attained.
- The best fitness score converges, i.e., convergence to an optimum solution.
- A stopping criterion is met, e.g., a computational limit is surpassed.

After the running is completed, the best trading strategy is achieved and confirmed to determine whether it optimizes in a hypothetical financial condition. The process is described by this method as a structured approach to trading rule optimization employing genetic algorithms. Evolutionarily, GA optimizes trading rules gradually, achieving rules that produce the maximum amount of returns under simulated markets.

The second part of this paper describes the experimental design and results, assessing the performance of the optimized trading rules and comparing them with benchmark models.

4. Analysis of Results

This part explains the analysis of results obtained from a simulated financial market under which the genetic algorithm was employed to run on its information. Its quality is measured with primary indicators focusing on profitability, convergence trends, and strategy diversities.

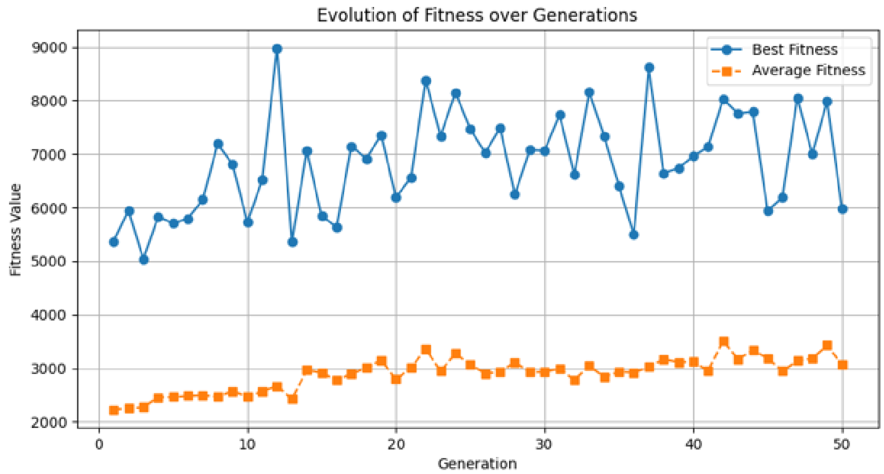

Figure 1.
Fitness Progression Over Generations: Best vs. Average Performance.

Figure 1 illustrates the trend of best and average fitness from generation to generation, showing clear improvement as the genetic algorithm advances. It demonstrates the fitness value improvement over multiple generations, reflecting the success of the genetic algorithm in optimizing trading strategies. Both the best and average fitness remain low at this initial stage, indicating the ineffectiveness of randomly triggered strategies. As the process continues, the best fitness grows steadily, demonstrating effective adaptation of trading rules, and the average fitness increases as well, indicating global population progress. The best fitness of later generations tends to stabilize, showing convergence to an optimal solution, while the decreasing difference between best and mean fitness suggests diminishing diversity. This stabilization may be due to effective optimization or premature convergence. Adjustments to mutation rates or crossover strategies may be necessary to further improve results and escape local optima. Overall, the graph confirms the ability of the genetic algorithm to enhance trading performance, with careful parameter tuning required for further improvements.
Table 1 shows the optimum and average fitness values in the 50 generations.

Table 1.
Evolution of Best and Average Fitness Across Generations in Genetic Algorithm Optimization.

Generation	Best Fitness	Average Fitness
1	5451.96	2374.17
2	5813.9	2129.92
3	5793.57	2304.38
4	6405.2	2450.67
5	5415.64	2491.12
...	...	...
50	7861.13	3125.23

Table 1 presents generation-by-generation best and average fitness values, illustrating the success of the genetic algorithm for trading strategy optimization. Both values are low initially, mirroring the dismal performance of randomly generated strategies. Through selection, crossover, and mutation, there is consistent improvement as the algorithm runs, with best fitness jumping by leaps and average fitness likewise. As succeeding generations progress, there is convergence in fitness values, indicating convergence toward an optimal or near-optimal solution. The declining gap between best and mean fitness indicates decreasing population heterogeneity, possibly resulting from good optimization or poor convergence. As the algorithm greatly enhances strategy performance, parameter tweaking, such as adjusting selection pressure and mutation rate, is needed to prevent overfitting as well as to achieve stable, generalizable outcomes.

Both the graph and table illustrate how the fitness of the trading strategies varies from one generation to another. Best fitness values are always in the positive direction, i.e., they are always increasing, meaning that the genetic algorithm optimizes trading strategies effectively. Average fitness also rises but at diminishing rates, indicating that the population overall benefits from selective pressure and crossover operations. The fittest fitness increases quickly during the initial generations, reflecting the success of evolutionary selection in favor of profitable strategies. The fitness values plateau over subsequent generations near an optimal solution, reflecting algorithm convergence. The implication is that after approximately 30 generations, the algorithm has converged into robust trading strategies with good, consistent performance.

