Decisions in Economics and Finance
https://doi.org/10.1007/s10203-025-00546-5

Low- and high-beta stock rotation trading by reinforcement learning policies

Eshagh Jahangiri[1] · **Marco Corazza**[1]

Received: 3 January 2024 / Accepted: 28 October 2025
© The Author(s), under exclusive licence to Associazione per la Matematica Applicata alle Scienze Economiche e Sociali (AMASES) 2025

Abstract
The financial markets are becoming more complicated and unstable, which makes it even more important to find portfolio management solutions that are more flexible and effective. Mean-Variance Optimization, Risk Parity, and Equal-Weighted Portfolio are examples of conventional models that do not always take into account how markets change over time. This is why more advanced methods like Reinforcement Learning (RL) are needed. This study looks at how three RL algorithms, SARSA, Q-Learning, and Greedy-GQ, can manage a portfolio with two stocks, one that is very volatile and one that is not. This study makes a unique contribution as it compares these algorithms and looks at how adding extra information, like technical indicators, the Relative Strength Index, and Moving Averages, can help the models make better decisions. This research evaluates how well these methods perform compared to a benchmark, the equally weighted portfolio, using data from the New York Stock Exchange (NYSE) from October 2021 to October 2022. During the financial crisis of 2007–2008, a robustness check showed that these methods still worked well even when the market was very unstable. A second robustness test with higher transaction costs showed once more that the reinforcement learning methods can work in more realistic crisis trading environments. The results showed that, on average, Greedy-GQ did better than SARSA and Q-Learning. But the performance was different for each configuration, and some choices of hyperparameters had a big effect on the results. When more market information was added to the state vector, all three algorithms worked better.

Keywords Rotation Trading · Reinforcement Learning · SARSA · Q-Learning · Greedy-GQ

JEL classification G11 · C49 · C63

✉ Eshagh Jahangiri
eshagh.jahangiri@unive.it

Marco Corazza
corazza@unive.it

[1] Department of Economics, Ca' Foscari University of Venice, Venice, Italy

Published online: 08 November 2025

1 Introduction

Investors are constantly trying to predict stock markets with high accuracy because it can have profitable consequences. Due to the recent progress of Artificial Intelligence (AI) in different fields, such as self-driving cars (Ni et al. 2020), speech recognition (Wolf et al. 2020), healthcare and medicine (Mishra 2022), and Chat-Bot (for example, ChatGPT[1]), investors are considering using AI algorithms, especially Machine Learning (ML) methods, for assessing financial markets. AI, and especially reinforcement learning (RL), makes predictions more accurate by letting models learn from their mistakes in real time as they interact with the environment. Unlike traditional models, such as Mean-Variance Optimization or linear regressions, that assume stationarity or require rigid parametric forms, RL methods adapt to changing market conditions by continuously updating their strategies based on new information. As a result, they are particularly beneficial for financial markets, which are dynamic, noisy, and influenced by agent and feedback loops (Hambly et al. 2023). By optimizing long-term cumulative returns, RL algorithms can assist investors in making decisions that consider both short-term and long-term market reactions (Sutton and Barto 2018).

In this study, we examine the application of reinforcement learning in the development of adaptive trading strategies that can react to market fluctuations in real time. We examine three unique RL algorithms: SARSA, Q-Learning, and Greedy-GQ, each representing a distinct learning paradigm within the reinforcement learning framework. By applying these methods to a two-stock portfolio composed of one high-β and one low-β asset, we aim to understand how their learning structures influence trading behavior and portfolio performance under changing market conditions. This study presents three principal contributions: it compares the efficacy of three reinforcement learning algorithms, SARSA, Q-Learning, and Greedy-GQ, in the management of a two-stock portfolio across diverse market conditions and feature sets; assesses the influence of integrating supplementary information, such as technical indicators, on the agent's decision-making and profitability; and evaluates the resilience of these methods during the 2007–2008 financial crisis to determine their adaptability to extreme market volatility.

There are three types of ML approaches: Supervised, Unsupervised, and RL. In supervised learning, the algorithm learns from labeled data to make predictions, while in unsupervised learning, it looks for patterns or groupings in data without any labels. RL, which is the ML method used in this research, maximizes an accumulative reward measure by learning from trial and error. RL's background will be explained in detail in Section 2.2. The problem of an agent having to maximize a cumulative reward in an unknown environment is also known as a sequential decision-making problem, where the agent discovers an optimal strategy (Littman 1996).

Financial markets are inherently unpredictable, complex, and subject to fast change, making it challenging to study them using traditional approaches (Tian et al. 2024). Traditional learning algorithms frequently fail when markets are volatile, resulting in methods that achieve short-term results but become unprofitable over time. In contrast, RL approaches have the advantage of being flexible and adaptable, allowing them to

[1] https://chat.openai.com.

produce strategies that change in response to market conditions. This capacity is critical in developing trading methods that respond to current market conditions and adapt to shifting settings, offering a sustained advantage.

One of the essential concerns that needs to be dealt with is the theoretical nature of the capital market. According to the Efficient Market Hypothesis (EMH) in its weak form, it is impossible to systematically gain profits in the financial markets because all agents are entirely rational and instantly process the current information of the market into the stocks' prices (Fama 1970; Chinas 2019). In this kind of world, there are no chances to make a profit, as it is impossible to forecast new financial market information (like stock prices and returns). However, based on several investigations that experimental economists have done since the EMH was introduced, neither the market is forecastable nor the economic agents of the market are entirely rational while they are deciding their trading strategies under uncertainty (Subramanian 2010). Consequently, the market offers profitable opportunities frequently. Hence, we need to choose the theoretical framework of the market in a way that we can use these opportunities for profitable trading.

The Adaptive Market Hypothesis (AMH) is the theory proposed by Andrew W. Lo, which demonstrates that financial markets are evolutionary markets that can be inefficient as produced by the interaction of not-fully rational economic agents (Lo 2017). Once the market is unstable, the agents interact with each other to achieve some level of satisfactory profitability and do not necessarily choose an optimal action. Because they are partly rational, they cannot promptly react to new information, making the markets inefficient. In order to take advantage of profitable trading opportunities, we have chosen the AMH as our theoretical framework of the market for this paper and our financial trading system.

This study compares three reinforcement learning algorithms—SARSA, Q-Learning, and Greedy-GQ—in managing a two-stock portfolio. We chose these three methods because they are examples of different RL paradigms: SARSA is an on-policy algorithm that changes its value estimates based on the agent's current strategy. Q-Learning is an off-policy method that finds the best policy no matter what the agent does, and Greedy-GQ is made to make sure that the learning process converges even when function approximation and off-policy learning are used (Maei et al. 2010). This diversity allows us to evaluate how the choice of learning framework influences trading performance and policy robustness in dynamic financial settings. We select a two-stock portfolio, comprising one high-volatility stock and one low-volatility stock, as it enables the evaluation of reinforcement learning methods in a simple yet realistic context that mirrors the actual portfolios in which traders try to diversify their portfolio by including both high- and low-risk stocks. Beta (ß) measures a stock's sensitivity to overall market movements and is commonly estimated by regressing the stock's returns against a market index. A beta greater than 1 indicates a stock is more volatile than the market (high-beta), while a beta below 1 indicates lower volatility (low-beta).

Feature engineering and assessing the effect of features on ML methods is one of the important ways to potentially improve the current algorithms (Heaton 2016). Enhancing RL algorithms with more information is crucial in this research, as financial markets are affected by numerous factors beyond simple price fluctuations. For instance, technical indicators incorporate trends, momentum, and overbought or over-

sold conditions, which help the agent make better-informed trading decisions. In this paper, we seek to evaluate the influence of these further features on the models' performance to ascertain if more intricate input data improves decision-making precision. Understanding the role of information in RL models is crucial because it could help us create more effective trading strategies, potentially improving their robustness and profitability. In practice, this can improve the model's ability to identify regime changes and optimize decisions accordingly.

This research contributes theoretically and practically by illustrating the application of reinforcement learning methods in real-world portfolio management, reflecting the increasing significance of machine learning in finance. In this paper, we aim to explore how the reinforcement learning algorithms SARSA, Q-Learning, and Greedy-GQ perform in managing a two-stock portfolio under normal market conditions, whether the inclusion of additional information like technical indicators enhances their effectiveness, and how robust these methods are during periods of significant market volatility, such as the 2007–2008 financial crisis. The models are tested on five pairs of NYSE-listed stocks, each consisting of one high-β and one low-β stock, using daily data from October 2021 to October 2022, reflecting a non-crisis market period with moderate volatility. Results show that all three RL algorithms generally outperform the benchmark equally weighted portfolio, with Greedy-GQ demonstrating the most stable and consistent performance.

The rest of this paper is as follows: Section 2.2 will cover the fundamentals of RL and some related studies. The details of the methodology and proposed RL methods will be given in Section 3. Section 4 will cover the results and their discussion. Section 5 will serve as the paper's conclusion.

2 Literature review and background

This section offers a thorough examination of the literature regarding Reinforcement Learning (RL) applications in financial decision-making, particularly in portfolio management. The discussion begins by reviewing related studies that have applied RL techniques to financial markets, highlighting both foundational and recent contributions. These studies show how RL-based models have changed to optimize asset allocation techniques by dynamic interactions with real-time market data.

After reviewing other works, the basic concepts and theories behind RL are explained. This includes an explanation about Markov Decision Processes (MDPs), value functions, and policy optimization, which are the core elements of RL applications in finance. The next sections will build on this theoretical base to help you better understand how RL techniques develop and are utilized in portfolio management.

2.1 Related studies

As described in the preceding section, RL is a learning method that involves an agent interacting with its surroundings (an environment), with the goal of maximizing the

discounted accumulation of the rewards the agent receives from its interactions with the environment.

The psychology of animal learning from the 1950s is where the idea of a learning system that interacts with the environment first emerged (Deese 1952). In the 1950s, the well-known Bellman equation and the value iteration method were introduced by Bellman (1952). Howard (1960) advanced the work of Bellman by introducing the concept of the policy iteration method and incorporating Markov decision processes into dynamic programming. After the publication of Minsky's research (Minsky 1961), RL was acknowledged in the literature. Harry Klopf expanded on the concept of RL in three papers, Klopf (1972), Klopf (1975), and Klopf (1982). With their book, Sutton and Barto (2018) made a significant contribution to the RL literature by unifying previous approaches and setting out the framework used today.

Portfolio management is a core problem in finance, which seeks the optimal distribution of capital among a group of assets in order to optimize some risk-adjusted return measure. Mean-variance theory is one of the main theories for discovering this problem (Markowitz 1952, 1959; Markowitz and Todd 2000). Another important theory for exploring this problem is the capital growth theory (Kelly 1956). While the capital growth theory works on multi-period portfolio management, mean-variance theory focuses on a single-period problem. Dynamic portfolio management, which is the focus of this paper, is fitted to the capital growth theory. The aim of dynamic portfolio management approaches is to maximize the wealth. There are different methods to manage portfolios dynamically. One of the well-known dynamic portfolio management methods is Constant Rebalanced Portfolio (Kelly 1956; Cover 1991), which is mostly used as a benchmark to assess other investment strategies. The constant rebalanced portfolio method attempts to keep the distribution of wealth among a group of assets at a constant level over the course of time, one trading period after another. The other categories of dynamic portfolio management are Follow-the-Winner (Agarwal et al. 2006; Gaivoronski and Stella 2000), Follow-the-Loser (Borodin et al. 2003), Pattern-Matching Methods (Györfi et al. 2006), and Meta-Learning Algorithms (Vovk 1990; Das and Banerjee 2011).

Due to the progress of AI in different fields, researchers started to explore it for portfolio management as well. RL has been used by researchers in the field of finance, particularly for automated trading and portfolio management. RL is, therefore, not a novel technique in finance, but it is still in its early stages of development. One of the first papers that used RL to manage a portfolio was Moody et al. (1998a). In their work, they used recurrent RL algorithms and utilized the Sharpe ratio and their proposed reward function, the differential Sharpe ratio. They showed that the trading system by recurrent RL performs better than the one by using Supervised learning. They assessed their model on a two-asset portfolio. In the other work, Moody et al. (1998) demonstrated that the QL method performs better than the classical buy-and-hold strategy. Gao and Chan (2000) employed the QL algorithm and Sharpe ratio maximization in portfolio management. They considered a risk-adjusted return and absolute profit as their reward functions and evaluated their method on the Forex market, which resulted in a positive return.

From the mid-2010s onward, rapid developments in deep learning significantly expanded the scope of RL in finance. Jin and El-Saawy (2016)'s paper developed

a model for managing a two-asset portfolio utilizing deep RL, a deep QL algorithm. The value function approximations were calculated using the classical Artificial Neural Network (ANN) approach, and the portfolio that they used for analyzing their method consisted of one high-risk stock and one low-risk stock. They adopted seven available actions in each state and considered a negligible transaction cost. They showed that their proposed model's performance was comparable to, if not better than, buy-and-hold and rebalanced benchmark strategies.

The research by Almahdi and Yang (2017) is another successful work using recurrent RL and the so-called Calmar ratio as their reward function in portfolio management. Another successful research using RL in portfolio management is Jiang et al. (2017), which used Convolutional Neural Networks (CNN) as a function approximation. Their RL agents decide the optimal weights for the portfolio's assets, which consist of 30 cryptocurrencies. All of their three methods performed better than the benchmarks (buy-and-hold, constant rebalanced portfolio, and best stock strategies). Similarly, Liang et al. (2018) employed three continuous RL methods, such as Deep Deterministic Policy Gradient, Proximal Policy Optimization, and Policy Gradient, in portfolio management.

Another study on a two-asset portfolio, the one by Du et al. (2015), considered the QL algorithm and Recurrent Reinforcement Learning (RRL) to reallocate optimally capital between two assets, which are cash and a risky portfolio. They employ interval profit, the Sharpe ratio and differential Sharpe ratio reward functions. For RRL, they used a policy iteration, and for the QL, they employed a value iteration method. They demonstrated that RRL outperformed QL considering stability. The paper by Srivastava et al. (2020) is another research in the field of portfolio management using Deep Reinforcement Learning (DRL) methods based on Long Short Term Memory Networks, CNN, and Recurrent Neural Networks. They compared their proposed methods with four traditional benchmarks named equally weighted portfolio, risk parity, mean-variance, and minimum variance. Their DRL methods performed better than the traditional benchmarks by testing on 24 stocks of the U.S. stock market.

The work by Pendharkar and Cusatis (2018) suggested both discrete and continuous action spaces for the management of a two-asset retirement portfolio. Their agents maximize the portfolio return and differential Sharpe ratio (as in Moody et al. (1998b)) as the reward function. In their model, SARSA and QL methods were utilized. Their proposed model produced greater profits than buy-and-hold and decision-tree trading strategies, which served as benchmarks. Corazza et al. (2019) proposed three agents using SARSA, QL, and Greedy-GQ for their financial trading system. Their model trade only one asset, and it can take one action amongst "Buy", "Sell", "Stay-Out-Of-Market" at each time step. They utilized the Sharpe and Calmar ratios for their reward function. They assessed their model by using five stocks from the Italian exchange market, which resulted in a positive return in all three SARSA, QL, and Greedy-GQ methods.

The paper by Patani et al. (2020) proposed three DRL methods named Deep Q-Networks, Double Deep Q-Networks, and Deep Q-Networks with fixed Target for managing a two-asset portfolio, and they chose two assets of their portfolio similar to the way Jin and El-Saawy (2016) has chosen. Furthermore, they used a continuous action space and stocks from the Indian stock market to assess their proposed

algorithms. The paper illustrated that DDQN performed better than others by far, as it had two separate estimators for calculating Q-values. Giorgi et al. (2024) recently proposed a reinforcement learning algorithm that is made just for trading commodities. Their research shows how to use the SARSA algorithm and a neural network to create a dynamic trading strategy that can handle continuous state-action spaces. The suggested method is compared to the analytical best solution found using the Garleanu-Pedersen model, which assumes that transaction costs are linear and factors are quadratic. RL was able to replicate the best policy in situations where the best strategy was known. But when the RL algorithm was used on a more realistic, non-linear model that was set to WTI spot prices, it did better than the GP model. This shows how useful RL could be in dynamic trading environments, especially when theoretical models do not work because the asset dynamics are not linear.

For portfolio optimization, Jiang et al. (2024) proposed a deep reinforcement learning framework based on the twin-delayed deep deterministic policy gradient (TD3) algorithm. Their model incorporates risk aversion and transaction cost limits directly into a mean-variance reward function. This enables the agent to better balance return and risk. Their proposed method outperformed both traditional strategies and other DRL-based benchmarks when it was tested on portfolios consisting of stocks from the Dow Jones and S&P 100.

Taken together, these studies demonstrate a clear evolution from early discrete-state RL models toward deep and continuous-action formulations capable of handling the complexity of modern financial markets. This research employs algorithms for dynamic portfolio management of two assets using SARSA, QL, and Greedy-GQ with function approximation methods. No reinforcement learning models were found (to the best of our knowledge) in the literature that use SARSA, QL, or Greedy-GQ with function approximation methods to manage a two-asset portfolio. Another novelty of this research is to assess the effect of giving more information to the agent, examining whether it will perform better with more information. The following section elaborates on the model used in this paper.

2.2 Basics of reinforcement learning (RL)

RL (see Sutton and Barto (2018)) is one of the main three paradigms of ML. RL is about an agent (agent and learner are used interchangeably in this field and this paper) interacting with an environment and how the agent learns. Imagine an infant who wants to start walking; she does not have a specific teacher for that, but she has sensors that communicate with the environment and tell her about the results of her movements. For instance, if she falls after moving one step forward, she receives a negative reward, and the probability of taking that step in that state of the environment will decrease for future movements. RL is learning about what action to choose in each situation of an environment. In most cases, the learner's actions will affect not only the immediate reward but also all following rewards. Furthermore, no supervisor tells the learner what actions to take, and the learner will learn which actions to choose just by trying them. Thus, trial-and-error learning and delayed rewards are two fundamental characteristics of RL (Sutton and Barto 2018).

