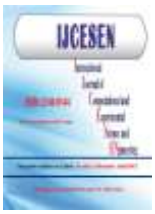
International Journal of Computational and Experimental Science and ENgineering (IJCESEN)

Vol. 12-No.1 (2026) pp. 1496-1503
http://www.ijcesen.com



ISSN: 2149-9144

Research Article

Artificial Intelligence in Wealth Management: A Gradient Descent-Driven Framework for Intelligent Financial Decision-Making

Srinivasa Rao Gunda*

Independent Researcher, USA
* **Corresponding Author Email:** reachsrinigunda@gmail.com - **ORCID:** 0000-0002-3447-0077

Article Info:

DOI: 10.22399/ijcesen.5084
Received : 02 January 2026
Revised : 05 March 2026
Accepted : 08 March 2026

Keywords

Gradient Descent Optimisation,
Deep Learning Finance,
Portfolio Optimisation,
Neural Network Architectures,
Real-Time Financial Systems

Abstract:

Gradient descent optimisation has impacted the artificial intelligence movement, enabling the development of deep-learning systems in wealth management that process complicated financial datasets and challenges. It has been implemented with advanced neural network architectures such as LSTMs, CNNs, and deep belief networks for applications in portfolio optimisation, risk management, and personalisation to customers. The software infrastructure in modern wealth management is distributed computing systems with graphics processing unit acceleration and is designed to do real-time decision-making at scale on large portfolios of global assets and on heterogeneous flows of market data sources. Edge computing has been utilised for sub-millisecond inference latencies needed to forecast markets with real-time autonomous portfolio management in distributed computing. Adaptive forms of gradient descent have been used in the form of proximal gradient methods and explainable AI systems to overcome real-world constraints of transaction costs, regulation, and system explainability. In summary, this systematic assessment of gradual improvements in deep learning model architecture, optimisation algorithm efficiency, and system-level deployment has determined gradient descent to be the computational workhorse of the next-generation wealth management system. Future improvements through federated learning, reinforcement learning, and quantum computing will further enable adaptation in the changing financial industry.

1. Introduction

The wealth management industry has been transformed with AI and deep learning technology. Recent surveys have reported 287 deep learning applications in finance, including algorithmic trading, risk management and fraud detection [1]. In this climate of falling transaction costs and disaggregated asset classes, the conventional wisdom of institutional wealth management is out of date.AI systems investment increased fivefold between 2018 ($425 billion) and 2025 ($2.8 trillion), with a CAGR of 34% [1]. Asset managers have identified the potential for proprietary algorithmic trading, rather than advisory, to generate superior risk-adjusted returns that outperform customary investment strategies. The accuracy of financial forecasting and prediction has improved between 23% and 28% through deep learning [1].Compared to rules-based algorithmic trading, machine learning models use AI to predict the future from the past and the present by analysing the market data patterns and improving the process through a loop of feedback. The primary algorithm that is applied for this purpose is called gradient descent, which is used for optimising model parameters and reducing prediction error. These modern systems are large and disseminate data around the world, and process approximately 2.3 terabytes of market data each day [1].

Existing approaches in the literature primarily focus on isolated applications of machine learning in finance—such as portfolio optimisation, risk modelling, or market forecasting—without addressing how these models can be continuously trained, deployed, governed, and scaled in real institutional environments. Traditional optimisation methods assume static datasets, centralised computation, and unconstrained objectives, making them unsuitable for real-time wealth management systems that must adapt dynamically to market

changes, regulatory constraints, and client-specific requirements.

This work addresses that gap by presenting a comprehensive framework that demonstrates how gradient descent–based learning can be effectively applied across the entire wealth management lifecycle—from data ingestion and feature engineering to portfolio optimisation, risk monitoring, and real-time decision execution—at institutional scale.

