University of Warsaw
Faculty of Economic Sciences

Working Papers
No. 22/2025 (485)

A Survey of Statistical Arbitrage Pair Trading with Machine Learning, Deep Learning, and Reinforcement Learning Methods

Yufei Sun

Warsaw 2025

ISSN 2957-0506

WORKING PAPERS 22/2025 (485)

UNIVERSITY OF WARSAW
FACULTY OF ECONOMIC SCIENCES
WORKING PAPERS

A survey of statistical arbitrage pair trading with machine learning, deep learning, and reinforcement learning methods

Yufei Sun[1] *

[1] *Faculty of Economic Sciences, University of Warsaw*
**Corresponding authors: y.sun2@uw.edu.pl*

Abstract: Pair trading remains a cornerstone strategy in quantitative finance, having consistently attracted scholarly attention from both economists and computer scientists. Over recent decades, research has expanded beyond traditional linear frameworks—such as regression- and cointegration-based models—to embrace advanced methodologies, including machine learning (ML), deep learning (DL), reinforcement learning (RL), and deep reinforcement learning (DRL). These techniques have demonstrated superior capacity to capture nonlinear dependencies and complex dynamics in financial data, thereby enhancing predictive performance and strategy design.
Building on these academic developments, practitioners are increasingly deploying DL models to forecast asset price movements and volatility in equity and foreign exchange markets, leveraging the advantages of artificial intelligence (AI) for trading. In parallel, DRL has gained prominence in algorithmic trading, where agents can autonomously learn optimal trading policies by interacting with market environments, enabling systems that move beyond price prediction to dynamic signal generation and portfolio allocation.
This paper provides a comprehensive survey of ML-, DL-, RL-, and DRL-based approaches to pair trading within quantitative finance. By systematically reviewing existing studies and highlighting their methodological contributions, it offers researchers a structured foundation for replication and further development. In addition, the paper outlines promising avenues for future research that extend the application of AI-driven methods in statistical arbitrage and market microstructure analysis.

Keywords: Pair Trading, Machine Learning, Deep Learning, Reinforcement Learning, Deep Reinforcement Learning, Artificial Intelligence, Quantitative Trading

JEL codes: C4, C45, C55, C65, G11

1. Introduction

Stock market forecasting concerns the prediction of future values of company shares or other financial assets traded on exchanges. While accurate forecasts can generate substantial financial gains, the task remains inherently complex. According to the Efficient Market Hypothesis (EMH), stock prices incorporate all publicly available information, implying that future movements cannot be reliably inferred from historical prices alone, as they are primarily driven by new information. Consequently, models that depend exclusively on past prices tend to have limited predictive power, especially given markets' high sensitivity to external shocks.

The proliferation of the Internet has introduced alternative sources of collective intelligence—such as Google Trends and Wikipedia usage—that may capture shifts in investor attention and sentiment. This has led to the view that prices are shaped not only by financial news but also by real-time public interest. Hence, a central question arises: to what extent can historical price data, possibly enhanced with such alternative sources, be used to improve forecasting accuracy?

Early investigations into the predictability of stock markets were deeply rooted in theories like Early investigations into market predictability were grounded in the EMH and the random walk hypothesis, which posit that prices adjust randomly to new information and therefore cannot be systematically predicted from past data. Under these frameworks, forecasting stock returns was regarded as no better than random guessing. Nevertheless, subsequent research has increasingly challenged these assumptions, suggesting that financial markets may display patterns that enable partial predictability. The sustained outperformance of certain investors, such as Warren Buffett, is often cited as suggestive evidence that markets are not perfectly efficient.

Developing robust forecasting models, however, remains difficult because asset prices are influenced by a wide range of factors, including firm fundamentals, macroeconomic conditions, market sentiment, and historical dynamics. Models based on a single predictor typically fail to capture this complexity, whereas incorporating multiple features—such as news sentiment, social media activity, and technical indicators—can improve predictive performance by reflecting the multifaceted drivers of market behavior.

Figure 1 illustrates the major domains of stock trading where AI techniques are increasingly applied. Quantitative trading employs mathematical and statistical models to exploit market inefficiencies. Algorithmic trading automates order execution according to predefined rules such as price or timing. High-Frequency Trading (HFT), a specialized subset of algorithmic trading, uses advanced algorithms to execute large volumes of trades within milliseconds. Automated Trading Systems (ATS) further extend algorithmic trading by directly routing orders to exchanges. Although these trading paradigms predate AI, recent developments have increasingly integrated AI methods to enhance pattern recognition, predictive modeling, and execution efficiency.

Figure 1. Utilization of Artificial Intelligence Technologies in Quantitative Finance and Equity Markets Trading.

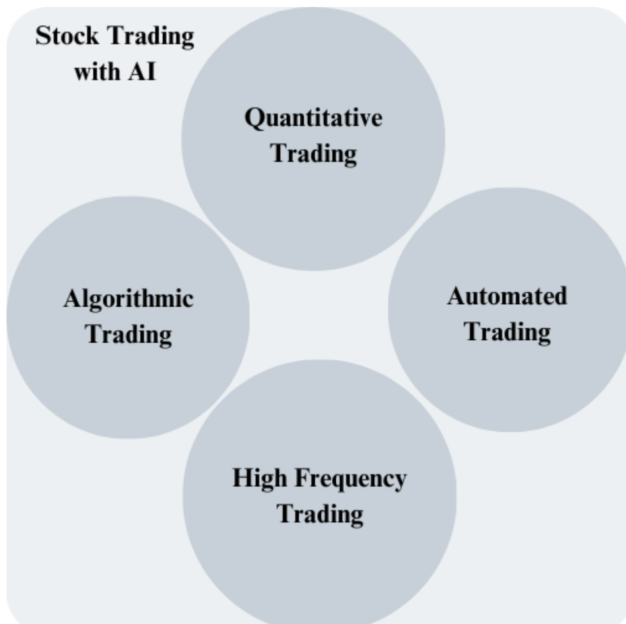


Figure 2 highlights major algorithmic trading strategies, ranging from trend-following and momentum trading to mean reversion, moving average crossovers, breakout strategies, and statistical arbitrage. These approaches reflect different assumptions about market behavior: some exploit the continuation of price trends, others rely on reversals to long-run equilibria, and still others focus on technical thresholds or temporary mispricings between related assets. Among them, statistical arbitrage—particularly pairs trading—has become one of the most prominent applications, employing tools such as cointegration and regression analysis to detect and exploit relative value opportunities.

While these strategies predate modern artificial intelligence, recent advances in machine learning and deep learning have enhanced their performance by enabling more sophisticated feature extraction, adaptive parameter tuning, and improved execution. In this sense, AI does not redefine these strategies but augments their predictive power and adaptability.

Figure 2. Diverse Approaches to AI-Enhanced Algorithmic Trading Strategies.

Among statistical arbitrage approaches, pairs trading has emerged as one of the most widely studied and implemented strategies. The basic idea is to identify pairs of assets whose prices have historically moved together and to exploit temporary deviations in their relative valuation. When the spread between two assets diverges beyond a predefined threshold, traders take offsetting positions with the expectation that the spread will revert to its long-run equilibrium. Despite its intuitive appeal, the strategy involves several practical challenges, including robust pair selection, optimal threshold determination, and accurate timing of trade execution.

Recent advances in ML, DL, and RL have the potential to substantially enhance pairs trading. ML algorithms can process large-scale datasets to uncover nonlinear dependencies and hidden correlations between securities that are often overlooked by traditional statistical methods. Both supervised and unsupervised learning have been applied, with the former forecasting spread dynamics and the latter clustering securities into candidate pairs. DL architectures, particularly recurrent neural networks (RNNs) and long short-term memory (LSTM) networks, are well-suited for modeling temporal dependencies and nonlinear spread behavior, while more recent approaches explore convolutional and transformer-based models. RL offers a different paradigm, focusing on sequential decision-making through interaction with the market environment. RL agents dynamically adjust trading policies by optimizing cumulative rewards, allowing adaptive refinement of entry and exit rules under evolving market conditions.

Overall, these advanced AI-driven techniques promise to improve the robustness and adaptability of pairs trading strategies. By enabling models that learn from data and adjust in real time, ML, DL, and RL provide new opportunities for capturing complex dynamics in increasingly data-rich and competitive financial markets. This motivates a growing body of literature that applies AI methods to pairs trading, which the present survey systematically reviews.

The primary objective of this literature review is to systematically examine and synthesize approximately 70 academic papers from the past decade that explore the use of ML, DL, and RL in the context of pair trading strategies. Specifically, this study aims to: 1. identify the current state of research by mapping out methodologies and techniques employed in AI-driven pair trading; 2. evaluate the effectiveness, efficiency, and scalability of various algorithmic and computational frameworks; 3. highlight recent innovations that improve upon traditional statistical arbitrage models, particularly those leveraging time-series forecasting, feature learning, or decision optimization; 4. analyze the performance metrics used to assess trading strategies, including risk-adjusted returns and prediction accuracy; 5. examine key challenges such as data limitations, computational demands, and model overfitting; 6. recommend best practices for designing and implementing robust AI-based pair trading systems; and 7. propose future research directions by identifying unresolved issues and emerging trends within this evolving field.

This survey makes several principal contributions. First, it reviews recent advancements in AI, with particular attention to innovations from the past decade. Second, it critically examines how ML, DL, and RL have been applied to pairs trading and stock market forecasting. Third, it situates these developments within the broader context of financial market mechanisms, drawing upon existing scholarly work. Fourth, it outlines future research directions in stock market prediction, providing guidance for emerging scholars. Finally, the survey highlights potential data sources that can be leveraged for further investigation.

The structure of the remainder of this document is organized as follows: Section 2 elaborates on the research works that inform this study. Section 3 provides a concise introduction to the scope of this survey. Section 4 discusses the methodologies for data processing and feature extraction.

In Section 5, we offer a detailed examination of the various forecasting techniques, enriched with relevant context. Section 6 focuses on the metrics used to assess prediction accuracy. The framework for implementation and the availability of data are discussed in Section 7. Section 8 envisions the prospective avenues for further inquiry in this field. The paper concludes with a summary in Section 9, encapsulating the essence of the study, and final thoughts are presented in Section 10.

2. Related Work

This section reviews the evolution of ML methodologies applied to pairs trading, illustrating how computational approaches have progressively transformed strategy design and implementation. Early studies predominantly relied on unsupervised learning techniques—such as clustering and cointegration analysis—to identify candidate trading pairs based on historical co-movements. While effective in capturing broad similarities, these approaches were limited by their inability to incorporate predictive signals or adapt to changing market dynamics. The subsequent introduction of supervised learning methods addressed some of these shortcomings by leveraging labeled financial data to refine pair selection and improve the timing of trade execution. With the advent of DL, researchers further advanced pairs trading by modeling nonlinear dependencies and temporal structures in financial time series, employing architectures such as recurrent and convolutional neural networks. More recently, RL has emerged as a promising paradigm, enabling adaptive strategy optimization through continuous interaction with market environments and dynamic adjustment of entry–exit rules.

Collectively, these methodological advances highlight the central role of ML in enhancing the robustness, adaptability, and profitability of pairs trading strategies, while also signaling a broader transition toward data-driven decision-making in financial markets.

2.1 Unsupervised Learning Methods

Unsupervised learning methods uncover latent patterns and structural relationships in data without relying on labeled outcomes. In financial markets, these techniques are particularly useful for detecting correlations and dependencies that arise naturally across assets, making them well suited for identifying candidate pairs in statistical arbitrage strategies.

The selection of studies reviewed here follows three main criteria: (i) a direct focus on the application of unsupervised learning to financial trading, with particular emphasis on pairs trading or statistical arbitrage; (ii) empirical validation using real-world market data to ensure practical relevance; and (iii) methodological novelty, especially where machine learning (ML) techniques enhance or replace conventional statistical approaches in pair selection.

Chen et al. (2012) Chen et al. (2012) investigate the application of machine learning techniques to enhance pairs trading, a statistical arbitrage strategy that exploits mean-reverting spreads between correlated assets. The study compares two modeling approaches: a conventional portfolio rebalancing and linear regression framework, and a more sophisticated method that integrates Kalman filtering with the Expectation–Maximization (EM) algorithm. Using data from the China futures market, the empirical results indicate that while both approaches show potential, the Kalman filter–based model delivers superior profitability. This work illustrates the benefits of incorporating adaptive, data-driven methods into traditional arbitrage frameworks, particularly in capturing dynamic co-movement between assets.

Sohail et al. (2020) apply the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm in combination with conventional pairs trading techniques to identify structurally similar stocks using firm size (market capitalization) and principal component analysis (PCA) of daily returns. Using data from the Pakistan Stock Exchange (PSX), the study demonstrates that this ML-enhanced approach generates statistically significant average monthly excess returns while preserving market neutrality and confirming mean-reversion dynamics. The findings highlight the potential of unsupervised clustering methods to improve pair selection, particularly within the context of emerging markets.

In a follow-up study, Sohail et al. (2022) examine the robustness of pairs trading during the COVID-19 pandemic—a period of heightened volatility and uncertainty. Using DBSCAN to form trading pairs from market and accounting features, the authors evaluate the continued viability of statistical arbitrage under disrupted conditions. They report that unsupervised-learning–guided pairs trading remains profitable and market-neutral, with spreads exhibiting mean reversion in adverse environments. Results are, however, sensitive to clustering hyperparameters (ε, minPts) and feature scaling; robustness is typically assessed through rolling windows, regime-specific subsamples, and multiple-testing adjustments such as the false discovery rate.

Han et al. (2023) move beyond pair selection based solely on return co-movements or cointegration by integrating firm-specific characteristics with price dynamics. Applied to U.S. equities (1980–2020), their framework identifies pairs with stronger co-movement propensity and more pronounced mean-reverting behavior, delivering robust out-of-sample performance across regimes. Mean-reversion strength is quantified via Ornstein-Uhlenbeck (OU) half-life and Hurst exponent, while performance evaluation accounts for transaction costs and employs White's reality check or the superior predictive ability test to mitigate data-snooping bias.

In summary, unsupervised learning has become increasingly influential in advancing pairs trading. By uncovering latent structure in high-dimensional data, approaches such as Kalman filtering, DBSCAN clustering, and firm-level feature integration enable more informative pair selection and more resilient arbitrage execution, particularly in volatile or non-stationary markets.

2.2 Supervised Learning Methods

This section examines supervised learning methods, where models are trained on historical datasets with labeled outcomes to predict future events. Such techniques are increasingly applied to pair trading, where accurate forecasting is critical for optimizing entry and exit decisions. By learning from past market behavior, supervised models can uncover patterns that inform both the timing and selection of trading pairs, thereby improving the precision and potential profitability of arbitrage strategies.

In the context of pair trading, supervised learning has been instrumental in developing predictive models that assess the likelihood of mean reversion and estimate the expected duration of spread convergence. Recent studies demonstrate that algorithms such as decision trees, support vector machines, and neural networks can capture complex, nonlinear relationships within financial time series. These models not only contribute to improved trading performance but also offer enhanced robustness under dynamic and volatile market conditions. Nonetheless, challenges remain—particularly regarding the construction of appropriate labels (e.g., defining convergence events) and addressing class imbalance in financial datasets—which require careful methodological design.

This subsection reviews key studies that apply supervised learning techniques to enhance pairs trading strategies. Madhavaram (2013) introduces an innovative framework that integrates dynamic PCA with Support Vector Machines (SVM) for financial market analysis. Applied to the Financial Select Sector SPDR Fund (XLF), PCA is employed to extract systematic risk factors, while SVM models classify and validate potential trading signals. The findings suggest that combining SVM with factor-based analysis can improve decision-making, highlighting the potential synergy between traditional quantitative finance and machine learning in algorithmic trading. However, the approach remains constrained by its sector-specific focus, leaving open questions about scalability and generalizability to other markets.

Nóbrega and Oliveira (2013) propose an intraday statistical-arbitrage framework that integrates Extreme Learning Machine (ELM) and Support Vector Regression (SVR) with classical linear regression. Applied to a financial-sector universe, the approach is further refined with a Kalman filter to dynamically update hedge ratios and smooth predictions. Empirical evidence suggests that such hybrid models improve pair selection and signal stability, leading to measurable gains in trading performance, though results remain sensitive to hyperparameters and intraday microstructure noise.

In a subsequent study, Nóbrega and Oliveira (2014) extend their earlier work by integrating ELM and SVR with Kalman Filter regression. Their findings confirm that such hybrid models deliver more accurate forecasts, enhancing annualized returns while simultaneously reducing portfolio volatility—further underscoring the benefits of combining statistical models with ML algorithms in financial trading.

Wu (2015) introduces a novel spread-based approach to pairs trading by applying the OU model in conjunction with SVM to trade GOOG/GOOGL stocks. The innovation lies in the incorporation of technical indicator spreads and the development of two new metrics designed to predict future prices rather than returns. Empirical results demonstrate a high win rate, highlighting the practical viability of this predictive framework.

Chaudhuri et al. (2017) examine the predictive capacity of multiple ML algorithms—including SVM, Random Forest (RF), and Adaptive Neuro-Fuzzy Inference System (ANFIS)—for forecasting the price ratio of stock pairs. Drawing on nine selected input features, their results show that these models effectively capture price dynamics, thereby illustrating the ability of ML techniques to refine trading decisions and improve financial forecasting accuracy.

Sutherland et al. (2018) apply a range of ML models—including logistic regression (LR), RF, deep neural networks (DNN), and gradient-boosted trees (GBT)—to statistical arbitrage on the KOSPI 200 index. Their comparative analysis reveals that all models outperform the benchmark, with classification-based approaches slightly surpassing regression-based ones. These results emphasize the capacity of ML to generate profitable trading signals in emerging markets.

Sun et al. (2019) extend ML applications to the cryptocurrency domain, employing RF algorithms combined with Alpha101 features to predict price movements using data from Binance and Bitfinex. The proposed strategy exhibits strong predictive performance in a highly volatile environment, demonstrating the adaptability of ML approaches beyond traditional equity markets.

Finally, Hong and Hwang (2023) assess the contribution of firm fundamentals to the robustness of pair selection. Addressing the problem of spurious pair identification through multiple hypothesis testing, they find that greater similarity in firm characteristics improves trading performance and reduces non-convergence risk. Their results suggest that integrating firm-level information enhances strategy stability, particularly during crisis periods when fundamentals play a more prominent role.

In summary, the reviewed studies highlight the substantial potential of supervised learning in advancing pairs trading. By leveraging labeled historical data, these models refine trade selection and timing, thereby optimizing entry and exit decisions. Core methodologies—including dynamic PCA, SVM, ELM, SVR, Kalman Filters, RF, and ANFIS—consistently demonstrate improved predictive accuracy and profitability across diverse asset classes. Moreover, the integration of firm characteristics provides an effective remedy for spurious pairings, offering more stable and reliable frameworks in volatile or crisis-prone markets.

2.3 Deep Learning Methods

This review next turns to DL methods—a specialized subset of ML that employs DNNs to capture complex, nonlinear patterns in large-scale datasets. In the context of pair trading, DL has emerged as a transformative tool, offering refined insights into market dynamics and enabling the development of more adaptive and sophisticated trading algorithms.

For clarity, DRL is not treated here as an isolated topic but rather as the intersection of DL and RL. RL refers to a class of ML methods in which an agent interacts with an environment and learns optimal actions through trial and error, guided by rewards and penalties to maximize cumulative returns. DL, in contrast, leverages multilayer neural networks to automatically extract high-level features from raw inputs, and has demonstrated strong capabilities in domains such as computer vision and natural language processing.