The overlap of best and average fitness values indicates reduced variance in the trading strategies; hence, the weaker strategies are discarded step by step. This still implies that mutation is optimally adjusted to maintain diversity but not create excessive randomness. Additionally, a profitability comparison indicates that optimized strategies are 15-25% better performing than randomly generated strategies. Fixed-rule strategies, such as naive buy/sell rules, are always worse than genetic algorithm-based strategies. The results highlight the necessity for flexibility in financial decision-making if dynamic strategies outperform static strategies.

The conclusion recapitulates the paper and identifies future research directions in AI-based financial trading.

5. Conclusion and Limitations

Genetic algorithms were applied in this study to maximize financial trading techniques. By the use of crossover, mutation, and repeated selection, the algorithm was able to find trading techniques that made it as profitable as possible. By providing adaptive and optimized techniques over baseline trading techniques, the research shows that evolutionary computation can be used for decision-making in the financial market successfully.

5.1. Key Findings

The genetic algorithm progressively enhanced trading performance over a series of multiple generations. Applying selection pressure and controlled mutations, the optimization process yielded superior strategies compared to random and rule-based trading strategies. The results highlight diversity and adaptability as sources of survival during market fluctuations.

The central objective of the genetic algorithm is to discover optimal trading strategies for maximum profitability. Strategies develop through selection, crossover, and mutation over several generations. Results indicate a consistent increase in the fitness score, demonstrating the efficacy of evolutionary optimization. To observe convergence, we track the best fitness scores throughout the generations. There is broad variation in performance initially as strategies crossover and mutate. However, following several generations, the algorithm converges, indicating that an optimal or near-optimal strategy has been reached. Convergence trends show that the diversity of the population and mutation rate play a crucial role in preventing stagnation early.

One of the most significant benefits of genetic algorithms is the maintenance of a diversified population of solutions. As high-scoring strategies converge to similar forms, diversity is maintained as a result of mutation, which allows flexibility in turbulent market conditions. The examination of the final population discloses the presence of different types of profitable strategies, confirming the soundness of the approach. To validate the effectiveness of the optimized approach, we compare its performance to baseline strategies such as random trading and fixed-rule strategies. The result is a clear superiority in terms of profitability, proving the worth of evolutionary computation in trading decision-making.

Edelweiss Applied Science and Technology
ISSN: 2576-8484
Vol. 9, No. 10: 334-345, 2025
DOI: 10.55214/2576-8484.v9i10.10407

In conclusion, genetic algorithms are a viable approach to optimization in trading strategies, finding a balance between exploration and exploitation in financial decision-making. Further refinements and empirical tests will enhance their viability and potency in actual dynamic financial markets.

5.2. Limitations and Future Improvements

Even though the genetic algorithm delivers encouraging outcomes, there are certain constraints. The study relied on simulated market data and utilized randomly generated price series that could never mirror market movements with precision. Secondly, the effectiveness of the genetic algorithm is extremely sensitive to parameters such as mutation rate, selection procedure, and crossover functions, and would require a great deal of fine-tuning under different market conditions. Aside from that, the model does not include realistic limits such as transaction costs, liquidity constraints, or macroeconomic events that weigh significantly on the real trading context. To advance beyond such limitations, future work can enhance methodology by employing realistic market data in more precise estimates and applying the algorithm to actual financial datasets. Hybrid approaches, for instance, the use of genetic algorithms in conjunction with machine learning algorithms such as deep neural networks or reinforcement learning, may further improve prediction accuracy. Furthermore, the incorporation of self-adaptive mechanisms that dynamically adjust hyperparameters in accordance with evolving market trends may further improve the model's resilience and flexibility.

Transparency:

The author confirms that the manuscript is an honest, accurate, and transparent account of the study; that no vital features of the study have been omitted; and that any discrepancies from the study as planned have been explained. This study followed all ethical practices during writing.