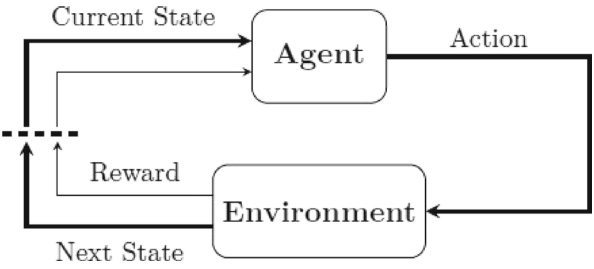

Fig. 1 Basic Framework of RL. Source: (Sutton and Barto 2018), page 48

To formulate the issues that RL aims to solve, it is necessary to consider Markov Decision Processes (MDPs) (Bellman 1957). MDPs are stochastic control processes that provide the mathematical framework for modeling decision-making problems in different situations. In detail, a MDP consists of four sets. The first one is the set of all possible states of the environment, S, which tells about the current environment. The agent receives $S_t \in S$ at each time step t. The second set is the set of all actions available in each state S_t, $\mathcal{A}(S_t)$, in which the agent takes one action of the set $\mathcal{A}(S_t)$ at S_t. Associated with the action that the learner takes at state S_t, there is a probability,

$$P_a(s, s') = Pr\,(s_{t+1} = s' \mid s_t = s,\ a_t = a) \tag{1}$$

where Pr$(\cdot)$ denotes the probability of the event and $P_a(s, s')$ shows the probability that the learner taking action $a_t = a$ in state $s_t = s$ at time t will reach s_{t+1} at time $t + 1$. After taking action $a_t \in A(S_t)$ in state s_t, the learner will receive a reward $R_{t+1} \in R_{a_t}(s_t, s_{t+1})$ at time t+1. The probability of reward R_{t+1} given the previous state S_t and taking action A_t is

$$P_a(s, r) = Pr\,(R_{t+1} = r \mid S_t = s,\ A_t = a) \tag{2}$$

Therefore, the joint transition probability, which merges $Pr\,(s_{t+1} = s' \mid s_t = s,\ a_t = a)$ and $Pr\,(R_{t+1} = r \mid S_t = s,\ A_t = a)$ is

$$p(s', r|s_t = s, a_t = a) \doteq Pr\,(S_{t+1} = s', R_{t+1} = r \mid S_t = s,\ A_t = a) \tag{3}$$

provides the probability of having state $S_{t+1} = s'$ and reward r given the previous state and action. The joint transition probability has the Markov property as the next state, s' given the current state and action, s and a, is independent of all previous states and actions. RL methods are approaches that solve problems that are formalized as MDPs. Consequently, RL heeds problems in which the following features are present: an agent, that is the decision-making learner; an environment with which the agent interacts; actions available to the agent at each state and time; and a reward that the agent receives from the environment as a result of the actions it chooses. Figure 1 shows the RL's basic framework.

The main aim of the RL is to maximize a cumulative reward when the agent is solving the problem. The formalization of this cumulative reward, also called the

value function, is as follows

$$G_t := \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{4}$$

which $0 < \gamma \leq 1$[2] is the discount rate that considers the time value of later rewards in the future, and R_{t+k} is the reward at time$t + k$.

Important characteristics of MDPs that should be given attention to include the fact that all MDP problems involve interactions between an environment and a decision-making agent, with the agent attempting to achieve the objective of maximizing the cumulative reward over time in an unpredictable environment. The second crucial characteristic is that the actions influence the future states of the environment, which in turn influences future rewards and actions. Therefore, taking one action in one state may result in low rewards but may yield substantial benefits in the long run. This is the role that value functions play. A reward represents the short-term value of an action, while a value function reveals its long-term value.

There are two types of value functions: state-value function and state-action value function (referred to as the action-value function in the rest of the paper). The state-value function of one state is the cumulative amount of discounted reward that the agent is expected to receive from that state onward. The action-value function for a state-action pair shows the expected total discounted reward that an agent will get for doing that action in that state. For instance, think about a robot moving through a grid where each cell stands for a state. The robot can move up, down, left, or right. Suppose the robot is currently in state s and takes action a (moving right). The action-value function $Q(s, a)$ estimates the expected sum of future discounted rewards the robot will receive if it moves right from s.

Furthermore, a policy, a reward, and a value function are the three primary subcomponents of RL. A policy is the specification of the agent's behavior, which consists of determining what action to take for each state at every time step. The policy is the map from states to the feasible actions in those states. In the RL system, the agent seeks to maximize the value function over time. Value functions reveal more about how favorable it is to be in a particular state of the environment and offer a more thoughtful perspective of being in that state.

A policy can be stochastic or deterministic. The former returns the conditional probability distribution over the set of all possible actions in one state, $\psi\ (A \mid S)$, and this distribution assigns a probability for every action. The latter returns an action and is a function from the set of states to the set of actions, $\psi : S \rightarrow A$.

Given a stochastic policy $\psi\ (a \mid s)$, where s is the current state of the environment and a is the taken action in that state, $a \in A(s)$, the action-value function is defined as follows:

$$q_{\psi}\ (s,\ a) := \mathbb{E}_{\psi}\ [\ G_t \mid S_t = s,\ A_t = a\ ] \tag{5}$$

where $\mathbb{E}_{\psi}$ is the expected value of the value function computed when the agent follows policy $\psi(a|s)$ after taking action $a \in \mathcal{A}(s)$ in the state s. As detailed in Sutton and

[2] When $\gamma = 1$, in the case of the infinite horizon, it is possible that the cumulative reward (value function) diverges. So, in this case, the value of such a cumulative reward has to be carefully monitored.

Barto (2018), the value functions satisfy the so-called Bellman equation (6), which follows from the recursive formulation $G_t = R_{t+1} + \gamma, G_{t+1}$ for the current state-action pair and its successors:

$$q_\psi\ (s,\ a) = \sum_{s' \in \mathcal{S}} \sum_{r \in \mathcal{R}} p(s', r \mid s, a) \left[r + \gamma \sum_{a' \in \mathcal{A}(s')} \psi\ (a' | s')\, q_\psi\ (s',\ a') \right] \tag{6}$$

where γ is the discount factor to consider the time value.

The agent makes decisions based on the value functions, not reward, as the value function indicates which action captures the greatest value function over a long period. So, the policy is chosen based on the action-value functions. The agent needs to calculate these values using the "experiences" it has had during its "life", which is one of the most fundamental characteristics of all RL approaches. Consequently, RL methods aim to identify the optimal policy, ψ', that we deem superior to other policies, $\psi' \succeq \psi$, if and only if its state-value function is greater than that of others, $v_{\psi'}(s) \geq v_\psi(s)$ for all states $s \in S$,

$$v_\psi(s) = \sum_{a \in \mathcal{A}(s)} \psi(a \mid s) q_\psi(s,\ a). \tag{7}$$

There is one action-value function for all optimal policies, which can be found as

$$q_*(s,\ a) = \max_\psi q_\psi(s,\ a). \tag{8}$$

The optimal action-value function satisfies the Bellman equation (6);

$$q_*(s,\ a) = \sum_{s', r} p(s',\ r \mid s,\ a) \left[r + \gamma \max_{a'} q_*(s',\ a') \right] \tag{9}$$

For dealing with equation (9), we need the transition probability $p(s', r \mid s, a)$, which is usually difficult to calculate, especially when the environment (which is a financial market in this paper) is dynamically changing. Furthermore, with having the transition probability, it is still time-consuming to derive the optimal $q_*\ (s,\ a)$ as the number of states and actions can be very high, and it can suffer from the curse of dimensionality (Bellman 2003).

The fourth major sub-element of RL is a model of an environment that describes its behavior. For example, it can provide information about the probability of each state occurring and the corresponding reward. On the basis of this criterion, RL methods are split into two groups. The first group is model-based methods, in which the environment's model is known, and the other group consists of model-free methods, which address the problem without employing an environment's model and instead rely on trial and error. Model-free approaches, which are applied in this research, learn the environment's model on the basis of its current state through trial and error and then utilize it to predict the next state and reward. In other terms, the RL tries to use

trial-and-error and its experiences gained through interaction with the environment to calculate the action-value function and then select the optimal policy, as previously described.

This research employs RL methods from the Temporal-Difference (TD) learning method family. These methods update the value functions step-by-step (bootstrap), similar to dynamic programming techniques (see Chapter 6 of Sutton and Barto (2018)).

In the previous sections, the evaluation and prediction of value functions were discussed; the following section addresses the identification of the optimal policy. Consequently, a control mechanism should be chosen, and it comes in two varieties: an on-policy method and an off-policy method. In the on-policy methods, the value function for the current policy is estimated using the same policy that was utilized to generate the sequences of state-action pairs. Conversely, off-policy methods directly estimate the optimal value function without adhering to the policy used to determine the state-action pairs. SARSA is an on-policy control mechanism that is utilized in this research (Rummery and Niranjan 1994). The SARSA method's updating rule is as follows:

$$Q(S_t,\ A_t) \leftarrow Q(S_t,\ A_t) + \alpha_t \left[R_{t+1} + \gamma\, Q(S_{t+1},\ A_{t+1}) - Q(S_t,\ A_t)\right] \tag{10}$$

where $Q(S_t, A_t)$ indicates the value function of being at the state S_t and taking an action A_t, R_{t+1} is the next reward, α is the learning rate, and γ stands for the discounted rate for the value functions of the next states.

There is a dilemma that the agent faces when selecting an action in each state, and it is called the exploration-exploitation tradeoff. Exploration is the process of looking for other actions (other policies) that have not yet been taken and which, if implemented, could yield a greater value function than that of following the current policy. In contrast, exploitation entails repeatedly selecting the action with the highest value function.

In the SARSA method, actions are chosen to utilize an ε-greedy policy that tries to tradeoff the dilemma of exploration and exploitation. At the time step $t + 1$, ε-greedy chooses action A_{t+1} randomly from a uniform distribution $U_t(\mathcal{A}\,(S_{t+1}))$ with probability ε and determines action using the current action-value faction, $A_{t+1} \in \arg\max_{a \in \mathcal{A}\,(S_{t+1})} Q\,(S_{t+1}, a)$ with probability $1 - \varepsilon$. ε is the parameter that manages the tradeoff between exploration and exploitation (see Section 6.4 of Sutton and Barto (2018) for more details about the SARSA method).

The Q-learning (QL) method is used as an off-policy control mechanism in this research (Watkins and Dayan 1992). The updating rule of the QL is as follows:

$$Q(S_t,\ A_t) \leftarrow Q(S_t,\ A_t) + \alpha_t \left[R_{t+1} + \gamma \max_a Q(S_{t+1},\ a) - Q(S_t,\ A_t)\right] \tag{11}$$

The distinction between SARSA and QL is that the former is an on-policy method that uses the quintuple $(S_t, A_t, R_{t+1}, S_{t+1}, A_{t+1})$ in each time step, which is the reason it is named "SARSA". The latter is an off-policy method that does not follow the current policy for selecting the next action and instead chooses the next action using the action-

value function and greedy behavior. These methods asymptotically converge to the optimal action-value function when all the state-action pairs are visited an infinite number of times and the parameters are chosen appropriately (Sutton and Barto 2018). Convergence can be demonstrated when the state space is not huge. When there are an infinite number of states or the state set is too large, visiting states an infinite number of times (or even just a large number of times) is hard to be satisfied. To address this issue, RL is improved using a generalization theory called function approximation, where the action-value function is the target function to be approximated. In these methods, examples of one function are utilized to generalize and predict the entire function.

This study uses linear function approximation, which means that the action-value function is shown as $\hat{Q}(s,\ a,\ \omega)$ which estimates the true value function, $Q(s,\ a)$, where ω is a weight vector. This method does not provide the actual value function but aims to produce a well-approximated one. As the dimension of the weight vector is less than the dimension of the state vector, when the value of one state is changed, it affects other states' values as well, and it is infeasible to control all states' values precisely. Therefore, the mean-square error (MSE) (12) is used to evaluate the $\hat{Q}(s,\ a,\ \omega)$, which is the approximation of the true value function $Q_{\psi}(s,\ a)$ given the policy ψ;

$$MSE(\omega) = \sum_{s \in \mathcal{S}} \mu(s) \sum_{a \in \mathcal{A}(s)} \psi(a|s) \left[Q_{\psi}(s,\ a) - \hat{Q}(s,\ a,\ \omega) \right]^2 \tag{12}$$

The optimal weight vector, ω^*, is the one which $MSE(\omega^*) < MSE(\omega)$ for all possible weight vectors ω.

A well-established method for finding the optimal weight vector in function approximation is Stochastic Gradient Descent (SGD). The update rule of the SGD is as follows:

$$\omega_{t+1} = \omega_t + \eta_t \left[\theta_t - \hat{Q}(S_t,\ A_t;\ \omega_t) \right] \nabla \hat{Q}(S_t,\ A_t;\ \omega_t) \tag{13}$$

where θ_t is an unbiased estimate for the true value function $Q(S_t, A_t)$; note that the true value function is frequently changing and is unknown in the RL. Therefore, θ_t typically reflects a target computed using the agent's current experience and policy. For example, the expected value of the cumulative return (4) can be used as an unbiased estimate of the value function. Therefore, the update rule of the SGD (13) is formulated as follows:

$$\omega_{t+1} = \omega_t + \eta_t \left[G_t - \hat{Q}(S_t,\ A_t;\ \omega_t) \right] \nabla \hat{Q}(S_t,\ A_t;\ \omega_t) \tag{14}$$

where $G_t = R_{t+1} + \gamma\, G_{t+1} = R_{t+1} + \gamma\, \hat{Q}(S_{t+1},\ A_{t+1};\ \omega_t)$. This recursive definition G_t shows the expected cumulative return from time t onward. It is also used as a sample-based target for approximating the true action-value function in the gradient update.

The TD methods do not converge surely with the SGD algorithm (Barnard 1993); however, they can converge consistently with Semi-Gradient methods. The term "Semi-Gradient" is used because in the update rule it considers only the gradient effect

of the weight vector and not the one of the target. The Semi-Gradient approaches converge consistently when the estimate of the value function is a linear combination of the weight vector, ω, and a so-called feature vector (Sutton and Barto 2018). A feature vector describes the state-action pair and is incorporated into the update rule using a basis function. Therefore, the estimated value function, $\hat{Q}\left(S_t,\ A_t;\ \omega_t\right)$, is characterized by the basis function, whose coefficients form the feature vector as follows:

$$\hat{Q}\left(S_t,\ A_t,\ \omega_t\right) = \omega^T\ x\left(S_t,\ A_t\right) \tag{15}$$

where $x\left(S_t,\ A_t\right)$ is the feature vector. When we use linear function approximation, the estimated value function (in expectation) gets closer to the projection of the true value function onto the space defined by these features. There is no closed-form error bound, but the approximation error remains on the order of the model's capacity to represent relevant patterns in the state-action space. Consequently, the gradient component of the SGD update rule, (14), consists solely of the weight vector ω. Hence, using the Semi-Gradient Algorithm and linear combination of the weight and the feature vectors for the estimated value function, the SGD update rule for the SARSA method is as follows:

$$\omega_{t+1} = \omega_t + \eta_t \left[R_{t+1} + \gamma\ \omega_t^T\ x\left(S_t,\ A_t\right) - \omega_t^T\ x\left(S_t,\ A_t\right)\right] x\left(S_t,\ A_t\right) \tag{16}$$

and for the QL method is as follows:

$$\omega_{t+1} = \omega_t + \eta_t \left[R_{t+1} + \gamma \max_{a \in \mathcal{A}(S_{t+1})} \omega_t^T\ x\left(S_t,\ a\right) - \omega_t^T\ x\left(S_t,\ A_t\right)\right] x\left(S_t,\ A_t\right) \tag{17}$$

Note that convergence is not guaranteed in off-policy methods when using function approximation with a feature vector. To address this issue, the method called Greedy-GQ is introduced (Maei et al. 2010). Notably, in this method, convergence to the optimal point is assured when using the function approximation, and it imposes no restrictions for choosing the feature vector and supports online incremental updating (Bootstrapping). The update rule for the Greedy-GQ method has one more weight vector κ_t and the step size parameter ξ_t, and it is as follows:

$$\omega_{t+1} = \omega_t + \eta_t \left[\delta_{t+1}\ x\left(S_t,\ A_t\right) - \gamma\ \kappa_t^T\ x\left(S_t,\ A_t\right)\right] x\left(S_{t+1},\ a'\right) \tag{18}$$

$$\delta_{t+1} \leftarrow \left[R_{t+1} + \gamma \omega_t^T\ x\left(S_{t+1},\ a'\right) - \omega_t^T\ x\left(S_t,\ A_t\right)\right] \tag{19}$$

$$\kappa_{t+1} = \kappa_t + \xi \left[\delta_{t+1} - \kappa_t^T\ x\left(S_t,\ A_t\right)\right] x\left(S_t,\ A_t\right) \tag{20}$$

where $a' \in \arg\max_{a \in \mathcal{A}(S_{t+1})} \omega_t^T x\left(S_{t+1},\ a\right)$

3 Methodology

This section describes the methodology used in this research and shows the application of the SARSA, QL, and Greedy-GQ algorithms for managing the two-asset portfolio. Stocks from the New York Stock Exchange[3] market are used to test the algorithms. The Yahoo Finance[4] website is utilized to download the daily adjusted close prices of the chosen stocks for one year, from the 1st of October 2021 to the 1st of October 2022. This time frame is chosen because the financial market was very volatile due to the Covid-19 pandemic and the Russia-Ukraine conflict. Applying the algorithms in this period is a good challenge to face. Learning the real oscillations of the market can be very beneficial for the agent, and the results will be more interesting. Although the analysis covers a relatively short period of one year, this duration provides a sufficient number of trading days (approximately 250) for the reinforcement learning agents to learn how to trade well by interacting with the environment. The goal was not to model long-term market cycles, but to see how well the agent could adapt to changing conditions in a realistic and focused time frame. The investor, the RL agent, is assumed to be risk-neutral (Kabanov et al. 2006).

Although Yahoo Finance is widely used in academic research for its accessibility and broad market coverage, it is recognized that it may not provide the same level of data accuracy and reliability as proprietary platforms like Bloomberg or Refinitiv. This study could not utilize more specialized commercial databases because they were not available to most institutions. Nevertheless, the selected stocks are all large-cap and highly liquid, which reduces the likelihood of discrepancies in pricing data.

The model is applied to a portfolio of two stocks, one of which is a high-volatility stock and the other a low-volatility stock. The portfolio is constructed in this manner because the majority of actual portfolios contain a mixture of high- and low-volatility stocks (Jin and El-Saawy 2016). Thus, our chosen portfolio can serve as a simplified representation of an actual financial portfolio. The Beta index (Sharpe 1964) is used to evaluate the stock's riskiness. High-volatility (high-risk) stocks are those with a beta index greater than 1, while low-volatility (low-risk) stocks are those with a beta index less than 1. Stock pairs are selected based on high market capitalization (over $10 billion) and a high average volume of transactions. Hence, there is no cause for concern regarding slippage (slippage happens when the bid/ask spread differs between the time that the market order is placed and the time it is actually executed). A high average volume of transactions indicates that the expected transaction price, the price that the agent decides to trade, and the actual transaction price are nearly identical (Chiang and Chi 2024). The goal is to show that state-of-the-art reinforcement learning algorithms can be applied to dynamic portfolio management.

In this paper, the RL problem is represented as a discrete-time Markov decision process with $\mathcal{S}$ indicating the set of states, $\mathcal{A}$ representing the set of actions, and $\mathcal{R}$ displaying the set of rewards. At each time step, the agent goes long on one stock and goes short on another stock if it has one of them in its portfolio. The agent has the possibility of having nothing and staying out of the market (Corazza et al. 2019).

[3] https://www.nyse.com/index.

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

In other words, at any given time, the RL agent can choose between high- or low-volatility stocks or avoid the market and have nothing. The action set consists of three actions, $\mathcal{A} = \{-1, 0, 1\}$, where -1 means buying the low-volatility stock and selling the high-volatility one, action 0 means going out of the market or staying out of the market, and 1 means buying the high-volatility stock and selling the low-volatility one at each time step. It can either open the new position and go long for the low-volatility or high-volatility stock or close the current position and stay out of the market and have none of the stocks. The action in each state is chosen based on the selected control mechanism (e.g., SARSA, QL, or Greedy-GQ).