DRL combines the representational power of DL with the sequential decision-making ability of RL by using neural networks to approximate value functions, policies, or transition models. Prominent DRL algorithms such as Deep Q-Networks (DQN), Deep Deterministic Policy Gradient (DDPG), and Proximal Policy Optimization (PPO) exemplify this integration. This allows DRL to handle high-dimensional input spaces—such as multivariate time series in financial markets—supporting end-to-end learning directly from raw data.

In financial applications, including pairs trading, DRL provides the flexibility to adaptively optimize trading policies in dynamic and uncertain environments where traditional RL methods face scalability constraints. Although applications of DRL to pairs trading remain relatively limited, its ability to integrate feature extraction and policy learning presents a promising frontier for future research.

Krauss et al. (2017) investigate the application of advanced ML models—including DNNs, GBTs, and RFs—to the S&P 500 index. By predicting the probability of individual stocks outperforming the market, the study demonstrates that these models, particularly when combined in ensemble form, generate substantial out-of-sample returns. Their findings challenge the semi-strong form of market efficiency and underscore the benefits of combining ML algorithms for robust predictive modeling.

Ruxanda and Opincariu (2018) propose a sophisticated framework that integrates Bayesian neural networks (BNNs) with Dirichlet process mixtures to model pairs trading strategies. This hierarchical model, with priors derived from a Dirichlet process mixture, enables dynamic adaptation to non-stationarity in financial data, thereby enhancing both the flexibility and predictive power of pairs trading. Through empirical application, the study illustrates the model's capacity to capture complex relationships between financial assets, offering a more nuanced and effective tool for navigating dynamic markets.

The paper by Brim (2019) employs DQN to improve pairs trading strategies, testing on 38 cointegrated stock pairs. Leveraging RL to exploit mean reversion in stock spreads, the study demonstrates that DQNs consistently generate positive returns. By training on 2014–2017 data and testing on 2018 data, the results highlight the adaptability of DRL in financial markets and its ability to learn from complex dynamics for profit maximization.

Huang et al. (2020) develop a hybrid DL model to detect structural breaks in stock markets, with a focus on pairs trading. The approach integrates Wavelet Transform for frequency-domain feature extraction with a combined CNN–LSTM architecture for time-domain analysis. Applied to minute-scale data from the Taiwan Stock Exchange, the model significantly outperforms traditional techniques, illustrating the potential of DL in identifying structural breaks and enhancing arbitrage strategies.

The paper by Flori and Regoli (2021) explores the application of DL, specifically LSTM networks, to identify pairs-trading opportunities in the stock market. The authors focus on the reversal effect, where market deviations are expected to correct over time, offering profitable trading signals. By comparing and combining LSTM predictions with traditional trading practices based on price or returns gaps, the study aims to improve portfolio performance under various investment scenarios. The analysis confirms that strategies incorporating LSTM predictions can enhance financial performance, providing valuable signals beyond those captured by price and returns gaps.

Relan (2021) investigates the predictive power of the USD/GBP exchange rate on the FTSE100 index using a Temporal Convolutional Network (TCN), achieving an accuracy of 89.96%. The research spans from December 31, 1985, to October 6, 2021, employing 13,028 data points to develop a ML model and subsequently testing two trading strategies: pairs trading using Bollinger Bands and a buy and hold strategy. The pairs trading strategy notably outperformed the buy and hold strategy in terms of Annual Average Return and Sharpe Ratio, despite a higher Maximum Drawdown, indicating a successful application of ML techniques in predicting stock market movements based on currency exchange rates.

Du (2022) delves into mean-variance portfolio optimization leveraging DL forecasts for stocks exhibiting cointegration, showcasing a novel approach in financial market analysis. This research pioneers the integration of advanced DL models to enhance forecast accuracy and subsequently inform portfolio allocation decisions. By focusing on cointegrated stocks, the study not only adheres to the classical portfolio theory but also embraces modern computational techniques, marking a significant stride in the application of AI in finance. The methodology's strength lies in its ability to dynamically adjust to market conditions, potentially offering superior returns on investment.

The paper by Platania et al. (2023) presents an innovative approach to pair trading by integrating multi-objective programming, cyclical insights, and neural networks. This strategy aims to exploit market inefficiencies through statistical arbitrage, enhancing the traditional pair trading framework by incorporating a broader range of analytical tools. By analyzing cyclical behaviors and employing neural networks for predictive accuracy, the approach seeks to optimize trading performance by balancing profitability with risk management.

In summary, these studies underscore the transformative potential of DL and DRL in refining pairs trading and statistical arbitrage. By leveraging architectures such as DNNs, CNNs, LSTMs, TCNs, and DQNs, these approaches capture complex temporal patterns, detect structural changes, and adaptively optimize trading policies. Collectively, the evidence highlights the growing role of DL in improving profitability, enhancing adaptability, and challenging long-standing assumptions of market efficiency in high-dimensional and dynamic environments.

2.4 Reinforcement Learning Methods

This section reviews the application of RL methods in pairs trading, where agents learn optimal decision-making policies through interactions with a dynamic environment. RL's sequential learning framework allows trading agents to adaptively adjust their strategies over time, balancing the trade-offs between exploration and exploitation to maximize cumulative returns.

Fallahpour et al. (2016) propose an innovative approach to optimizing pairs trading strategies through RL. By dynamically adjusting trading parameters in response to market conditions, the study demonstrates substantial improvements in performance compared with traditional methods. Incorporating both cointegration and RL, the research provides a novel perspective on maximizing profits and managing risks in pairs trading, with empirical validation using S&P 500 constituent stocks.

Brim (2020) applies Double Deep Q-Networks (DDQNs) to pairs trading, leveraging the mean-reversion property of stock prices for profit generation. Training the DDQN model on historical stock-pair data, the study shows its capacity to predict profitable trading signals, emphasizing the utility of RL in financial applications. Notably, the research introduces a Negative Rewards Multiplier (NRM) to regulate the model's risk-taking behavior, marking a significant advance in applying DRL to complex market dynamics.

Sermpinis et al. (2020) present a pioneering Cointegration Approach–Deep Reinforcement Learning (CA-DRL) framework for pairs trading in commodities markets. The framework combines cointegration-based pair selection with DRL for trade execution, achieving superior returns compared with traditional strategies while maintaining similar risk levels. A genetic algorithm is further incorporated to optimize trading thresholds, providing incremental improvements and highlighting the advantages of integrating advanced ML techniques with traditional financial methods.

Zong et al. (2021) extend the CA-DRL framework to pairs trading on the Dalian Commodity Exchange (DCE), focusing on soybean futures and their derivatives. Their results demonstrate that CA-DRL consistently outperforms traditional models across different pair-formation horizons (one-, two-, or three-year periods), underscoring its effectiveness in designing profitable commodity trading strategies.

Kim et al. (2022) introduce HDRL-Trader, a hybrid DRL framework that optimizes both trading actions and stop-loss boundaries through two independent RL networks. By incorporating dimensionality reduction, clustering, regression, and behavior cloning, HDRL-Trader significantly outperforms state-of-the-art benchmarks on the S&P 500 index, delivering a 25.7% higher average return. This framework exemplifies the benefits of integrating TD3 and DDQN algorithms, offering a promising direction for algorithmic trading.

Lu et al. (2022) propose a two-phase ML framework, Structural Adjustment and Policy Testing (SAPT), designed to enhance pairs trading strategies by integrating structural break detection. The first phase employs a hybrid model to identify structural breaks, while the second phase optimizes the trading strategy, accounting for transaction costs and market-closing risks. Empirical tests in the Taiwan stock market show that SAPT significantly outperforms existing strategies, highlighting the value of incorporating structural shifts into trading models for improved profitability and risk control.

Xu and Luo (2023) develop an enhanced pairs trading strategy using a two-level RL framework. Pair selection is conducted via the Extended Option-Critic (EOC) method, while trade thresholds are optimized using a Multi-Agent Deep Deterministic Policy Gradient (MADDPG) approach. Applied to the Chinese futures market, this integrated framework achieves superior returns compared with traditional methods by dynamically selecting trading pairs and optimizing trade thresholds. The study demonstrates the potential of DRL to address the complexities of modern financial markets.

In summary, the reviewed RL-based studies highlight the transformative role of reinforcement learning in pairs trading. By enabling adaptive decision-making and continuous strategy refinement, RL models such as DDQNs, CA-DRL, and hybrid frameworks like HDRL-Trader empower trading agents to respond to evolving market conditions with greater precision. Furthermore, the integration of RL with traditional financial tools—such as cointegration analysis and structural break detection—enhances both robustness and profitability across diverse markets, including equities, commodities, and futures. These advancements emphasize RL's pivotal role in shaping the next generation of intelligent, data-driven trading systems.

3. Landscape Overview

This section outlines the core body of literature that forms the foundation of this review. Through comprehensive database searches, a central set of studies was identified and further complemented by additional scholarly contributions sourced from reputable academic platforms, including Google Scholar, Scopus, Springer, IEEE Xplore, ScienceDirect, and Web of Science. To ensure broad coverage of machine learning applications in financial decision-making, a wide range of keyword combinations was employed, such as "Pair Trading with Deep Learning," "Pair Trading with Reinforcement Learning," "Pair Trading with Deep Reinforcement Learning," "Pair Trading with Machine Learning," "Pair Trading with Supervised Learning," and "Pair Trading with Unsupervised Learning." Articles not directly contributing to the thematic scope were excluded from the final sample.

In total, 68 peer-reviewed studies published between 2012 and 2023 were selected for in-depth analysis. Table 1 reports the yearly distribution of these publications, together with their corresponding reference indices.

Table 1. Counts of publication frequencies in the studies over the years analyzed, from 2012 to 2023.

Year	Count	Article
2023	12	[1-12]
2022	18	[13-30]
2021	8	[31-38]
2020	14	[39-52]
2019	3	[53-55]
2018	3	[56-58]
2017	4	[59-62]
2016	1	[63]
2015	1	[64]
2014	1	[65]
2013	2	[66-67]
2012	1	[68]

Table 2 describes of the Integration of Artificial Intelligence in Pair Trading: This table encapsulates the various elements involved in the application of AI to stock market operations, including the Categories of Stock Market Data, Types of ML Applications, Empirical and Prediction Models, Model Performance Evaluation Metrics, and Measures for Evaluating Portfolio Outcomes.

Among ML models, RF stands for Random Forest, a versatile and widely-used ensemble method. AdaBoost refers to Adaptive Boosting, a technique that combines multiple weak learners into a stronger model. DT is the abbreviation for Decision Tree, a fundamental classification and regression approach. Boosting is a family of algorithms that enhance the performance of ML models. LM denotes Linear Model, often used in the context of regression, or Lasso Model when it involves L1 regularization. LR stands for Logistic Regression, a staple method for binary classification tasks. SVM and SVR represent Support Vector Machine and Support Vector Regression, respectively, both of which are powerful in finding the optimal boundary between different classes. NB is short for Naive Bayes, a probabilistic classifier that assumes independence between predictors. EN means Elastic Net, a regularized regression method that combines both L1 and L2 penalties. GDA stands for Gaussian Discriminant Analysis (GDA), a generative model for classification, while ELM is a fast-learning algorithm for single-hidden layer feedforward neural networks.

In Deep Learning, NN is the abbreviation for Neural Network, a basic structure of interconnected nodes mimicking the human brain. DNN stands for Deep Neural Network, which is a complex network with multiple layers, enabling the model to learn high-level features from data. RNN, or Recurrent Neural Network, is a type of neural network where connections between nodes form a directed graph along a temporal sequence, allowing it to exhibit temporal dynamic behavior. LSTM is Long Short-Term Memory Network, a special kind of RNN capable of learning long-term dependencies. CNN represents Convolutional Neural Network, highly effective in processing data with a grid-like topology, such as images. TCN stands for Temporal Convolutional Network, known for its use in sequence modeling tasks.

For RL, DQN is a groundbreaking algorithm that combines Q-Learning with DNN. DDQN is an improvement over DQN that reduces overestimation of action values. PPO refers to Proximal Policy Optimization, a policy gradient method for training DNN. HDRL stands for Hierarchical Deep Reinforcement Learning, which structures the learning process in a hierarchical fashion for complex tasks. Lastly, DPG is used for Deep Policy Gradient, often called Deep Deterministic Policy Gradient when it is used in a context of continuous action spaces.

Performance evaluation typically relies on statistical error metrics such as MAE, MAPE, MSE, RMSE, and model loss; classification indicators such as accuracy, precision, recall, F-score, confusion matrix, AUC, and ROC curve; and reward-based measures in the reinforcement learning context. Portfolio-level performance is assessed using financial indicators including cumulative and daily returns, Sharpe and Treynor ratios, maximum drawdown, skewness, kurtosis, profit–loss ratio, volatility, and standard deviation.

In our exploration of recent trends, we conducted a comprehensive methodological analysis of the 68 selected studies. The results reveal a growing diversification in the types of AI methods employed. Approximately 20% of the studies utilized RL exclusively, another 20% adopted DL-based approaches, while traditional ML techniques dominated with around 48%. Notably, about 10% of the papers implemented hybrid methods, with the remainder distributed across miscellaneous approaches. This distribution indicates an emerging interest in integrating multiple AI paradigms for enhanced performance.

Table 2. The Categories of Stock Market Data, Types of ML Applications, Empirical and Prediction Models, Model Performance Evaluation Metrics, and Measures for Evaluating Portfolio Outcomes.

Types of Data	Tasks	Models[1]	Model Performance Evaluation Metrics	Portfolio Performance Evaluation Metrics
Company Stocks	Empirically investigate the pairs trading performance	Machine Learning (RF, AdaBoost, DT, Boosting, LM, LR, SVM, SVR, NB, EN, GDA, ELM)	MAE	Cumulative Return
Stock Index			MAPE	Daily Return
Stock Index Futures			MSE	Monthly Return
Commodities			RMSE	Average Daily Return
Cryptocurrencies			Confusion Matrix	Maximum Drawdown
Currencies		Deep Learning (NN, DNN, RNN, LSTM, CNN, TCN)	Accuracy	Skewness
			Precision	Kurtosis
	Price prediction		Recall	Sharpe Ratio
		Reinforcement Learning (DQN, DDQN, PPO, HDRL, DPG)	F-Score	Treynor ratio
			Model Loss	Profit-loss ratio
			AUC	Annual Volatility
			ROC Curve	Standard Deviation
			Reward	

The reviewed studies span multiple regions and markets, offering a global perspective on pairs trading strategies. For the United States, indices such as the S&P 500, Russell 2000, NYSE Composite, and AMEX Composite were frequently analyzed. Asian markets were represented through the Taiwan TAIEX, KOSPI 200, CSI 300, PSX, and SSE indices, while European markets were evaluated using indices such as the FTSE 100. Table 3 provides a detailed overview of the indices and their respective countries.

Table 3. presenting a compilation of key stock indices and markets evaluated in this study

Index	Country
S&P 500	U.S.
S&P 400	U.S.
S&P 100	U.S.
Russell 2000	U.S.

[1] A detailed introduction of the method will be provided in Chapter 5.

Index	Country
New York Stock Exchange (NYSE)	U.S.
American Stock Exchange (AMEX)	U.S.
CSI 300	China
CSI 300 index futures (IF)	China
CSI 300 exchange traded fund (ETF)	China
Shanghai Stock Exchange (SSE)	China
Taiwan Stock Exchange Capitalization Weighted Stock Index (TAIEX)	Taiwan
FTSE 100	UK
Pakistan Stock Exchange (PSX)	Pakistan
KOSPI 200	South Korea
1027 NSE (National Stock Exchange of India)	India

4. Data Processing and Analysis

Most of the studies included in our review relied on daily historical stock datasets, typically comprising the Opening, High, Low, and Closing prices together with trading Volume—collectively referred to as OHLCV data. A subset of researchers further employed higher-frequency intraday datasets, with sampling intervals of 1, 5, or 15 minutes, to capture finer-grained market microstructure dynamics and short-term price fluctuations. In addition, several studies incorporated sentiment indicators derived from unstructured textual sources, such as Twitter feeds and online discussion forums, thereby enriching financial models with measures of public opinion and investor sentiment.

4.1 Historical Price Data with Daily Interval

Daily price data forms the cornerstone of quantitative financial analysis, particularly in the domain of pairs trading. A standard daily dataset includes five key attributes for each stock: Open, High, Low, Close, and Volume. Many datasets also provide Adjusted Close, which accounts for dividends and stock splits, thereby enabling more accurate backtesting and performance evaluation.

An illustration of such data is shown below using Tesla Inc. as an example, consistent with practices observed in the reviewed literature [1, 2, 5–8, 11–13, 15–23, 25–33, 35–44, 46, 49, 50, 52, 53, 56, 58–62, 65–68]. Table 4 displays a typical sample of daily OHLCV data for Tesla.

Table 4. An example of Tesla Inc.'s daily historical data.

Date	Open	High	Low	Close	Adj Close	Volume
4/4/2023	197.320007	198.740005	190.320007	192.580002	192.580002	126463800
4/5/2023	190.520004	190.679993	183.759995	185.520004	185.520004	133882500
4/6/2023	183.080002	186.389999	179.740005	185.059998	185.059998	123857900
4/10/2023	179.940002	185.100006	176.110001	184.509995	184.509995	142154600
4/11/2023	186.690002	189.190002	185.649994	186.789993	186.789993	115770900
4/12/2023	190.740005	191.580002	180.309998	180.539993	180.539993	150256300
4/13/2023	182.960007	186.500000	180.940002	185.899994	185.899994	112933000
4/14/2023	183.949997	186.279999	182.009995	185.000000	185.000000	96438700
4/17/2023	186.320007	189.690002	182.690002	187.039993	187.039993	116662200
… …	… …	… …	… …	… …	… …	… …

In our dataset analysis, 51 out of the 68 reviewed studies (approximately 75%) utilized daily frequency data as the primary input for modeling and testing. This dominance reflects a strong methodological preference in the academic community, driven by the balance between temporal resolution and computational feasibility. Daily data provides sufficient granularity to capture meaningful price movements while avoiding the noise and complexity of intraday or tick-level datasets.

Furthermore, practical considerations—such as the limited accessibility of high-frequency data, which often requires costly subscriptions or institutional-level access—further explain the prevalence of daily data in academic research. Accordingly, the popularity of daily frequency datasets stems from both methodological robustness and practical feasibility, making them the preferred choice for constructing and evaluating pairs trading strategies across diverse markets.

4.2 Historical Price Data with Tick Interval

Tick-level historical data represents the most granular form of market information, capturing every individual transaction executed on an exchange. Unlike intraday datasets aggregated over fixed intervals (e.g., 1-minute or 5-minute bars), tick data provides a continuous stream of time-stamped records, each detailing the exact execution time, trade price, and trade size.

This high-resolution dataset is particularly valuable for HFT strategies, which rely on executing a large number of trades within milliseconds or seconds. Researchers and practitioners use tick data to construct fine-grained models of market behavior, enabling the detection of short-term price patterns and microstructure dynamics with high precision. Moreover, tick data supports detailed examinations of order flow, liquidity provision, and price discovery mechanisms, offering crucial insights into intraday market conditions.

Despite these advantages, the use of tick data poses significant challenges in terms of storage, processing speed, and computational requirements, due to the massive volume of records generated. Nonetheless, for algorithmic and intraday traders, the predictive power of tick data often justifies these computational demands, particularly in markets characterized by rapid price fluctuations and frequent quote updates.