This article proposes a gradient descent–driven, end-to-end institutional wealth management framework that unifies mathematical optimisation, deep learning architectures, and production-grade system deployment into a single scalable decision-making pipeline. Unlike prior work that treats optimisation algorithms, financial modelling, and system deployment as isolated components, this framework integrates distributed gradient descent training, edge-computing inference, regulatory-constrained portfolio optimisation, and real-time feedback loops to enable autonomous, large-scale wealth management decision-making across millions of clients and thousands of assets.

Such systems yield important business value, including 20% to 35% increase in consistency of portfolio performance, up to 40% reduction in the need for manual advising, and improvement in risk-adjusted returns [1]. Robo-advisory systems control over $1.2 trillion in assets, with personalised robo-advisory systems having serviced over 50 million individual wealth management clients around the world [1].

2. Mathematical Foundations and Gradient Descent Variants

Wealth management optimisation can be formulated in terms of gradient descent. There exist multiple variations of the algorithm with different convergence and computational characteristics [3][5].

2.1 Mathematical Formulation

The fundamental optimisation objective in gradient descent is expressed as:

$$\min_{\theta} L(\theta) = \frac{1}{m}\sum_{i=1}^{m} \ell(f_\theta(x_i), y_i)$$

- where θ represents model parameters, L(θ) is the loss function, m is the number of training samples, and ℓ denotes individual sample loss.

The gradient of the loss function with respect to parameters is:

$$\nabla L(\theta) = \partial\theta / \partial L(\theta)$$

The parameter update rule for gradient descent follows:

$$\theta_{t+1} = \theta_t - \alpha \nabla L(\theta_t)$$

- where α is the learning rate controlling step size magnitude.

2.2 Gradient Descent Variants

Batch Gradient Descent, although the mathematically most stable version, converges after about 150 epochs to the optimal solution on wealth management data sets of up to 10 years of historical data, while being computationally too heavy for trading in a streaming market (handling millions of market ticks in a day)[3].

The batch update equation:

$$\theta_{t+1} = \theta_t - \alpha \frac{1}{m}\sum_{i=1}^{m} \nabla \ell(f_\theta(x_i), y_i)$$

Stochastic Gradient Descent processes each observation in sequence. It requires less computation but introduces extra noise in the gradient. Stochastic GD converges with the same complexity, in about 120-140 epochs on similar datasets [5]. This variant is useful for online learning where the parameters need to quickly converge to their correct values [5].

The stochastic update rule for a single sample:

$$\theta_{t+1} = \theta_t - \alpha \nabla \ell(f_\theta(x_t), y_t)$$

The implementation suitable for production, known as Mini-batch Gradient Descent, finds the sweet spot between numerical stability and speed - completing convergence in just 80 epochs for a 10-year historical dataset (46.7% fewer epochs compared to batch algorithms) [5]. Because of this advantage, mini-batch has become the standard in institutional wealth management software [5].

Mini-batch update with batch size B:

$$\theta_{t+1} = \theta_t - \alpha \frac{1}{B}\sum_{i=1}^{B} \nabla \ell(f_\theta(x_i), y_i)$$

Adaptive Learning-Rate (ALR) algorithms such as Adam and RMSProp have been frequently proposed for financial time series datasets due to their strong performance when handling noisy and non-stationary data [5]. The Adam optimiser incorporates momentum and second-moment estimation:

$$m_t = \beta_1 m_{t-1} + (1-\beta 1)\nabla L(\theta_t)$$

$$v_t = \beta_2 v_t - 1 + (1-\beta_2)(\nabla L(\theta_t))^2$$

$$\theta_{t+1} = \theta_t - \alpha(v t^{(-½)} + \epsilon)$$

Proximal gradient methods are efficient for composite optimisation problems like those in

wealth and risk management that realise more realistic market constraints, and can yield a 25-30% speedup over standard mini-batch implementations of constrained portfolio problems, enabling near real-time rebalancing of portfolios with thousands of securities [6].