When using linear combinations of feature and weight vectors, it is very important to choose the right feature representation of the states for RL function approximation methods. So, choosing the right state representation can significantly impact model performance. The aim of the state vector is to summarize recent market behavior and past trading performance in a compact form, which the RL agent uses to make decisions. Including both price-based features and feedback from past trades helps the agent learn from recent outcomes. Following the approach introduced by Corazza et al. (2019), the state vector at each time step t is constructed using the last n log-returns of the two selected stocks combined with the profit or loss of the most recent trade.

It is necessary to apply a suitable transformation of the state vector elements (basis function) in order to enhance numerical stability and facilitate convergence. In this research, the logistic function is chosen as the basis function, defined as:

$$\phi(x) = \frac{a}{1 + be^{-cx}} + d \tag{21}$$

where x denotes the state vector and parameters are set as $a = 1, b = 2, c = 10^2$, and $d = -1$, consistent with Corazza et al. (2019). This transformation makes it possible for the state vector to store recent market activity for both stocks and the last trade outcome in a standardized way. The choice of structure is in line with the goal of providing the learning algorithm with symmetric, bounded, and informative inputs, which facilitates convergence and stability in training.

The log-return function is defined as follows:

$$\text{logret}\,(v) = \begin{cases} \ln\left(\dfrac{P^j_{t-n+i}}{P^j_{t-n+i-1}}\right), & \text{if } v = S \text{ and } j = 1, 2 \\ \ln\left(\dfrac{\text{current capital}}{\text{entry capital}}\right), & \text{if } v = \bar{R} \end{cases} \tag{22}$$

where P^j_t is an adjusted closed price at time t of stock j, and v indicates the type of return being computed. The basis vector $\overrightarrow{\phi} \in \mathbb{R}^{2n+1}$, which will be used in the feature vector later, is defined as follows:

$$\phi^j_i = \phi(\text{logret}\,(S^j)), \quad \text{if } i = 1, 2, \ldots, n \text{ and } j = 1, 2 \tag{23}$$

$$\phi_{n+1} = \phi(\text{logret}\,(\bar{R})) \tag{24}$$

where "logret" is the log-return function (22), j = 1 is for the low-volatility stock, and j = 2 is for the high-volatility stock, and $\bar{R}$ is the profit or loss of the last trade.

After applying the logistic function, the resulting basis vector, denoted by $\overrightarrow{\phi}$, is constructed as follows:

$$\overrightarrow{\phi} = \left[\phi(S^1_{t-n+1}), \ldots, \phi(S^1_t), \phi(S^2_{t-n+1}), \ldots, \phi(S^2_t), \phi(\bar{R})\right]_{2n+1} \tag{25}$$

where S^1 and S^2 represent the low-volatility and high-volatility stocks, respectively.

The dimension of the basis vector $\overrightarrow{\phi}$ is $2n + 1$, and the dimension of the feature vector is $|\mathcal{A}| * (2n + 1)$, which in this case is $3 * (2n + 1)$. Given that the action set is defined by $\mathcal{A} = -1, 0, 1$, indicating the three possible portfolio positions, the feature vector $x(S_t, A_t)$ incorporates action-specific representation through the block feature expansion technique introduced by Geramifard et al. (2011). This technique is widely employed within the field of linear function approximation of RL. Using this method, the block representation of the feature vector is utilized, and based on the currently selected action, one of the three slots in the feature vector is filled with the basis vector $\overrightarrow{\phi}$. This block structure makes sure that each action has its own feature subspace, which lets the model learn different value functions for each possible decision. The structure is defined as follows:

$$\mathbf{x}(S_t, A_t) = \begin{cases} [\boldsymbol{\phi}\ \mathbf{0}\ \mathbf{0}], & \text{if } A_t = -1 \\ [\mathbf{0}\ \boldsymbol{\phi}\ \mathbf{0}], & \text{if } A_t = 0 \\ [\mathbf{0}\ \mathbf{0}\ \boldsymbol{\phi}], & \text{if } A_t = 1 \end{cases} \tag{26}$$

where $\overrightarrow{0} \in \mathbb{R}^{2n+1}$.

This structured and explicit representation ensures clarity and facilitates the efficient implementation of the reinforcement learning algorithms employed in this research.

A fixed transaction cost of 2.28 cents per trade is included in the model to make it more like real-life trading. All three reinforcement learning algorithms take this cost into account when figuring out how much they will gain during training and testing. Every time a position is opened or closed, there is a transaction cost. If both actions happen at the same time, the cost is doubled. This structure makes sure that the impact of transaction fees is reflected in the reward signals received by the agent.

The transaction cost δ is incorporated directly into the portfolio return, which serves as the base for the reward. The adjusted return g_t is computed as:

$$g_t = \text{PortfolioReturn}_t - \delta \cdot |a_t - a_{t-1}|,$$

where $a_t \in \{-1, 0, 1\}$ denotes the action at time t, and $\delta = 0.0228$ represents the fixed transaction cost in USD per trade.

The model employs three reward functions. The first reward function used is the Calmar ratio, which is a risk-adjusted return measure,

$$R_{t+1} = \frac{\text{mean}(g_{l,t+1}) - r}{1 + \max DD_{l,t+1}} \tag{27}$$

Table 1 Hyperparameters and their values

Hyperparameters	Values
n	1, 5, 10, 40
l	1, 5, 20
$\eta = \xi$	0.01, 0.1, 0.25
ε	0.01, 0.05, 0.1
γ	0.95, 1
reward function	Calmar, Sharpe, Penalized

where $g_{l,t+1}$ represents the vector of the most recent l daily returns up to time $t+1$ and mean$(g_{l,t+1})$ represents the average profit and loss over the last l trading days at time $t+1$, max $DD_{l,t+1}$ indicates the maximum drawdown over the last l trading days at time $t+1$, and r is a daily risk-free return. The daily risk-free rate is negligible over the short evaluation horizon, r is assumed to be zero in this research. The maximum drawdown is the maximum loss value from the equity line's highest value to its lowest value over the past l trading days.

The second reward function (penalized reward) used is the portfolio return with a volatility penalty, as in Jin and El-Saawy (2016). The penalized reward function is defined as

$$R_{t+1} = g_{t+1} - \lambda\sigma(g_{l,t+1}) \tag{28}$$

where $\lambda = 0.5$, g_{t+1} denotes the portfolio return on day $t+1$ (last portfolio return), and $\sigma(g_{l,t+1})$ is the standard deviation of profit and loss over the past l trading days.

The third reward function utilized in this model is the well-known Sharpe ratio, defined as follows:

$$R_{t+1} = \frac{mean(g_{l,t+1}) - r}{\sigma(g_{l,t+1})} \tag{29}$$

In the field of AI, hyperparameters play an important role and must be set before the method begins. In this paper, three RL algorithms are considered, each having seven hyperparameters whose values are derived from the literature. The performance of the algorithms is evaluated across various combinations of hyperparameters. The parameters varied include: η (learning rate), ξ (secondary learning rate used in Greedy-GQ), γ (discount factor), ε (exploration probability in the ε-greedy policy), n (number of past returns used in the state vector), l (lookback window for computing rewards), and the choice of reward function. Table 1 summarizes the hyperparameters and their corresponding values utilized in this research. The values were chosen based on the literature (Corazza et al. 2019). These ranges were chosen to explore both conservative and aggressive learning settings and to ensure robustness across varying market conditions.

According to Table 1, there are 648 configurations for each RL algorithm (i.e. SARSA, QL, and Greedy-GQ). For each configuration, 1000 simulations are executed, resulting in 1000 distinct actions at each time step t. However, only one action, $\tilde{A}_t$, is required at each time step to take. Two aggregation techniques are used to achieve this: a threshold-based rule and a statistical significance test. The first method involves calculating the average of actions at each time step t (1000 actions), $\bar{A}_t$, determining

a threshold $\iota \in (0, 1)$, and then defining the following rules:

$$\bar{A}_t = \frac{\sum_{i=1}^{N=1000} \mathcal{A}_{i,t}}{N} \tag{30}$$

$$\tilde{A}_t = \begin{cases} -1, & \text{if } \bar{A}_t \in [-1, -\iota) \\ 0, & \text{if } \bar{A}_t \in [-\iota, \iota) \\ 1, & \text{if } \bar{A}_t \in [\iota, 1) \end{cases} \tag{31}$$

The second method used in this model considers $\{A_{i,t}\}_{i=1}^{N=1000}$ independent and identically distributed random variables. An average of the random variables, $\bar{A}_t$, is calculated, and from statistics, $\bar{A}_t$ is known to tend toward a normal distribution. Then, a statistical test is performed on $\bar{A}_t$ under the null hypothesis to determine whether it follows a Student's t-distribution at a given significance level (Corazza et al. 2019). Therefore, $\tilde{A}_t = 0$ if the hypothesis is accepted and if the hypothesis is rejected, the $\tilde{A}_t$ is the sign of $\bar{A}_t$. Based on the literature, ι is set to 0.05, and the significance level to 5%.

To evaluate how well RL algorithms work, we compare their results to those of the equally weighted portfolio, which is one of the most common benchmark strategies used in financial research. This strategy has been shown to be strong and often outperforming in a variety of market conditions (Palit and Prybutok 2024; Swade et al. 2022). The equally weighted portfolio is rebalanced at each trading period using the updated portfolio value. This makes sure that the two assets share capital evenly throughout the investment period. An annualized return is considered the evaluation metric to compare the performance of the RL algorithms with the benchmark. This metric is particularly relevant for risk-neutral investors who are primarily concerned with overall returns rather than risk-adjusted measures. $1000 is chosen as the initial capital to start trading.

In the next section, the results of the RL algorithms managing a two-stock portfolio will be presented and compared with those of the benchmark.

4 Results and discussion

The RL methods were applied to five pairs of stocks, each consisting of one high-volatility stock and one low-volatility stock, which are ("BALL", "APA"), ("WMT", "F"), ("TM", "AAPL"), ("VZ", "NVDA"), and ("NEM", "JBL"). The stock pairs were selected based on their historical beta values, which measure each stock's sensitivity to market movements. Specifically, five high-beta (high-volatility) and five low-beta (low-volatility) stocks were randomly selected from the S&P 500 index and then paired so that each pair consisted of one high-beta and one low-beta stock. This approach allows the analysis to capture differences in risk profiles while maintaining representative large-cap U.S. equities. Here, the RL methods will be demonstrated and evaluated for one selected pair (additional results for the remaining pairs are provided in Appendix 6). The selected pair consists of Ball Corporation shares with the symbol "BALL" representing the low-volatility stock and American Psychological

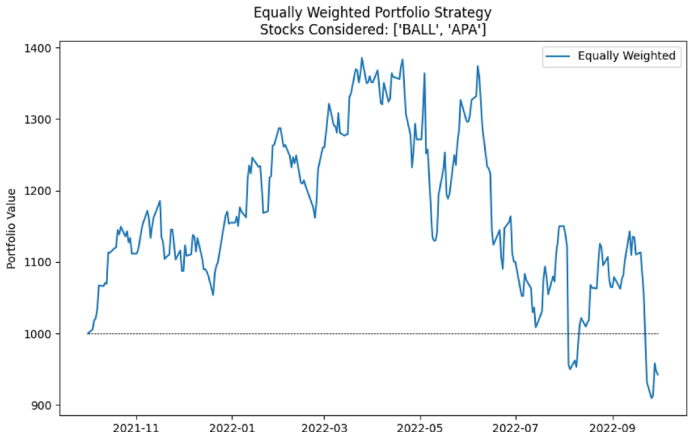

Fig. 2 Portfolio Value of the benchmark, the equally weighted portfolio, including two stocks, BALL and APA, for the period of 01/10/2021-01/10/2022

Association shares with the symbol "APA" representing the high-volatility stock. As stated previously, the implementation utilizes the daily adjusted-closed price for a duration of one year, from the 1st of October 2021 to the 1st of October 2022. As described at the end of Section 3, two aggregation techniques were tested to determine the final action at each time step: (i) a threshold-based averaging rule and (ii) a statistical significance test on the mean of simulated actions. After assessing the performance of two methods for determining actions for each time step, it was observed that the first method performed slightly better. That is why only the results of the first method are reported here.

Figure 2 illustrates the equity line for the benchmark, the equally weighted portfolio, and its annualized return over the period was -5.74%. The equally weighted benchmark portfolio was rebalanced monthly to maintain equal capital allocation between the two stocks.

Most of the previous studies neglected to consider transaction costs in their models, but to make the environment in this research more realistic, a fixed transaction cost of 2.28 cents per transaction is included in the model, consistent with the average magnitude of fees reported by the New York Stock Exchange[5]. This fixed-fee approximation represents a realistic aggregate cost for small retail-size trades and is adopted for simplicity in the simulation. After running all configurations, the most profitable configuration is for the case where the SARSA method is used with the following parameters; $n = 10, l = 5, \eta = 0.01, \varepsilon = 0.01, \gamma = 0.95$, and the Sharpe ratio as the reward function, which results in a 187.78% annualized return. Tables 2 and 3 present the top twenty best and the top twenty worst configurations, respectively. Figure 3 shows the performance of the best configuration, the actions chosen by the method, and its equity line over the specified period.

Looking at the tables of the top twenty best and worst configurations (Tables 2 and 3), it could be possible to say that the best performance happened by good luck,

[5] https://www.nyse.com/publicdocs/nyse/markets/nyse/NYSE_Price_List.pdf.

Table 2 The top 20 best configurations of the model based on the annualized return for the two-stock portfolio of BALL and APA over the period from October 1, 2021, to October 1, 2022

	Hyperparameters	Annualized Return %
1	SARSA, n=10, l=5, $\eta = 0.01$, $\varepsilon = 0.01$, $\gamma = 0.95$, Sharpe	187.78
2	SARSA, n=5, l=5, $\eta = 0.1$, $\varepsilon = 0.01$, $\gamma = 1$, Calmar	144.36
3	SARSA, n=5, l=5, $\eta = 0.1$, $\varepsilon = 0.01$, $\gamma = 0.95$, Calmar	132.61
4	SARSA, n=10, l=5, $\eta = 0.01$, $\varepsilon = 0.05$, $\gamma = 0.95$, Sharpe	109.35
5	SARSA, n=40, l=5, $\eta = 0.25$, $\varepsilon = 0.01$, $\gamma = 0.95$, Penalized	93.59
6	SARSA, n=10, l=5, $\eta = 0.01$, $\varepsilon = 0.01$, $\gamma = 0.95$, Calmar	93.08
7	SARSA, n=10, l=5, $\eta = 0.01$, $\varepsilon = 0.01$, $\gamma = 1$, Sharpe	91.41
8	SARSA, n=10, l=5, $\eta = 0.01$, $\varepsilon = 0.05$, $\gamma = 0.95$, Calmar	89.59
9	SARSA, n=10, l=5, $\eta = 0.01$, $\varepsilon = 0.01$, $\gamma = 1$, Calmar	87.19
10	QL, n=10, l=5, $\eta = 0.1$, $\varepsilon = 0.01$, $\gamma = 0.95$, Sharpe	83.75
11	SARSA, n=10, l=5, $\eta = 0.01$, $\varepsilon = 0.1$, $\gamma = 1$, Calmar	79.48
12	SARSA, n=10, l=1, $\eta = 0.25$, $\varepsilon = 0.01$, $\gamma = 0.95$, Calmar	76.68
13	SARSA, n=5, l=5, $\eta = 0.1$, $\varepsilon = 0.05$, $\gamma = 0.95$, Calmar	75.39
14	SARSA, n=10, l=5, $\eta = 0.01$, $\varepsilon = 0.1$, $\gamma = 0.95$, Calmar	71.16
15	SARSA, n=10, l=5, $\eta = 0.01$, $\varepsilon = 0.05$, $\gamma = 1$, Calmar	66.27
16	QL, n=5, l=1, $\eta = 0.1$, $\varepsilon = 0.05$, $\gamma = 1$, Calmar	63.61
17	SARSA, n=5, l=5, $\eta = 0.01$, $\varepsilon = 0.1$, $\gamma = 0.95$, Calmar	58.24
18	SARSA, n=5, l=5, $\eta = 0.25$, $\varepsilon = 0.05$, $\gamma = 0.95$, Calmar	57.67
19	QL, n=40, l=1, $\eta = 025$, $\varepsilon = 0.01$, $\gamma = 1$, Calmar	56.66
20	SARSA, n=10, l=5, $\eta = 0.01$, $\varepsilon = 0.05$, $\gamma = 1$, Sharpe	56.02

and the worst one by bad luck, due to the problem of selection bias under multiple testing (López de Prado and Lewis 2019). This refers to the risk of falsely identifying a strategy as successful due to random chance when many configurations are tested. To mitigate this, we report performance metrics averaged across all configurations. Table 4 reports the performance of the methods for each hyperparameter on average. The first two columns correspond to the SARSA method: the first column presents the annualized return (%), and the second one shows the percentage of configurations with that hyperparameter which resulted a higher annualized return than the benchmark (-5.74%). The second two and third pairs of columns correspond to the QL and Greedy-GQ methods, respectively. It is evident from Table 4 that on average, Greedy-GQ demonstrated stronger performance, particularly in terms of return stability and robustness to parameter changes.