Table 5 provides an illustration of tick-level data, using trading records from the Gold Futures contract AU2106. As observed in a limited number of studies [34, 57], tick data remains underutilized in the pairs trading literature, largely due to accessibility constraints and technical complexity. Nevertheless, its potential to enhance the accuracy and responsiveness of trading algorithms in fast-paced environments is considerable.

Table 5. A sample of historical tick-level data from the Gold Futures AU2106 contract.

Contract ID	Trade Date	Previous Settle	Timestamp	Volume	Price	Bid Price	Ask Price	Last Settle	Turnover
au2106	20200610	20200609	21:00:00	500	391.88	391.88	391.88	391.88	391880
au2106	20200610	20200609	21:00:01	0	391.88	391.88	391.88	391.88	391880
au2106	20200610	20200609	21:00:01	500	391.88	391.88	391.88	391.88	391880
au2106	20200610	20200609	21:00:02	0	391.88	391.88	391.88	391.88	391880
au2106	20200610	20200609	21:00:02	500	391.88	391.88	391.88	391.88	391880
au2106	20200610	20200609	21:00:03	0	391.88	391.88	391.88	391.88	391880
au2106	20200610	20200609	21:00:03	500	391.88	391.88	391.88	391.88	391880
au2106	20200610	20200609	21:00:04	0	391.88	391.88	391.88	391.88	391880
au2106	20200610	20200609	21:00:04	500	391.88	391.88	391.88	391.88	391880
au2106	20200610	20200609	21:00:05	0	391.88	391.88	391.88	391.88	391880
… …	… …	… …	… …	… …	… …	… …	… …	… …	… …

4.3 Historical Price Data with One-Minute Interval

One-minute historical data represents a practical compromise between the ultra-high granularity of tick-level data and the broader perspective of hourly or daily intervals. Each record captures the OHLC prices and trading volume for a specific minute, thereby offering a more detailed view of market activity than daily data while avoiding the overwhelming scale of tick data.

This resolution is particularly well-suited for medium-frequency trading strategies, as it enables analysts and traders to track short-term price dynamics and assess evolving market conditions with enhanced clarity. At the minute level, researchers can investigate phenomena such as volatility clustering, price momentum, and short-term reversals, all of which are essential for intraday decision-making and predictive modeling.

Although one-minute data is less demanding than tick-level data, it still requires considerable computational resources and analytical sophistication, especially when applied to large cross-sectional datasets or extended time horizons. Nevertheless, it strikes a valuable balance between informational richness and computational manageability, making it a popular choice in empirical studies of intraday trading strategies.

Table 6 illustrates one-minute historical data for Tesla Inc. stock. This level of data granularity has been adopted in a select number of studies [10, 24, 48, 54, 55, 64], underscoring its utility in modeling short-horizon market behavior and enhancing predictive accuracy for time-sensitive trading decisions.

Table 6. An example of Tesla Inc.'s one-minute historical data.

Datetime	Open	High	Low	Close	Adj Close	Volume
2024-04-04 09:30:00	170.0000	170.7400	169.8000	170.0303	170.0303	2425141
2024-04-04 09:31:00	170.0000	170.0301	169.3100	169.4850	169.4850	637577
2024-04-04 09:32:00	169.5050	169.6800	168.8741	168.9050	168.9050	537015
2024-04-04 09:33:00	168.8900	169.4800	168.7800	168.8800	168.8800	494559
2024-04-04 09:34:00	168.8800	168.9867	168.5200	168.5600	168.5600	466847
2024-04-04 09:35:00	168.5800	168.6350	168.2500	168.3400	168.3400	487605
2024-04-04 09:36:00	168.3875	168.7100	168.2613	168.4650	168.4650	447779
2024-04-04 09:37:00	168.4600	168.8000	168.3400	168.5400	168.5400	441173
2024-04-04 09:38:00	168.5464	168.7000	168.3201	168.4117	168.4117	308755
2024-04-04 09:39:00	168.4324	168.8500	168.3927	168.7199	168.7199	384769
… …	… …	… …	… …	… …	… …	… …

4.4 Historical Price Data with Five-Minute Interval

Five-minute historical data provides an intermediate level of temporal granularity by aggregating OHLC prices and trading volume within five-minute windows. This format offers a condensed yet informative perspective on intraday market behavior, striking a balance between the noise sensitivity of one-minute or tick-level data and the broader abstraction of hourly or daily intervals.

Such data is particularly valuable for identifying short- to medium-term trends, including support and resistance levels, breakout opportunities, and overall market momentum. The five-minute interval facilitates smoother visualization of price action, filtering out micro-level fluctuations while still retaining sufficient detail to support effective intraday decision-making.

Although less voluminous than higher-frequency formats, five-minute data still requires robust analytical frameworks to uncover meaningful patterns, particularly in high-volatility assets or multi-asset portfolios. It is frequently employed in algorithmic trading, signal generation, and predictive modeling tasks where reduced noise and enhanced pattern stability are advantageous.

Table 7 illustrates this data format using Tesla Inc.'s trading records. This interval has been adopted in several empirical studies [3, 14, 22, 45, 54], underscoring its relevance for strategies that require a balanced view of intraday dynamics across moderately extended time horizons.

Table 7. An example of Tesla Inc.'s five-minute historical data.

Datetime	Open	High	Low	Close	Adj Close	Volume
2024-04-04 09:30:00	170.0000	170.7400	168.5200	168.5600	168.5600	4561139
2024-04-04 09:35:00	168.5800	168.8500	168.2500	168.7199	168.7199	2070081
2024-04-04 09:40:00	168.7200	169.1600	168.4500	168.7500	168.7500	1467289
2024-04-04 09:45:00	168.7342	169.6300	168.0100	169.3650	169.3650	2099000
2024-04-04 09:50:00	169.3900	169.8600	169.0300	169.4500	169.4500	1527195
2024-04-04 09:55:00	169.4300	169.5000	168.9200	169.0999	169.0999	1159740
2024-04-04 10:00:00	169.0800	169.1300	168.3736	168.7200	168.7200	1309723
2024-04-04 10:05:00	168.7600	169.7500	168.7200	169.6550	169.6550	1288829
2024-04-04 10:10:00	169.6600	170.1400	169.4300	170.0437	170.0437	1449532
2024-04-04 10:15:00	170.0274	170.1200	169.7200	169.8956	169.8956	978258
… …	… …	… …	… …	… …	… …	… …

4.5 Historical Price Data with Hourly Interval

Hourly historical data provides a macro-intraday perspective by capturing aggregated price movements and trading volumes within each trading hour. This frequency represents a compromise between the microstructure-oriented granularity of tick or minute-level data and the broader abstraction of daily summaries.

Each observation in an hourly dataset typically contains the OHLC, together with the trading volume for that hour. Such temporal aggregation is particularly relevant for identifying intraday price dynamics—such as momentum shifts, breakout formations, and volatility spikes—while avoiding the noise and computational intensity associated with higher-frequency formats.

Hourly data is often employed in event-driven strategies, where price responses to scheduled economic announcements, corporate earnings releases, or shifts in market sentiment can be detected more clearly than at daily intervals. Although not as fine-grained as tick or minute data, hourly data retains sufficient detail to uncover meaningful short- to medium-term patterns and to generate actionable trading signals.

From a computational perspective, hourly data requires less storage and processing power than tick or minute-level datasets, yet still necessitates advanced modeling techniques for effective use. This makes it suitable for algorithmic trading strategies that must remain responsive to intraday dynamics while maintaining scalability in real-time applications.

As shown in Table 8, which illustrates Tesla Inc.'s hourly trading records, this data frequency has been adopted in a limited number of studies [51, 54]. Its application highlights its value for strategies that demand a broader yet still responsive perspective on market behavior, including pairs trading frameworks where signal stability and reduced noise are essential.

Table 8. An example of Tesla Inc.'s hourly historical data.

Datetime	Open	High	Low	Close	Adj Close	Volume
2024-04-01 09:30:00	176.1600	176.3900	171.6000	171.7280	171.7280	23947033
2024-04-01 10:30:00	171.7400	172.1899	170.2100	172.1050	172.1050	15778216
2024-04-01 11:30:00	172.0900	172.4300	170.8900	172.2800	172.2800	8095873
2024-04-01 12:30:00	172.2882	173.2771	171.7400	171.8900	171.8900	7020081
2024-04-01 13:30:00	171.8600	173.1050	171.7500	172.9400	172.9400	5705376
2024-04-01 14:30:00	172.9200	174.0868	172.5901	173.9550	173.9550	7463119
2024-04-01 15:30:00	173.9500	175.2900	173.1901	175.1200	175.1200	8080117
2024-04-02 09:30:00	164.6890	167.1900	163.4300	165.5701	165.5701	45944518
2024-04-02 10:30:00	165.5800	165.8707	163.9000	165.6700	165.6700	18110912
2024-04-02 11:30:00	165.6700	166.6500	165.1129	166.2600	166.2600	11561980
… …	… …	… …	… …	… …	… …	… …

4.6 Historical Price Data with Monthly Interval

Monthly historical data provide a broad, long-term perspective on market performance by aggregating trading activity over one-month intervals. Each record typically includes the opening price at the start of the month, the highest and lowest prices during the period, the closing price at month-end, and the total trading volume. In addition, monthly datasets generally contain the adjusted closing price, which incorporates dividends and stock splits to provide a more accurate measure of long-term returns. Compared with high-frequency or intraday data, monthly data are less granular but offer valuable insights for strategic investment decisions and macroeconomic analysis.

This type of data is particularly beneficial for long-term investors, as it facilitates the detection of seasonal patterns, responses to quarterly earnings announcements, and the influence of broader economic cycles. Analysts frequently employ monthly datasets to assess the overall direction and health of stocks or indices, forming the basis for portfolio rebalancing, asset allocation, and risk management over extended horizons.

Although the volume of data is significantly reduced relative to tick or daily frequencies, monthly data still require a robust analytical framework to capture cyclical and trend-driven dynamics. The lower frequency mitigates short-term market noise, making it a suitable input for models that emphasize stability, directionality, and macro-level indicators. However, in the context of pairs trading, monthly data are rarely employed for direct signal generation, since trading opportunities typically unfold at higher frequencies. Instead, they are more commonly used to validate long-run equilibrium relationships or to support cross-market and cross-asset hedging strategies.

Table 9 presents an example of monthly historical data for Tesla Inc., as adopted in selected studies [4, 9, 47]. This dataset is particularly appropriate for predictive models aimed at guiding investment decisions over months or quarters, where the focus is on strategic positioning in response to economic cycles or policy changes rather than daily price fluctuations.

In Section 4, we examined various types of historical stock price data across different temporal resolutions—from daily to monthly—each providing distinct analytical advantages for understanding market behavior. Daily data, which include OHLCV, are the most widely adopted due to their accessibility and ability to capture comprehensive market activity within a trading session. Tick-level data, on the other hand, record every market transaction, offering high-resolution insights into market microstructure and liquidity. However, their use is constrained by computational demands and limited accessibility due to licensing costs.

Table 9. An example of Tesla corporation's monthly historical data.

Datetime	Open	High	Low	Close	Adj Close	Volume
2023-05-01 00:00:00	163.1700	204.4800	158.8300	203.9300	203.9300	2681994800
2023-06-01 00:00:00	202.5900	276.9900	199.3700	261.7700	261.7700	3440477900
2023-07-01 00:00:00	276.4900	299.2900	254.1200	267.4300	267.4300	2392089000
2023-08-01 00:00:00	266.2600	266.4700	212.3600	258.0800	258.0800	2501580900
2023-09-01 00:00:00	257.2600	278.9800	234.5800	250.2200	250.2200	2439306100
2023-10-01 00:00:00	244.8100	268.9400	194.0700	200.8400	200.8400	2590570100
2023-11-01 00:00:00	204.0400	252.7500	197.8500	240.0800	240.0800	2650798400
2023-12-01 00:00:00	233.1400	265.1300	228.2000	248.4800	248.4800	2294598400
2024-01-01 00:00:00	250.0800	251.2500	180.0600	187.2900	187.2900	2343784600
2024-02-01 00:00:00	188.5000	205.6000	175.0100	201.8800	201.8800	2019907700
……	……	……	……	……	……	……

Intermediate-frequency data, such as one-minute, five-minute, and hourly intervals, provide a practical compromise between granularity and manageability. These datasets enable analysts to detect intraday trends and short-term volatility, making them well-suited for medium-frequency trading strategies that demand timely insights without the overwhelming volume and noise of tick data. Monthly data, by contrast, offer a macro-level view, capturing long-term trends and structural market shifts, thus serving the needs of strategic, long-horizon investors.

Using Tesla's historical data as a reference, we observe that tick data reveals subtle, rapid price changes essential for modeling high-frequency trading behaviors. In contrast, daily data smooth out such micro-level noise, emphasizing broader market dynamics and trend-based signals. Five-minute and hourly datasets act as effective intermediaries, enabling traders to exploit intraday momentum while maintaining data tractability. Monthly data, with their low frequency, capture broad economic cycles, making them instrumental for portfolio reallocation and macroeconomic alignment.

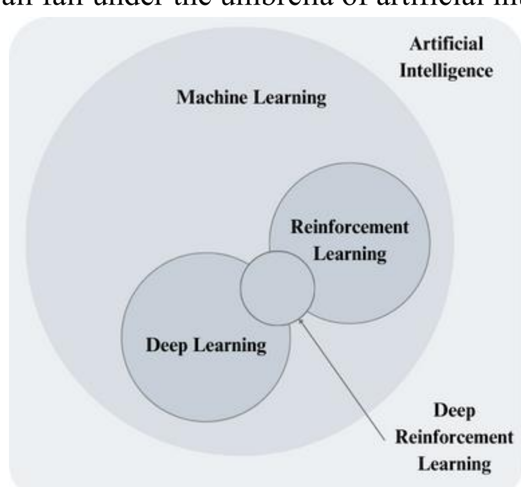
Comparing Tesla's price data across these intervals also reveals notable differences in statistical characteristics—such as mean, variance, and volatility. Higher-frequency data typically exhibit greater variance and noise due to rapid market reactions, whereas lower-frequency data reduce short-term fluctuations, making long-term patterns more visible. Recognizing these differences is crucial for aligning the data frequency with the design and objective of a trading strategy. In the specific context of pairs trading, this alignment is particularly critical: high-frequency data may uncover transient arbitrage opportunities but at the cost of higher noise and transaction costs, while lower-frequency data highlight more stable long-term mean-reverting relationships. Ultimately, the effectiveness of any predictive model depends heavily on the chosen data resolution, underscoring the importance of selecting time intervals that best suit the intended analytical or trading goals.

5. Artificial Intelligence Models in Pair Trading

AI, particularly its subfields of ML, DL, and RL, has attracted growing attention within the financial industry. Banks, asset managers, hedge funds, and securities firms are increasingly incorporating AI-based techniques—including supervised and unsupervised learning, NLP, and advanced data analytics—into their investment processes. The goal is to enhance predictive accuracy, extract insights from complex and large-scale datasets, and ultimately improve profitability while sustaining competitive advantages.

Figure 3 illustrates the hierarchical relationships among AI, ML, DL, and RL. Within this framework, ML is a subdomain of AI, DL constitutes a specialized branch of ML, and RL, while overlapping with both, operates under distinct principles of sequential decision-making. DRL emerges at the intersection of DL and RL, combining representational power with adaptive learning mechanisms.

Figure 3. Relationship between AI, ML, DL, RL, and DRL. Machine learning, deep learning, and reinforcement learning all fall under the umbrella of artificial intelligence.



AI methods are now widely applied in quantitative trading, where financial data are modeled through mathematical, statistical, and algorithmic techniques. Among these strategies, pairs trading has proven particularly well-suited to AI applications. Pairs trading involves identifying two historically correlated assets and exploiting temporary deviations in their relative valuations. ML, DL, and RL provide complementary approaches that improve the precision, adaptability, and robustness of such strategies.

Machine Learning in Pairs Trading

Machine learning plays a central role in identifying suitable asset pairs for statistical arbitrage by analyzing historical data to uncover stable, long-term relationships. The primary mechanism involves modeling the typical spread behavior between two assets and detecting significant deviations from historical norms. A critical step is feature engineering, in which inputs such as price ratios, price differences, volatility measures, and mean-reversion indicators are constructed. These features are then fed into classification or regression models—such as SVM, LR, or RF—which are trained to predict the likelihood of convergence or divergence in the asset pair. This data-driven approach moves beyond rigid econometric assumptions, enabling traders to make more adaptive and statistically grounded decisions.

Deep Learning in Pairs Trading

Deep learning extends the capabilities of ML by capturing nonlinear and complex dependencies in financial data. DNNs, CNNs, and recurrent architectures such as LSTMs are particularly effective at identifying subtle patterns that traditional linear models fail to capture. In data-rich environments, DL models excel at processing large-scale, high-dimensional inputs to uncover latent structures that signal profitable trading opportunities. Moreover, DL methods are useful for anomaly detection, flagging unusual deviations in price relationships that may indicate trading signals. By learning from the broader distribution of market behavior, DL models enhance robustness, reduce false positives, and improve the reliability of signal generation in pairs trading.

Reinforcement Learning in Pairs Trading

Reinforcement learning introduces a fundamentally different paradigm, making it highly suitable for sequential decision-making in dynamic and uncertain markets. In the context of pairs trading, RL agents interact with a simulated or real trading environment, receiving rewards based on the profitability of their actions. Through this trial-and-error process, the agent progressively refines its trading policy, determining optimal entry and exit points in real time. RL's adaptability is a key advantage: agents can update strategies incrementally as new data arrive, reducing the need for complete retraining when market conditions evolve. Furthermore, RL frameworks facilitate continuous improvement through simulation and back-testing, allowing agents to be trained across diverse market scenarios without incurring real financial risk. This makes RL particularly effective in addressing non-stationarity and evolving asset relationships—persistent challenges in financial markets.

To provide a structured overview of existing approaches, the methodologies documented in the literature are categorized into the tables presented below. Table 10 summarizes the supervised learning models employed, while Table 11 lists the unsupervised models. Table 12 outlines commonly applied DL models, Table 13 details RL approaches, and Table 14 highlights hybrid methodologies. These categorizations establish the foundation for the more detailed discussions presented in the subsequent sections of this chapter.

Table 10. Comprehensive overview of supervised learning models employed.

Article	Supervised Learning Models
[2, 30, 47, 64, 67]	SVM
[62, 65, 66]	SVR
[4]	Elastic Net Regression
[16]	AdaBoost
[19]	XGBoost
[25, 39]	LGBM, RF, SVR
[54]	RF
[55]	LR, RF
[59]	GBDT
[60]	SVM, RF, ANFIS

SVM and SVR are among the most widely adopted supervised models in the reviewed studies. Their popularity stems from robustness in high-dimensional settings and the ability to handle outliers effectively. While SVM is primarily designed for classification tasks, SVR extends the framework to regression problems, enabling the prediction of continuous outputs.

Tree-based ensemble methods constitute another major category. RF constructs multiple decision trees and aggregates their predictions, reducing variance and mitigating overfitting. Gradient Boosting Decision Trees (GBDT), along with advanced variants such as XGBoost and

LightGBM (LGBM), follow an iterative approach where each tree corrects the residual errors of its predecessors. Compared to RF, boosting methods aim to reduce bias and generally achieve higher predictive accuracy. XGBoost and LGBM further improve scalability and computational efficiency, making them particularly suitable for large-scale financial datasets.

