Are Reinforcement Learning Based Algorithms a Viable Alternative to Traditional Wealth Management Strategies?

Ștefan-Constantin RADU
Romanian Economic Studies Academy
Piata Romana, Sector 1, Bucharest, Romania
stefan.radu98@yahoo.com

Lucian Claudiu ANGHEL
National University of Political Studies and Public Administration
30a Expoziției Blvd., Sector 1, 012104 Bucharest, Romania
lucian.anghel@facultateademanagement.ro

Ioana Simona ERMIȘ
Romanian Economic Studies Academy
Piata Romana, Sector 1, Bucharest, Romania
simonaermis@yahoo.com

Abstract
One of the fundamental questions of finance and economics is stated by how should we preserve money, and in a more specific way their purchasing power. Due to the way that money markets and fiscal policy work the wealth of every one of us is eroded daily by inflation, taxes, and currency exchange rates depreciation. A solution for preventing this erosion of the purchasing power of our capital is investing, the most simple form of investment is considered to be the traditional bank deposit, but in the context of today's market, at least in the developed countries, the returns can be smaller than the rate of inflation, and if taking into account other factors such as the increase in prices and the stagnation of the real wage growth, one of the only viable solutions seem to be investing in the stock market. The present article answers the question of wealth management by implementing the new ideas of reinforcement learning to solve the problem. In recent years reinforcement learning has become one of the most important fields of study in science, its use has been seen in almost all academic fields with many applications marking a paradigm shift from the classic methods used. The enrichment brought to science by this field of knowledge has not escaped the grasp of economists, and the reinforcement learning-based algorithms started to see use in the solving of non-trivial problems. One such problem is the automatic management of an investment account by selling and buying stocks, this type of solution is implemented by developing a trading algorithm that analyzes the conditions of the market and decides if selling or buying a certain stock is the most suitable action in the given time. The results of the algorithm will be compared to a benchmark set by three portfolios which are optimized by minimizing a risk ratio (i.e. Standard Deviation, Mean Absolute Error, and Semi Standard Deviation).

Keywords
Modern methods of wealth management; reinforcement learning; q-learning; stock market; investment strategies; portfolio optimization.

Introduction

In the domain of management one of the most important areas of study is that of wealth management. Taking into account that nowadays, the average level of wealth has been increasing at an extraordinary pace, one of the most discussed topics by people all over the world has become the proper and most efficient way to manage your wealth. In such a climate, where interest in this field of study is on the rise, many researchers are looking to move past the Modern Portfolio Theory and other passive investment strategies, and are starting to consider active management. This change in perception is first due to the economic conditions of today's markets and to the fact that financial deregulation brought about more investors and investment opportunities than it was originally believed.

In this article, we will present a wealth management strategy based on active investment in stocks that will be compared to active management strategies as the simple moving average and the "turtle" algorithm and to three passive investment strategies which are based on investing in a portfolio that minimizes a certain risk ratio, the ratios of the portfolios are calculated at the end of the sample period and are used as a benchmark for testing the results of the algorithms, in this way we can compare the results of the algorithms to the best possible results in the market. In this way, we will see what type of investment strategy is the most suitable for the markets and can help the investor to obtain the biggest return for the money invested. The data used for the application of the algorithms are from the NYSE and NASDAQ market, the companies that were used are: General Motors, Apple, Microsoft, Amazon, Ford, American Express, IBM, Visa, J.P. Morgan, and Home Depot, the period for which the data was collected, is from the 1st of April 2016 to the 1st of April 2021, and the data was made available by the Yahoo Finance portal.

Literature review

Since the beginning of the last decade, the interest in the application of reinforcement learning algorithms in finance has been increasing. This preoccupation with this field of study has been materialized in an array of papers, one of them, that is considered to be of great importance is „Hands-On Machine Learning for Algorithmic Trading" written by Stefan Jansen (2018), this work describes in a detailed way how an array of trading algorithms can be programmed by using Python language-based software. Another work of importance is „Python for Finance – Mastering Data-Driven Finance" by Yves Hilpsich (2018) that is written with the idea of popularizing the Python programming language for use in the economic field of study. Other works that have helped develop the program are „Python for Finance" by Yuxing Yan (2017), „Python for Finance – Analyze Big Financial Data" by Yves Hilpsich (2015) which describe in detail the way to write a functional program. The present article was written with the idea of comparing several wealth management techniques to determine if active portfolio management is better than passive portfolio management, this idea is presented through graphs and tables. The first papers that are written with the application of algorithmic trading on the stock market are the ones written by

Neuneier in 1995 and in 1997 in which a multiple asset portfolio is managed using Q-learning. In the same period, other notable studies are the one written by Moody, Wu, Liao, and Saffell (1998) and the one made by Moody and Saffell (2001), which was done by using reinforcement learning for buying and selling a share, these studies were instrumental in introducing the Sharpe and Sterling ratio to compare the strategies and the performance of the algorithms. Another notable article is the one written by Jangmin, Lee, Lee, and Zhang (2006) which develops a transaction system based on q-learning which classifies a series of prices of shares in four different categories, and applies different rules for each one of them. The article applies this strategy on the KOSPI index from South Korea for the period between 2002 and 2003 and has registered a return of 258%.