The performance of RL methods varied significantly depending on the hyperparameters chosen (Table 4). The state vector's number of past returns (n) played a crucial role in determining strategy effectiveness. The values $n = 5$ and $n = 10$ consistently outperformed other choices across all RL methods. In particular, $n = 5$ provided the best performance for SARSA and QL, while $n = 10$ was optimal for Greedy-GQ. It was found that the worst performance happened when $n = 1$. This suggests that making decisions based on very short historical data is not sufficient. Similarly, $n = 40$

Table 3 The top 20 worst configurations of the model based on the annualized return for the two-stock portfolio of BALL and APA over the period from October 1, 2021, to October 1, 2022

	Hyperparameters	Annualized Return %
1	QL, n=40, l=5, $\eta = 0.1$, $\varepsilon = 0.01$, $\gamma = 1$ Calmar	−72.81
2	SARSA, n=40, l=1, $\eta = 0.01$, $\varepsilon = 0.05$, $\gamma = 1$, Calmar	−67.36
3	QL, n=40, l=5, $\eta = 0.1$, $\varepsilon = 0.05$, $\gamma = 1$, Calmar	−66.54
4	SARSA, n=40, l=1, $\eta = 0.01$, $\varepsilon = 0.01$, $\gamma = 1$, Calmar	−66.30
5	SARSA, n=40, l=20, $\eta = 0.25$, $\varepsilon = 0.01$, $\gamma = 1$, Penalized	−65.50
6	SARSA, n=40, l=1, $\eta = 0.01$, $\varepsilon = 0.01$, $\gamma = 0.95$, Calmar	−62.76
7	QL, n=10, l=20, $\eta = 0.1$, $\varepsilon = 0.01$, $\gamma = 0.95$, Calmar	−61.19
8	QL, n=1, l=20, $\eta = 0.1$, $\varepsilon = 0.01$, $\gamma = 0.95$, Calmar	−60.98
9	SARSA, n=40, l=5, $\eta = 0.01$, $\varepsilon = 0.01$, $\gamma = 0.95$, Sharpe	−60.63
10	SARSA, n=40, l=1, $\eta = 0.01$, $\varepsilon = 0.05$, $\gamma = 0.95$, Calmar	−58.50
11	QL, n=40, l=20, $\eta = 0.1$, $\varepsilon = 0.01$, $\gamma = 1$, Calmar	−58.25
12	SARSA, n=40, l=20, $\eta = 0.01$, $\varepsilon = 0.1$, $\gamma = 1$, Sharpe	−57.80
13	SARSA, n=1, l=20, $\eta = 0.01$, $\varepsilon = 0.01$, $\gamma = 1$, Sharpe	−57.61
14	SARSA, n=40, l=5, $\eta = 0.01$, $\varepsilon = 0.05$, $\gamma = 0.95$, Sharpe	−57.13
15	SARSA, n=40, l=5, $\eta = 0.01$, $\varepsilon = 0.01$, $\gamma = 1$, Sharpe	−57.04
16	SARSA, n=40, l=5, $\eta = 0.01$, $\varepsilon = 0.1$, $\gamma = 0.95$, Calmar	−56.85
17	QL, n=10, l=20, $\eta = 0.1$, $\varepsilon = 0.01$, $\gamma = 0.95$, Penalized	−56.34
18	QL, n=10, l=20, $\eta = 0.01$, $\varepsilon = 0.01$, $\gamma = 0.95$, Calmar	−56.19
19	QL, n=10, l=5, $\eta = 0.01$, $\varepsilon = 0.05$, $\gamma = 0.95$, Calmar	−56.05
20	SARSA, n=10, l=20, $\eta = 0.1$, $\varepsilon = 0.01$, $\gamma = 1$, Penalized	−55.23

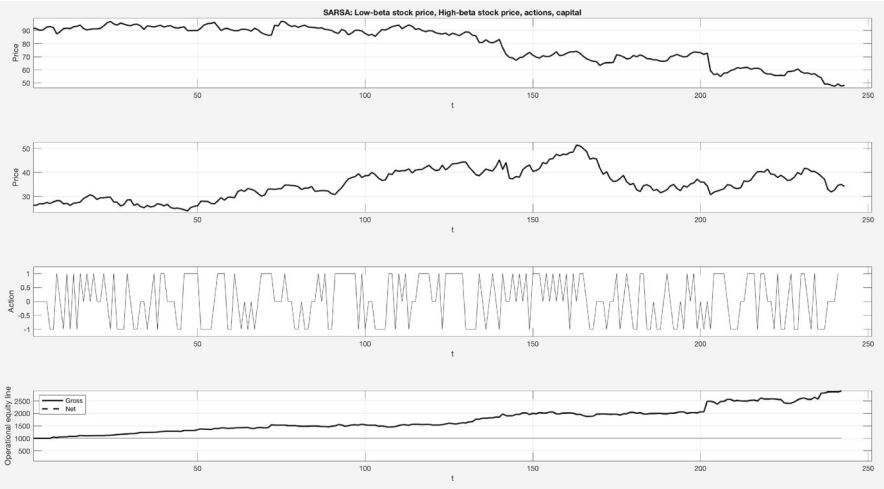

Fig. 3 Performance of the best configuration (its hyperparameters are n = 10, l = 5, $\eta = 0.01$, $\varepsilon = 0.01$, $\gamma = 0.95$, reward function = Sharpe, and the model is SARSA): The first two graphs show the adjusted-closed price of the low–β stock, and high–β stock, respectively. The third graph shows the actions chosen by the method over the period. The graph at the bottom illustrates the value of the portfolio over the period

did not yield strong results, likely due to an excessive amount of past data leading to inefficient learning.

The lookback window for calculating rewards (l) also had an effect on the results. For Greedy-GQ, $l = 1$ worked best, while for SARSA, $l = 5$ worked best. The longest window, $l = 20$, always led to worse performance across all methods. This suggests that a longer reward horizon may make the model less responsive to changes in the market. The learning rate (η) showed that, in most cases, higher values improved performance. The learning rate that worked best for all methods was $\eta = 0.25$. This made convergence happen faster while keeping things stable. Values like $\eta = 0.01$ and $\eta = 0.1$ made performance worse, especially for QL. This suggests that slow learning rates make it harder to adapt to changes in the market.

The exploration-exploitation balance, which was controlled by the exploration rate (ε), showed that SARSA and QL worked best with moderate exploration ($\varepsilon = 0.1$), while Greedy-GQ worked better with a lower exploration rate ($\varepsilon = 0.01$). This means that off-policy methods like Greedy-GQ may not need as much exploration, while on-policy methods like SARSA do better when they explore more often. The discount factor (γ) had an effect on optimizing long-term rewards. $\gamma = 0.95$ always performed slightly better than $\gamma = 1$ across all methods. This suggests that discounting future rewards slightly helps stabilize learning without overly prioritizing immediate gains.

Finally, the choice of reward function had a significant impact on overall performance. The Sharpe ratio was the best reward function for all RL methods, giving better returns than the Calmar and penalized reward functions. The penalized reward function did the worst, which shows that punishing volatility too much led to strategies that were too cautious.

Lastly, the performance of each hyperparameter across all configurations is presented in Table 5. The hyperparameter $n = 5$ generated the highest annualized return over the period and was the only positive one. Regarding the reward function, the Sharpe ratio outperformed the other reward functions and was the most stable one with performing better than the benchmark in 61% of its configurations.

Table 6 illustrates the overall performance of each RL method for the pair of BALL and APA during the considered period. According to the table, the methods SARSA and Greedy-GQ beat the benchmark on average. The method QL could not perform better than the benchmark in this period.

To improve the performance of the methods, the state vector was expanded to have more elements and provide the agent with more information about the environment. These additions introduce trend and momentum signals, such as moving averages and relative strength, which can help the agent better anticipate short-term reversals or continuations. From now on, the original state vector used previously is named "base state vector". Firstly, the β–indices of two stocks were added to the state vector, and this modification is referred to as "Type 1". The second attempt to improve the methods by modifying the state vector was incorporating two technical indicators, Moving Average[6] and Relative Strength Index [7], to the base state vector. This approach is named "Type 2". The third modification to improve the methods by changing the base

[6] https://www.investopedia.com/terms/m/movingaverage.asp.

[7] https://www.investopedia.com/terms/r/rsi.asp.

Table 4 Hyperparameter statistics, reported on average, for the model performed for the two-stock portfolio of BALL and APA over the period from October 1, 2021, to October 1, 2022

	SARSA		QL		Greedy-GQ	
	$R_{\text{annualized}}\%$	$\% > -5.74$	$R_{\text{annualized}}\%$	$\% > -5.74$	$R_{\text{annualized}}\%$	$\% > -5.74$
All Configurations	−5.57	51	−13.05	40	−0.06	74
n = 1	−17.96	31	−23.34	22	−0.3	81
n = 5	5.71	66	−5.10	52	−0.24	77
n = 10	−0.94	57	−10.12	46	2.26	76
n = 40	−9.11	50	−13.67	39	−1.96	64
l = 1	−2.80	64	−4.02	65	0.11	75
l = 5	−1.59	50	−14.50	33	−0.89	69
l = 20	−12.33	40	−20.66	21	0.60	79
$\eta = 0.01$	−10.69	40	−18.24	31	0.39	76
$\eta = 0.1$	−6.14	52	−16.32	32	0.31	77
$\eta = 0.25$	0.11	61	−4.62	56	−0.88	70
$\varepsilon = 0.01$	−6.13	48	−14.87	39	0.08	76
$\varepsilon = 0.05$	−6.57	49	−12.25	39	0.02	76
$\varepsilon = 0.1$	−4.02	57	−12.05	41	−0.28	71
$\gamma = 0.95$	−5.21	52	−13.22	37	0.73	76
$\gamma = 1$	−5.93	51	−12.9	42	−0.85	73
Sharpe	−3.54	59	−11.18	45	0.71	79
Calmar	−3.30	50	−17.13	30	−0.01	75
Penalized	−9.87	44	−10.87	44	−0.88	70

state vector was a combination of the first and second types, adding both the β–indices and the technical indicators of two stocks to the base state vector. Methods using this state vector are referred to as "Type 3".

Furthermore, an attempt was made to improve the methods by using adaptive learning rates. Hence, to evaluate the impact, the performance of Type 3 (having technical indicators and the β–indices) was compared with its variant using adaptive learning rates. After each 60 days, the learning rates were divided by 2. Methods utilizing the Type 3 state vector and adaptive learning rates are named "Type 4".

Figures 4, 5, and 6 illustrate the results of the SARSA, Q-Learning, and Greedy-GQ methods, respectively, when using different state vector types and adaptive learning rates. The red line in the figures represents the benchmark's annualized return. In all cases, adding more informative state variables led to improved average performance, particularly for SARSA and Q-Learning. Greedy-GQ remained strong but showed diminishing returns from added features. Figure 4 presents the results for the SARSA method using the proposed state vectors. Across all state vector types, the SARSA method performed better than the benchmark on average. Moreover, it achieved a higher annualized return when it had more information in its state vector. SARSA method Type 4 (having the state vector Type 3 and using adaptive learning rates) performed on average slightly better than Type 3, but its spread was narrower.

Table 5 Hyperparameters performance, reported on average, for all configurations of the model performed for the two-stock portfolio of BALL and APA over the period from October 1, 2021, to October 1, 2022

	$R_{\text{annualized}}\%$	$\% > -5.74$
n = 1	−13.87	45
n = 5	0.13	65
n = 10	−2.93	60
n = 40	−8.25	51
l = 1	−2.24	68
l = 5	−5.65	50
l = 20	−10.80	47
$\eta = 0.01$	−9.51	49
$\eta = 0.1$	−7.38	54
$\eta = 0.25$	−1.80	62
$\varepsilon = 0.01$	−6.97	54
$\varepsilon = 0.05$	−6.27	55
$\varepsilon = 0.1$	−5.45	56
$\gamma = 0.95$	−5.90	55
$\gamma = 1$	−6.56	55
Sharpe	−4.67	61
Calmar	−6.81	52
Penalized	−7.21	53

Table 6 Annualized return of the RL methods and the equally weighted portfolio strategy for the two-stock portfolio of BALL and APA over the period from October 1, 2021, to October 1, 2022

RL algorithms	Annual Return %
SARSA	−5.57
QL	−13.05
Greedy−GQ	−0.06
Equally−weighted portfolio (benchmark)	−5.74

Figure 5 illustrates the performance of the QL method using different proposed state vectors. Similar to the SARSA method, it performed better when it has more information in its state vector. For instance, the QL method with the base state vector could not beat the benchmark on average; however, using state vector Type 3 (which incorporates technical indicators into the state vector), it surpassed the benchmark on average. Furthermore, having adaptive learning rates could help the QL method to perform better on average.

Figure 6 illustrates the performance of the Greedy-GQ method using different types of state vectors and adaptive learning rates. In this method, using more information in the state vector did not lead to the improved results. However, across all types, Greedy-GQ outperformed the benchmark and the other two methods. One possible explanation regarding the lack of positive effect of having more information in the state vector for the Greedy-GQ method is that, due to the greater complexity of Greedy-GQ compared

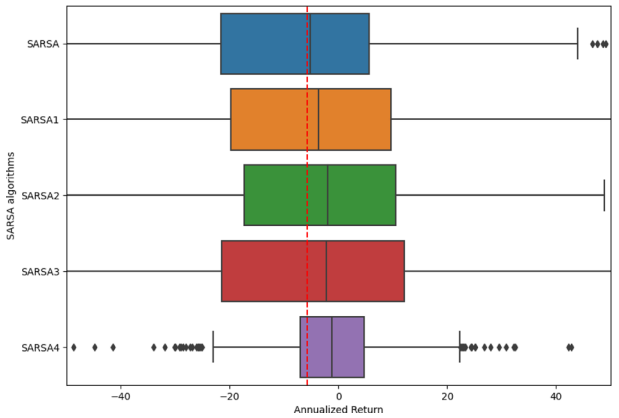

Fig. 4 Performance of the SARSA method (with different state vectors) for the two-stock portfolio of BALL and APA over the period from October 1, 2021, to October 1, 2022. Red line is the benchmark's annualized return

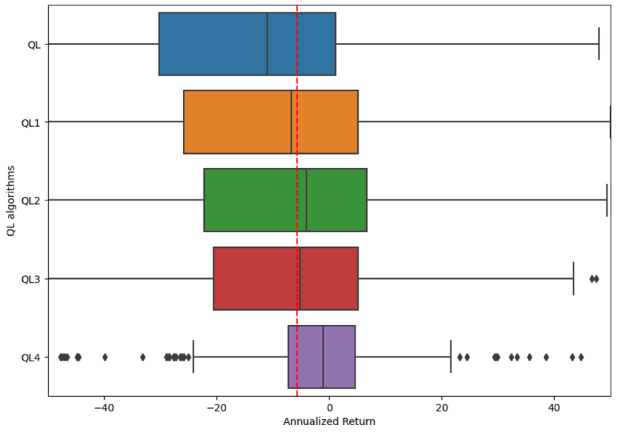

Fig. 5 Performance of the QL method (with different state vectors) for the two-stock portfolio of BALL and APA over the period from October 1, 2021, to October 1, 2022. Red line is the benchmark's annualized return

to the other two methods, it requires more time to effectively use this information and decide better over the period. This aspect will be further evaluated in the robustness check.

The pie charts in Fig. 7 present the percentage of all configurations for each RL method (with Type 3 state vector) that outperformed the benchmark (equally-weighted portfolio). Based on the pie charts, Greedy-GQ demonstrates better and more stable performance with resulting better than the benchmark in 70% of all configurations. The second-best method is SARSA, which surpassed the benchmark in 58% of its configurations. In contrast, the QL method did not perform well with almost half of its configurations exceeding the benchmark.

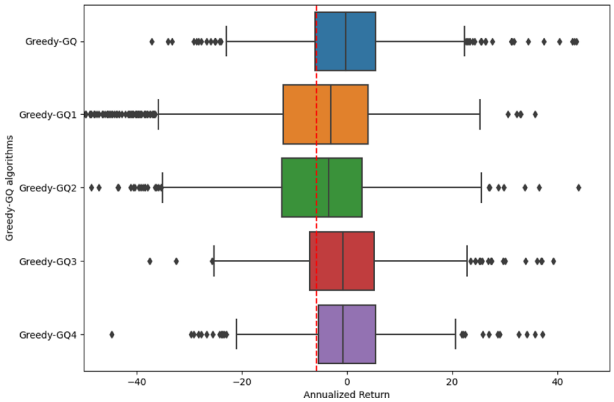

Fig. 6 Performance of the Greedy-GQ method (with different state vectors) for the two-stock portfolio of BALL and APA over the period from October 1, 2021, to October 1, 2022. Red line is the benchmark's annualized return

4.1 Robustness check

For the robustness check of the model, the methods were run for the period of the 2007-2008 financial crisis. This period can be a meaningful challenge for the RL methods, as the financial market was very volatile during this time. Hence, the methods were run for the stock pair BALL and APA over the period from August 1, 2007, to November 1, 2008 (319 days). Again, the model is compared with the equally weighted portfolio as the benchmark.

Figure 8 presents the results of using the benchmark for this period. The equally weighted portfolio benchmark generated approximately -13.13% by the end of this period.

The performance of the methods using the base state vector is presented in Table 7. All methods outperformed the benchmark on average. Greed-GQ performed the best among the other two methods.

The performance of the methods, on average, for each hyperparameter is reported in Table 8. The results indicate that the Greedy-GQ method remains the best-performing method on average during this period. Another notable finding from the table is that the penalized reward function outperformed the other two reward functions.

Moreover, the methods were run using other types of proposed state vectors over this period, similar to the approach taken for the period from October 2021 to October 2022. The results are shown and compared in Fig. 9.

Fig. 9 confirms that including technical indicators significantly enhances the models' returns, especially during the financial crisis, by enabling the agent to respond to changing market regimes more effectively. Across all types of state vector, the methods outperformed the benchmark (equally weighted portfolio strategy) on average. Moreover, except for the base state vector methods, Greedy-GQ demonstrated the best performance. Using the two technical indicators in the state vector had a significant positive impact on performance. Type 2, Type 3, and Type 4 methods, all of which

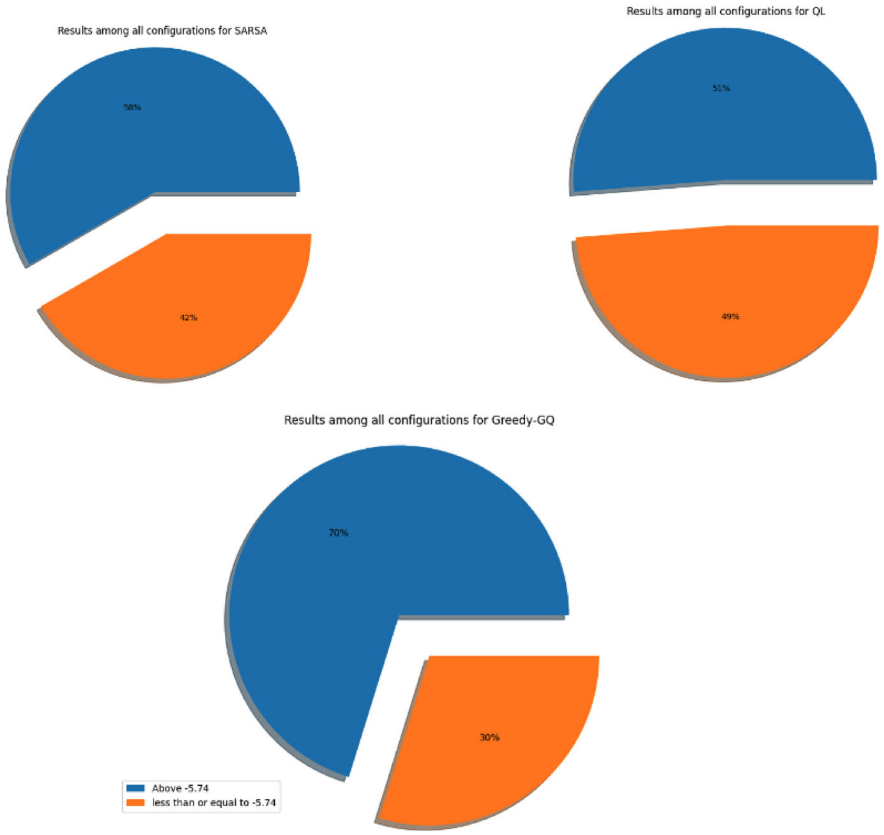

Fig. 7 Pie plots illustrate the percentage of all configurations for each RL method outperformed the benchmark. The configurations are run for the two-stock portfolio of BALL and APA over the period from October 1, 2021, to October 1, 2022. The top left graph represents the results for SARSA, and the top right one shows the same for the QL method. The bottom graph demonstrates the results of the Greedy-GQ method

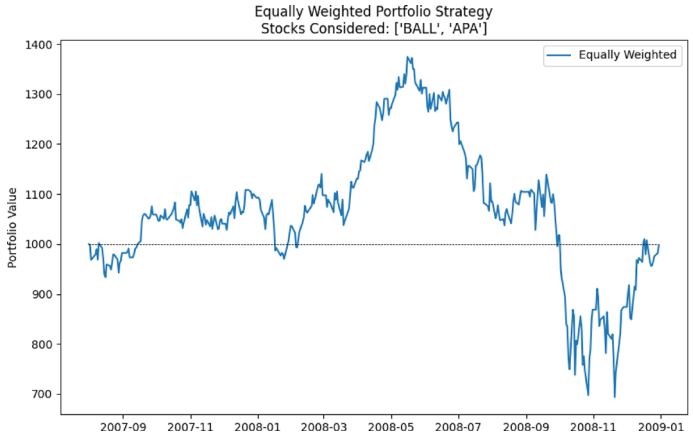

Fig. 8 Portfolio Value of the benchmark, the equally weighted portfolio strategy, for the two-stock portfolio of BALL and APA over the period from August 1, 2007, to November 1, 2008

Table 7 Annualized return of the RL methods and the benchmark for the two-stock portfolio of BALL and APA over the period from August 1, 2007, to November 1, 2008

Methods	Annualized Return %
SARSA	−4.22
QL	−7.57
Greedy-GQ	−2.13
Equally-weighted portfolio (benchmark)	−13.13

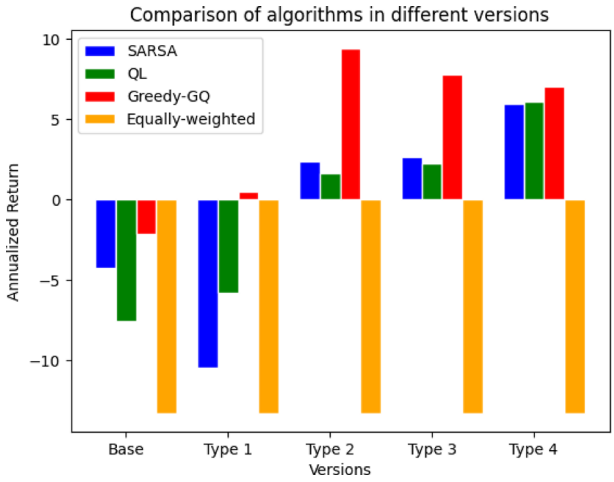

Fig. 9 Performance and comparison of the methods (using different state vectors) for the two-stock portfolio of BALL and APA over the period from August 1, 2007, to November 1, 2008

utilize the technical indicators in their state vectors, resulted in a positive annualized return.