Linear models also appear in the literature. Elastic Net Regression (EN) addresses multicollinearity by combining the penalties of Lasso (L1) and Ridge (L2), improving both generalization and feature selection. LR, by contrast, is frequently used in binary classification problems, modeling the probability of specific trading outcomes through the logistic function.

AdaBoost represents another ensemble approach. By sequentially reweighting misclassified examples, it builds a strong composite classifier from weak learners, which is especially effective in noisy datasets.

Finally, the Adaptive Neuro-Fuzzy Inference System (ANFIS), though less common, integrates neural networks with fuzzy logic to capture nonlinear dynamics and handle uncertainty—characteristics that align closely with financial market conditions.

In summary, supervised learning models in pairs trading vary in complexity and computational requirements. Ensemble methods such as RF, GBDT, XGBoost, and LGBM typically deliver superior predictive performance and robustness, especially in high-dimensional or large-sample settings. However, they are more resource-intensive compared to simpler models like LR or SVM. Ultimately, the choice of model depends on data characteristics, the volatility of the trading environment, and the specific objectives of the strategy.

Table 11. Comprehensive overview of unsupervised learning models employed.

Article	Unsupervised Learning Models
[13, 49]	DBSCAN
[7]	PCA, DBSCAN
[9]	K-Means, DBSCAN, AHC
[18]	OPTICS Clustering Algorithm (OPTICS)
[68]	EM Algorithm (EM)
[25, 39]	LGBM, RF, SVR

Unsupervised learning models are widely applied in pairs trading to uncover latent structures in unlabeled data, making them particularly valuable for exploratory analysis and pair selection.

The reviewed studies predominantly employ clustering algorithms and dimensionality reduction techniques.

Among the clustering approaches, density-based methods such as DBSCAN are frequently used due to their ability to detect clusters of arbitrary shape and handle noisy financial data. This property is particularly useful when market data exhibit irregular distributions. However, DBSCAN's performance is less reliable in high-dimensional spaces or when cluster densities vary significantly. OPTICS, an extension of DBSCAN, addresses this limitation by generating a reachability plot that captures hierarchical cluster structure without requiring a predefined number of clusters or fixed density thresholds.

Centroid- and hierarchy-based clustering also appear in the literature. K-Means provides a computationally efficient means of partitioning assets into a fixed number of clusters by minimizing within-cluster variance. Yet, its reliance on convex, equally sized clusters may not align with the complexities of real financial markets. Agglomerative Hierarchical Clustering (AHC), in contrast, constructs a nested cluster hierarchy that offers intuitive visualization through dendrograms. While this method reveals multi-level relationships between assets, it becomes computationally intensive with larger datasets. Probabilistic clustering is represented by the EM algorithm, typically applied within Gaussian Mixture Models (GMM). EM provides a flexible, probabilistic description of asset groupings, which is advantageous for modeling overlapping structures in financial data. Nevertheless, its sensitivity to initialization and higher computational demands limit scalability.

In addition to clustering, dimensionality reduction techniques are also applied. PCA serves as a critical preprocessing tool by transforming correlated variables into orthogonal components, thereby reducing dimensionality while preserving most of the variance. This not only enhances clustering performance but also mitigates noise, making it particularly relevant when dealing with high-dimensional stock return datasets.

Interestingly, some studies also report the use of models such as LGBM, RF, and SVR in this context. Although these are conventionally classified as supervised methods, their inclusion likely reflects applications within semi-supervised or hybrid frameworks, where clustering results are integrated with supervised prediction tasks.

Overall, unsupervised learning methods provide powerful tools for detecting hidden relationships in financial markets. Density-based models like DBSCAN and OPTICS excel at handling irregular structures and noisy environments, whereas K-Means and AHC offer more structured but assumption-dependent clustering. EM enables probabilistic modeling of latent asset groups, while PCA plays a complementary role in reducing dimensionality and improving model efficiency. The choice of technique ultimately depends on data characteristics—such as dimensionality, noise, and heterogeneity—as well as the objectives of the trading strategy, whether exploratory pair identification or enhanced predictive modeling.

Table 12. Comprehensive overview of deep learning models employed.

Article	Deep Learning Models
[24, 36, 37, 38, 48]	LSTM
[1]	Neural Network
[17]	KalmanNet Bollinger Trading (KalmanBOT)
[27]	Stochastic Neural Network (SNN)
[28]	RNN, LSTM, TCN
[31]	TCN
[45]	LSTM, LSTM Encoder-Decoder
[46]	LSTM, CNN, Multilayer perceptron (MLP)
[50]	DNN
[56]	BNN, DP-BNN, DDP-BNN
[57]	Filterbank CNN

The DL models employed in the reviewed studies encompass a wide range of architectures, each addressing specific challenges inherent in financial time series analysis. RNNs and their advanced variants, particularly LSTM networks, dominate the literature due to their capacity to capture long-term temporal dependencies while mitigating vanishing gradient problems. More sophisticated designs, such as LSTM encoder–decoder architectures, extend this capacity to sequence-to-sequence forecasting, which is crucial for multi-step financial prediction. TCNs provide an alternative to RNNs by employing dilated convolutions, offering stable gradients and parallelizable computation for modeling long-range dependencies.

Feedforward-based models, including DNNs, MLPs, and other standard neural network structures, are applied to capture nonlinear mappings in asset price data. CNNs have also been adapted from computer vision to financial contexts, where variants such as Filterbank CNNs are used to extract local temporal patterns and repetitive structures from market signals.

Beyond conventional neural networks, several innovative models have emerged. KalmanBOT integrates Kalman filtering into a neural framework, allowing dynamic state estimation and adaptation to evolving market conditions. SNNs incorporate controlled randomness into the training process, enhancing generalization under noisy environments. BNNs, together with extensions such as DP-BNN and DDP-BNN, embed probabilistic reasoning into DL models, enabling uncertainty quantification—a critical aspect in financial risk assessment and decision-making.

In summary, DL models for pairs trading differ in their primary strengths: RNN-based architectures (LSTM, TCN) excel at sequential dependency modeling; CNNs capture local temporal features; probabilistic approaches such as BNNs and SNNs provide robustness under uncertainty; and hybrid innovations like KalmanBOT address dynamic and non-stationary market conditions. The choice of model is ultimately contingent upon the characteristics of the dataset, the required balance between predictive accuracy and interpretability, and the adaptability needed for volatile financial environments.

Table 13. Comprehensive overview of reinforcement learning models employed.

Article	Reinforcement Learning Models
[32, 53]	DQN
[5, 33, 41]	DRL, CA-DRL
[3]	Two-Level Reinforcement Learning
[6]	CA-DRL, NEWS-CO-DRL
[10]	PPO, PPO-PT, PPO-PT w/o Demo, SAPT, PTDQN
[12]	Hierarchical Reinforcement Learning (HRL)
[15]	P-DDQN, PTDQN
[29]	SAPT, SAPT w/o Break, SAPT w/o Time, SAPT w/o Hold, PTDQN, SAPT-3-std, SAPT-ADF, SAPT-BCD, SAPT-LSTM
[34]	RL-0, RL-0.02, RL-0.05, RL-0.1
[35]	Deep Reinforcement Learning (DRL)
[42]	DQN, Double Deep Q-Network (DDQN)
[63]	RL

The RL approaches applied in the reviewed literature span a diverse set of frameworks, reflecting the complexity of financial decision-making environments. Value-based methods such as the DQN employ neural networks to approximate Q-values, enabling agents to learn optimal policies through iterative exploration and exploitation. Its extension, DDQN, addresses the overestimation bias inherent in standard DQN, yielding more stable and accurate convergence.

Beyond these foundations, DRL integrates neural architectures with RL to handle high-dimensional market data. Domain-specific variants such as CA-DRL embed econometric knowledge directly into the learning process, improving the agent's capacity to exploit long-term equilibrium dynamics between asset pairs. Policy gradient methods, particularly PPO and its derivatives (e.g., PPO-PT), offer improved training stability by constraining policy updates within clipped bounds, a feature especially critical in financial markets where instability can lead to substantial losses.

Hierarchical RL (HRL) and multi-layered frameworks such as Two-Level RL and SAPT introduce additional structural flexibility. These approaches decompose trading tasks into layered sub-decisions—such as entry, holding, and exit—enabling agents to operate effectively across multiple levels of abstraction. SAPT and its numerous variants (e.g., SAPT-ADF, SAPT-3-std, SAPT-BCD) further illustrate how statistical criteria can be integrated into RL frameworks to adapt strategies under different market conditions.

Advanced Q-learning extensions like Prioritized Double DQN (P-DDQN) and Prioritized Twin Delayed Q-Network (PTDQN) refine the learning process by assigning greater weight to informative experiences via prioritized replay, thereby accelerating convergence and enhancing adaptability in non-stationary environments.

In summary, RL models in pairs trading differ not only in algorithmic design but also in their suitability for specific trading objectives. Value-based methods (DQN, DDQN) are effective for discrete action problems; PPO are advantageous in continuous and high-dimensional settings; hierarchical and layered frameworks (HRL, SAPT) provide modularity for complex strategies; and advanced replay-based Q-learning variants (P-DDQN, PTDQN) enhance efficiency in volatile markets. The choice of approach ultimately depends on the complexity of the trading environment, the stationarity of asset relationships, and the need for adaptability in dynamic financial contexts.

The hybrid models summarized in Table 14 reflect a growing consensus in the literature that no single approach is sufficient to capture the multifaceted dynamics of financial markets. These designs can be broadly categorized into three groups. First, feature-combination approaches employ dimensionality reduction techniques (e.g., PCA, CAE) and clustering algorithms (e.g., DBSCAN) as preprocessing steps, before passing the transformed data into reinforcement learning models such as PPO. This improves the signal-to-noise ratio and enhances the stability of policy learning. Second, model-fusion frameworks integrate conventional ML methods (e.g., LR, XGBoost, SVM) with DL architectures (e.g., LSTM, CNN), thereby combining the interpretability of shallow learners with the representational power of deep networks. Third, multi-objective frameworks seek to jointly optimize signal generation and portfolio management—for example, by combining Kalman filtering with LSTM architectures, or coupling predictive models with allocation rules such as 1/N or MVF.

Table 14. Comprehensive overview of combined models employed.

Article	Combined Models
[8]	PCA, Convolutional AutoEncoders (CAE), DBSCAN, PPO
[14]	LR, XGBoost, CNN, LSTM
[20]	A-LSTM+1/N, RF+MVF, RF+1/N, SVM+MVF, SVM+1/N
[22]	LSTM, OPTICS
[23]	XGBoost, TCAN
[26]	SVM, XGBoost, DNN, LSTM, RAF
[40]	LSTM, ANN, LR, GBDT, FeedForward NN
[42]	DQN, Double Deep Q-Network (DDQN)
[44]	LSTM, CNN, Deterministic Policy Gradient (DPG)
[51]	ANN, XGBoost
[52]	LR, Gaussian Discriminant Analysis (GDA), SVM, NN
[61]	DNN, GBDT, RF

These hybrid approaches go beyond mere technical integration: they embody an attempt to reconcile diverse temporal, structural, and behavioral aspects of market data within a unified framework. Particularly noteworthy is the increasing incorporation of reinforcement learning into hybrid systems, which shifts the methodological focus from static prediction toward adaptive, sequential decision-making under uncertainty.

Nevertheless, such models introduce new challenges. They often demand significantly greater computational resources, reduce transparency, and are rarely benchmarked against simpler baselines, raising concerns about overfitting and limited generalizability. Future research may thus benefit from formalizing hybrid architectures within principled frameworks such as meta-learning or modular reinforcement learning, while simultaneously prioritizing model interpretability, robustness, and real-world deployability.

5.1 Machine Learning

ML refers to the process of training algorithms to learn patterns from data and make predictions or decisions without explicit rule-based programming. Within ML, algorithms are typically categorized into supervised and unsupervised approaches. In supervised learning, models are trained on labeled datasets to map input features to known outputs, whereas unsupervised learning focuses on uncovering hidden structures within unlabeled data. Both paradigms have been widely applied in financial research, particularly in the prediction of stock price movements and the identification of trading opportunities. The following sections examine the methods most frequently reported in the literature, with reference to Table 10, 11 and 14.

5.1.1 Supervised Learning

Supervised learning involves training models to capture the relationship between explanatory variables and corresponding target labels, with the objective of minimizing prediction error. Applications in finance typically fall into two categories. The first category is regression, where the output variable is continuous, such as predicting stock price levels or expected returns. The second category is classification, which involves categorical outputs, such as forecasting whether the price of a stock will rise or fall.

Among the supervised learning models applied in quantitative finance, linear regression, support vector machines, and random forests are the most commonly used. Evidence from the reviewed literature summarized in Table 10 and 14 shows that support vector machines appear in nine studies, accounting for roughly 13 percent of the reviewed articles, while random forests are reported in six studies, representing about 9 percent. These frequencies highlight the strong preference for these methods, largely due to their robustness in handling high-dimensional, noisy, and nonlinear financial data.

The subsequent discussion provides a closer examination of how these models, particularly support vector machines, are employed in pairs trading. Attention is given to their predictive capacity, their effectiveness in generating trading signals, and their role in supporting statistical arbitrage strategies.

5.1.1.1 Introduction to Support Vector Machine

SVM is a widely used and versatile supervised learning algorithm that can be applied to both classification and regression tasks, although it is predominantly utilized for classification. The fundamental objective of SVM is to identify an optimal decision boundary, or hyperplane, that separates data points belonging to different classes with the greatest possible margin. The data points that lie closest to this hyperplane are referred to as support vectors, and they play a critical role in determining its orientation and position.

The principle of margin maximization lies at the core of SVM. By selecting the hyperplane that maximizes the distance to the nearest support vectors, the algorithm enhances its ability to generalize to unseen data, thereby reducing the risk of overfitting. For linearly separable data, this hyperplane takes the form of a line in two dimensions, a plane in three dimensions, or a higher-dimensional analogue. However, financial data and other real-world datasets are rarely linearly separable. To address this limitation, SVM employs the kernel trick, which maps the original feature space into a higher-dimensional one where linear separation becomes feasible. This transformation is performed implicitly by kernel functions, avoiding the computational burden of explicit feature expansion. Commonly used kernels include the polynomial kernel, the radial basis function (RBF) kernel, and the sigmoid kernel, each offering different capacities to capture nonlinear structures in the data.

An additional and equally important aspect of SVM is regularization, which is governed by a penalty parameter commonly denoted as C. This parameter balances the trade-off between maximizing the margin and minimizing classification errors on the training data. A large value of C forces the model to prioritize correct classification of training samples, often at the expense of generalization, whereas a smaller value of C allows some misclassifications but typically yields a more robust model. This flexibility is particularly relevant in financial applications, where noise and outliers are common in market data.

The optimal separating hyperplane in a linear SVM is:

$$w^T \mathrm{x} + \mathrm{b} = 0 \tag{5.1}$$

where w is the normal vector (determining the orientation of the hyperplane), x denotes a data point in feature space, and b is the bias term shifting the hyperplane along w.

Training a linear SVM chooses w and b to maximize the margin, i.e., the minimum distance from the hyperplane to the nearest data points (the support vectors). For linearly separable data (hard margin), this is equivalent to the convex program,

$$\min_{w,b} \frac{1}{2}||w||^2 \quad s.t. \quad y_i(w^T \mathrm{x}_i + \mathrm{b}) \geq 1,\ \mathrm{i} = 1, \dots, \mathrm{n} \tag{5.2}$$

with labels $y_i \in \{-1, +1\}$.

In practice, real data are rarely perfectly separable. The soft-margin formulation introduces slack variables $\xi_i \geq 0$ and the regularization parameter $C > 0$ (which you discussed earlier) to balance margin size and training errors:

$$\min_{w,b,\xi} \frac{1}{2}||w||^2 + C\sum_{i=1}^{n} \xi_i \quad s.t. \quad y_i(w^T \mathrm{x}_i + \mathrm{b}) \geq 1 - \xi_i,\ \xi_i > 0 \tag{5.3}$$

Larger C penalizes violations more strongly (lower bias, higher variance); smaller C allows a wider margin at the cost of more training errors (higher bias, lower variance).

For nonlinearly separable data, SVM employs the kernel trick, replacing inner products by a kernel $K(x_i, x) = \phi(x_i)^T \phi(x)$ without explicitly computing the mapping $\phi(\cdot)$. The decision function is

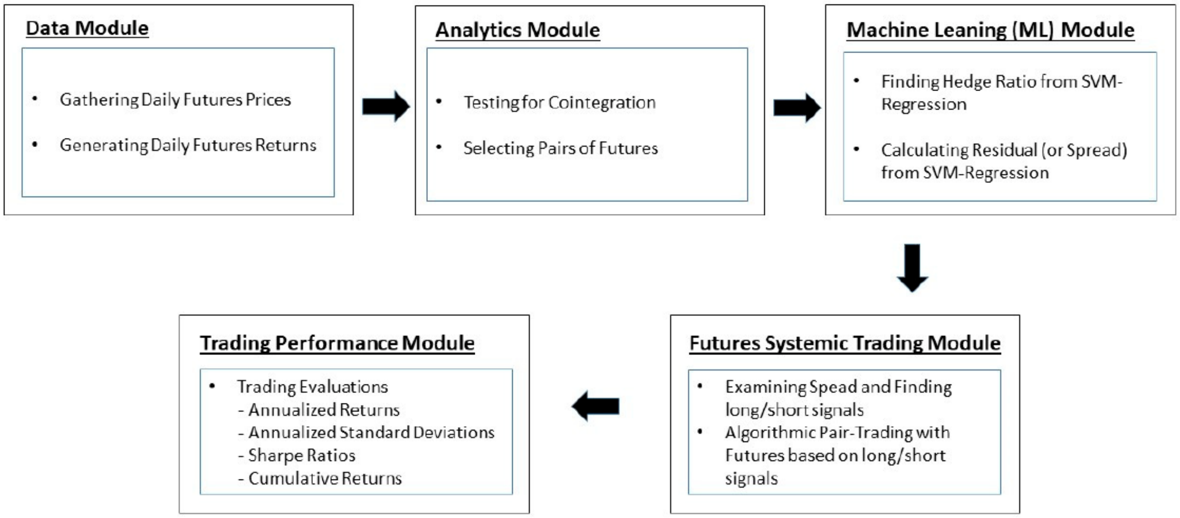
$$f(x) = sgn(\sum_{i=1}^{n} \alpha_i y_i K(x_i,\ x) + b) \tag{5.4}$$

where α_i are the Lagrange multipliers from the dual optimization; only support vectors have $\alpha_i > 0$ (by KKT conditions). Common choices of K include the polynomial and RBF kernels (the sigmoid kernel is also used in some applications).

5.1.1.2 Support Vector Machine in Pair Trading

Speaking to the steps of implementing SVM in pair trading, as shown in Figure 4 from article [47], the model architecture consists of five modules: data acquisition, analytics, hedging based on the ML method, pairs trading, and trading performance evaluation.

Figure 4. Systematic pair trading model architecture using SVM for futures in article [47].



The specific steps for applying SVM to pairs trading are also outlined in article [30], where model performance under different parameter settings is presented in Table 15.

Table 15. Model performance with different parameters in article [30].