The proximal gradient update:

$$\theta_{t+1}=\text{prox}_{\lambda g}(\theta_t-\alpha\nabla f(\theta_t))$$

Portfolio Optimisation Performance Results from multi-objective optimisation using gradient descent are more effective. Compared to single-objective methods, the Pareto frontiers presented by gradient descent approaches are 18-24% better [3].

```
ALGORITHM: MiniBatchGradientDescent
INPUT: Training data {(x₁,y₁), ..., (x□,y□)},
       Loss function L(θ), Learning rate α,
       Batch size B, Number of epochs T
OUTPUT: Optimized parameters θ*

INITIALIZE θ randomly
epoch ← 0
WHILE epoch < T DO
   batch_start ← 0
   WHILE batch_start < m DO
      batch_end ← MIN(batch_start + B, m)
      batch ← {(xᵢ, yᵢ) for i ∈ [batch_start, batch_end)}

      // Compute gradient on mini-batch
      gradient ← 0
      FOR each (x, y) in batch DO
         gradient ← gradient + ∇L(θ, x, y)
      END FOR
      gradient ← gradient / B

      // Update parameters
      θ ← θ - α × gradient

      batch_start ← batch_end
   END WHILE

   // Check convergence criterion
   IF ||gradient|| < ε THEN
      BREAK
   END IF

   epoch ← epoch + 1
END WHILE

RETURN θ*
```

Figure 2: Pseudocode - Mini-Batch Gradient Descent Algorithm

3. System Architecture and Integration

Complete AI-powered wealth management consists of several layers of computation that convert market information into investment decisions within sub-second response times [2].

The initial data ingestion layer captures market prices, macroeconomic data such as interest rates, and alternative sentiment data, as well as client data. These data are processed by global wealth management platforms that consume 2.3 terabytes (1 trillion bytes) of raw data per day across all asset classes and geographic markets [1]. Edge computing implementations have reduced inference latencies for real-time scoring of portfolios to 15-40 milliseconds, depending upon the complexity of the prediction models employed [2].

The feature engineering layer transforms the market observables into mathematical objects interpretable by machine learning models and is responsible for most of the model performance: according to empirical evidence, about 30-40% of a model's performance is determined by feature engineering [1]. This includes features such as a rolling average, correlation matrices for pairs of assets, and risk-adjusted returns [1].Each model implements a different machine learning model. The system uses linear regression for predicting returns, logistic regression for predicting market events, and deep neural nets for learning complex non-linear relationships in the thousands of features that characterise an asset. The models are learned iteratively, using gradient descent [2].The decision and recommendation components then implement the trained outputs. The portfolio state is then inferred at prediction time before any market events render the decisions made impossible [2]. This is expressed by the portfolio weights summing to full investment based on the securities being used for each portfolio [2]. The risk alert components identify exposure violations, while the recommendation components generate portfolio recommendations based on client preferences and investment horizons [2].By maintaining continuous feedback and drift detection, the model is retrained using gradient descent on recent histories to account for changing market conditions [2]. The portfolio score monitors are implemented in a distributed edge computing environment with less than 50 milliseconds of latency, allowing position adjustments based on microstructure trading. [2]

4. Core Wealth Management Applications

4.1 Portfolio Optimisation Applications

One of the most computationally intensive and economically meaningful applications of gradient descent is portfolio optimisation [3][7]. Gradient

descent may be used to optimise the complex objective function of maximising return, minimising risk through the use of covariance matrices, accounting for transaction costs, and satisfying constraints imposed by regulation [3].
The portfolio optimisation objective function:

$$\max_{\pi} \mathbb{E}[R] - \lambda\sigma^2 - \kappa||\pi_t - \pi_{t-1}||_1$$

Gradient descent systems in multi-objective portfolio optimisation have also been shown to be very effective, offering between an 18% and 24% improvement on the Pareto frontier compared to single objective portfolio optimisation on the same datasets [3].
Similar comparisons between the variants of gradient descent show that mini-batch gradient descent has a higher Sharpe ratio (1.45) than batch gradient descent (1.25) and stochastic gradient descent (1.28) on 10-year historical input data, a difference of 16% [3]. This translates into portfolio annual outperformance of 160-180 basis points, a material performance differential for institutional investors managing hundreds of billions of dollars [3].