The pie charts in Fig. 10 illustrate the percentage of all configurations for each RL method (Type 3) that performed better than the benchmark.

4.1.1 Robustness Under higher transaction costs

We used the same fixed transaction cost (2.28 cents per trade) that we used during the non-crisis period for the main robustness test that looked at the 2007–2008 financial crisis. But when there is a financial crisis, markets usually have less liquidity and more slippage, which makes transactions more expensive. We did another robustness analysis to better reflect these conditions by doubling the transaction costs for the same crisis period (August 1, 2007, to November 1, 2008).[8].(See Fig. 11,12,13).

Even though there were more trading frictions, all RL methods still did better than the benchmark, equally weighted portfolio strategy. Also, the performance of each algorithm got better and better as richer state representations were added, from Base

[8] The transaction cost was increased from $0.0228 to $0.0456 per trade.

Table 8 Hyperparameter Statistics, reported on average, for the model performed for the two-stock portfolio of BALL and APA over the period from August 1, 2007, to November 1, 2008

	SARSA		QL		Greedy-GQ	
	$R_{\text{annualized}}\%$	$\% > -13.3$	$R_{\text{annualized}}\%$	$\% > -13.3$	$R_{\text{annualized}}\%$	$\% > -13.3$
All Configurations	−4.23	72	−7.57	63	−2.13	91
n = 1	−14.45	49	−18.71	43	−0.25	98
n = 5	−8.32	65	−6.93	63	−0.21	95
n = 10	−1.57	81	−3.75	75	−2.86	88
n = 40	7.43	91	−0.89	69	−5.21	85
l = 1	−2.21	83	−1.66	85	−0.62	94
l = 5	−5.18	67	−6.59	55	−2.28	91
l = 20	−5.29	65	−14.45	49	3.50	89
$\eta = 0.01$	−0.43	75	−0.89	67	−2.14	92
$\eta = 0.1$	−5.21	68	−12.84	56	−2.35	90
$\eta = 0.25$	−7.03	72	−8.98	65	−1.91	92
$\varepsilon = 0.01$	−4.54	68	−5.03	66	−1.95	91
$\varepsilon = 0.05$	−4.12	74	−9.09	62	−2.46	90
$\varepsilon = 0.1$	−4.02	74	−8.58	66	−1.99	92
$\gamma = 0.95$	−4.10	72	−8.76	59	−1.84	94
$\gamma = 1$	−4.35	72	−6.37	66	−2.42	88
Sharpe	−5.99	74	−9.05	62	−1.71	94
Calmar	−9.56	56	−12.90	48	−3.19	88
Penalized	2.88	85	−0.74	78	−1.29	92

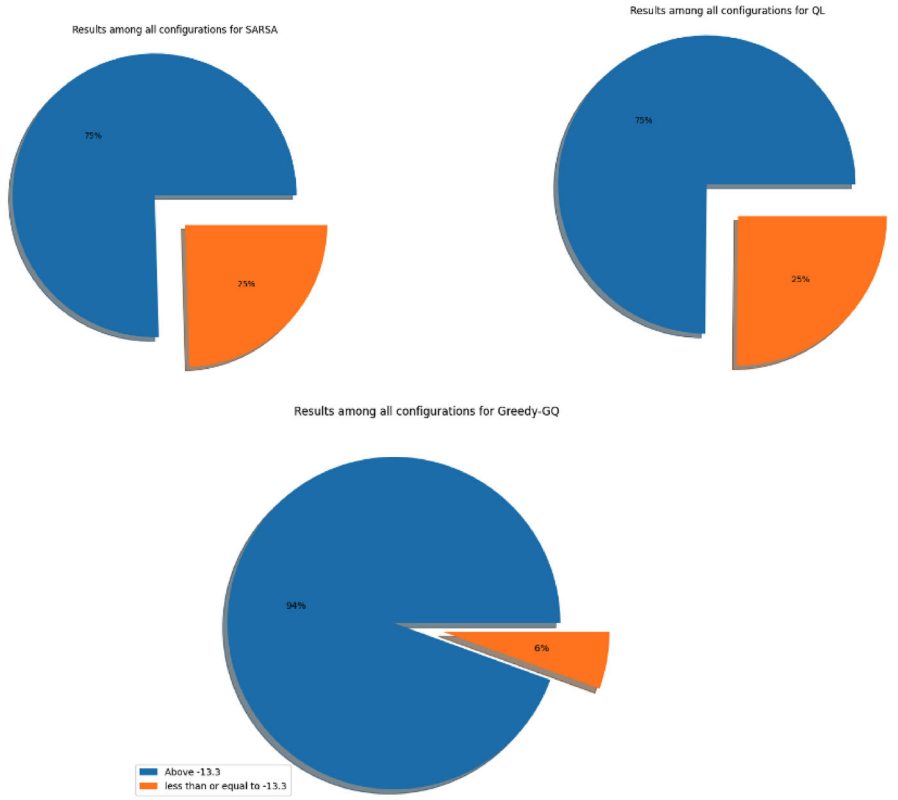

Fig. 10 Pie plots illustrate the percentage of all configurations for each RL method (Type 3) outperformed the benchmark. The configurations are run for the two-stock portfolio of BALL and APA over the period from August 1, 2007, to November 1, 2008. The top left graph represents the results for SARSA, and the top right one shows the same for the QL method. The bottom graph demonstrates the results of the Greedy-GQ method

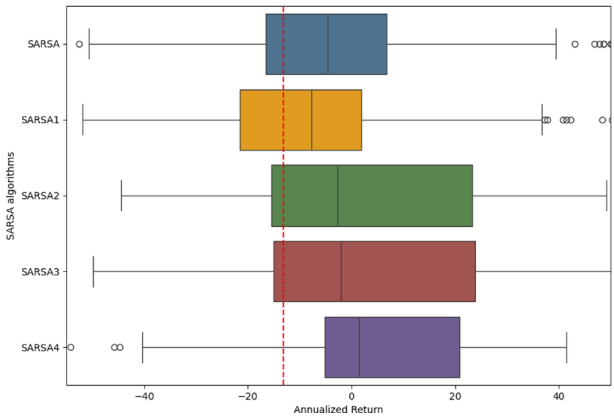

Fig. 11 Performance of the SARSA method under higher transaction costs during the 2007–2008 financial crisis using different state vectors

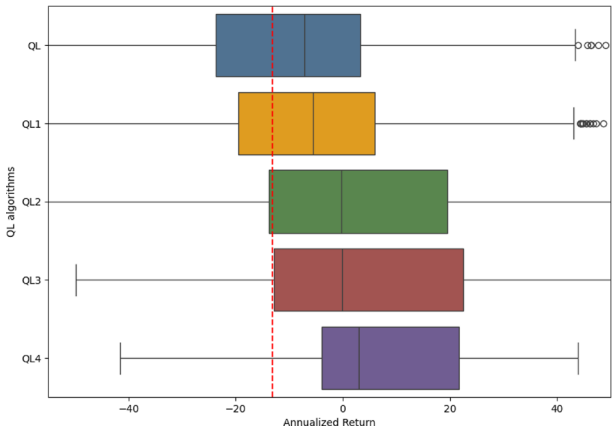

Fig. 12 Performance of the Q-Learning method under higher transaction costs during the 2007–2008 financial crisis using different state vectors

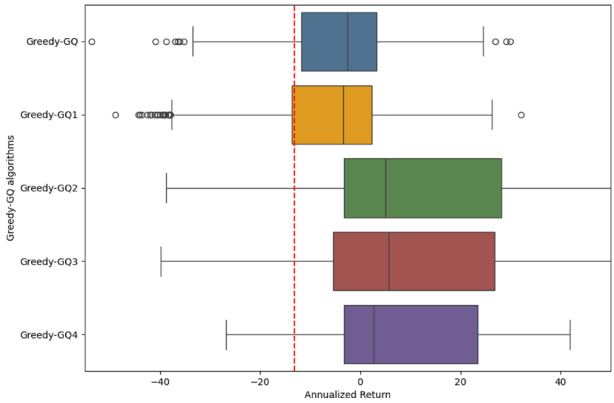

Fig. 13 Performance of the Greedy-GQ method under higher transaction costs during the 2007–2008 financial crisis using different state vectors

to Type 4. This supports what the paper said before and makes the conclusion stronger that informative state vectors make the agent much better at learning, especially during volatile conditions.

Although the higher transaction costs meant that absolute returns were slightly lower than what was reported under normal cost assumptions, the relative ranking of methods and the dominance over the benchmark remained unchanged. These results demonstrate that the reinforcement learning algorithms, particularly when combined with more informative state vectors, are still strong and flexible even in tougher trading conditions, like financial crises with higher transaction costs.(See Fig. 14)

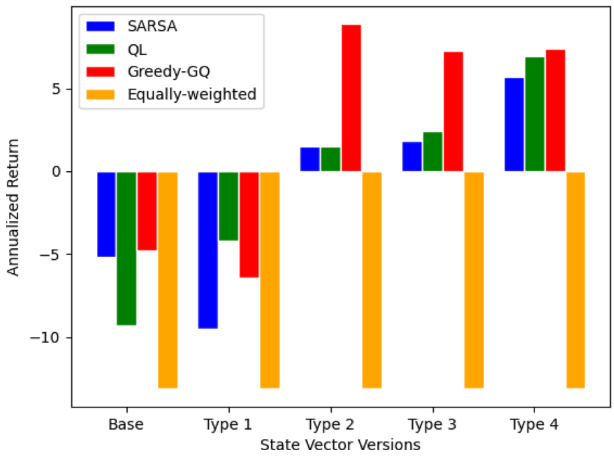

Fig. 14 Comparison of all RL methods (SARSA, QL, Greedy-GQ) using different state vectors under higher transaction costs during the 2007–2008 financial crisis

Greedy-GQ was always the strongest and most stable of the three reinforcement learning algorithms we examined, even when transaction costs were higher during the financial crisis. Greedy-GQ not only did better than the benchmark strategy across all state vector types, but it also had a smaller range of performance, showing that it was less affected by changes in transaction costs. It was able to give competitive returns with less volatility in outcomes because it converged well under function approximation and made good use of richer state information. These results support the idea that Greedy-GQ is a good and flexible way to manage a portfolio that changes over time, especially in markets that are very volatile and illiquid market environments.

To provide a more comprehensive evaluation of the robustness of the reinforcement learning methods under increased trading frictions, we extended the high transaction cost analysis to four additional stock pairs: (“NEM”, “JBL”), (“WMT”, “F”), (“TM”, “AAPL”), and (“VZ”, “NVDA”). The results, shown in Appendix 6.3, show that all three RL algorithms did better than the equally weighted portfolio strategy, even when transaction costs were high. Greedy-GQ was the most stable and consistently profitable method for most stock pairs. SARSA, on the other hand, did well in some configurations. In these tough market conditions, adding more market information, like technical indicators and beta indices, continued to improve performance. These results show that RL-based trading strategies are even stronger and more flexible, particularly in periods characterized by high volatility and low liquidity.

5 Conclusion

This research examined the potential of three RL methods, SARSA, QL, and Greedy-GQ, to manage a portfolio of two stocks. One of the stocks stands for a high-volatility stock, and the other one serves as a low-volatility stock. The methods were evaluated on five pairs of NYSE stocks over two periods (from October 2021 to October 2022 and from August 2007 to November 2008 as a robustness test). For each configuration, 1000 simulations were run, and two different techniques were considered for selecting one action from the 1000 simulations. A fixed transaction cost of 2.28 cents for each transaction was also incorporated to make it more similar to real trading conditions.

The results showed the potential of the RL methods to generate returns higher than the benchmark and to make optimal decisions regarding when to trade and which actions to take at each time step. On average, all three RL methods outperformed the benchmark across both periods (in most experiments), reinforcing their effectiveness in dynamic portfolio management. Although Greedy-GQ exhibited greater performance in the majority of instances, its efficacy fluctuated among various stock combinations. Particular market situations favored SARSA or QL, emphasizing the significance of algorithm selection based on asset characteristics and market dynamics. In addition, three different reward functions were evaluated. During period 1 (October 2021 to October 2022), which exhibited lower volatility than period 2 (the financial crisis period), the Sharpe ratio function outperformed the other two reward functions. During the financial crisis, the penalized reward function produced superior performance compared to the other reward functions.

The results show that the RL algorithms do not work the same way on all pairs of stocks. Some pairs showed bigger improvements with SARSA or QL, probably because of differences in how volatile they are and how well they use information. These results show that portfolio management strategies based on RLs should be adapted to suit the market conditions instead of using the same method for all markets. SARSA and QL both got a lot better with more market information. Greedy-GQ, on the other hand, did not get much better with new state features, even though it was the best overall. This could be because it was more complicated and needed more time to converge.

In the robustness check, where the methods were tested over the financial crisis period, all methods outperformed the benchmark on average. The RL methods represented their ability to exceed the benchmark even in the period of financial crisis (highly volatile market conditions). During this turbulent period, all three methods did better than the benchmark on average. Greedy-GQ was the most stable and profitable of the three. Also, the penalized reward function, which includes a volatility penalty, was found to be the best during the crisis. This suggests that the market conditions should be taken into account when choosing the reward function.

The additional robustness analysis with higher transaction costs also showed that reinforcement learning methods can be adapted to different situations. The Greedy-GQ algorithm, in particular, kept doing better than the benchmark, even in tougher trading conditions. Adding more detailed state vectors, especially technical indicators, continued to make the models perform better in different market conditions. This confirmed that adding more market information to the decision-making process is beneficial.

For future research, it would be valuable to extend the analysis to emerging markets, since their market structures and trading frictions are different from those of developed markets. This would help us learn more about how adaptable reinforcement learning methods are. More tests with different hyperparameter settings could help prove certain hypotheses, like that the penalized function is a better reward function when the market is volatile, and the Sharpe ratio function is a better choice when the market is less volatile. Testing over longer and more varied time horizons, as well as incorporating additional pairs of stocks, would provide further insight into the consistency of the results. One potential improvement to the methods is the use of other functions as a basis function. Expanding the action space is another way to improve the performance of the models.

6 Appendix: Additional results

6.1 Period October 2021 – October 2022

This subsection presents the performance of the reinforcement learning methods SARSA, Q-Learning, and Greedy-GQ during the market period from October 2021 to October 2022. The methods are applied to four stock pairs, each composed of one high-volatility and one low-volatility stock. Performance is compared against the equally weighted portfolio benchmark.

Stock Pair: TM - AAPL

See Figs. 15,16,17,18.