Algorithm methodology and results

In this paper, the methodology used is similar to the one in the research papers that are described in the literature review section. In this way the "turtle" algorithm was developed by Richard Dennis and William Eckhart through the project named the „Turtle Trading Experiment" in the latter half of the 20th century this is a trend following algorithm. It works in such a way that when the price of a share is smaller than the average of the last 126 days (this number was obtained by taking 10% out of the number of observations) the program buys and when it is smaller - the program sells, the algorithm can't buy or sell more than one share a day, this limit was imposed because the observed intervals should be equal.

The simple moving average trading algorithm works by calculating two moving averages for two-time intervals which are compared to figure out if a certain stock should be bought or sold. In the present case, we calculated the short window of the moving average at a period of 32 days (0.025 of the selected time interval) and the long window at a period of 63 days (0.05 of the selected interval). When the average of the short window is bigger than the average for the long window, the algorithm will buy a unit of the observed shares, when is smaller the algorithm will sell one unit of the share.

The Q-Learning process is defined by its capacity to learn iteratively by maximizing the value of a reward function that is programmed in the algorithm. The program has a training period in which it explores the function in a process that is called „epsilon-greedy", in this way the algorithm selects a random action that has an assigned probability of ε when in the opposite case the algorithm chooses the action with the probability 1-ε. In the applied algorithm we choose the value to be ε=0.5, in this way the algorithm has equal probabilities for both options (i.e. exploring or taking the route it has taken before). In this way, the algorithm estimates the Q value of an action, that corresponds to action a, in the s state according to policy π. During the training period, the agent updates the Q value for the state-action combination, in this way the Q values matrix is updated having action and a state value. These values are updated using the following function:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha * [R_{t+1} + \delta * Q(s_{t+1}, a) - Q(s_t, a_t)] \qquad (1)$$

The Q value of a pair formed out of a state and action can be considered a good estimation of how a certain action is useful in the given state. With the increase of the Q value, the reward function value (noted with r) increases, this being the way the algorithm is learning the way to solve the problem in the most efficient way. In the formula numbered with 1, we can observe that the present value of the function is determined by the value of the following period, this is possible because at the start of the program the Q values are initialized with random values for all the states. In the first iteration of training, the algorithm updates the Q value in the present state by taking into account the randomly generated variable in the next state. Because the algorithm maximizes the reward and not the Q value, the program will reach after a certain number of iterations the best result.

In the described way the Q learning algorithm has the following structure: at first, it initializes all the Q values in the matrix randomly, in the following way:

$$Q(s,a) = n, \forall\, s \in S, a \in A \qquad (2)$$

And the final state is initialized by the following formula:

$$Q(s - final, a) = 0 \qquad (3)$$

In the written functions S and A are the sets of actions and of states, the algorithm chooses the action a from set A, and it acts the appropriate action and it observes the r value for the following state s_{t+1}. From all the possible actions the algorithm selects the one with the greatest Q value. And then it updates the states according to formula 1. The steps are repeated until the final state is reached. What was described until now is the iteration of an algorithm, for the majority of the programs more than one iteration, is necessary to obtain conclusive results. In the following pages, we described the results of the algorithms for the time interval that was analyzed.

By applying the turtle algorithm to the analyzed time interval and stocks, we obtain the following results for all the stocks at the end of the five years.

Table 1. Results of the "turtle" algorithm for the analyzed interval (own calculations)

Share of company	GM	AAPL	MSFT	AMZN	Ford	AXP	IBM	Visa	JPM	Home Depot	Average
Return	2.68%	1.67%	0.22%	30.26%	-1.28%	2.69%	-25.95%	3.92%	4.04%	8.32%	2.66%
Profit	268.39	167	22	3026	-128	269	-2595	392	404	832	265.7

The following graph describes the actions of the algorithm for the shares of the Amazon Company, the green arrows are used to signal a buying signal and the red arrows signal a selling signal that was perceived by the algorithm and the program acted on.