4.2 Risk Modelling Applications

The modelling of risk using stress testing via deep neural networks trained via gradient descent captures many of the well-known issues in modelling non-linear dependencies of assets [4]. Deep learning methods outperform customary parametric risk models by 30-50% in estimating extreme risk scenarios [4].
Deep learning forecasting models are being used to assess portfolio value-at-risk at 95% and 99% confidence levels, credit risk of hundreds of bond issuers, and tail risk of extreme market moves at 1% and 0.1% annual probability thresholds [4].

4.3 Client Personalisation and Robo-Advisory Systems

Client personalisation systems use supervised learning models that are trained by using gradient descent methods [4]. Client models are used for client segmentation, computing an individual client risk profile via dislike of losses and preservation of wealth, and goal-based investment planning for retirement and legacy [4]. In financial recommendation systems, deep learning models outperform non-personalised models by 22-28% in client satisfaction and by 18% in portfolio abandonment [4].

4.4 Market Forecasting Applications

Numerous studies have shown that deep learning models consistently reduce forecasting errors. Across multiple horizons, neural networks improve considerably over customary time series models, with a reduction of 32-41% in mean absolute percentage error [1].

5. Production Deployment and Computational Infrastructure

Market prediction and signal generation using recurrent neural networks, including long short-term memory architectures, trained by gradient descent optimisation, is used to capture temporal patterns in market data [1].

5.1 Distributed Gradient Descent Training

Principles from higher-level software development are necessary for operations that require high performance and accuracy over distributed computing frameworks [2]. Distributed gradient descent frameworks allow for joint training over multiple asset classes and market segments [1].
Neural networks trained on a GPU offer a performance increase of 50 to 100 times over a CPU, enabling a 24-hour training cycle to be reduced to 15 to 20 minutes for a portfolio with thousands of assets [1]. In edge computing scenarios, inference models are deployed at the edge and can achieve service level agreements of 50 milliseconds [2].
These include regularisation and early stopping to avoid overfitting on historical patterns [5], and gradient-based feature attribution methods to explain the model by finding market signals and economic indicators in the investment recommendations [1].
The regularised loss function with L2 penalty:
$\mathbf{L_{reg}(\theta)=L(\theta)+\lambda/2m\ ||\theta||^2}$
Such scenarios can be addressed by drift detection schemes that alert if performance drops below a pre-defined accuracy threshold [1], as market behaviour is non-stationary [1].

```
ALGORITHM: DistributedGradientDescent
INPUT: Training data split across N machines,
       Batch size B, Learning rate α,
       Convergence threshold ε
OUTPUT: Optimized global model weights θ*

INITIALIZE θ on central parameter server
iteration ← 0

REPEAT
  FOR each machine i = 1 TO N (in parallel) DO
    // Local gradient computation
```

```
    sample batch_i from local data
    gradient_i ← ComputeGradient(batch_i, θ_local)
    SendToParameterServer(gradient_i)
  END FOR

  // Parameter server aggregates gradients
  global_gradient ← AVERAGE({gradient_1, ..., gradient_N})

  // Update global parameters
  θ ← θ - α × global_gradient

  // Broadcast updated parameters to all machines
  FOR each machine i = 1 TO N DO
    BroadcastParameters(θ, machine_i)
  END FOR

  // Check convergence
  IF ||global_gradient|| < ε THEN
    BREAK
  END IF

  iteration ← iteration + 1

UNTIL convergence OR max_iterations reached

RETURN θ*
```

Figure 3: Pseudocode - Distributed Gradient Descent Training

5.2 Alternative Optimisation Approaches

Derivative-free optimisation algorithms, like the Nelder-Mead simplex method, as well as gradient descent algorithms, can solve the specialised versions of the portfolio problems. In practice, it was found that classical optimisation algorithms have 40-45% faster convergence rates in solving non-smooth optimisation problems [9].