Pair	Accuracy	AUC of ROC	Parameters
600797.SH-603421.SH	64%	0.63	kernel=poly, C=0.1, degree=3, coef0=2
002807.SZ-601288.SH	57%	0.57	kernel=poly, C=1, degree=3, coef0=0.5
000025.SZ-000715.SZ	56%	0.57	kernel=rbf, C=0.1, gamma=0.01, coef0=0
600029.SH-600115.SH	58%	0.59	kernel=poly, C=4, degree=3, coef0=5
002696.SZ-600975.SH	66%	0.67	Kernel=poly, C=2, degree=2, coef0=2

Data Acquisition and Preprocessing

The implementation of an SVM-based pairs trading strategy begins with data acquisition and preprocessing. Historical daily price data for selected stock pairs are retrieved from the Chinese market using the Tushare API. This dataset typically includes open, high, low, close, and volume information for each trading day. The raw data are then cleansed to address missing values, correct anomalies, and exclude non-trading days, thereby ensuring consistency and reliability. To enhance model convergence and stability, Min-Max normalization is applied, scaling all feature values to a uniform range, typically between 0 and 1.

Feature Engineering and Selection

Following preprocessing, feature engineering and selection are conducted. For each stock pair, absolute and relative price spreads are calculated to quantify divergence between the two assets. Volatility measures, such as rolling standard deviations of the spread, are also computed to capture dynamic changes in co-movement. Additionally, technical indicators—including moving averages, RSI, and MACD—are constructed to embed trend-following and momentum-based information. To manage dimensionality and focus on informative predictors, techniques such as PCA or feature importance scores from tree-based models are employed.

SVM Configuration and Model Training

Once features are defined, the SVM is configured and trained. Different kernel functions—linear, polynomial, and RBF—are tested via cross-validation to identify the most effective choice in terms of accuracy and computational efficiency. A grid search is then performed across hyperparameters, including the penalty term (C), kernel coefficient (gamma), and polynomial degree, to determine optimal settings. Using the selected configuration, the model is trained on the full training dataset to capture nonlinear dependencies and complex feature interactions.

Strategy Formulation and Trading Signal Generation

Based on the trained SVM, trading signals are generated by predicting whether the price spread between a stock pair will widen or narrow. Execution criteria are then defined to translate these predictions into trading actions. For instance, a long-short position is initiated when the predicted spread change exceeds a specified threshold. The resulting strategy is back-tested on historical data to validate its performance and confirm the decision rules.

Back-testing and Performance Evaluation

Historical simulations are conducted under varying market conditions to rigorously evaluate performance. Key performance indicators—including cumulative return, Sharpe ratio, maximum drawdown, and beta-adjusted return—are calculated to assess both profitability and risk-adjusted performance relative to benchmarks.

Implementation and Real-Time Execution

For live trading, the system continuously fetches and processes real-time data to update predictions. Trade execution is automated through brokerage APIs, ensuring efficiency. A dynamic risk management framework is also incorporated, applying stop-loss and take-profit thresholds, along with volatility-adjusted position sizing, to limit potential losses.

Ongoing Monitoring and Model Updating

Finally, the system integrates monitoring and maintenance mechanisms. Live trading outcomes are tracked against historical benchmarks to detect deviations. The SVM is periodically retrained using the latest data to adapt to evolving market dynamics. A feedback loop further refines trading logic and model parameters, improving robustness and adaptability over time.

5.1.2 Unsupervised Learning

Unsupervised learning is a type of ML in which algorithms are trained using input data without corresponding output labels. The primary objective is to uncover the underlying structure or distribution of the data, thereby gaining insights into hidden patterns and relationships. This paradigm is particularly useful when the outcomes are unknown in advance and the goal is exploratory analysis rather than direct prediction.

Unsupervised learning tasks are typically categorized into clustering and dimensionality reduction. Clustering algorithms, such as k-means or DBSCAN, aim to identify inherent groups within the data, assigning unlabeled points to meaningful subgroups based on similarity. This process is widely applied in finance for tasks such as grouping assets with similar dynamics or risk profiles. Dimensionality reduction techniques, most notably PCA, serve to project high-dimensional financial data into lower-dimensional representations while retaining the most informative variance components. Although association rule mining is also a branch of unsupervised learning, its applications are more prevalent in domains such as retail and recommendation systems rather than financial markets.

The reviewed literature, as summarized in Table 11 and 14, shows that DBSCAN and PCA are the dominant unsupervised approaches adopted. Specifically, DBSCAN has been implemented in four studies, accounting for approximately 6% of applications, while PCA has been applied in two studies, representing a smaller proportion. Although their overall usage is limited compared to supervised learning models, these methods play an important role in enhancing pairs trading strategies. DBSCAN contributes by identifying asset clusters with strong co-movement properties, while PCA facilitates noise reduction and the extraction of latent factors driving asset returns. The following sections explore the integration of DBSCAN in pairs trading, critically evaluating its effectiveness and distinct contributions to strategy refinement.

5.1.2.1 Introduction to Density-Based Spatial Clustering of Applications with Noise

Developed in 1996 by Ester et al. (1996), DBSCAN is a widely recognized clustering algorithm that groups points according to the density of their neighborhoods. Unlike partition-based methods such as k-means, DBSCAN distinguishes between dense regions and sparse regions, treating the latter as noise. As a non-parametric, density-based method, it does not require prior knowledge of the number of clusters, which makes it particularly well suited for applications involving complex data distributions.

The algorithm relies on two key parameters: ε (epsilon), which defines the radius of the neighborhood around a point, and minPts, which specifies the minimum number of neighboring points required for a region to be considered dense. Formally, for a given dataset D, the ε-neighborhood of a point p is defined as:

$$N_{\varepsilon}(p) = \{q \in D \mid dist(p, q) \leq \varepsilon\}$$

A point p is classified as a core point if

$$N_{\varepsilon}(p) \geq \text{minPts}$$

Based on these definitions, DBSCAN categorizes points as follows. A point q is said to be directly reachable from p if $q \in N_{\varepsilon}(p)$ and p is a core point. A point q is reachable from p if there exists a chain of points $p_1, \ldots, p_n$ with $p_1 = q$ and $p_n = q$, where each point is directly reachable from the previous one. Points that are not reachable from any other point are designated as outliers or noise.

In practice, this means that a cluster is formed by a core point together with all points (both core and non-core) that are reachable from it. Each cluster must contain at least one core point, while non-core points may belong to a cluster but lie on its “edge,” as they cannot extend reachability further.

The overall DBSCAN procedure unfolds in three stages. First, the ε-neighborhood of each point is evaluated to determine whether it qualifies as a core point. Second, a connectivity graph is constructed by linking core points that lie within each other’s ε-neighborhoods, thereby initiating cluster formation. Finally, non-core points are assigned to clusters if they fall within the ε-radius of any core point; otherwise, they are labeled as noise. Through this iterative process, DBSCAN is able to discover clusters of arbitrary shapes and effectively separate noise without requiring the number of clusters as an input.

5.1.2.2 Density-Based Spatial Clustering of Applications with Noise in Pair Trading

In the context of pair trading, DBSCAN can be effectively utilized to identify groups of stocks that exhibit similar movement patterns over time. It plays a critical role in the preliminary stages of strategy development by clustering stocks based on their historical price behavior, which allows traders to systematically filter potential trading pairs from large datasets.

The study presented in article [9] introduces a novel pair trading framework that incorporates DBSCAN, moving beyond traditional strategies that rely solely on price data by integrating firm-level characteristics. This approach, applied to the U.S. stock market over a 40-year period, demonstrates superior performance in terms of annualized returns and Sharpe ratios. The strategy remains profitable even after accounting for transaction costs and exhibits robustness against data-snooping concerns.

How DBSCAN Works

The application of DBSCAN in pair trading begins with feature selection. In this step, relevant features are derived from historical return data, augmented by firm-level characteristics such as size, book-to-market ratio, and other accounting-based attributes. Incorporating both price dynamics and firm-specific fundamentals enables the formation of groups consisting of stocks with historically similar movements and comparable structural features, thereby reducing the likelihood of spurious pairings.

A suitable distance metric is then required to quantify similarity between stocks. In article [9], cosine similarity is employed to capture the relative alignment of stocks' return vectors in the feature space. Based on this measure, DBSCAN is applied to identify clusters of stocks with common patterns. A known challenge of this method is the imbalance in cluster sizes, where one large high-density cluster may dominate, leaving many stocks classified as outliers. This highlights the importance of parameter calibration, as the number and structure of clusters have a direct impact on the trading strategy.

Once the distance metric is established, DBSCAN is applied to the feature set to identify clusters. A known challenge of this method is the imbalance in cluster sizes, where a dominant high-density cluster often emerges, leaving the remaining stocks classified as outliers. This phenomenon reflects the concentration of similarities in the data but also underscores the need for careful tuning of DBSCAN parameters. The number and structure of resulting clusters can significantly influence the strategy's performance, making parameter calibration essential. In financial contexts, this clustering process is particularly valuable, as it helps differentiate between common price movements and anomalous behaviors.

After clustering, stock pairs are selected from within the same group, as they are expected to share common dynamics. Trading rules are then based on one-month return differentials: stocks with relatively lower returns are treated as undervalued, while those with higher returns are considered overvalued. A contrarian strategy is employed, buying the undervalued stock and shorting the overvalued one when the spread exceeds one cross-sectional standard deviation of the past month's return difference. Positions are rebalanced monthly. The performance of this strategy is summarized in Table 16.

Table 16. Annualized risk-return metrics with DBSCAN in article [9].

DBSCAN	Long	Short	Long-Short
Mean return	0.291	0.053	0.238
Standard deviation	0.244	0.188	0.152
Sharpe ratio	1.195	0.281	1.571
t-statistic	2.296	2.215	2.875
Downside deviation	0.171	0.162	0.066
Sortino ratio	1.703	0.327	3.591
Gross profit	17.544	9.051	12.733
Gross loss	-5.603	-6.882	-2.96
Profit factor	3.131	1.315	4.302
Profitable years	37	27	38
Unprofitable years	4	14	3
Maximum drawdown	-0.477	-0.624	-0.129
Calmar ratio	0.611	0.085	1.846
Turnover	0.932	0.981	1.913

Also, article [9] conducted robustness tests, in Table 17, to assess the sensitivity to clustering parameters and to ensure results are not driven by data snooping.

Table 17. Parameter sensitivity of the strategies, DBSCAN with different percentiles (α) for the outlier threshold in article [9].

α	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Mean return	0.256	0.2	0.173	0.154	0.148	0.148	0.138	0.134	0.129
Sharpe ratio	2.039	1.763	1.526	1.328	1.19	1.142	1.036	0.954	0.867
Maximum drawdown	-0.149	-0.146	-0.163	-0.206	-0.197	-0.242	-0.238	-0.285	-0.332
Number of clusters	2	2	2	2	2	2	2	2	1
Number of stocks in clusters	376(12.05)	749(24.06)	1092(35.02)	1418(45.44)	1729(55.41)	2034(65.14)	2328(74.56)	2609(83.54)	2878(92.2)
Number of outliers	2781(87.95)	2408(75.94)	2065(64.98)	1739(54.56)	1428(44.59)	1123(34.86)	829(25.44)	548(16.46)	279(7.8)
Number of stocks traded	75(2.47)	188(6.17)	314(10.18)	450(14.52)	594(19.09)	743(23.82)	896(28.67)	1053(33.67)	1217(38.86)

Practical Implementation Summarize

In practical implementation, the process begins with data preparation. This involves collecting and preprocessing historical price data for a broad set of stocks, ensuring that the dataset is clean and consistent for analysis. Once the data is prepared, attention shifts to parameter selection. The two key parameters of DBSCAN—the neighborhood radius (epsilon) and the minimum number of points required to form a dense region (min_samples)—must be carefully tuned, as they critically determine the clustering results and, by extension, the effectiveness of the trading strategy.

Following parameter calibration, DBSCAN is executed on the processed dataset to identify meaningful stock clusters. These clusters are then examined to select potential trading pairs based on their historical co-movements. To validate the strategy's performance, back-testing is conducted using historical data, evaluating both feasibility and profitability of the identified trading opportunities.

Benefits of Using DBSCAN in Pair Trading

There are several notable advantages to applying DBSCAN in pair trading. One significant advantage is efficiency. The algorithm reduces the dimensionality of the trading universe by filtering out stocks that do not exhibit strong similarities, thereby focusing attention on the most promising pair candidates. Moreover, DBSCAN's ability to detect and exclude outliers improves the robustness of the strategy by eliminating stocks that might introduce excessive noise or volatility. Another important strength lies in its adaptability: unlike many clustering methods, DBSCAN does not require a predefined number of clusters, making it well-suited to dynamic financial markets where structural assumptions are often unreliable.

Challenges and Considerations

Despite these advantages, DBSCAN also presents challenges. A key issue is parameter sensitivity, as the selection of epsilon and min_samples substantially influences the clustering outcomes and often requires adjustment in response to changing market conditions. Furthermore, financial markets are inherently noisy and influenced by numerous unpredictable factors. This external volatility can reduce the consistency and reliability of clusters generated by DBSCAN, particularly during periods of structural breaks or regime shifts.

5.2 Deep Learning

DL refers to a subset of machine learning techniques that employ artificial neural networks (ANNs) with multiple layers, where the term "deep" reflects the use of these hierarchical layers. Such architectures enable representation learning, allowing the model to capture and exploit abstract patterns in data. The most common DL architectures include DNNs, DBNs, RNNs, CNNs, and transformers. These models are highly versatile and can be trained under supervised, semi-supervised, or unsupervised paradigms.

DL has been successfully applied across a wide spectrum of fields. In computer vision and speech recognition, it enables machines to interpret visual and auditory signals. In NLP and machine translation, it facilitates communication and understanding between humans and machines. In bioinformatics and drug design, DL aids in predicting molecular activities and interactions, while in medical imaging, it assists in the accurate diagnosis of diseases. Furthermore, DL contributes to climate modeling and prediction, material inspection for quality control, and has even been employed to master complex board games, frequently achieving or surpassing human-level performance.

In quantitative finance, and particularly in quantitative trading, DL has also gained increasing attention. In the subsequent sections, I will examine in detail how DL is applied to pairs trading. The discussion expands upon the methodologies briefly referenced in the literature, with a particular emphasis on the techniques summarized in Table 12 and 14.

The evidence from Table 12 and 14 clearly indicates that LSTM and CNN are the most frequently applied DL architectures in the reviewed works. Specifically, LSTM was utilized in eleven studies, accounting for 16% of applications, whereas CNN was adopted in four studies, representing 6% of the sample. These findings highlight a strong reliance on these two methods within the academic community. The following sections will further analyze the application of LSTM in pairs trading, evaluating its effectiveness and illustrating the role it plays in enhancing trading strategies.

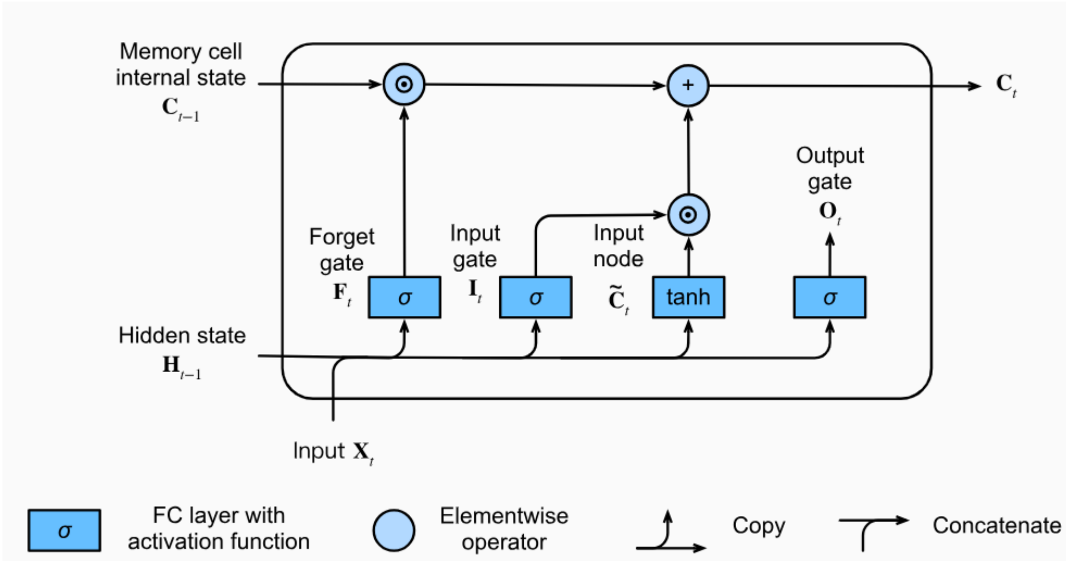
5.2.1 Introduction to Long Short-Term Memory

RNNs are designed to process sequential data, distinguishing them from standard feedforward neural networks that treat inputs as independent. By maintaining a hidden state, RNNs are able to capture temporal dependencies, making them suitable for tasks where the meaning of current inputs depends on prior context—for example, understanding how the interpretation of a word depends on its surrounding text.

LSTM networks are a special class of RNNs, shown in Figure 5, developed to mitigate the issues of vanishing and exploding gradients that frequently arise when training over long sequences. Compared to conventional RNNs, LSTMs are more effective at modeling long-term dependencies, making them a widely adopted solution in sequence modeling tasks. A schematic of the LSTM architecture is presented in Figure 5.

What differentiates LSTMs from traditional RNNs is the incorporation of memory cells, which allow information to be stored and accessed over extended periods. Each LSTM unit consists of a cell, an input gate, a forget gate, and an output gate. These gates regulate the flow of information into and out of the memory cell, controlling how the cell state evolves across time steps.

Figure 5. Schematic Representation of the LSTM Model.



Source: Zhao, W. and Fan, L. (2024). Short-Term Load Forecasting Method for Industrial Buildings Based on Signal Decomposition and Composite Prediction Model

The cell state C_t is the central component of the LSTM, functioning like a conveyor belt that runs through the entire sequence of units. It allows information to be carried forward largely unchanged, with selective modifications introduced by the gates. This enables the network to preserve essential information while discarding irrelevant details.

The forget gate f_t determines which parts of the previous cell state should be discarded. It takes as input the previous hidden state h_{t-1} and the current input x_t, passing them through a sigmoid function:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{5.5}$$

where σ denotes the sigmoid activation that outputs values between 0 and 1, W_f is the weight matrix of the forget gate, b_f is the bias term, h_{t-1} is the hidden state from the previous time step, and x_t is the current input vector. The resulting vector f_t is then applied element-wise to the previous cell state C_{t-1} to determine which components are retained.

The input gate i_t determines which values will be updated, while a $tanh$ layer generates a vector of new candidate values $\tilde{C}_t$ that may be added to the cell state. These components jointly contribute to updating the cell state:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{5.6}$$

$$\tilde{C}_t = tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{5.7}$$

Here, the input gate uses a sigmoid activation to determine which elements of $\tilde{C}_t$ are accepted. The candidate vector $\tilde{C}_t$, produced by the $tanh$ transformation, represents potential updates to the cell state, thereby allowing flexible integration of new information through non-linear transformations.

The old cell state C_{t-1} is then updated to the new state C_t. Specifically, the forget gate's output scales the old state (controlling what is retained), while the input gate's output scales the candidate values (controlling what is updated):

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{5.8}$$

This formulation combines the previous cell state C_{t-1}, modulated by f_t, with the candidate values $\tilde{C}_t$, modulated by i_t. Such selective updating enables LSTMs to propagate relevant information across long sequences while mitigating the risks of gradient vanishing or explosion.