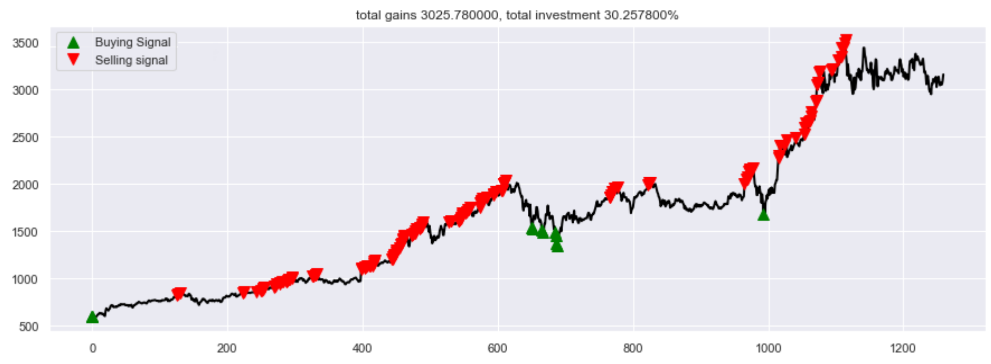

Figure 1. Evolution of the "turtle" algorithm for AMZN
(own calculations)

In the following table, we present the results obtained by calculating the Simple Moving Average algorithm for the same sample of stocks in the same period.

Table 2. Results of the SMA algorithm for the analyzed interval (own calculations)

Share of company	GM	AAPL	MSFT	AMZN	Ford	AXP	IBM	Visa	JPM	Home Depot	Average
Return	-0.24%	-0.57%	-0.95%	-9.42%	-0.12%	-1.41%	-1.74%	0.55%	-0.85%	-2.96%	-1.77%
Profit	-24	-57	-95	-942	-12	-141	-174	55	-85	-296	-177.1

The next figure illustrates the way the algorithm acted when analyzing the given period for the share of Apple Company, the red arrows represent selling signals and the green ones, buying signals.

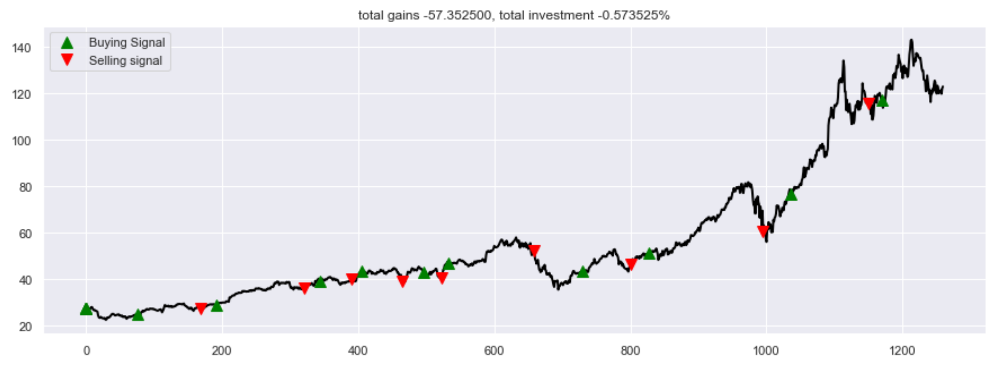

Figure 2. Evolution of the SMA algorithm for AAPL
(own calculations)

In the following part of the paper, we will present the results obtained by applying the Q-learning algorithm for the sample period. In the implementation of the Q-learning algorithm, 500 iterations were used. These results are presented in Table 3.

Table 3. Results of the Q-learning algorithm for the analyzed interval (own calculations)

Share of company	GM	AAPL	MSFT	AMZN	Ford	AXP	IBM	Visa	JPM	Home Depot	Average
Return	13.25%	103.46%	218.00%	517.61%	-3.13%	52.73%	30.49%	142.20%	77.77%	114.87%	126.72%
Profit	1324.98	10346	21800.04	51761.18	-313.085	5272.87	3048.98	14219.63	7776.54	11486.84	12672.4

It can be observed that the results obtained from the Q-learning-based algorithm are significantly better than the ones that were obtained by applying the “turtle” algorithm and the Simple Moving Average. The biggest return was registered in the case of the shares issued by Amazon the algorithm made a return of 517.61%.

Figure 3 presents the actions of the algorithm for the used sample and the moments that the algorithm considered to be suitable to buy or to sell the shares of the Apple Company. The total returns registered in the sampled period are 103.46%.