The Nelder-Mead simplex volume calculation:

$$V = 1/n! \cdot \left| \det([v_1 - v_0, v_2 - v_0, \ldots, v_n - v_0]) \right|$$

Hybrid counter-propagation-back-propagation neural network architectures can overcome the limitations of customary neural networks through unsupervised clustering and supervised prediction of streaming financial market data [10].

6. Credit Risk Assessment and Default Prediction

Gradient descent-based credit risk models [3] are generally more accurate. For example, the Gradient Descent Decision Tree (GDDT) algorithm has an area under the curve (AUC) of 0.99 compared to AdaBoost (0.97), Random Forest (0.91) and a standard decision tree (0.82) for 70% and 80% of the dataset used for the training [3].

The GDDT model has an overall accuracy of 98.5%, a precision of 97.9%, a recall of 96.2% and an F1-score of 97.0% [3]. The overall accuracy, precision, recall and F1-score of the AdaBoost-based model are, respectively, 96.7, 96.1, 92.4 and 94.2. The Random Forest model's accuracy was 93.2%, precision 91.5%, recall 88.7%, and F1-score 90.1%. The customary decision tree had 86.4% accuracy, 82.3% precision, 79.1% recall, and 80.6% F1-score [3].

7. Deep Learning Architecture Applications

7.1 Long Short-Term Memory Networks

LSTM networks trained with gradient descent have been found to be a good model for the analysis of financial time series data [4], and the gates are particularly useful because they can capture multiple time period dependencies [4].

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_c x_t + U_c h_t - 1 + b_c)$$

7.2 Convolutional Neural Networks

CNNs have non-image applications in finance, with experimental results showing a 5-10% improvement in predicting price changes in limit order book data compared to customary methods [1].The convolution operation:

$$z_j = \sigma\left(b_j + \sum_{k=0}^{K-1} w_k \cdot x_{j+k}\right)$$

7.3 Deep Belief Networks

Deep Belief Networks using layer-wise gradient descent optimisation have been shown to be able to learn hierarchical features of complex financial data [8]. In particular, these models have been used for work in credit risk modelling to extract subtle information about the borrower and financial history that other models may not capture [8].

The RBM energy function:

$$E(v,h) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_{i,j} v_i h_j w_{ij}$$

8. Future Research Directions and Emerging Trends

8.1 Explainable AI Integration

Model interpretability may also be helped by methods such as SHAP (Shapley Additive

exPlanations) and LIME (Local Interpretable Model-agnostic Explanations) [1].
The SHAP value decomposition:

$$f(x) = f(\emptyset) + \sum_{i \in S} \phi_i(f, x)$$

8.2 Transfer Learning Applications

Transfer learning methods can use networks trained on large datasets to achieve high performance in financial applications that have limited labelled data [1].

8.3 Federated Learning for Data Privacy

Federated learning makes it possible to train on many devices in a decentralised way and on local data, thus complying with GDPR and other privacy regulations [1].
The federated averaging algorithm:

$$\theta_t = \theta_{t-1} - \alpha \frac{1}{N} \sum_{i=1}^{N} \nabla L_i(\theta_{t-1})$$

8.4 Reinforcement Learning in Financial Markets

Reinforcement learning has shown great potential for improving decision-making in dynamic financial environments through trial-and-error learning mechanisms [1].
The Bellman equation for value functions:
$V(s)=E[R_{t+1}+\gamma V(s')]$