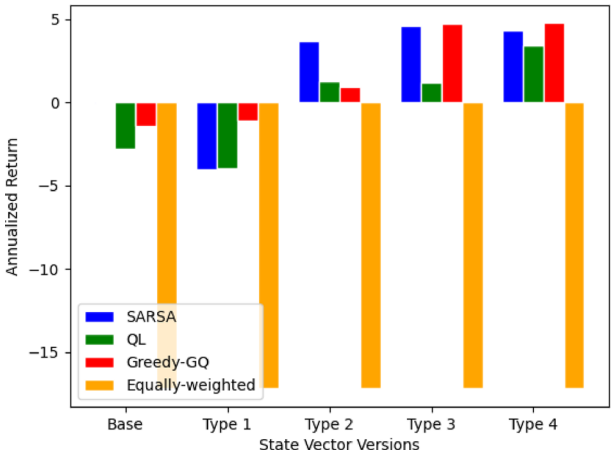

Fig. 15 Comparison of SARSA, Q-Learning, and Greedy-GQ methods (with different state vectors) for the TM and AAPL stock pair during the normal period (October 1, 2021 – October 1, 2022)

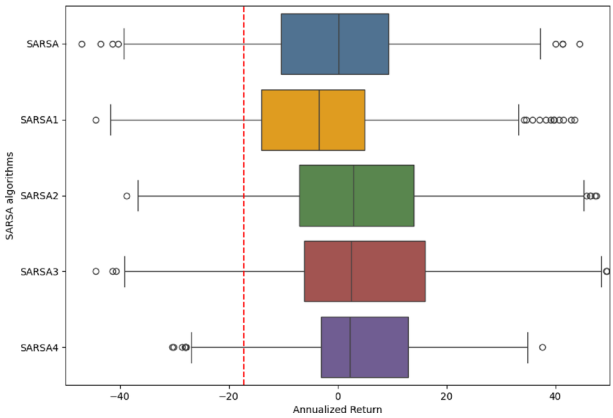

Fig. 16 Performance of the SARSA method (with different state vectors) for the TM and AAPL stock pair during the normal period (October 1, 2021 – October 1, 2022). The red line indicates the benchmark annualized return

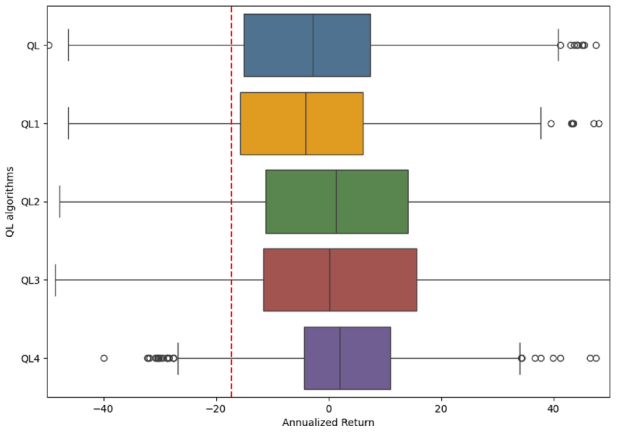

Fig. 17 Performance of the Q-Learning method (with different state vectors) for the TM and AAPL stock pair during the normal period (October 1, 2021 – October 1, 2022). The red line indicates the benchmark annualized return

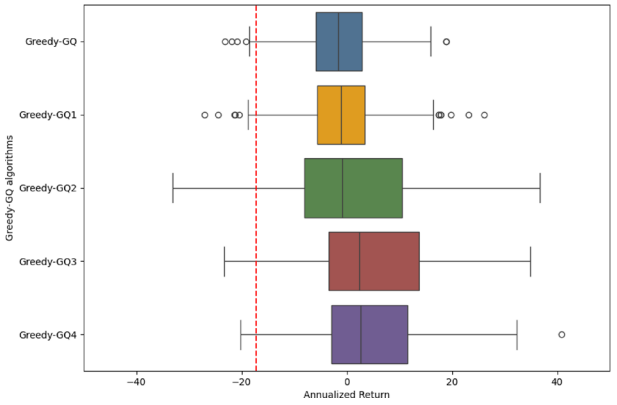

Fig. 18 Performance of the Greedy-GQ method (with different state vectors) for the TM and AAPL stock pair during the normal period (October 1, 2021 – October 1, 2022). The red line indicates the benchmark annualized return

Stock Pair: NEM - JBL

See Figs. 19,20,21,22.

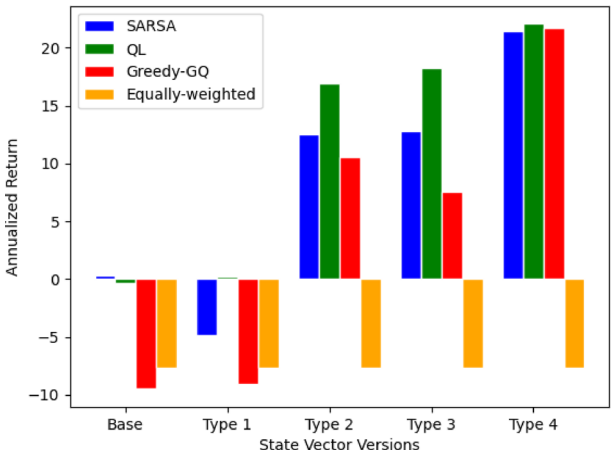

Fig. 19 Comparison of SARSA, Q-Learning, and Greedy-GQ methods (with different state vectors) for the NEM and JBL stock pair during the normal period (October 1, 2021 – October 1, 2022)

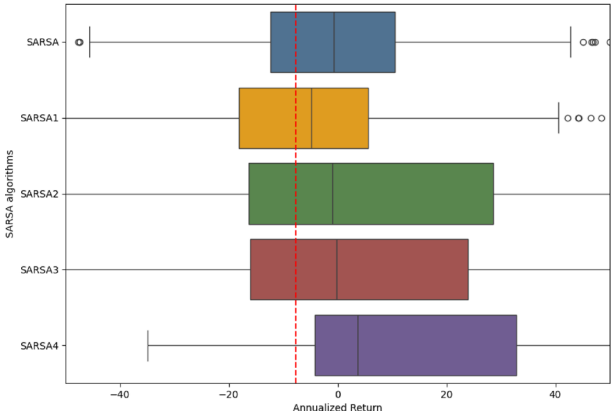

Fig. 20 Performance of the SARSA method (with different state vectors) for the NEM and JBL stock pair during the normal period (October 1, 2021 – October 1, 2022). The red line indicates the benchmark annualized return

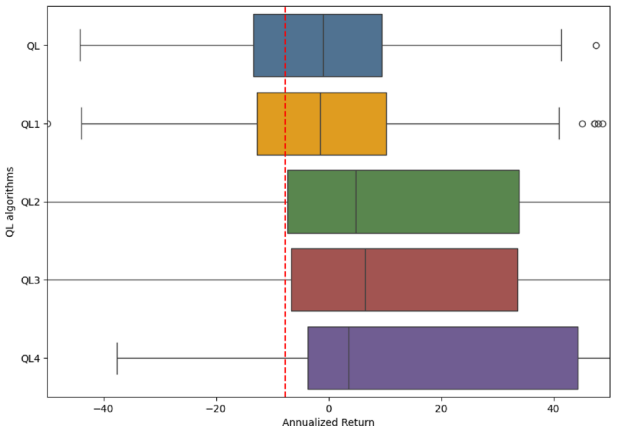

Fig. 21 Performance of the Q-Learning method (with different state vectors) for the NEM and JBL stock pair during the normal period (October 1, 2021 – October 1, 2022). The red line indicates the benchmark annualized return

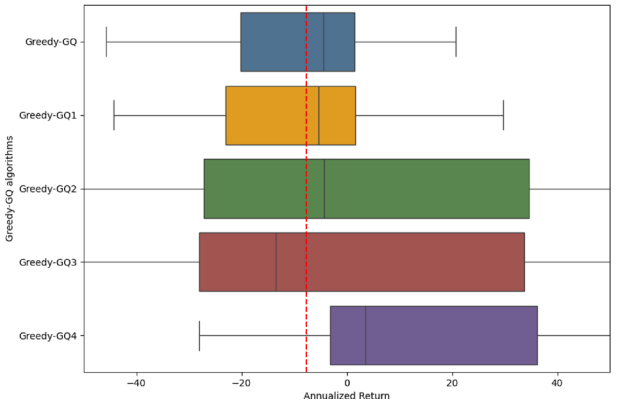

Fig. 22 Performance of the Greedy-GQ method (with different state vectors) for the NEM and JBL stock pair during the normal period (October 1, 2021 – October 1, 2022). The red line indicates the benchmark annualized return

Stock Pair: WMT - F

See Figs. 23,24,25,26.

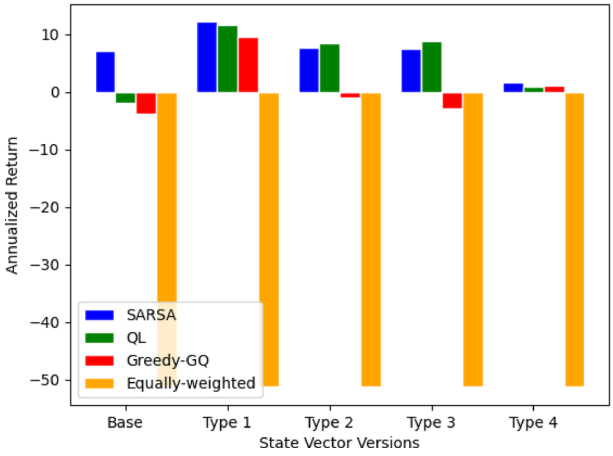

Fig. 23 Comparison of SARSA, Q-Learning, and Greedy-GQ methods (with different state vectors) for the WMT and F stock pair during the normal period (October 1, 2021 – October 1, 2022)

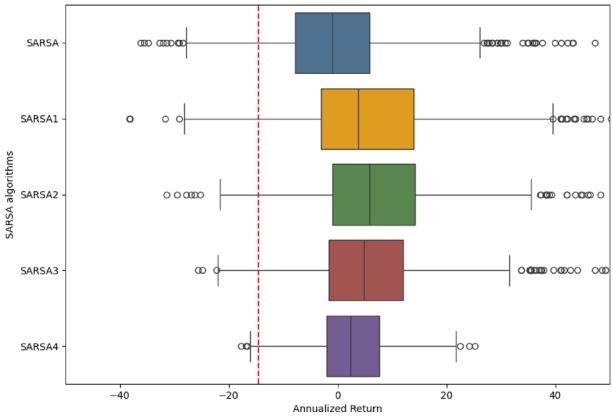

Fig. 24 Performance of the SARSA method (with different state vectors) for the WMT and F stock pair during the normal period (October 1, 2021 – October 1, 2022). The red line indicates the benchmark annualized return

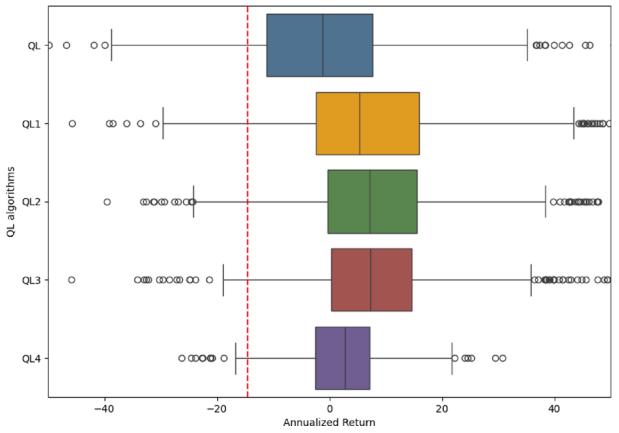

Fig. 25 Performance of the Q-Learning method (with different state vectors) for the WMT and F stock pair during the normal period (October 1, 2021 – October 1, 2022). The red line indicates the benchmark annualized return

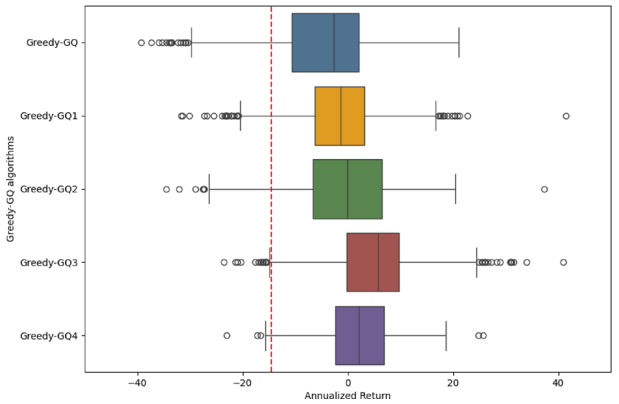

Fig. 26 Performance of the Greedy-GQ method (with different state vectors) for the WMT and F stock pair during the normal period (October 1, 2021 – October 1, 2022). The red line indicates the benchmark annualized return

Stock Pair: VZ - NVDA

See Figs. 27,28,29,30.

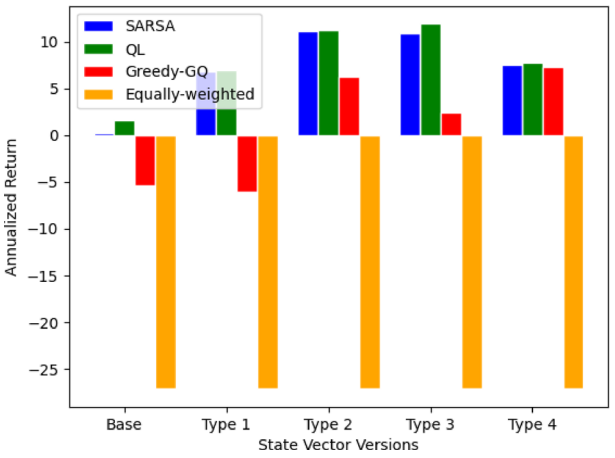

Fig. 27 Comparison of SARSA, Q-Learning, and Greedy-GQ methods (with different state vectors) for the VZ and NVDA stock pair during the normal period (October 1, 2021 – October 1, 2022)

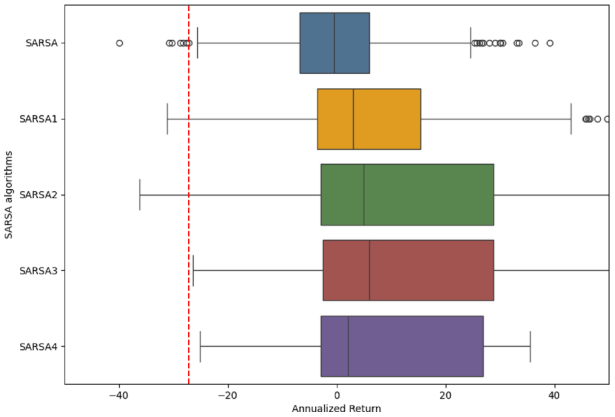

Fig. 28 Performance of the SARSA method (with different state vectors) for the VZ and NVDA stock pair during the normal period (October 1, 2021 – October 1, 2022). The red line indicates the benchmark annualized return

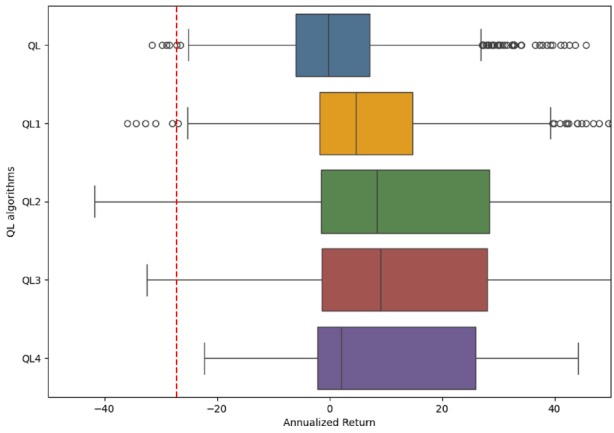

Fig. 29 Performance of the Q-Learning method (with different state vectors) for the VZ and NVDA stock pair during the normal period (October 1, 2021 – October 1, 2022). The red line indicates the benchmark annualized return

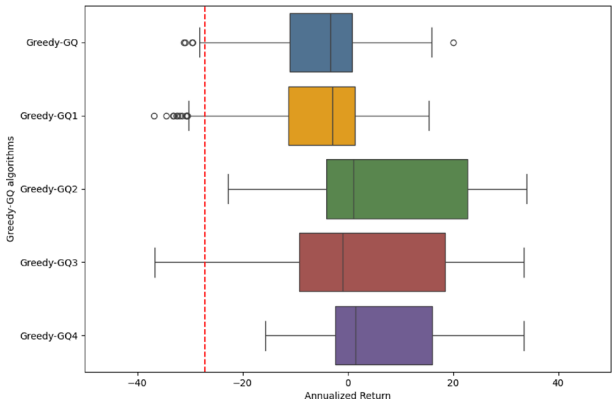

Fig. 30 Performance of the Greedy-GQ method (with different state vectors) for the VZ and NVDA stock pair during the normal period (October 1, 2021 – October 1, 2022). The red line indicates the benchmark annualized return

6.2 Financial Crisis Period (August 2007 – November 2008)

In this subsection, we report the results during the financial crisis period with a transaction cost of 2.28 cents per trade. The performance of the RL methods is evaluated on the same set of stock pairs, offering insights into their adaptability during periods of high market volatility.

Stock Pair: TM - AAPL

See Figs. 31,32,33,34.

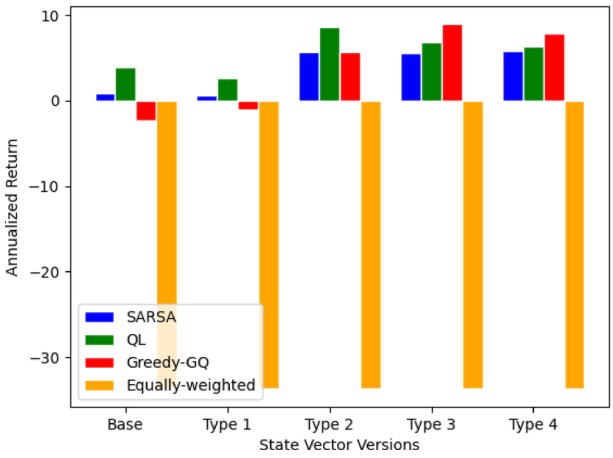

Fig. 31 Comparison of SARSA, Q-Learning, and Greedy-GQ methods (with different state vectors) for the TM and AAPL stock pair during the financial crisis period (August 1, 2007 – November 1, 2008)

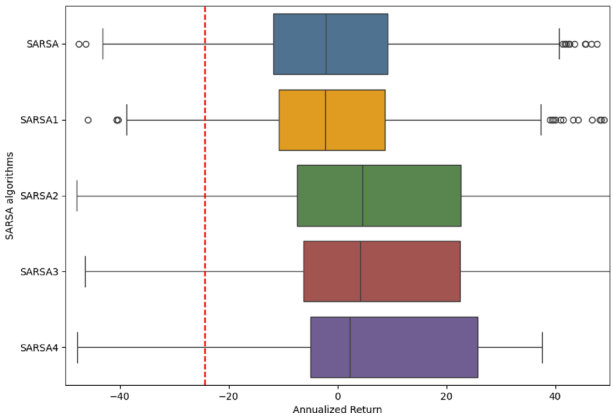

Fig. 32 Performance of the SARSA method (with different state vectors) for the TM and AAPL stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

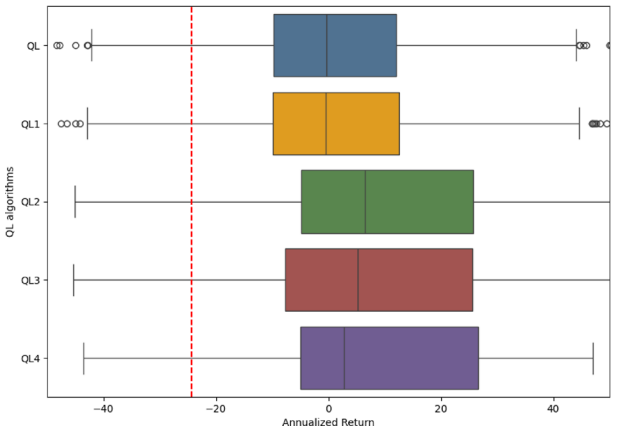

Fig. 33 Performance of the Q-Learning method (with different state vectors) for the TM and AAPL stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

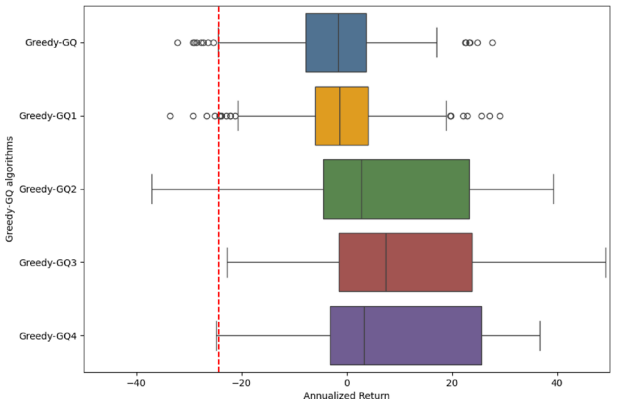

Fig. 34 Performance of the Greedy-GQ method (with different state vectors) for the TM and AAPL stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

Stock Pair: NEM - JBL

See Figs. 35,36,37,38.