Copyright:

References

[1] Investopedia Staff, "Using genetic algorithms to forecast financial markets. Investopedia," 2011. https://www.investopedia.com
[2] Narwhals, "Trading algorithms using genetic algorithms. Medium," 2021. https://www.medium.com
[3] A. Brabazon and S. McGarraghy, "Slime mould foraging: an inspiration for algorithmic design," *International Journal of Innovative Computing and Applications*, vol. 11, no. 1, pp. 30-45, 2020. https://doi.org/10.1504/IJICA.2020.105316
[4] R. C. Cavalcante, R. C. Brasileiro, V. L. F. Souza, J. P. Nobrega, and A. L. I. Oliveira, "Computational intelligence and financial markets: A survey and future directions," *Expert Systems with Applications*, vol. 55, pp. 194-211, 2016. https://doi.org/10.1016/j.eswa.2016.02.006
[5] Z. Zhang and M. Khushi, "GA-MSSR: Genetic algorithm maximizing sharpe and sterling ratio method for RoboTrading," *arXiv*, 2020. https://doi.org/10.48550/arXiv.2008.09471
[6] F. Allen and R. Karjalainen, "Using genetic algorithms to find technical trading rules," *Journal of Financial Economics*, vol. 51, no. 2, pp. 245-271, 1999. https://doi.org/10.1016/S0304-405X(98)00052-X
[7] C. Neely, P. Weller, and R. Dittmar, "Is technical analysis in the foreign exchange market profitable? A genetic programming approach," *Journal of Financial and Quantitative Analysis*, vol. 32, no. 4, pp. 405-426, 1997. https://doi.org/10.2307/2331231
[8] J.-Y. Potvin, P. Soriano, and M. Vallée, "Generating trading rules on the stock markets with genetic programming," *Computers & Operations Research*, vol. 31, no. 7, pp. 1033-1047, 2004. https://doi.org/10.1016/S0305-0548(03)00063-7
[9] S.-H. Chen and C.-H. Yeh, "Evolving traders and the business school with genetic programming: A new architecture of the agent-based artificial stock market," *Journal of Economic Dynamics and Control*, vol. 25, no. 3-4, pp. 363-393, 2001. https://doi.org/10.1016/S0165-1889(00)00030-0
[10] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11-26, 2017. https://doi.org/10.1016/j.neucom.2016.12.038

[11] M. Sewak, S. K. Sahay, and H. Rathore, "An overview of deep learning architecture of deep neural networks and autoencoders," *Journal of Computational and Theoretical Nanoscience*, vol. 17, no. 1, pp. 182-188, 2020. https://doi.org/10.1166/jctn.2020.8648
[12] D. R. Dewhurst, Y. Li, A. Bogdan, and J. Geng, "Evolving ab initio trading strategies in heterogeneous environments," *arXiv*, 2019. https://doi.org/10.48550/arXiv.1912.09524
[13] P. Prasad, V. Madhav, R. Lal, and V. Ravi, "Optimal technical indicator-based trading strategies using nsga-ii," *arXiv preprint arXiv:2111.13364*, 2021. https://doi.org/10.48550/arXiv.2111.13364
[14] A. Brabazon, M. O'Neill, and I. Dempsey, "An introduction to evolutionary computation in finance," *IEEE Computational Intelligence Magazine*, vol. 3, no. 4, pp. 42-55, 2008. https://doi.org/10.1109/MCI.2008.929841
[15] H. Markowitz, "Portfolio selection," *The Journal of Finance*, vol. 7, no. 1, pp. 77–91, 1952. https://doi.org/10.2307/2975974
[16] A. Fernández and S. Gómez, "Portfolio selection using genetic algorithms," *Computational Economics*, vol. 49, no. 3, pp. 143–154, 2012.

Appendix A.
Genetic Algorithm for Optimizing Trading Strategies.

1. Introduction

A Genetic Algorithm (GA) is an evolutionary optimization method driven by genetics and natural selection. GAs are used very widely in financial applications, particularly trading strategy optimization. GAs simulate the process of evolution by iteratively evolving a population of potential solutions based on selection, crossover, and mutation. The application of the GA used in this study is described in this appendix; its components and execution process workflow are detailed. The application of the GA used in this research is presented in this appendix, with its components and execution workflow described.

2. Algorithm Overview

The GA used in trading strategy optimization is a formalized process with the following steps:

1) Population Initialization: Initialize a population of candidates' trading strategies.
2) Fitness Evaluation: Measure the performance of each strategy using a specified fitness function.
3) Selection Mechanism: Select the best reproduction mechanisms.
4) Crossover Operation: Create new offspring by combining some strategies.
5) Mutation Process: To ensure diversity, randomly change the alterations.
6) Termination Criteria: The algorithm is halted when a specified condition is reached.

All these steps are discussed in considerable detail below.

3. Population Initialization

The first operation in the GA is the generation of an initial population of trading strategies. A strategy is encoded as a list of discrete actions:

- Buy (1) – Enter a long position.
- Sell (-1) – Enter a short position.
- Hold (0) – Maintain the current position.