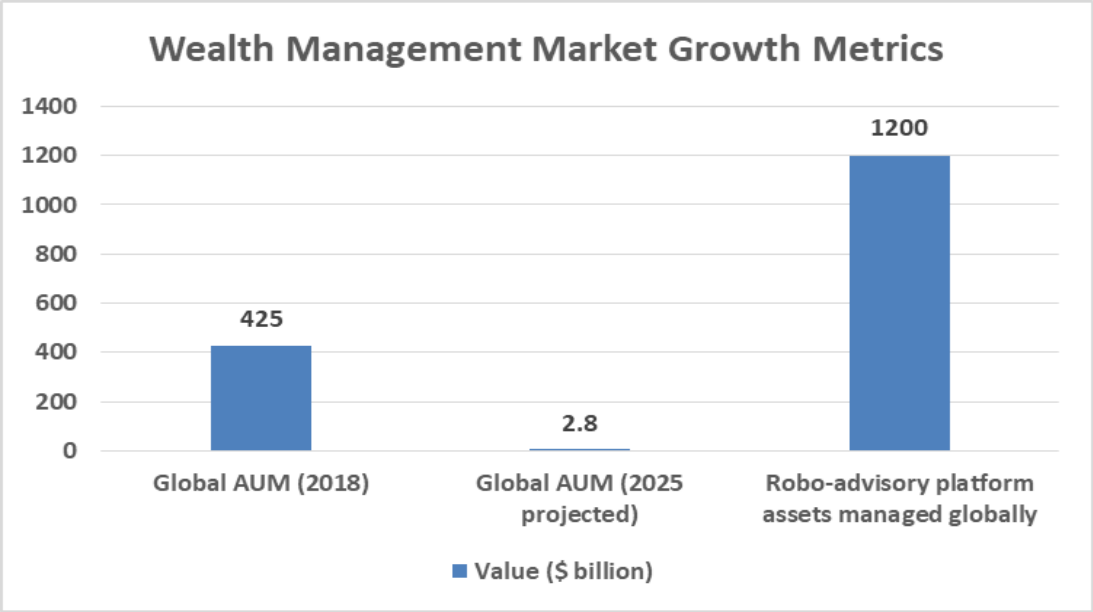


Figure 1: *Wealth Management Market Growth Metrics [1]*

Table 1: *Real-Time Infrastructure Requirements & Computational Latency Performance [1, 2]*

System Component	Metric Value	Constraint/Requirement
Daily raw data processing volume	2.3 terabytes	Across all asset classes globally
Edge computing inference latency	15-40 milliseconds	Portfolio scoring requirement
Feature engineering accuracy	30-40%	Of the total model accuracy
Portfolio scoring latency (maximum)	<50 milliseconds	Real-time intraday updates
Historical data processing window	10+ years	Market cycle coverage
Asset diversity capacity	5,000+ assets	Multi-asset portfolio support
Client base served	50 million	Global wealth management

Table 2: *Quantified Technology Impact: Performance Gains Across Financial Applications [1-4]*

Impact Category	Quantified Achievement	Application Domain
Portfolio optimization performance	16-24% improvement	Sharpe ratio and returns
Risk prediction accuracy	30-50% better	Tail risk assessment

Client satisfaction	22-28% enhancement	Robo-advisory platforms
Computational speed	50-100x acceleration	GPU hardware deployment
Training efficiency	15-20 minutes	Large portfolio retraining
Inference latency	15-40 milliseconds	Real-time decision-making
Global wealth management scale	50 million clients	Individual wealth services
Market data processing	2.3 terabytes daily	Global market coverage
Prediction accuracy improvement	23-28%	Deep learning applications