Finally, the output gate o_t determines the next hidden state h_t. The hidden state encapsulates information from prior inputs and is also used for predictions. A sigmoid layer regulates which portions of the current cell state contribute to the output. The cell state is then passed through a $tanh$ function, constraining values to the range [−1, 1], and multiplied element-wise by the gate's output:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{5.9}$$

$$h_t = o_t \odot tanh(C_t) \tag{5.10}$$

Here, the output gate o_t filters the cell state C_t to determine the information carried forward to the next hidden state h_t, and potentially to the output layer of the network. By applying the $tanh$ activation, this step ensures non-linear scaling and controlled representation of the state information.

The detailed dynamics and operational flow of LSTMs are as follows. Each gate within the architecture serves a distinct role in managing the flow of information. By preserving valuable historical data, incorporating relevant new inputs, and discarding redundant or obsolete information, LSTMs are able to maintain stable performance across sequences of varying lengths.

One of the key advantages of LSTMs is their ability to regulate gradient flow during backpropagation. The gating mechanisms help mitigate the well-known issues of vanishing and exploding gradients by selectively controlling how much information—and thus how much gradient—is propagated through each time step. This selective control ensures that critical signals are preserved during training while minimizing instability.

Moreover, the structure of LSTMs allows them to effectively capture long-term dependencies. Their ability to determine what information should be retained or forgotten makes them particularly well-suited for tasks where historical context strongly influences current outcomes. A common example is language modeling, in which earlier words in a sentence provide essential context for predicting subsequent words.

Through the integration of these mechanisms, LSTM networks provide a powerful and flexible framework for modeling sequential data. Their capacity to preserve temporal context, adaptively manage memory, and maintain gradient stability makes them a preferred choice for a wide range of applications, including time-series analysis, natural language processing, and financial prediction, where long-term dependencies play a critical role.

5.2.2 Long Short-Term Memory in Pair Trading

Using LSTM networks in pairs trading involves leveraging their ability to remember and utilize historical financial data over extended periods, which is crucial for identifying and exploiting market inefficiencies between pairs of stocks, commodities, or other financial instruments.

The paper [38] provides a detailed methodology for using LSTM networks to identify and capitalize on pairs-trading opportunities. Here are the specific steps based on the information from the article.

Data Collection

The process begins with data collection. The study focuses on stocks within the S&P 500 index, which are updated annually to reflect changes in the index composition. Daily price data, including open, high, low, close, and volume, are gathered for the period from January 2000 to June 2019. Data retrieval is typically performed via APIs like yfinance, and adjusted close prices are used to account for dividends and stock splits.

Data Preprocessing

In the preprocessing stage, the authors apply Z-score normalization, standardizing each stock's price series using its mean and standard deviation. Cointegration tests are then conducted using tools such as the Engle-Granger or Johansen procedures, implemented in Python or R, to identify stock pairs with statistically significant long-term equilibrium relationships—an essential condition for effective pairs trading.

LSTM Model Development

For model development, the LSTM network takes several input features, including price gaps, trading volumes, and returns. Price gaps are calculated as the difference between a stock's price and the average price of its cointegration group. Trading volume data is included to capture liquidity fluctuations, which can influence the persistence or reversal of price gaps. Historical returns are incorporated to help the model recognize trend and volatility dynamics.

Importantly, the task is formulated as a three-class classification problem: predicting whether the price gap will expand, shrink, or remain stable. The LSTM outputs these probabilities via a SoftMax activation function, which are later converted into trading signals.

Model Training

For training, historical sequences of fixed lengths—such as 240 trading days—are fed into the model. The training process uses backpropagation through time (BPTT), with model performance validated on a reserved portion of the data to avoid overfitting. Optimizers such as RMSprop or Adam are employed due to their effectiveness in handling stochastic updates in sequential models, while the loss function is categorical cross-entropy, suitable for multi-class classification tasks.

Prediction

After training, the LSTM model is used to generate predictions based on new data. This involves feeding in recent sequences of stock information and interpreting the SoftMax output to derive trading signals. For instance, a high predicted probability that a price gap will widen could trigger a long position in the underperforming stock and a short position in the outperforming one.

Strategy Implementation

The signals produced by the model are then used to construct a portfolio. Positions are opened based on the model's directional predictions, with appropriate risk and money management controls in place. These controls may include stop-loss orders, exposure limits, and position sizing rules based on the model's confidence scores.

Back-testing and Optimization

The strategy is rigorously evaluated through back-testing on historical data. This simulation assesses the model's performance and helps optimize key parameters, such as the lookback window, trading thresholds, and the architecture of the LSTM model itself. Adjustments are made based on observed performance to refine both the model and the strategy. The results are benchmarked against traditional cointegration-based strategies, showing that the LSTM consistently achieves higher Sharpe ratios and profitability by better capturing nonlinear and temporal dependencies.

Execution

Finally, although the original study evaluates the method in a back-testing framework rather than live deployment, the proposed approach could in principle be extended to execution in a real-time environment. Such implementation would require integration with live data feeds, broker APIs, and automated trading systems. Continuous retraining on new data would allow the model to adapt to evolving market conditions, ensuring robustness and consistency in performance.

Overall, the study demonstrates that LSTM networks significantly enhance the identification of profitable trading pairs. Compared with benchmark models, the LSTM-based strategy achieves superior performance in terms of annualized returns, Sharpe ratio, and hit ratio, highlighting its potential as a powerful tool in systematic pairs trading.

5.3 Reinforcement Learning

RL is a pivotal area at the intersection of machine learning and control theory, centered on how an intelligent agent can learn optimal strategies through interactions with a dynamic environment. As one of the three foundational paradigms of machine learning—alongside supervised and unsupervised learning—RL is distinct in its reliance on trial-and-error learning rather than labeled data or direct guidance.

In contrast to supervised learning, RL does not require predefined input–output pairs or explicit feedback after each action. Instead, it focuses on maximizing cumulative rewards over time by balancing two competing objectives: exploration (trying new actions to discover their potential) and exploitation (choosing known actions that yield high rewards). This balance makes RL particularly suitable for complex environments where feedback may be delayed, indirect, or noisy.

Many RL problems are formally modeled as Markov Decision Processes (MDPs), which provide a rigorous mathematical framework for sequential decision-making under uncertainty. While classical dynamic programming techniques assume full knowledge of the MDP's transition dynamics and reward structure, modern RL algorithms are capable of learning effective policies without such prior information. This flexibility allows RL to be applied in large-scale or partially observed environments where traditional approaches are infeasible.

Formally, an MDP is defined as:

$$M = (S, A, P, R, \gamma) \tag{5.11}$$

where S denotes the state space, A is the action space, P is the transition probability of moving from state s to s' given action a, $R(s, a)$ is the reward function, and $\gamma \in [0,1)$ is the discount factor balancing immediate and future rewards.

In the context of pairs trading, this mapping can be specified as follows:

- States (S): feature vectors representing market conditions, such as normalized spreads between stock pairs, historical moving averages, volatility, and trading volume.
- Actions (A): discrete trading decisions, including going long the spread, shorting the spread, closing positions, or remaining inactive.
- Transition Dynamics (P): the stochastic evolution of market prices, typically unknown and learned implicitly through experience.
- Rewards (R): realized profits or losses, adjusted for transaction costs and risk penalties such as volatility or drawdowns.
- Discount Factor (γ): represents the trader's preference for short-term versus long-term profitability.

A central concept in RL is the action-value function, or Q-function, which represents the expected cumulative discounted reward when starting from a given state s, taking an action a, and thereafter following policy π:

$$Q^{\pi}(s, a) = \mathbb{E}_{\pi}[\textstyle\sum_{t=0}^{\infty} \gamma^{t} R(s_t, a_t) | s_0 = s, a_0 = a] \tag{5.12}$$

The optimal Q-function satisfies the Bellman optimality equation:

$$Q^*(s,a) = \mathbb{E}\left[R(s,a) + \gamma \max_{a'} Q^*(s',a') \middle| s,a\right] \tag{5.13}$$

where s' denotes the next state after taking action a in state s. This recursive structure provides the theoretical foundation for modern RL algorithms such as DQNs, which approximate $Q^*(s,a)$ using deep neural networks.

Quantitative trading, and pairs trading in particular, can be naturally framed as an RL problem. Trading decisions are inherently sequential: at each time step, the agent evaluates the current market state (e.g., spread deviation, volatility) and selects an action (e.g., long, short, hold). The outcomes of these actions generate rewards (profits or losses), which in turn guide the refinement of the trading policy.

The data extracted from Table 13 and 14 indicates that DQN and deep reinforcement learning (DRL) methods more broadly are the most frequently adopted approaches in this domain. Specifically, DQN was applied in six studies (9% of the reviewed literature), while DRL was used in five papers (7%). This prevalence suggests that these methods are increasingly favored by researchers for developing adaptive and autonomous trading strategies.

5.3.1 Introduction to Deep Q-network

RL operates through the interaction between an agent, a set of states S, and a set of possible actions A that the agent can take in each state. When the agent performs an action $a \in A$, it transitions to a new state and receives a reward, a numerical signal that reflects the immediate utility of that action. The ultimate objective of the agent is to learn a policy that maximizes cumulative rewards over time.

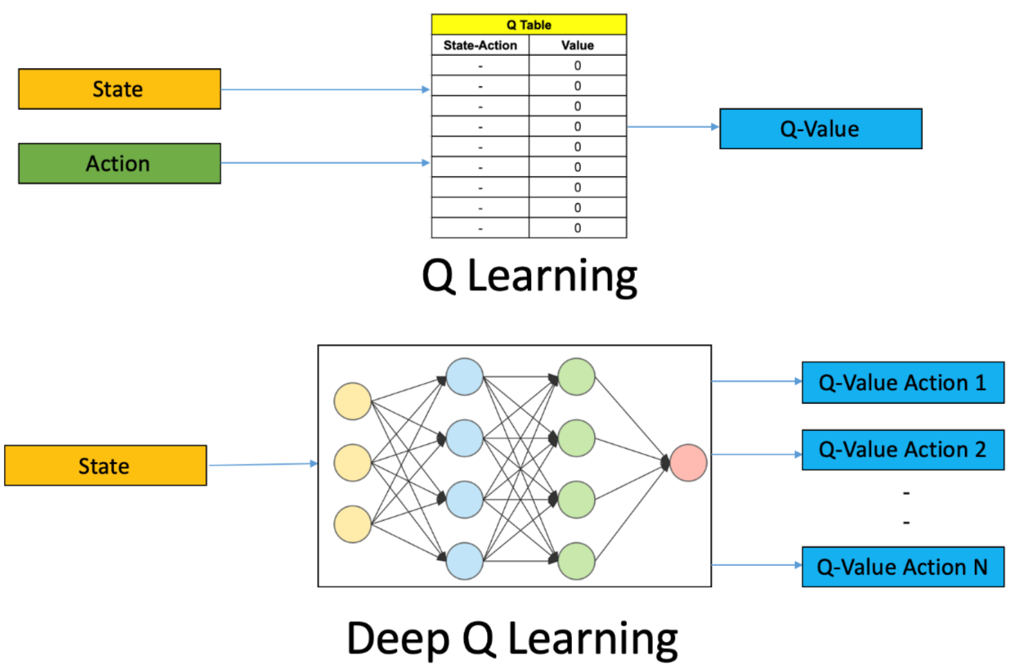
To achieve this goal, the agent must evaluate the potential future rewards that may result from its current decisions. These rewards are typically represented as the expected value of discounted returns from subsequent states, allowing the agent to balance immediate gains against long-term benefits. One of the most widely used algorithms in this context is Q-learning, a model-free RL technique designed to estimate the value of taking a particular action in a given state. The framework of Q-learning is illustrated in Figure 6.

A key advantage of Q-learning is that it does not require prior knowledge of the environment's transition dynamics—hence the term model-free. It can effectively handle environments with stochastic transitions and rewards, and, given sufficient exploration and a partially random policy, it is guaranteed to converge to the optimal policy in any finite MDP. At its core, Q-learning relies on the computation of a Q-function, which estimates the expected cumulative reward of performing an action in a given state and then following the optimal policy thereafter.

While Q-learning performs well in small, discrete state–action spaces, it faces significant challenges in high-dimensional or continuous domains due to its reliance on lookup tables. In such settings, maintaining and updating a complete Q-table becomes computationally infeasible. This limitation motivated the use of function approximation, where a parameterized function $Q_{(s,\ a)}$ is employed to estimate Q-values. Here, θ denotes the parameters of the function, typically learned through gradient-based optimization.

This development gave rise to the DQN, which integrates Q-learning with DNNs to approximate the Q-function. By leveraging the representational power of DNNs, DQN significantly enhances the scalability of Q-learning, enabling it to learn policies directly from raw, high-dimensional input data—such as images or long sequences—without explicit feature engineering. Figure 6 also depicts the structure of a typical Deep Q-learning setup, in which the neural network processes state representations and outputs Q-values for all possible actions.

Figure 6. Schematic Representation of Q-learning and Deep Q-learning**.**



Source: [A Hands-On Introduction to Deep Q-Learning using OpenAI Gym in Python]

Therefore, Deep Q-learning, as an extension of traditional Q-learning integrated with DNNs, has fundamentally transformed the way high-dimensional observation spaces are handled. By leveraging the representational power of DNNs, this method enables the learning of optimal policies directly from raw sensory data—an otherwise intractable task for classical Q-learning, which relies on discrete state–action space representations. The following provides a more detailed overview of the core components and operations of Deep Q-learning.

Q-learning Review

At its core, Q-learning seeks to learn the action-value function $Q(s,a)$, which represents the expected utility of taking an action a in a state s and subsequently following the optimal policy. The objective is to maximize the sum of rewards, discounted over time, by updating Q-values according to the Bellman equation:

$$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)] \quad (5.14)$$

Here, α is the learning rate, γ is the discount factor, r is the reward, s' is the next state, and a' is a possible action in state s'.

Deep Neural Network as a Function Approximator

In deep Q-learning, a DNN—often referred to as the Q-network—is used to approximate the Q-function. The network takes the state$s$as input and outputs a vector of action values $Q(s,\cdot)$ for all available actions. This approach mitigates the curse of dimensionality by replacing a discrete Q-table with a parameterized function that generalizes across similar states.

Experience Replay

To break harmful correlations between consecutive updates, deep Q-learning employs experience replay. Transitions (s,a,r,s') are stored in a replay buffer, and the algorithm randomly samples mini-batches from this buffer to train the network. Such stochastic sampling approximates i.i.d. training data and stabilizes learning by exposing the network to a diverse set of past experiences.

Fixed Q-targets

A central challenge in training neural networks for RL is the moving-target problem, where continuously changing targets can induce oscillations or divergence. Deep Q-learning addresses this by introducing a separate target network to compute target Q-values. The target network shares the architecture of the Q-network but its parameters are updated only intermittently (e.g., every few thousand steps), thereby providing a more stable target for Q-updates.

Loss Function and Optimization

The loss function typically minimizes the mean squared temporal-difference error between predicted Q-values and target Q-values:

$$L(\theta) = E_{(s,a,r,s')\sim replay\ buffer}\,[(r + \gamma \boldsymbol{\max_{a'}} Q(s', a'; \theta^-) - Q(s, a; \theta))^2] \quad (5.15)$$

where θ are the parameters of the Q-network and θ^- are the (periodically updated) parameters of the target network. This loss is optimized with standard stochastic gradient methods, such as SGD or Adam.

Practical Challenges and Enhancements

In practice, deep Q-learning is highly sensitive to hyperparameter choices, such as the learning rate, replay buffer size, batch size, and the update frequency of the target network. To mitigate these sensitivities and enhance both stability and performance, several extensions have been proposed, including Double DQN, which reduces overestimation bias; Dueling DQN, which separately estimates state values and action advantages; and Prioritized Experience Replay, which improves sample efficiency by replaying more informative transitions.

Collectively, these refinements strengthen the robustness and scalability of the base DQN framework, enabling it to efficiently learn policies in complex, high-dimensional environments. As a result, deep Q-learning has been successfully applied to a wide range of domains, from achieving superhuman performance in video games to enabling autonomous control in real-world robotic systems.

5.3.2 Deep Q-network in pair trading

To incorporate DQN into pairs trading, one must develop a sophisticated approach that leverages DRL to optimize sequential decision-making processes. This paper outlines a comprehensive method for applying DQNs in pairs trading, including the setup of the trading environment, the design of the network architecture, and the fine-tuning of the learning algorithms.

Paper [53] outlines a structured approach for employing DQN networks to pinpoint and leverage opportunities in pairs trading. The following are detailed procedures drawn from the insights of the article.

Defining the Trading Strategy and Environment Setup

Pairs trading relies on identifying two cointegrated stocks whose price spread exhibits mean-reverting behavior. In this framework, a DQN is trained to learn the temporal dynamics of the spread and to determine optimal entry and exit points. The experimental setup begins with the S&P 500 universe, from which approximately 78,000 potential stock pairs are generated. An Augmented Dickey–Fuller (ADF) test is applied, and pairs with p-values ≤ 0.05 are retained, yielding 145 candidates. To further ensure sufficient volatility, pairs are filtered by requiring a ratio of standard deviation to mean (std/mean) ≥ 0.5, resulting in a final set of 38 pairs. The trading environment is implemented using the OpenAI Gym framework, which provides a standardized interface for reinforcement learning tasks. A custom environment is created to simulate the trading process, exposing the agent to a set of state observations that characterize the behavior of each selected stock pair.

Data Handling and Feature Engineering

The state space of the DQN consists of ten standardized features designed to represent the mean-reversion behavior of the spread. These features include: (i) the current spread between the two stocks, (ii) the daily spread return, (iii) moving averages of the spread at multiple horizons, and (iv) ratios of the spread to its moving averages over the same horizons. Standardization ensures that all features have mean zero and unit variance, preventing scale imbalances and allowing the DQN to focus on learning the underlying dynamics of the spread.

Defining Actions and Reward Mechanism

The DQN selects one of three possible actions based on the observed state, aiming to maximize the expected cumulative reward by exploiting the mean-reversion characteristic of the price spread. The action space includes initiating a long position—buying the underperforming stock and shorting the outperforming one—when the spread exceeds an upper threshold. Conversely, when the spread falls below a lower threshold, the model takes a short position by selling the underperforming stock and buying the outperforming one. If the spread lies within the thresholds, indicating no strong trading signal, the DQN opts to hold and take no action.

The reward function is designed to align with profitability. At each time step, the immediate reward is defined as the profit or loss generated from the chosen action, directly reflecting the trading outcome.

Network Architecture and Training Process

The DQN architecture is composed of an input layer, two hidden layers, and an output layer. The input layer receives ten features representing the state of the trading environment, including the spread between the two stocks, moving averages, and return-based measures. The two hidden layers each contain 50 neurons and employ Rectified Linear Unit (ReLU) activation functions to capture nonlinear relationships in the data. The output layer produces Q-values corresponding to the three possible actions (long, short, hold), thereby mapping observed states to potential trading decisions.

To enhance learning stability, the training process incorporates experience replay. Transition tuples—comprising the current state, the action taken, the resulting reward, and the next state—are stored in a replay buffer. Mini-batches are randomly sampled from this buffer during training, which breaks correlations between consecutive observations and improves generalization across diverse scenarios.