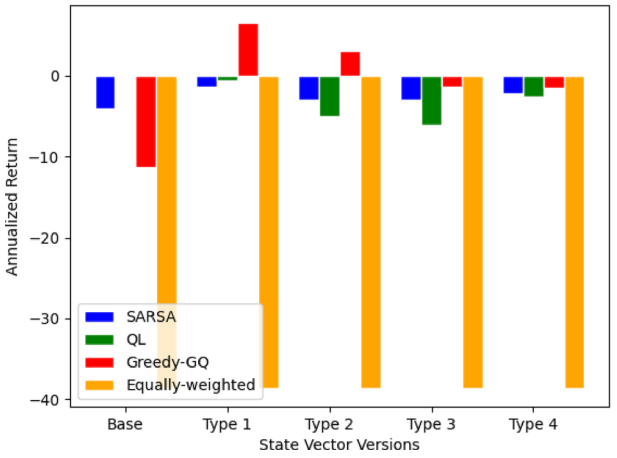

Fig. 35 Comparison of SARSA, Q-Learning, and Greedy-GQ methods (with different state vectors) for the NEM and JBL stock pair during the financial crisis period (August 1, 2007 – November 1, 2008)

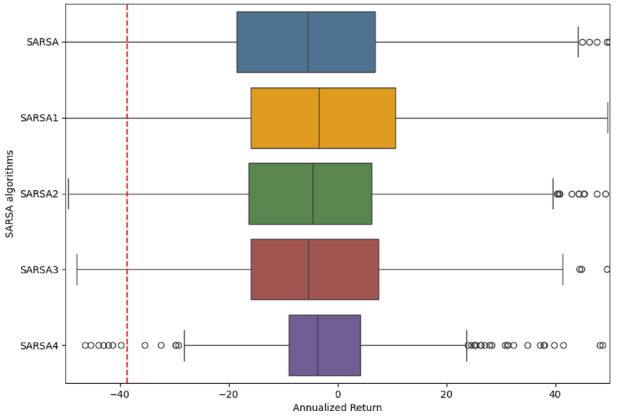

Fig. 36 Performance of the SARSA method (with different state vectors) for the NEM and JBL stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

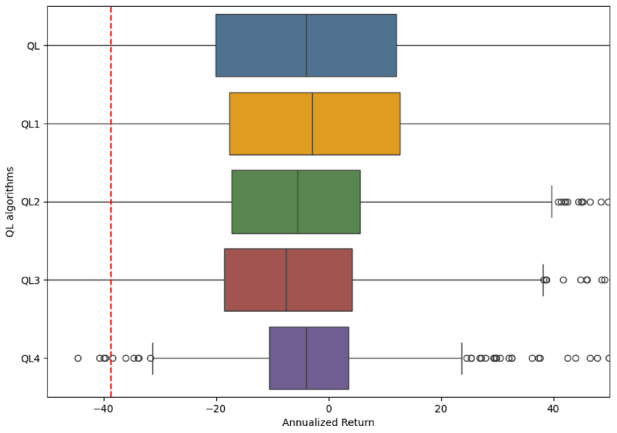

Fig. 37 Performance of the Q-Learning method (with different state vectors) for the NEM and JBL stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

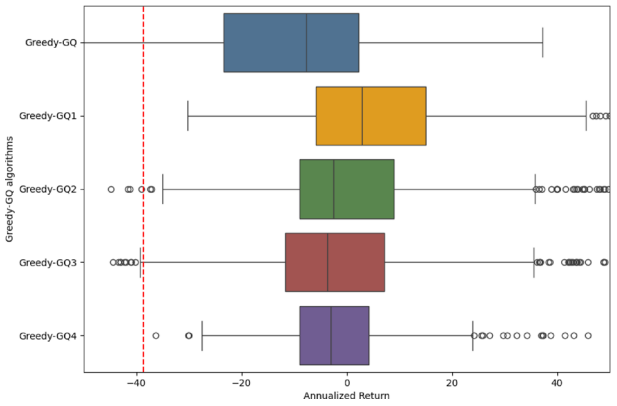

Fig. 38 Performance of the Greedy-GQ method (with different state vectors) for the NEM and JBL stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

Stock Pair: WMT - F

See Figs. 39,40,41,42.

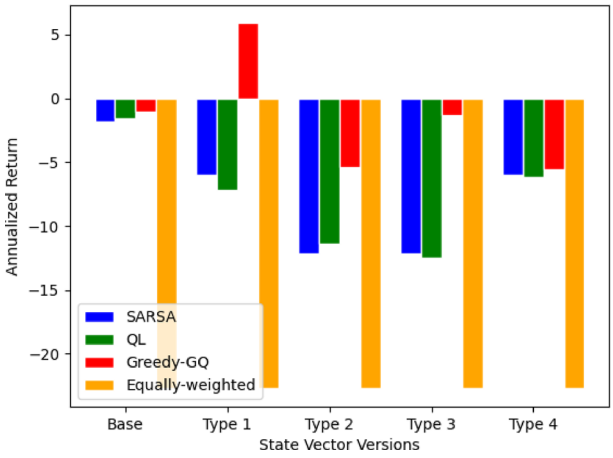

Fig. 39 Comparison of SARSA, Q-Learning, and Greedy-GQ methods (with different state vectors) for the WMT and F stock pair during the financial crisis period (August 1, 2007 – November 1, 2008)

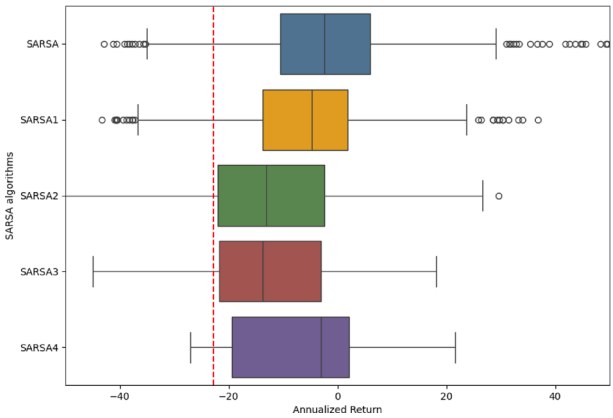

Fig. 40 Performance of the SARSA method (with different state vectors) for the WMT and F stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

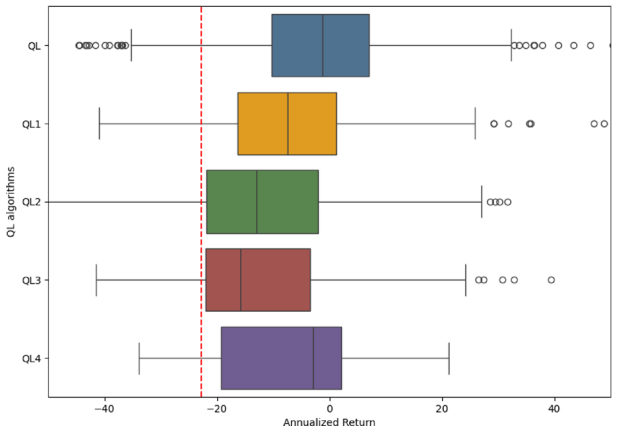

Fig. 41 Performance of the Q-Learning method (with different state vectors) for the WMT and F stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

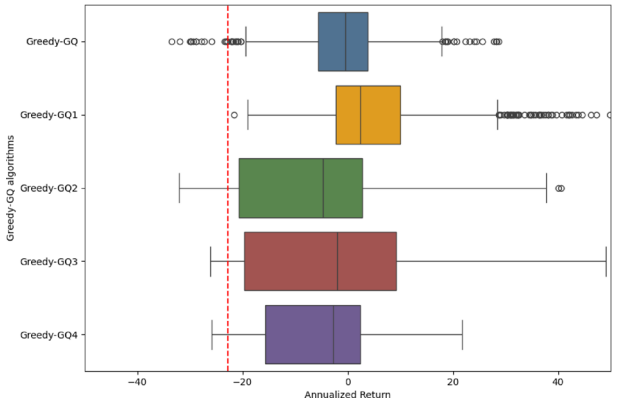

Fig. 42 Performance of the Greedy-GQ method (with different state vectors) for the WMT and F stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

Stock Pair: VZ - NVDA

See Figs. 43,44,45,46.

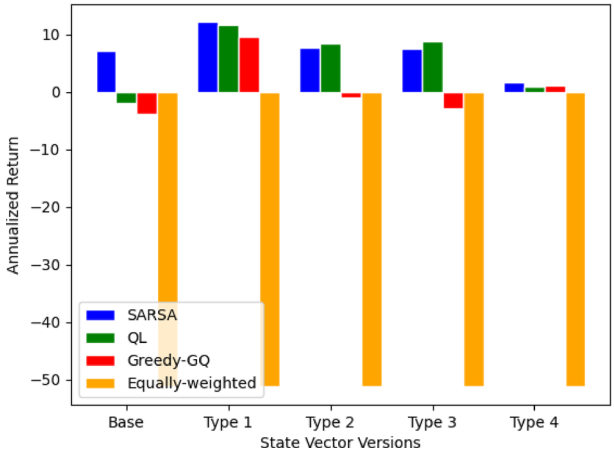

Fig. 43 Comparison of SARSA, Q-Learning, and Greedy-GQ methods (with different state vectors) for the VZ and NVDA stock pair during the financial crisis period (August 1, 2007 – November 1, 2008)

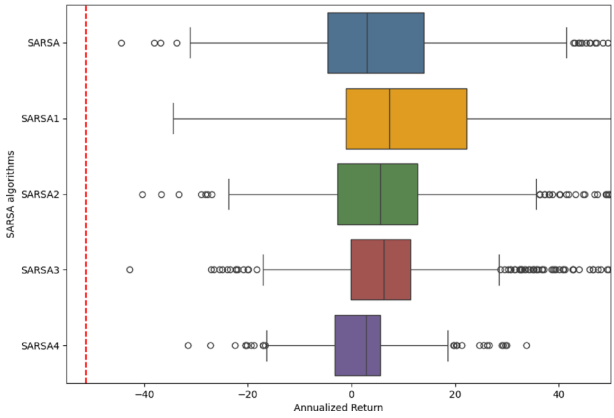

Fig. 44 Performance of the SARSA method (with different state vectors) for the VZ and NVDA stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

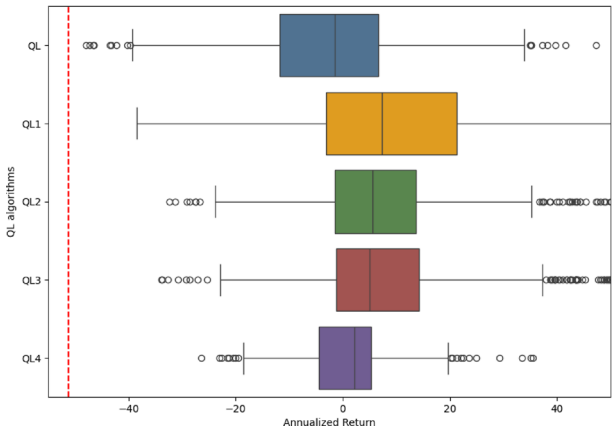

Fig. 45 Performance of the Q-Learning method (with different state vectors) for the VZ and NVDA stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

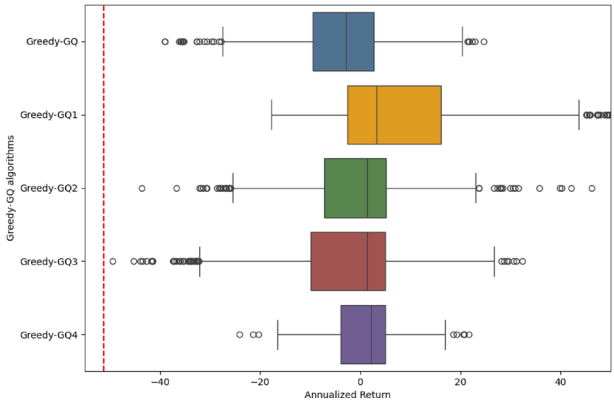

Fig. 46 Performance of the Greedy-GQ method (with different state vectors) for the VZ and NVDA stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

6.3 Financial Crisis Period (August 2007 – November 2008) – Higher Transaction Cost

This subsection introduces additional robustness checks by increasing the transaction cost to 4.56 cents per trade, representing harsher trading environments typical of financial crises. The analysis is extended to five stock pairs, including BALL and APA, to assess the resilience of the RL agents under elevated trading frictions.

Stock Pair: TM - AAPL

See Figs. 47,48,49,50.

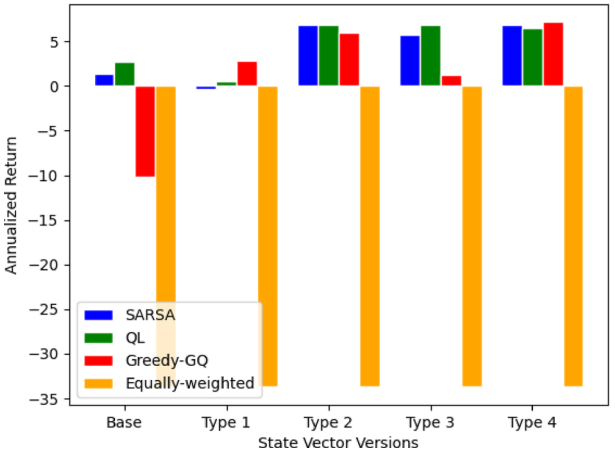

Fig. 47 Comparison of SARSA, Q-Learning, and Greedy-GQ methods (with different state vectors) for the TM and AAPL stock pair during the financial crisis period (August 1, 2007 – November 1, 2008) under higher transaction costs

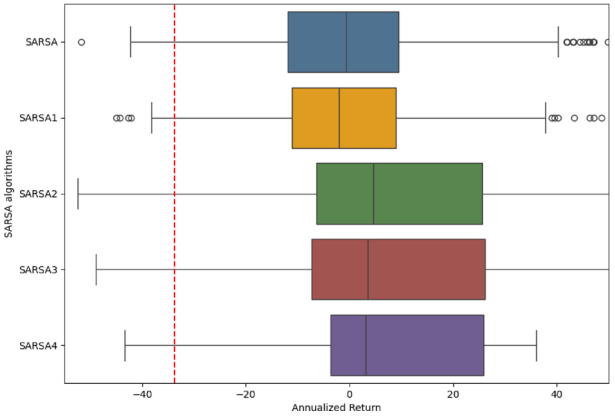

Fig. 48 Performance of the SARSA method (with different state vectors) for the TM and AAPL stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

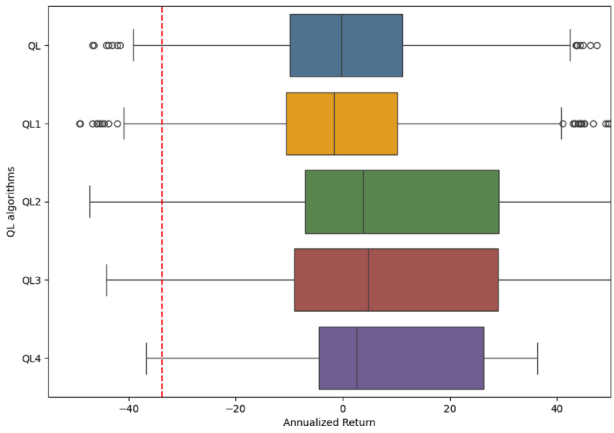

Fig. 49 Performance of the Q-Learning method (with different state vectors) for the TM and AAPL stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

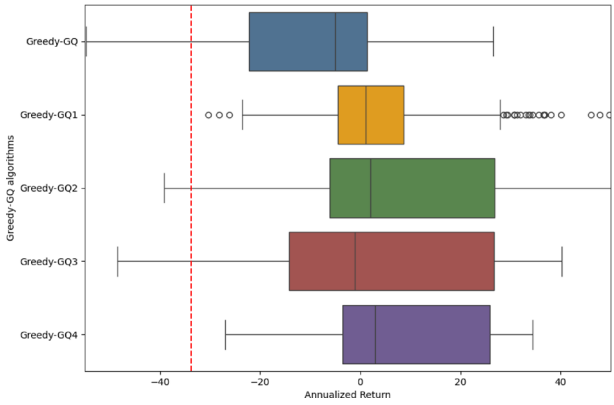

Fig. 50 Performance of the Greedy-GQ method (with different state vectors) for the TM and AAPL stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

Stock Pair: NEM - JBL

See Figs. 51,52,53,54.

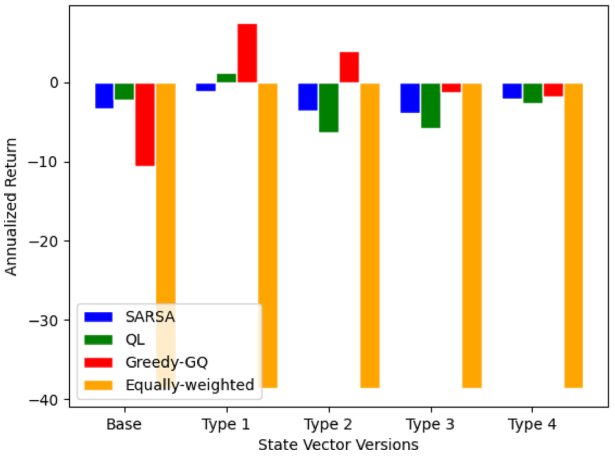

Fig. 51 Comparison of SARSA, Q-Learning, and Greedy-GQ methods (with different state vectors) for the NEM and JBL stock pair during the financial crisis period (August 1, 2007 – November 1, 2008) under higher transaction costs

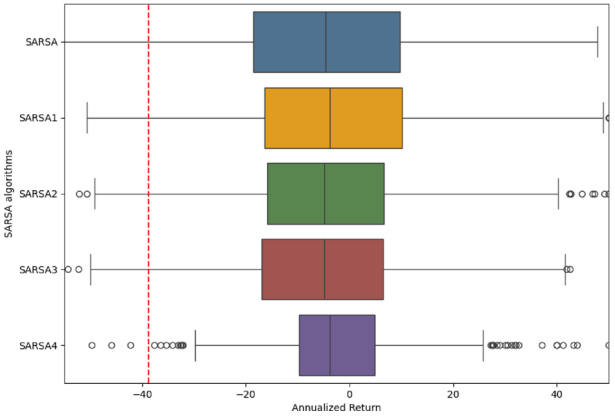

Fig. 52 Performance of the SARSA method (with different state vectors) for the NEM and JBL stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

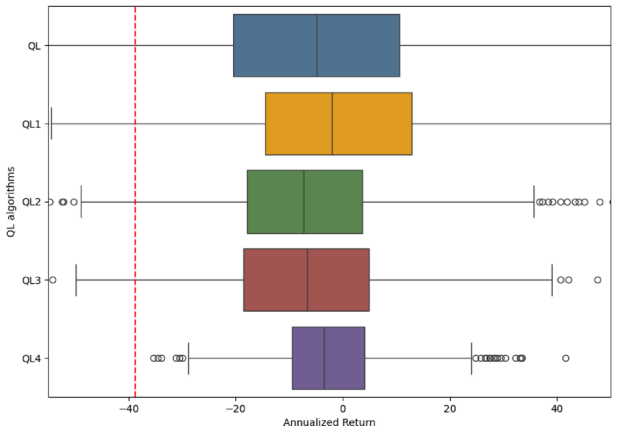

Fig. 53 Performance of the Q-Learning method (with different state vectors) for the NEM and JBL stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

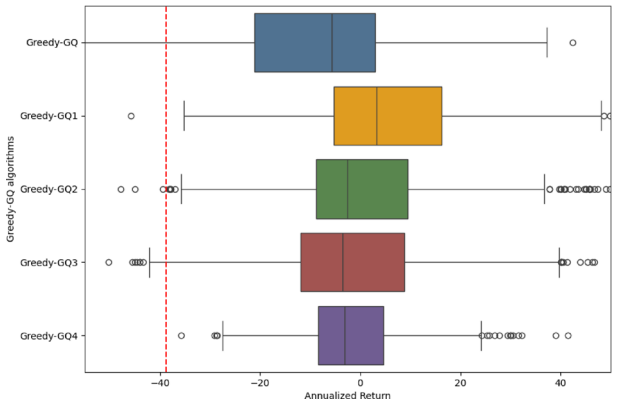

Fig. 54 Performance of the Greedy-GQ method (with different state vectors) for the NEM and JBL stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

Stock Pair: WMT - F

See Figs. 55,56,57,58.