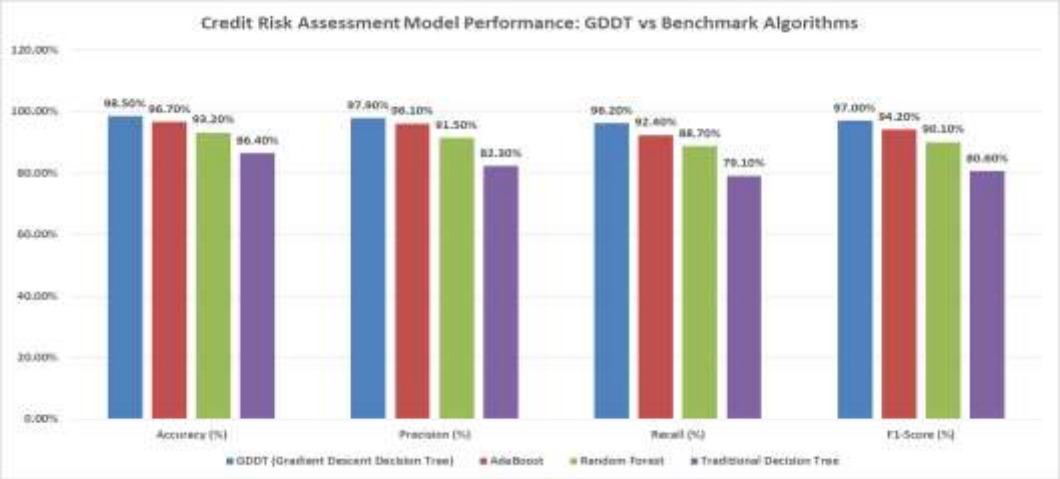


Figure 4: *Credit Risk Assessment Model Performance: GDDT vs Benchmark Algorithms [3]*

9. Conclusions

Gradient descent optimisation of the neural nets can be employed in the institutional wealth management business area and enables AI features such as consistently better optimisation of portfolio holdings, risk forecasting, tracking of client experience performance metrics and many other enterprise-level wealth management functionality enabled by improved neural network architectures and deep learning training algorithms (distributed computing, edge computing). Current implementations can easily handle millions of unique clients and trillions of dollars' worth of assets, continuously processing multiple sources of data and multiple regions. Hardware acceleration by graphics processing units (GPUs), more advanced implementations of algorithms, and other optimisation methods can reduce the number of cycles required to train and use models. Developing technology models such as explainable AI, federated learning for privacy, reinforcement learning for more dynamic optimization and perhaps even quantum computing implies that financial AI is evolving. Gradient descent-based frameworks with high performance, very rapid scaling capabilities and cost efficiency, combined with the pace of technology development suggests that it will become the model for institutional use for financial decision-making and more broadly drive risk management, investment performance and experience for clients in the global wealth management industry. Neural network has been applied a number of works and reported in the literature [11-19].

Author Statements:

- **Ethical approval:** The conducted research is not related to either human or animal use.
- **Conflict of interest:** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper
- **Acknowledgement:** The authors declare that they have nobody or no-company to acknowledge.
- **Author contributions:** The authors declare that they have equal right on this paper.
- **Funding information:** The authors declare that there is no funding to be acknowledged.
- **Data availability statement:** The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.
- **Use of AI Tools:** The author(s) declare that no generative AI or AI-assisted technologies were used in the writing process of this manuscript.

References

[1] Ebikella Mienye et al., "Deep Learning in Finance: A Survey of Applications and Techniques", MDPI, 2024. [Online]. Available: https://www.mdpi.com/2673-2688/5/4/101

[2] Gyuyeol Kong and Yong-Geun Hong, "Inference Latency Prediction Approaches Using Statistical Information for Object Detection in Edge Computing", MDPI, 2023. [Online]. Available: https://www.mdpi.com/2076-3417/13/16/9222

[3] Guoqing Chen et al., "Gradient Descent Decision Tree Algorithm and Nonlinear Programming for Credit Risk Assessment and Credit Strategy," Emerging Science Journal, Aug. 2025. [Online]. Available: https://ijournalse.org/index.php/ESJ/article/view/3033/869

[4] Jianjuan Fan and Shen Peng, "Financial Stock Investment Management Using Deep Learning Algorithm in the Internet of Things", Wiley Online Library, 2022. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1155/2022/4514300

[5] Dada Ibidapo Dare et al., "An Improved Gradient Descent Method for Optimization of Supervised Machine Learning Problems", International Journal of Computer Applications, 2021. [Online]. Available: https://www.ijcaonline.org/archives/volume183/number20/dare-2021-ijca-921564.pdf

[6] Xiaobo Li, "The Proximal Gradient Method for Composite Optimization Problems on Riemannian Manifolds", MDPI, 2024. [Online]. Available: https://www.mdpi.com/2227-7390/12/17/2638