The Q-values are updated using a mean squared error (MSE) loss, which compares predicted Q-values with target values derived from the Bellman equation:

$$Q_{target} = r + \gamma \boldsymbol{\max_{a'}} Q(s', a')$$

where γ is the discount factor emphasizing the importance of future rewards. The network parameters are optimized using the Adam optimizer, chosen for its adaptive learning rate and efficiency in handling noisy financial data.

Evaluation and Continuous Learning

After training, the DQN is evaluated on out-of-sample data, specifically using market data from 2018. The evaluation primarily focuses on cumulative returns and Sharpe ratios, which measure profitability and risk-adjusted performance. Robustness is assessed by testing the model under different market conditions to ensure consistent behavior.

Based on the evaluation results, further optimization is achieved by fine-tuning hyperparameters such as the learning rate, replay buffer size, and network architecture. Iterative retraining ensures that the model adapts to new market dynamics.

Through careful trading environment design, systematic data preprocessing, and continuous refinement of the neural network, the DQN-based approach to pairs trading provides a powerful framework for exploiting mean-reversion opportunities. By capturing temporal dependencies in spreads and adapting to evolving market conditions, it enhances profitability while maintaining robust risk control in dynamic financial markets.

6. Evaluation Metric

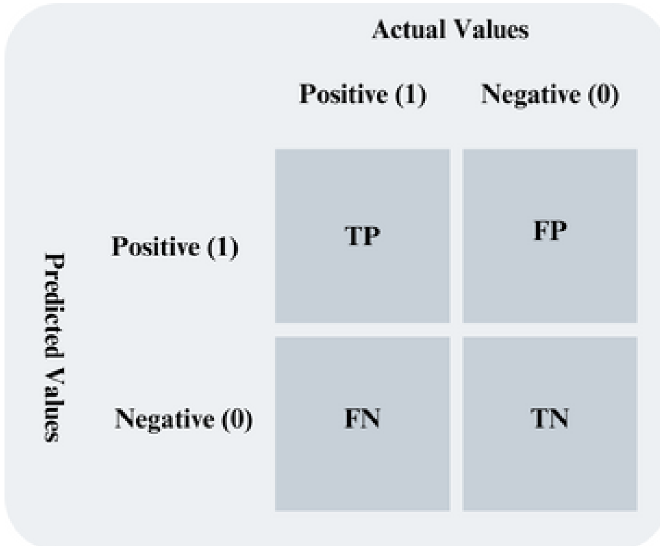
Two distinct categories of performance evaluation metrics are commonly employed. The first category assesses the predictive effectiveness of the model (as shown in Table 18), while the second evaluates portfolio performance (reported in Table 19). This section first focuses on model evaluation.

To measure the accuracy of classification models on a given test dataset, a confusion matrix is widely used. Figure 7 presents a confusion matrix for binary classification, where outcomes are categorized as either positive or negative. In the context of pairs trading, the confusion matrix provides valuable insights into the effectiveness of classification-based strategies, such as predicting whether the spread between two assets will converge or diverge.

A confusion matrix is a structured tabular representation that compares predicted outcomes with actual values, thereby enabling the calculation of performance metrics such as accuracy, precision, recall, and specificity. For a binary classification problem, it consists of two rows and two columns reporting the following cases:

- True Positives (TP): instances where the model correctly predicts the positive class.
- True Negatives (TN): instances where the model correctly predicts the negative class.
- False Positives (FP): instances where the model incorrectly predicts the positive class (Type I error).
- False Negatives (FN): instances where the model fails to predict the positive class (Type II error).

Figure 7. Confusion matrix.



6.1 Evaluation Metrics of Efficacy

In assessing the efficacy of models produced by machine learning algorithms, it is crucial to apply appropriate evaluation metrics that provide insights into predictive accuracy and reliability. These metrics allow for an evaluation of how well the models generalize to unseen data, based on known outcomes. Let Y_i denote the $i - th$ actual value and $\hat{Y}_i$ the $i - th$ predicted value, where N is the total number of predictions.

Table 18 summarizes the evaluation metrics frequently adopted in the reviewed literature. These metrics can be broadly categorized into error-based measures (e.g., MAE, MSE, RMSE, MAPE), which quantify the deviation between predicted and actual values, and classification-based measures (e.g., Accuracy, Precision, Recall, F1-score, AUC), which evaluate the ability of models to correctly classify outcomes. Error-based metrics are particularly useful for continuous prediction tasks such as forecasting spreads or returns, while classification-based metrics are more appropriate when strategies are framed as binary decisions (e.g., convergence versus divergence signals in pairs trading).

Table 18. Evaluation metrics for assessing model efficacy in reviewed articles.

Evaluation Metrics	Description	Formula	Article
Mean Absolute Error (MAE)	Measures the average magnitude of the errors in a set of predictions, without considering their direction. It is the mean over the test sample of the absolute differences between predicted values and actual values.	$MAE = \frac{1}{N}\sum_{i=1}^{N} \lvert Y_i - \hat{Y}_i \rvert$	[20, 45]
Mean Squared Error (MSE)	The average squared difference between the estimated values and the actual value. MSE is more sensitive to outliers than MAE as it squares the differences.	$MSE = \frac{1}{N}\sum_{i=1}^{N} (Y_i - \hat{Y}_i)^2$	[16, 20, 25, 37, 45, 60]
Root Mean Squared Error (RMSE)	RMSE is the square root of the mean of the squared errors. RMSE is particularly useful when large errors are particularly undesirable.	$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (Y_i - \hat{Y}_i)^2}$	[23, 28, 40, 45,56, 65]
Mean Absolute Percentage Error (MAPE)	MAPE is a measure of prediction accuracy of a forecasting method in statistics. It usually expresses the accuracy as a ratio.	$MAPE = \frac{100\%}{N}\sum_{i=1}^{N} \lvert \frac{Y_i - \hat{Y}_i}{Y_i} \rvert$	[16, 28, 31, 37, 40, 60]
Accuracy	Measures the overall effectiveness of the model in predicting correct outcomes.	$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$	[14, 24, 51, 52, 64]
Error rate	It quantifies the frequency at which the model accurately forecasts the outcome.	$Error\ rate = \frac{FP + FN}{TP + TN + FP + FN}$	[48]
Precision	Evaluates the reliability of the model in predicting positive (converging) outcomes.	$Precision = \frac{TP}{TP + FP}$	[14, 24, 52, 64]
Recall	Assesses the model's capability to identify actual positive (converging) instances.	$Recall = \frac{TP}{TP + FN}$	[14, 24, 52, 64]
F1-Score	Balances precision and recall, particularly useful when the cost of false positives and false negatives are high.	$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$	[14, 24, 52, 64]
AUC	AUC represents the area under the ROC curve and ranges from 0 to 1. The higher the AUC value, the better the classification performance of the model.	N.A.	[30, 64]

6.2 Performance of Evaluation Metrics

Evaluating the performance of an investment portfolio is a critical step in the investment decision-making process. The primary dimensions of this evaluation are return and risk, which together form the foundation for assessing portfolio outcomes. Within these dimensions, a variety of specific indicators have been employed in the literature to capture different aspects of portfolio behavior.

Table 19 summarizes the most frequently adopted performance and risk measures. Return-based indicators, such as Accumulated Return and Annual Return, provide a direct measure of profitability, while risk-adjusted measures such as the Sharpe ratio and Sortino ratio account for the variability of returns, with the latter placing greater emphasis on downside risk. On the risk dimension, Maximum Drawdown (MDD) captures the worst peak-to-trough decline over the investment horizon, making it particularly relevant for assessing vulnerability during market downturns. Volatility (standard deviation) measures overall variability, whereas higher-moment statistics such as Skewness and Kurtosis provide insights into the asymmetry and tail risks of return distributions.

Taken together, these indicators enable a comprehensive assessment of portfolio performance, balancing the dual objectives of maximizing returns while controlling for risk.

Table 19. Evaluation metrics for assessing model performance in reviewed articles.

Evaluation Metrics	Description	Article
Accumulated Return	Accumulated Return denotes the overall percentage gain or declines in the value of an investment throughout its duration. Ideally, the Accumulated Return should be positive and maximized.	[3, 6, 7, 8, 10, 17, 18, 29, 39, 42, 44, 53, 62, 65]
Annual Return	Annual return is the percentage change in the value of an investment over a one-year period, reflecting the compound rate of return earned or lost by the investment annually.	[2, 4, 5, 8, 12, 22, 25, 31, 32, 33, 35, 39, 41, 46, 47, 55, 58, 59, 63, 68]
Sharpe ratio (SR)	The Sharpe ratio is a measure used to evaluate the risk-adjusted return of an investment by comparing its excess return over the risk-free rate to the standard deviation of those returns.	[2, 3, 4, 5, 6, 8, 9, 10, 12, 13, 14, 15, 18, 20, 21, 22, 26, 29, 31, 33, 34, 38, 39, 40, 41, 43, 44, 45, 47, 49, 50, 55, 58, 62, 65, 66, 68]

Evaluation Metrics	Description	Article
Maximum Drawdown (MDD)	Maximum Drawdown is a risk measure that indicates the largest single drop from peak to trough in the value of a portfolio or an investment, before a new peak is achieved.	[2, 3, 5, 6, 8, 9, 10, 12, 15, 18, 20, 21, 22, 25, 26, 27, 29, 31, 33, 35, 38, 39, 41, 43, 45, 47, 58, 61, 64, 68]
Standard Deviation (Volatility)	quantifies the variability or dispersion of a set of data points from their mean, typically used to assess the consistency of an investment's returns.	[2, 4, 5, 8, 12, 33, 36, 38, 40, 41, 55, 58, 61, 62, 63, 65, 66, 68]
Skewness	Refers to the asymmetry in the distribution of returns for an investment, indicating whether the returns are biased towards higher or lower values than the average, which can reveal the potential for unpredictable extreme outcomes in investment performance.	[4, 58, 61, 62]
Kurtosis	Measures the "tailedness" of the return distribution of an investment, indicating the likelihood of extreme positive or negative returns compared to a normal distribution.	[4, 58, 61, 62]
Sortino ratio	Measures the excess return of an investment relative to the downward deviation, focusing specifically on the volatility of negative asset returns, thus providing a risk-adjusted measure of a portfolio's performance under negative fluctuations.	[9, 10, 20, 21, 26, 29, 41, 55, 58, 63]

7. Data Availability and Implementation

Access to reliable data sources is essential for conducting research on stock market forecasting. The internet provides a wide range of freely accessible platforms that supply historical and real-time market information. Among these, Yahoo! Finance is one of the most widely used resources, offering free access to stock quotes, market news, and international market statistics. It is particularly valuable for obtaining historical price and volume data, and was cited in 25 out of 68 reviewed studies, underscoring its prevalence in empirical research.

In addition to Yahoo! Finance, other platforms have gained traction in recent years. Kaggle, for instance, not only provides curated datasets but also hosts machine learning competitions, some of which are sponsored by quantitative firms to stimulate the development of forecasting models. GitHub serves as an important repository for open-source code and datasets, enabling the replication and extension of previous studies. Region-specific sources, such as the National Stock Exchange of India (NSE) website, provide high-quality local market data, while more general platforms like Wikipedia can supplement datasets with corporate or macroeconomic information.

Representative links to such resources include: https://finance.yahoo.com, https://github.com/, https://www.kaggle.com/, https://www.nseindia.com/, and https://www.wikipedia.org/ (accessed on 15 April 2024).

8. Empirical Result Analysis and Future Research Direction

8.1 Empirical Result Analysis of Pair Trading

Algorithmic pairs trading has become a mainstream technique in quantitative finance, attracting increasing participation from both institutional and individual investors. This intensifying competition has made arbitrage opportunities more difficult to exploit, underscoring the necessity of developing effective approaches to ensure sustainable profitability [14]. A review of the literature indicates that pairs trading remains a viable and, in many cases, profitable strategy, although a minority of studies report diminishing returns in more recent years.

Recent advances in ML have substantially enhanced the performance of pairs trading strategies. Neural networks, by capturing non-linear dependencies and uncovering complex patterns, significantly improve the predictive accuracy of trading signals [1]. Empirical results suggest that LSTM models achieve superior accuracy across most metrics, particularly in identifying positive arbitrage opportunities, and generate the highest annualized return on investment (ROI). CNNs, while less effective on arbitrage samples, perform better in detecting non-arbitrage situations, exhibiting higher precision and F-measure scores. LR tends to deliver relatively balanced outcomes across different categories. XGBoost, despite identifying fewer opportunities, achieves the highest positive average ROI, suggesting stronger performance in arbitrage timing. Overall, ML-based strategies consistently outperform traditional cointegration-based approaches in both profitability and predictive power [14].

Beyond supervised ML models, reinforcement learning and optimization frameworks have also demonstrated promise. For example, combining the Entropy Optimization Criterion (EOC) for pair selection with MADDPG for trading thresholds has yielded notable improvements in the Chinese futures market. Simulations show that Dynamic Q-Network-based pair selection improved cumulative returns by 50% relative to static benchmarks, while the EOC method added a further 23% improvement. Integrating MADDPG enhanced performance by an additional 5–15%, and combining both EOC and MADDPG resulted in a 29% improvement over static methods and 16% over DQN alone [3]. These findings highlight the potential of advanced learning-based approaches to significantly optimize pairs trading outcomes.

TCN also exhibit stronger predictive ability compared with traditional financial algorithms, translating into substantial profit gains in trading. Further extensions, such as Knowledge-Driven TCN (KDTCN) and Temporal Convolutional Attention Networks (TCAN), which incorporate natural language processing (NLP), demonstrate further accuracy improvements and suggest a fruitful direction for future research [28].

Nevertheless, not all studies report positive results. Some find that the profitability of pairs trading has significantly declined since 2003, even when fundamental similarity is considered [4]. These findings imply that market efficiency may have eroded traditional arbitrage opportunities. Moreover, the costs associated with trading spurious pairs remain substantial, reinforcing the importance of integrating fundamental information in order to identify reliable pairs.

8.2 Future Research Direction

The domain of algorithmic pairs trading continues to evolve rapidly, driven by advances in ML and DRL. While existing studies demonstrate that these methods can significantly improve predictive accuracy and profitability, future research must go beyond incremental refinements and systematically address the challenges posed by increasingly complex data environments, methodological limitations, and dynamic financial markets. A forward-looking research agenda can be organized into four interconnected dimensions: data-driven innovations, methodological innovations, market applications, and governance and validation frameworks.

Advances in ML and DRL offer substantial opportunities for refining pairs trading strategies. The application of advanced anomaly detection techniques to financial time series could strengthen risk management by identifying irregular patterns, flash crashes, or manipulative behaviors in real time. Another critical frontier is the development of adaptive models that adjust automatically to shifting market regimes, incorporating signals related to volatility, liquidity, and investor sentiment. Such dynamic models could enhance robustness across different market conditions, including crises. Reinforcement learning methods can also be extended by integrating meta-learning approaches, enabling agents to generalize across trading environments and adapt to non-stationary market dynamics more efficiently. Moreover, the emergence of quantum computing offers a novel pathway for tackling high-dimensional optimization problems in portfolio construction and risk assessment, although practical implementation remains constrained by the immaturity of current quantum hardware. Collectively, these methodological advances promise significant improvements, but they also raise concerns regarding computational complexity, sample efficiency, and interpretability.

Expanding pairs trading beyond equities toward cross-asset arbitrage is another promising direction. Identifying pricing inefficiencies between equities, commodities, currencies, and digital assets requires models capable of handling heterogeneous trading environments, liquidity conditions, and regulatory regimes. Such approaches could generate new opportunities for diversification and risk hedging. At the same time, the incorporation of game theory and agent-based modeling provides a valuable framework for simulating the strategic interactions of multiple market participants. Modeling markets as multi-agent systems may yield insights into equilibrium behaviors, competitive dynamics, and emergent liquidity patterns. However, cross-asset strategies face practical challenges related to execution, transaction costs, and heterogeneous market microstructures, while multi-agent frameworks often struggle with calibration and validation against real-world data.

As algorithmic trading strategies become increasingly sophisticated, ethical considerations and regulatory compliance must be integrated into future research agendas. Developing transparent and auditable algorithms will not only facilitate alignment with regulatory frameworks but also strengthen trust in financial markets. Automated compliance monitoring systems represent one possible solution to ensure that trading models operate within ethical and legal boundaries. Furthermore, rigorous real-world validation remains essential. While back-testing provides initial evidence of profitability, forward-testing in simulated or paper trading environments, as well as stress-testing under extreme scenarios, is critical for assessing robustness and practical viability. Ensuring reproducibility and mitigating biases—such as survivorship bias or look-ahead bias—remain ongoing challenges that require systematic attention.

By pursuing these interconnected research directions, the field of algorithmic pairs trading can evolve in both sophistication and applicability. Data-driven and methodological innovations have the potential to significantly enhance predictive power, while cross-asset applications and multi-agent frameworks can broaden the scope of arbitrage opportunities. At the same time, embedding governance and rigorous validation practices will be essential to balance technological advancement with market stability and transparency.

9. Discussion

This study has examined the evolution of pairs trading strategies, with particular attention to the growing role of AI techniques in quantitative finance. Early approaches were predominantly grounded in classical time-series methods, including cointegration and mean-reversion models, which provided a robust statistical framework for identifying arbitrage opportunities. These techniques, while foundational, were constrained by their linear assumptions and limited ability to capture the nonlinear dependencies frequently observed in financial markets.

The development of ML methods marked a significant advancement. Algorithms such as RF and SVM demonstrated improved capacity to model nonlinearities and high-dimensional interactions. In particular, SVM's kernel-based transformations enabled the identification of subtle decision boundaries in complex data. Nevertheless, these methods often required extensive feature engineering and were sensitive to parameter tuning, which limited their scalability in rapidly changing market conditions.

ANNs represented another turning point by offering greater flexibility in capturing latent structures within financial time series. Building upon this foundation, RNNs and their LSTM variants further improved the modeling of temporal dependencies, allowing for more accurate forecasts of spread dynamics. Despite these improvements, challenges such as overfitting, high computational costs, and limited interpretability remain critical concerns in their practical implementation.

More recently, RL has introduced a dynamic framework for decision-making in pairs trading. By learning iteratively from market feedback, RL algorithms are capable of adapting strategies in response to shifting conditions, thereby offering a mechanism to balance profitability and risk in real time. However, issues of sample efficiency, stability of training, and robustness under extreme market stress continue to constrain the widespread adoption of RL in live trading systems.

A particularly promising direction is the integration of DL and RL, creating hybrid models that combine powerful feature extraction with adaptive decision-making. These approaches have shown potential in capturing complex market dynamics and improving resilience under volatile conditions. Yet, their reliance on vast amounts of data and computational resources, as well as their limited transparency, highlight important areas for future refinement.

In summary, our analysis highlights both the potential and limitations of AI in reshaping pairs trading strategies. ML and DL methods significantly enhance predictive accuracy, while RL contributes adaptability to evolving markets. The convergence of these methods offers a powerful toolkit for traders; however, practical deployment requires addressing challenges of interpretability, computational efficiency, and robustness. Continued research into these areas is essential for translating methodological advances into sustainable, real-world trading strategies.

10. Conclusion

This survey has sought to provide a comprehensive account of the evolution of pairs trading strategies, with a particular emphasis on the transformative role of ML, DL, and RL. By systematically reviewing the literature, we have highlighted how traditional statistical methods have gradually been complemented—and in some cases outperformed—by advanced AI techniques capable of modeling nonlinearities, capturing temporal dependencies, and dynamically adapting to shifting market conditions.