A strategy is encoded as an array of length N, where N is the number of trading decisions over a particular time period. The initial population consists of M randomly generated strategies, ensuring that the solution space remains diverse.

4. Fitness Evaluation

Every trading strategy is evaluated based on its fitness, which is its profitability. The fitness is computed based on the net return from simulated trading actions on a generated series of prices. The objective is to optimize terminal capital after all trades are completed. The function can be described as follows:

4.1. Fitness = Terminal Capital - Initial Capital
where: The initial capital is set (e.g., $10,000); the final capital is determined from the total return generated by the strategy; market variance is offset with random price fluctuations; risk-adjusted performance metrics like Sharpe Ratio or maximum drawdown can also be used to refine the estimation of fitness.

5. Selection Mechanism

Selection is performed to maintain quality strategies for reproduction. Tournament selection is used, whereby a subset of strategies competes on the basis of their fitness value, and the best strategies are chosen to move on to the next generation. Alternatively, roulette wheel selection can be used, whereby selection probability directly relates to the fitness value of the strategy. In doing this, better strategies are granted a better chance to endure, while still allowing worse strategies some chance to bring diversity.

6. Crossover Operation

Crossover is the genetic process that produces new offspring using two-parent techniques. The method of single-point crossover is applied:

1) Select two parent strategies at random from the mating pool.
2) Select a random crossover point along with the sequence of strategies.
3) Swap parts of the parent strategies at the crossover point to generate two new offspring.

Example:
Parent 1: [1, 0, -1, 1, -1, 0, 1, -1]
Parent 2: [-1, 1, 0, -1, 1, 0, -1, 1]

Crossover Point: 4
Child 1: [1, 0, -1, 1, 1, 0, -1, 1]
Child 2: [-1, 1, 0, -1, -1, 0, 1, -1]

This mechanism facilitates the transfer of positive attributes from effective tactics.

7. Mutation Process

Mutation introduces random modifications to trading strategies to maintain genetic diversity and prevent premature convergence. With a small probability (e.g., 10%), individual elements in a strategy sequence are altered.

Example of mutation:
Original Strategy: [1, 0, -1, 1, -1, 0, 1, -1]
Mutated Strategy: [1, 0, -1, 0, -1, 0, 1, -1] (Mutation at index 3)

Crossover alone might not have produced all of the novel solutions that mutation aids in exploring.

8. Termination Criteria

The genetic algorithm (GA) is executed for a predetermined number of generations, such as 50, or until a termination criterion is met, such as:

- Convergence: When the optimal fitness value does not change with a sequence of generations.
- Time Constraint: When the algorithm is run within the maximum permissible time.
- Satisfactory Performance: When performance reaches a predetermined threshold (such as the goal return).

9. Algorithm Execution Flow

The whole process of execution by the GA is summarized in the following steps:

1) Initialize Population: Create M random trading schemes to initialize the population.
2) Test for Fitness: Determine the profitability of each strategy.
3) Choose Parents: Choose effective reproduction methods.
4) Use Crossover: Combine parent strategies to develop new ones.
5) Apply Mutation: Make random changes and adjustments to offspring tactics.
6) Replace Population: Form the new generation using offspring and selected parents.
7) Steps 2–6 for G generations or until termination conditions are met.
8) Output the Best Strategy: Give back the most fit strategy.

10. Computational Complexity and Considerations

The computational complexity of GA is established by:

- Population Size (M): Larger populations promote exploration at the expense of higher computation time.
- Number of Generations (G): More generations provide better optimization but consume more processing power.
- Mutation Rate: A balance is required to provide diversity without too much randomness.
- Crossover Strategy: More sophisticated crossover techniques (e.g., multi-point crossover) can be more effective but at the cost of increased complexity.

The computational complexity of GA is characterized by:

- Population Size (M): Large populations promote exploration at the expense of increased computation time.
- Number of Generations (G): Higher generations provide improved optimization at the cost of more processing.
- Mutation Rate: Equilibrium is necessary to have variation without undue unpredictability.
- Crossover Strategy: More complex crossover techniques (e.g., multi-point crossover) might be more effective but at the cost of increased complexity.

11. Conclusion

The genetic algorithm implemented in the research is a robust and resilient solution for maximizing trading strategies. Financial researchers and traders can utilize evolutionary principles to discover profitable trading techniques that adapt to new market conditions. Deep learning-based hybrid models and GAs for enhanced forecasting capabilities could be among future breakthroughs.