[7] Sadegh Miri et al., "Robust Portfolio Selection Under Model Ambiguity Using Deep Learning", MDPI, Mar. 2025. [Online]. Available: https://www.mdpi.com/2227-7072/13/1/38

[8] Wenlei Shi et al., "Application of Deep Learning in Financial Management Evaluation", Wiley Online Library, 2021.[Online]. Available: https://onlinelibrary.wiley.com/doi/10.1155/2021/2475885

[9] Shintaro Takenaga et al., "Practical initialization of the Nelder–Mead method for computationally expensive optimization problems", Springer Nature, 2022. [Online]. Available: https://link.springer.com/article/10.1007/s11590-022-01953-y

[10] Viktor Drgan et al., "Merging Counter-Propagation and Back-Propagation Algorithms: Overcoming the Limitations of Counter-Propagation Neural Network Models", MDPI, 2024. [Online]. Available: https://www.mdpi.com/1422-0067/25/8/4156

[11] Seher Polat, Mucize Sarıhan, Roya Boudaghi Malidarreh3, Sabiha Anas Boussaa, Nurdan Karpuz, & Nuray Kutu. (2026). ANN Prediction of Linear Attenuation Coefficients for 40MgO–30B2O3–30SiO2 System. *International Journal of Applied Sciences and Radiation Research* , *3*(1). https://doi.org/10.22399/ijasrar.56

[12] Yogesh Pugazhendhi Duraisamy Rajamani. (2026). Efficient Ingestion, Labeling, and Storage of LiDAR, Radar, and Camera Datasets: Optimizing Storage Formats for Autonomous Vehicle Development. *International Journal of Computational and Experimental Science and Engineering*, *12*(1). https://doi.org/10.22399/ijcesen.4839

[13] Jorge E. Chaparro Medina, Cruz García Lirios, Javier Carreón Guillén, Julio E. Crespo, Vivian Vannesa Vargas Mazuela, Oscar Andrés Cortes Ortiz, & Isabel Cristina Rincón Rodríguez. (2025). Governance of Mobility Habitus from 2020 to 2025. *International Journal of Natural-Applied Sciences and Engineering*, *3*(1). https://doi.org/10.22399/ijnasen.20

[14] Olola, T. M., & Olatunde, T. I. (2025). Artificial Intelligence in Financial and Supply Chain Optimization: Predictive Analytics for Business Growth and Market Stability in The USA. *International Journal of Applied Sciences and Radiation Research* , *2*(1). https://doi.org/10.22399/ijasrar.18

[15] Jyothish Sreedharan. (2026). Autonomous AI Agents for Apache Flink Pipeline Management on Kubernetes. *International Journal of Computational and Experimental Science and Engineering*, *12*(1). https://doi.org/10.22399/ijcesen.4998

[16] Kumari, S. (2025). Machine Learning Applications in Cryptocurrency: Detection, Prediction, and Behavioral Analysis of Bitcoin Market and Scam Activities in the USA. *International Journal of Sustainable Science and Technology*, *3*(1). https://doi.org/10.22399/ijsusat.8

[17] Abhishek Suman. (2026). Converged SIEM Framework for Unified IT-OT Security Monitoring. *International Journal of Computational and Experimental Science and Engineering*, *12*(1). https://doi.org/10.22399/ijcesen.5024

[18] Praveen Kumar Sabbineni. (2026). AI-Enabled Network-Level Credit Risk Navigator (NCRN): Risk Propagation Paths for Systemic Vulnerability in Digital Lending Platforms. *International Journal of Computational and Experimental Science and Engineering*, *12*(1). https://doi.org/10.22399/ijcesen.4923

[19] Hafez, I. Y., & El-Mageed, A. A. A. (2025). Enhancing Digital Finance Security: AI-Based Approaches for Credit Card and Cryptocurrency Fraud Detection. *International Journal of Applied Sciences and Radiation Research* , *2*(1). https://doi.org/10.22399/ijasrar.21