From a theoretical perspective, our findings underscore the significance of ML and DL as foundational tools in modern quantitative finance. These technologies extend beyond linear assumptions, enabling the identification of latent structures and complex interactions in financial time series. At the same time, hybrid approaches that integrate DL and RL provide a promising pathway toward the development of adaptive trading strategies that can adjust to evolving market environments.

On the practical side, this study provides a structured overview of evaluation metrics, implementation frameworks, and empirical outcomes. By synthesizing results across a broad set of studies, we offer a set of reference benchmarks that can inform both academic research and practical applications. Importantly, the findings suggest that while ML- and RL-based strategies show considerable promise in improving profitability and risk management, their effectiveness is highly dependent on data quality, market regime, and implementation constraints.

Nonetheless, several limitations remain. The reliance on historical data for training exposes models to risks of overfitting and survivorship bias, while the interpretability of complex models such as neural networks continues to be a major concern for both researchers and regulators. Furthermore, transaction costs, liquidity constraints, and market frictions may erode the theoretical profitability observed in back-testing environments.

Looking forward, the field presents fertile ground for further exploration. Promising avenues include the integration of multi-modal and sector-specific data, the development of more efficient and transparent algorithms, the extension of pairs trading into cross-asset domains, and the incorporation of advanced computational techniques such as meta-learning and quantum optimization. At the same time, embedding ethical considerations and compliance mechanisms into algorithmic design will be essential to align technological innovation with market integrity.

In conclusion, while the predictive capacity of ML, DL, and RL offers unprecedented opportunities for enhancing pairs trading, their success in real-world applications will ultimately depend on striking a balance between innovation, robustness, interpretability, and regulatory compliance. Addressing these challenges will ensure that algorithmic pairs trading continues to evolve not only as a profitable strategy but also as a sustainable component of modern financial markets.

Abbreviations

ADR - Average Daily Return

AHC - Hierarchical Clustering Algorithm

AI - Artificial Intelligence

AMEX - American Stock Exchange

ANFIS - Adaptive Neuro Fuzzy Inference System

ANN - Artificial Neural Network

ATS - Automated Trading System

AUC - Area Under the Curve

AdaBoost - Adaptive Boosting

BNN - Bayesian Neural Network

BP - Back Propagation

BPTT - Backpropagation through time

CA-DRL - Cointegration Approach-Deep Reinforcement Learning

CAE - Convolutional AutoEncoders

CNN - Convolutional Neural Network

DBSCAN - Density-Based Spatial Clustering of Applications with Noise

DCE - Dalian Commodity Exchange

DDQN - Double Deep Q-Network

DL - Deep Learning

DNN - Deep Neural Network

DPG - Deep Policy Gradient

DPG - Deterministic Policy Gradient

DQN - Deep Q-network

DRL - Deep Reinforcement Learning

DT - Decision Tree

ELM - Extreme Learning Machine

EM - Expectation Maximization

EMH - Efficient Market Hypothesis

EN - Elastic Net

EOC - Extended Option-Critic

FN - False Negatives

FP - False Positives

GBDT - Gradient Boosting Decision Tree

GBT - Gradient-boosted Tree

GDA - Gaussian Discriminant Analysis

GMM - Gaussian Mixture Model

HDRL - Hierarchical Deep Reinforcement Learning

HFT - High-Frequency Trading

KalmanBOT - KalmanNet Bollinger Trading

LGBM - LightGBM

LR - Logistic Regression

LSTM - Long Short-Term Memory

MADDPG - Multi-Agent Deep Deterministic Policy Gradient

MAE - Mean Absolute Error

MAPE - Mean Absolute Percentage Error

MDD - Maximum Drawdown

MDP - Markov Decision Process

ML - Machine Learning

MLP - Multilayer perceptron

MSE - Mean Squared Error

NB - Naive Bayes

NLP - Natural Language Processing

NRM - Negative Rewards Multiplier

NYSE - New York Stock Exchange

OU - Ornstein-Uhlenbeck

OHLCV - Open, High, Low, Close and Volume

PCA - Principal Component Analysis

PPO - Proximal Policy Optimization

PSX - Pakistan Stock Exchange

QT - Quantitative Trading

RBF - Radial Basis Function

RF - Random Forest

RL - Reinforcement Learning

RMSE - Root Mean Squared Error

RNN - Recurrent Neural Network

ROC - Receiver Operating Characteristic

ReLU - Rectified Linear Unit

SAPT - Structural Adjustment and Policy Testing

SNN - Stochastic Neural Network

SR - Sharpe ratio

SSE - Shanghai Stock Exchange

SVM - Support Vector Machine

SVR - Support Vector Regression

TAIEX - Taiwan Stock Exchange Capitalization Weighted Stock Index

TCN - Temporal Convolutional Network

TN - True Negatives

TP - True Positives

XGBoost - eXtreme Gradient Boosting

References[2]

[1] Platania, F., Appio, F., Toscano Hernandez, C., El Ouadghiri, I., Peillex, J., 2023. A multi-objective pair trading strategy: integrating neural networks and cyclical insights for optimal trading performance. Annals of Operations Research, pp. 1-20.

[2] He, J., Liu, J., Fu, Y., Zhu, Y., Zhou, Z., Tong, Y., 2023. Applications of Copulas and Machine Learning Algorithms on Pairs Trading.

[3] Xu, Z., Luo, C., 2023. Improved pairs trading strategy using two-level reinforcement learning framework. Engineering Applications of Artificial Intelligence, 126, p. 107148.

[4] Hong, S., Hwang, S., 2023. In search of pairs using firm fundamentals: is pairs trading profitable? The European Journal of Finance, 29(5), pp. 508-526.

[5] Liu, J., Lu, L., Zong, X., Xie, B., 2023. Nonlinear relationships in soybean commodities Pairs trading-test by deep reinforcement learning. Finance Research Letters, 58, p. 104477.

[6] Liu, J., Lu, L., Zong, X., Luo, Z., 2023. Nonlinear Relationships in Stock News Co-occurrence: A Pairs Trading Test on the Constituent Stocks of the CSI 300 Index Based on Deep Reinforcement Learning Methods. Available at SSRN 4573596.

[7] Gupta, V., Kumar, V., Yuvraj, Y., Kumar, M., 2023, April. Optimized pair trading strategy using unsupervised machine learning, in: 2023 IEEE 8th International Conference for Convergence in Technology (I2CT) (pp. 1-5). IEEE.

[8] Roychoudhury, R., Bhagtani, R., Daftari, A., 2023. Pairs Trading Using Clustering and Deep Reinforcement Learning. Available at SSRN 4504599.

[9] Han, C., He, Z., Toh, A.J.W., 2023. Pairs trading via unsupervised learning. European Journal of Operational Research, 307(2), pp. 929-947.

[10] Chen, Y.F., Shih, W.Y., Lai, H.C., Chang, H.C., Huang, J.L., 2023, February. Pairs Trading Strategy Optimization Using Proximal Policy Optimization Algorithms, in: 2023 IEEE International Conference on Big Data and Smart Computing (BigComp) (pp. 40-47). IEEE.

[11] Afzal, M., Usman, M., Raheman, A., 2023. Comparative Study of Pair Trading Techniques in Pakistan's Financial and Non-Financial Sector. Reviews of Management Sciences, 5(1), pp. 38-49.

[2] Since the order in which the papers are used in the article is chronological, they are not sorted alphabetically.

[12] Han, W., Zhang, B., Xie, Q., Peng, M., Lai, Y., Huang, J., 2023, August. Select and trade: Towards unified pair trading with hierarchical reinforcement learning, in: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (pp. 4123-4134).

[13] Sohail, M.K., Raheman, A., Iqbal, J., Sindhu, M.I., Staar, A., Mushafiq, M., Afzal, H., 2022. Are Pair Trading Strategies Profitable During COVID-19 Period? Journal of Information & Knowledge Management, 21(Supp01), p. 2240010.

[14] Zhan, B., Zhang, S., Du, H.S., Yang, X., 2022. Exploring statistical arbitrage opportunities using machine learning strategy. Computational Economics, 60(3), pp. 861-882.

[15] Kim, S.H., Park, D.Y., Lee, K.H., 2022. Hybrid deep reinforcement learning for pairs trading. Applied Sciences, 12(3), p. 944.

[16] Zhijie, Z., 2022. Identify Arbitrage Using Machine Learning on Multi-stock Pair Trading Price Forecasting (Doctoral dissertation, Tohoku University).

[17] Deng, H., Revach, G., Morgenstern, H., Shlezinger, N., 2023, June. Kalmanbot: Kalmannet-Aided Bollinger Bands for Pairs Trading, in: ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 1-5). IEEE.

[18] Sviridenko, E.V., Filipchenkov, V.D., Zuyonok, R.V., 2022. Machine learning algorithms in pair trading.

[19] Figueira, M., Horta, N., 2022. Machine Learning-Based Pairs Trading Strategy with Multivariate. Available at SSRN 4295303.

[20] Du, J., 2022. Mean–variance portfolio optimization with deep learning based-forecasts for cointegrated stocks. Expert Systems with Applications, 201, p. 117005.

[21] Zhang, Y., 2022. Multivariate Bilateral Gamma Process with Financial Application and Machine Learning in Corporate Bond Market (Doctoral dissertation, University of Maryland, College Park).

[22] Huang, K., Sun, J., Zhang, Z., Li, Q., 2022. Pair trading of Commodity Futures in China through the lens of intraday data A Machine Learning Framework and Empirical Analysis. Available at SSRN 4117105.

[23] Zhang, Y., Shen, Q., Guo, J., Jia, Y., 2022, September. Portfolio Trading of Financial Products Based on Machine Learning, in: 2022 International Conference on Machine Learning and Cybernetics (ICMLC) (pp. 72-79). IEEE.

[24] Su, C.H., Lai, H.C., Shih, W.Y., Wang, J.Z., Huang, J.L., 2022, January. Spread Movement Prediction for Pairs Trading with High-Frequency Limit Order Data, in: 2022 IEEE International Conference on Big Data and Smart Computing (BigComp) (pp. 64-71). IEEE.

[25] Carta, S., Consoli, S., Podda, A.S., Recupero, D.R., Stanciu, M.M., 2022. Statistical arbitrage powered by explainable artificial intelligence. Expert Systems with Applications, 206, p. 117763.

[26] Zhang, M., Tang, X., Zhao, S., Wang, W., Zhao, Y., 2022. Statistical arbitrage with momentum using machine learning. Procedia Computer Science, 202, pp. 194-202.

[27] Kalariya, V., Parmar, P., Jay, P., Tanwar, S., Raboaca, M.S., Alqahtani, F., Tolba, A., Neagu, B.C., 2022. Stochastic neural networks-based algorithmic trading for the cryptocurrency market. Mathematics, 10(9), p. 1456.

[28] Li, J., Wang, S., Zhu, Z., Liu, M., Zhang, C., Han, B., 2022, July. Stock Prediction Based on Deep Learning and its Application in Pairs Trading, in: 2022 International Symposium on Networks, Computers and Communications (ISNCC) (pp. 1-7). IEEE.

[29] Lu, J.Y., Lai, H.C., Shih, W.Y., Chen, Y.F., Huang, S.H., Chang, H.H., Wang, J.Z., Huang, J.L., Dai, T.S., 2022. Structural break-aware pairs trading strategy using deep reinforcement learning. The Journal of Supercomputing, 78(3), pp. 3843-3882.

[30] Yu, Z., 2022. The Implementation of Support Vector Machine into Pairs Trading Strategy. Frontiers in Business, Economics and Management, 4(3), pp. 159-165.

[31] Relan, T., 2021. Applying Machine Learning Techniques to Assess Whether a Country's Currency can Predict the Movement of their Respective Stock Market Index.

[32] Patole, D., Gupta, I., Jain, P., Gupta, V., Gada, Y., 2021, September. Controlled Risk Pairs Trading using ReinforcementLearning, in: 2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO) (pp. 1-5). IEEE.

[33] Zong, X., Liu, J., Wang, C., 2021. Deep Reinforcement Learning for Pairs Trading: Evidence from Soybean Commodities. Available at SSRN 3831581.

[34] Wang, C., Sandås, P., Beling, P., 2021, May. Improving pairs trading strategies via reinforcement learning, in: 2021 International Conference on Applied Artificial Intelligence (ICAPAI) (pp. 1-7). IEEE.

[35] Zong, X., 2021. Machine learning in stock indices trading and pairs trading (Doctoral dissertation, University of Glasgow).

[36] Zhang, L., 2021, May. Pair trading with machine learning strategy in china stock market, in: 2021 2nd International Conference on Artificial Intelligence and Information Systems (pp. 1-6).

[37] Chang, V., Man, X., Xu, Q., Hsu, C.H., 2021. Pairs trading on different portfolios based on machine learning. Expert Systems, 38(3), p. e12649.

[38] Flori, A., Regoli, D., 2021. Revealing pairs-trading opportunities with long short-term memory networks. European Journal of Operational Research, 295(2), pp. 772-791.

[39] Carta, S., Recupero, D.R., Saia, R., Stanciu, M.M., 2020, July. A general approach for risk controlled trading based on machine learning and statistical arbitrage, in: International Conference on Machine Learning, Optimization, and Data Science (pp. 489-503). Cham: Springer International Publishing.

[40] Osifo, E., Bhattacharyya, R., 2020. Cryptocurrency trading-pair forecasting, using machine learning and deep learning technique. Using Machine Learning and Deep Learning Technique (March 5, 2020).

[41] Sermpinis, G., Stasinakis, C., Zong, X., 2020. Deep reinforcement learning and genetic algorithm for a pairs trading task on commodities. Available at SSRN 3770061.

[42] Brim, A., 2020, January. Deep reinforcement learning pairs trading with a double deep Q-network, in: 2020 10th Annual Computing and Communication Workshop and Conference (CCWC) (pp. 0222-0227). IEEE.

[43] Navghare, N.D., Kulkarni, H.P., Kedar, P.R., Thakre, S.A., Patil, P.S., 2020, July. Design of pipeline framework for pair trading algorithm, in: 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA) (pp. 630-635). IEEE.

[44] Xu, F., Tan, S., 2020, November. Dynamic portfolio management based on pair trading and deep reinforcement learning, in: Proceedings of the 2020 3rd International Conference on Computational Intelligence and Intelligent Systems (pp. 50-55).

[45] Sarmento, S.M., Horta, N., 2020. Enhancing a pairs trading strategy with the application of machine learning. Expert Systems with Applications, 158, p. 113490.

[46] Ospino, J.F., Fernandez, L., Carchano, O., 2020. Improving pairs trading using neural network techniques and fundamental ratios. Available at SSRN 3653071.

[47] Baek, S., Glambosky, M., Oh, S.H., Lee, J., 2020. Machine learning and algorithmic pairs trading in futures markets. Sustainability, 12(17), p. 6791.

[48] Huang, S.H., Shih, W.Y., Lu, J.Y., Chang, H.H., Chu, C.H., Wang, J.Z., Huang, J.L., Dai, T.S., 2020, February. Online structural break detection for pairs trading using wavelet transform and hybrid deep learning model, in: 2020 IEEE International Conference on Big Data and Smart Computing (BigComp) (pp. 209-216). IEEE.

[49] Sohail, M.K., Raheman, A., Adil, I.H., Rizwan, M.F., Khan, S.U., 2020. Pair Trading Strategies Using Machine Learning: A Case of PSX Firms. Pakistan Business Review, 22(3), pp. 340-351.

[50] Yoshikawa, D., 2020. Pairs trading with deep learning, in: the 34th Annual Conference of the Japanese Society for Artificial Intelligence (2020) (pp. 3I1GS1301-3I1GS1301).

[51] Jirapongpan, R., Phumchusri, N., 2020, April. Prediction of the profitability of pairs trading strategy using machine learning, in: 2020 IEEE 7th International Conference on Industrial Engineering and Applications (ICIEA) (pp. 1025-1030). IEEE.

[52] Fuster, A., Zou, Z., 2020. Using Machine Learning Models to Predict S&P500 Price Level and Spread Direction.

[53] Brim, A., 2019. Deep reinforcement learning pairs trading.

[54] Sun, J., Zhou, Y., Lin, J., 2019, May. Using machine learning for cryptocurrency trading, in: 2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS) (pp. 647-652). IEEE.

[55] Fischer, T.G., Krauss, C., Deinert, A., 2019. Statistical arbitrage in cryptocurrency markets. Journal of Risk and Financial Management, 12(1), p. 31.

[56] Ruxanda, G., Opincariu, S., 2018. BAYESIAN NEURAL NETWORKS WITH DEPENDENT DIRICHLET PROCESS PRIORS. APPLICATION TO PAIRS TRADING. Economic Computation & Economic Cybernetics Studies & Research, 52(4).

[57] Chen, Y.Y., Chen, W.L., Huang, S.H., 2018, July. Developing arbitrage strategy in high-frequency pairs trading with filterbank CNN algorithm, in: 2018 IEEE International Conference on Agents (ICA) (pp. 113-116). IEEE.

[58] Sutherland, I., Jung, Y., Lee, G., 2018. Statistical arbitrage on the KOSPI 200: An exploratory analysis of classification and prediction machine learning algorithms for day trading. Journal of Economics and International Business Management, 6(1), pp. 10-19.

[59] Li, Y., Bu, H., Wu, J., 2017, June. A two-stage multi-view prediction method for investment strategy, in: 2017 International Conference on Service Systems and Service Management (pp. 1-6). IEEE.

[60] Chaudhuri, T.D., Ghosh, I., Singh, P., 2017. Application of Machine Learning Tools in Predictive Modeling of Pairs Trade in Indian Stock Market. IUP Journal of Applied Finance, 23(1).

[61] Krauss, C., Do, X.A., Huck, N., 2017. Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500. European Journal of Operational Research, 259(2), pp. 689-702.

[62] Kim, S., Heo, J., 2017. Time series regression-based pairs trading in the Korean equities market. Journal of Experimental & Theoretical Artificial Intelligence, 29(4), pp. 755-768.

[63] Fallahpour, S., Hakimian, H., Taheri, K., Ramezanifar, E., 2016. Pairs trading strategy optimization using the reinforcement learning method: a cointegration approach. Soft Computing, 20, pp. 5051-5066.

[64] Wu, J., 2015. A pairs trading strategy for GOOG/GOOGL using machine learning.

[65] Nóbrega, J.P., Oliveira, A.L., 2014, October. A combination forecasting model using machine learning and kalman filter for statistical arbitrage, in: 2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC) (pp. 1294-1299). IEEE.

[66] Nóbrega, J.P., De Oliveira, A.L.I., 2013, November. Improving the statistical arbitrage strategy in intraday trading by combining extreme learning machine and support vector regression with linear regression models, in: 2013 IEEE 25th International Conference on Tools with Artificial Intelligence (pp. 182-188). IEEE.

[67] Madhavaram, G.R., 2013. Statistical arbitrage using pairs trading with support vector machine learning.

[68] Chen, Y., Ren, W., Lu, X., 2012. Machine Learning in Pairs Trading Strategies.

[69] Ester, M., Kriegel, H.P., Sander, J., Xu, X., 1996, August. A density-based algorithm for discovering clusters in large spatial databases with noise. In kdd (Vol. 96, No. 34, pp. 226-231).

University of Warsaw
Faculty of Economic Sciences
44/50 Długa St.
00-241 Warsaw
www.wne.uw.edu.pl
ISSN 2957-0506