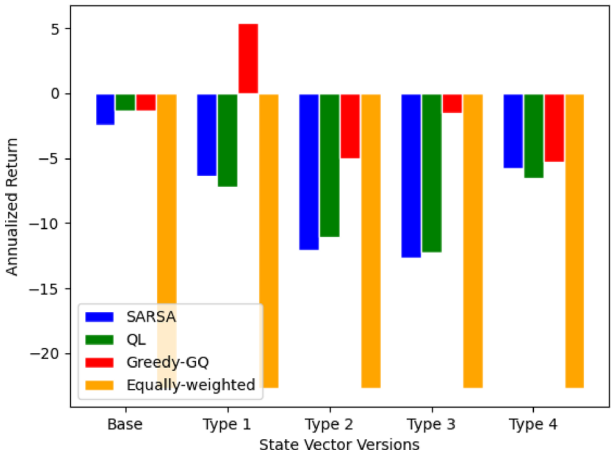

Fig. 55 Comparison of SARSA, Q-Learning, and Greedy-GQ methods (with different state vectors) for the WMT and F stock pair during the financial crisis period (August 1, 2007 – November 1, 2008) under higher transaction costs

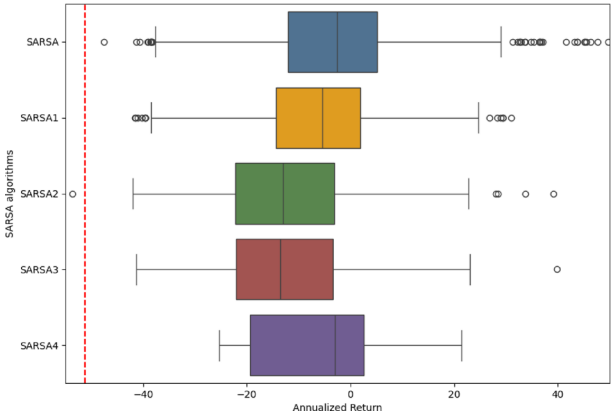

Fig. 56 Performance of the SARSA method (with different state vectors) for the WMT and F stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

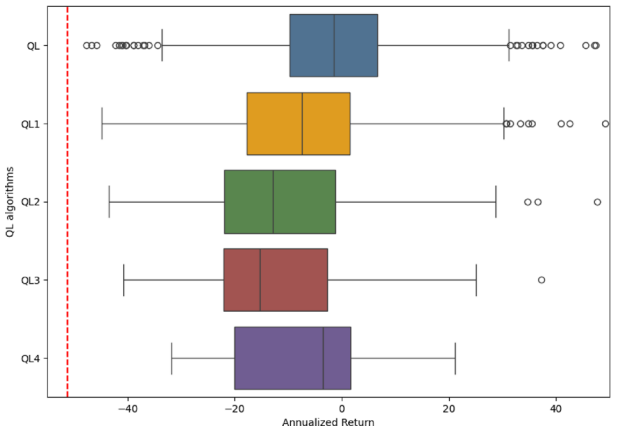

Fig. 57 Performance of the Q-Learning method (with different state vectors) for the WMT and F stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

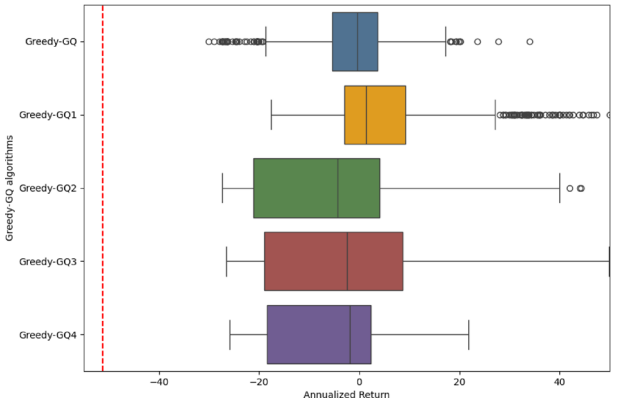

Fig. 58 Performance of the Greedy-GQ method (with different state vectors) for the WMT and F stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

Stock Pair: VZ - NVDA

See Figs. 59,60,61,62.

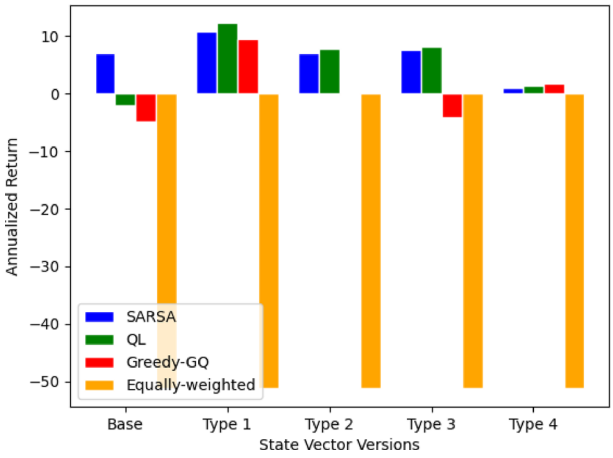

Fig. 59 Comparison of SARSA, Q-Learning, and Greedy-GQ methods (with different state vectors) for the VZ and NVDA stock pair during the financial crisis period (August 1, 2007 – November 1, 2008) under higher transaction costs

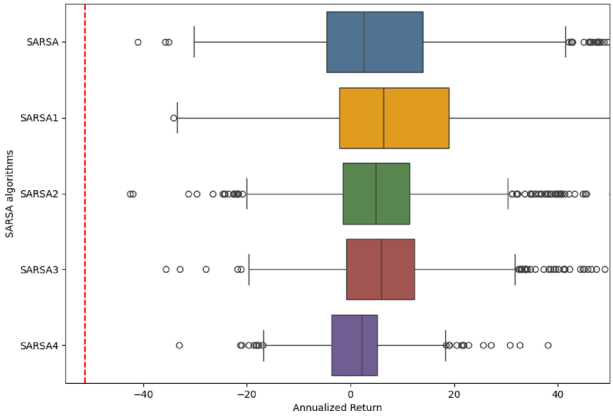

Fig. 60 Performance of the SARSA method (with different state vectors) for the VZ and NVDA stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

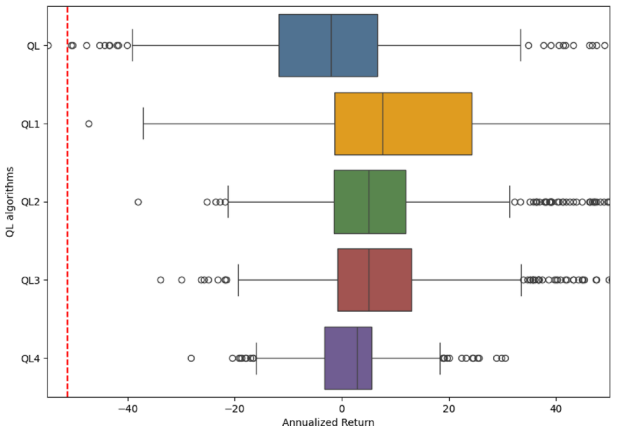

Fig. 61 Performance of the Q-Learning method (with different state vectors) for the VZ and NVDA stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

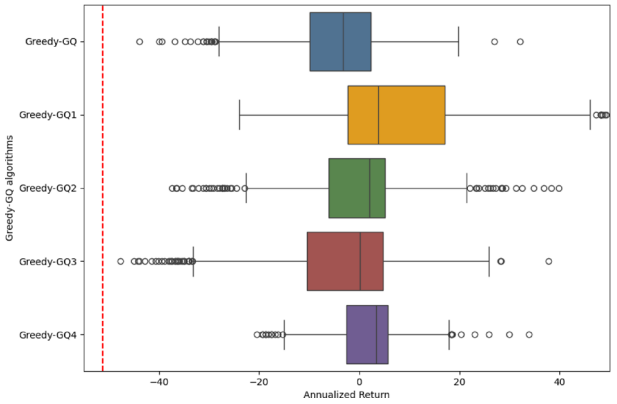

Fig. 62 Performance of the Greedy-GQ method (with different state vectors) for the VZ and NVDA stock pair during the financial crisis period (August 1, 2007 – November 1, 2008). The red line indicates the benchmark annualized return

Funding No funding was received for conducting this study.

Declarations

Competing Interests The authors have no competing interests to declare that are relevant to the content of this article.

Ethical approval Not applicable.

References

Agarwal, A., Hazan, E., Kale, S., and Schapire, R.E.: Algorithms for portfolio management based on the newton method. In Proceedings of the 23rd international conference on Machine learning, 9–16 (2006)

Almahdi, S., Yang, S.Y.: An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown. Expert Syst. Appl. **87**, 267–279 (2017)

Barnard, E.: Temporal-difference methods and markov models. IEEE Trans. Syst. Man Cybern. **23**(2), 357–365 (1993)

Bellman, R.: On the theory of dynamic programming. Proc. Natl. Acad. Sci. **38**(8), 716–719 (1952)

Bellman, R.: A markovian decision process. J. Math. Mech. **6**(5), 679–684 (1957)

Bellman, R.: Dynamic Programming. Dover Books on Computer Science Series, Dover Publications (2003)

Borodin, A., El-Yaniv, R., Gogan, V.: Can we learn to beat the best stock. Advances in Neural Information Processing Systems, **16**, (2003)

Chiang, J.K., Chi, R.: A novel stock price prediction and trading methodology based on active learning surrogated with cyclegan and deep learning and system engineering integration: A case study on tsmc stock data. FinTech **3**(3), 427–459 (2024)

Chinas, M.: Efficient Market Hypothesis: Weak Form Efficiency: An Examination of Weak Form Efficiency. Amazon Digital Services LLC - KDP Print US (2019)

Corazza, M., Fasano, G., Gusso, R., Raffaele, P.: A comparison among reinforcement learning algorithms in financial trading systems. University Ca' Foscari of Venice, Dept. of Economics Research Paper Series No. No. 33/WP/2019 (2019)

Cover, T.M.: Universal portfolios. Math. Finan. **1**(1), 1–29 (1991)

Das, P., Banerjee, A.: Meta optimization and its application to portfolio selection. In Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, 1163–1171 (2011)

Deese, J.: The psychology of learning (1952)

Du, X., Zhai, J., Lv, K.: Algorithm trading using q-learning and recurrent reinforcement learning. Positions **1**(1), 1–7 (2016)

Fama, E.F.: Efficient capital markets: A review of theory and empirical work. J. Financ. **25**(2), 383–417 (1970)

Gaivoronski, A.A., Stella, F.: Stochastic nonstationary optimization for finding universal portfolios. Ann. Oper. Res. **100**, 165–188 (2000)

Gao, X., Chan, L.: An algorithm for trading and portfolio management using q-learning and sharpe ratio maximization. 832–837 (2000)

Geramifard, A., Doshi, F., Redding, J., Roy, N., How, J.P.: Online discovery of feature dependencies. 881–888 (2011)

Giorgi, F., Herzel, S., Pigato, P.: A reinforcement learning algorithm for trading commodities. Appl. Stoch. Model. Bus. Ind. **40**(2), 373–388 (2024)

Györfi, L., Lugosi, G., Udina, F.: Nonparametric kernel-based sequential investment strategies. Math. Finan.: An Int. J. Math. Stat. Finan. Econ. **16**(2), 337–357 (2006)

Hambly, B., Xu, R., Yang, H.: Recent advances in reinforcement learning in finance. Math. Financ. **33**(3), 437–503 (2023)

Heaton, J.: An empirical analysis of feature engineering for predictive modeling. In SoutheastCon 2016, 1–6 (2016)

Howard, R.A.: Dynamic programming and markov processes (1960)

Jiang, Y., Olmo, J., Atwi, M.: Deep reinforcement learning for portfolio selection. Glob. Financ. J. **62**, 101016 (2024)

Jiang, Z., Xu, D., Liang, J.: A deep reinforcement learning framework for the financial portfolio management problem. (2017). arXiv preprint arXiv:1706.10059

Jin, O., El-Saawy, H.: Portfolio management using reinforcement learning (2016)

Kabanov, Y., Liptser, R., Stoyanov, J., Di Masi, G.B., Stettner, L.: Remarks on risk neutral and risk sensitive portfolio optimization. From Stochastic Calculus to Mathematical Finance: The Shiryaev Festschrift, 211–226 (2006)

Kelly, J.L.: A new interpretation of information rate. The bell Syst. Tech. J. **35**(4), 917–926 (1956)

Klopf, A.H.: Brain function and adaptive systems: a heterostatic theory. Number 133. Air Force Cambridge Research Laboratories, Air Force Systems Command, United .... (1972)

Klopf, A.H.: A comparison of natural and artificial intelligence. ACM SIGART Bull. **52**, 11–13 (1975)
Klopf, A.H.: The Hedonistic Neuron: A Theory of Memory, Learning, and Intelligence. Hemisphere Pub., New York (1982)
Liang, Z., Chen, H., Zhu, J., Jiang, K., Li, Y.: Adversarial deep reinforcement learning in portfolio management. (2018). arXiv preprint arXiv:1808.09940
Littman, M.L.: Algorithms for sequential decision-making. Brown University (1996)
Lo, A.W.: Adaptive Markets. Financial Evolution at the Speed of Thought. Princeton University Press, Princeton (2017)
López de Prado, M., Lewis, M.J.: Detection of false investment strategies using unsupervised learning methods. Quant. Finan. **19**(9), 1555–1565 (2019)
Maei, H., Szepesvári, C., Bhatnagar, S., Sutton, R.: Toward off-policy learning control with function approximation. ICML 2010 - Proceedings, 27th International Conference on Machine Learning, 719–726 (2010)
Markowitz, H.: Portfolio selection, J. Finan. **7**(1), (1952)
Markowitz, H.: Portfolio Selection: Efficient Diversification of Investments. Wiley, New york (1959)
Markowitz, H.M., Todd, G.P.: Mean-Variance Analysis in Portfolio Choice and Capital Markets, vol. 66. John Wiley & Sons, Hoboken (2000)
Minsky, M.: Steps toward artificial intelligence. Proc. IRE **49**(1), 8–30 (1961)
Mishra, S.: Artificial intelligence: A review of progress and prospects in medicine and healthcare. J. Electron. Electromed. Eng. Med. Inform. **4**(1), 1–23 (2022)
Moody, J., Saffell, M., Liao, Y., Wu, L.: Reinforcement learning for trading systems and portfolios. In KDD, 279–283 (1998)
Moody, J., Wu, L., Liao, Y., Saffell, M.: Performance functions and reinforcement learning for trading systems and portfolios. J. Forecast. **17**(5–6), 441–470 (1998a)
Moody, J., Wu, L., Liao, Y., Saffell, M.: Performance functions and reinforcement learning for trading systems and portfolios. J. Forecast. **17**(5–6), 441–470 (1998b)
Ni, J., Chen, Y., Chen, Y., Zhu, J., Ali, D., Cao, W.: A survey on theories and applications for self-driving cars based on deep learning methods. Appl. Sci. **10**(8), 2749 (2020)
Palit, D., Prybutok, V.R.: A comparative study of the equal-weight method and hierarchical risk parity in portfolio construction. Finan. Econ. Rev. **6**(1), 1–12 (2024)
Patani, J., Nair, S., Mehta, K., Sankhe, A., Kanani, P., Sanghvi, D.: Financial portfolio management using reinforcement learning. Int. J. Adv. Sci. Technol. **29**, 9740–9751 (2020)
Pendharkar, P., Cusatis, P.: Trading financial indices with reinforcement learning agents. Expert Systems with Applications **103**, 1–13 (2018). Publisher Copyright: © 2018 Elsevier Ltd
Rummery, G.A., Niranjan, M.: On-line Q-learning using connectionist systems, vol. 37. University of Cambridge, Department of Engineering Cambridge, UK (1994)
Sharpe, W.F.: Capital asset prices: A theory of market equilibrium under conditions of risk. J. Financ. **19**(3), 425–442 (1964)
Srivastava, S. et al.: Deep reinforcement learning for asset allocation in us equities. (2020). arXiv preprint arXiv:2010.04404
Subramanian, K.: Efficient market hypothesis: The model that failed. Econ. Pol. Wkly **45**(31), 20–22 (2010)
Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction, 2nd edn. The MIT Press, Cambridge (2018)
Swade, A., Shackleton, M.B., Lohre, H., et al.: Why do equally weighted portfolios beat value-weighted ones? J. Portf. Manag. **49**(5), 167–187 (2022)
Tian, Y., Gao, M., Gao, Q., Peng, X.-H.: Trading in fast-changing markets with meta-reinforcement learning. Intell. Autom. Soft Comput. **39**(2), 175–188 (2024)
Vovk, V.G.: Aggregating strategies. In Annual Workshop on Computational Learning Theory: Proceedings of the third annual workshop on Computational learning theory, 1990. Association for Computing Machinery, Inc. (1990)
Watkins, C.J., Dayan, P.: Q-learning. Machine Learn. **8**(3), 279–292 (1992)
Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., et al.: Transformers: State-of-the-art natural language processing. In Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations, 38–45 (2020)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.