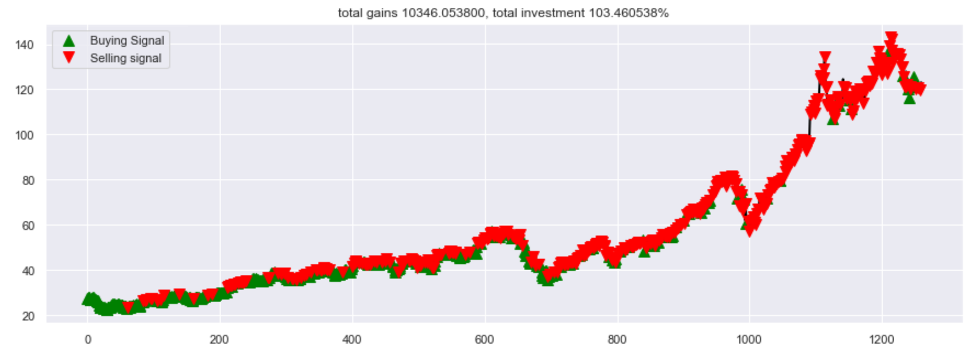

Figure 3. Evolution of the Q-learning algorithm for AAPL
(own calculations)

For the comparison with the classic portfolio management techniques, we choose to compute, with the help of several packages available for Python, three portfolios that minimize a certain risk measure that can be calculated for the sample shares. The first portfolio minimizes the Standard Deviation, the second one the Mean Absolute Deviation, and the third one minimizes the Semi Standard Variance. In the following figure, we illustrated the value of the portfolio that minimizes the Standard Deviation by minimizing the Sharpe Mean-Variance.

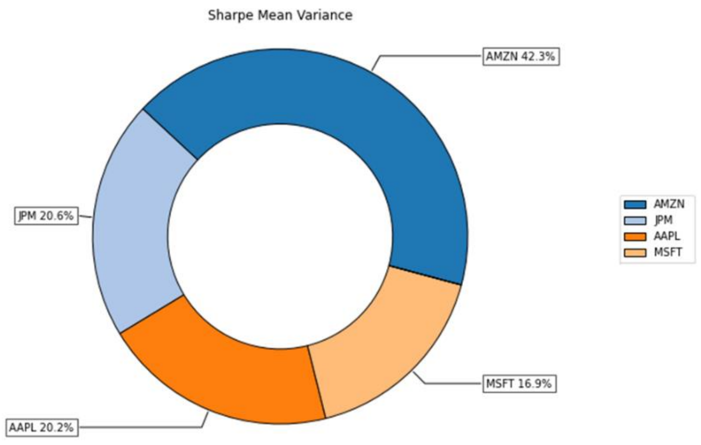

Figure 4.The composition of the portfolio that minimizes the standard deviation
(own calculations)

In the following table, we presented the results for the portfolio optimized by the risk measures selected.

Table 4. Optimized portfolio for the selected risk measures (own calculations)

Share of company	GM	AAPL	MSFT	AMZN	Ford	AXP	IBM	Visa	JPM	Home Depot
SD	0.00%	20.24%	16.92%	42.26%	0.00%	0.00%	0.00%	0.00%	20.58%	0.00%
MAD	0.00%	16.62%	21.97%	25.07%	0.00%	0.00%	0.00%	0.00%	23.67%	12.68%
MSV	0.00%	17.37%	15.74%	47.82%	0.00%	0.00%	0.00%	0.00%	19.07%	0.00%

The risk measures are calculated by using the following formulas: for the Standard Deviation the used formula is:

$$DS = \sqrt{\frac{1}{N-1} * \sum_{i-1}^{N} (x_i - \bar{x})^2} \qquad (4)$$

And for the Mean Absolute Deviation the formula used for calculation is:

$$MAD = \frac{\sum_{i=1}^{N} |x_i - \bar{x}|}{N} \qquad (5)$$

The formula used for the Semi-Standard Deviation is:

$$MSV = \sqrt{\frac{1}{N} * \sum_{i<\bar{x}}^{N} (\bar{x} - x)^2} \qquad (6)$$

We start at the presumption that the investor will keep the money in the portfolio for the period, acting as a long-term investment. The scope of this calculation is comparing the values obtained by having a long-term passive investment strategy. The comparison can be observed in the 5th Table.

Table 5. Comparison between active and passive investment management (own calculation)

Strategy	Active			Passive		
	"Turtle"	SMA	Q-Learning	SD	MAD	MSV
Return at the end of sample	2.66%	-1.77%	126.72%	411.53%	364.65%	414%

From the table we can see that the optimized portfolios have had a better performance than all the real-time algorithms, these results however should not be taken as a failure of the algorithms, the optimization portfolios have been calculated at the end of the sample, so their role is one of a benchmark, they represent the best possible results in the given period. Taking these facts into account and that the average obtained for the algorithms has an equal investment in all shares, we can state that for the active investment of capital the Q-learning algorithms represent a viable option for wealth management.

Conclusions

This article demonstrates the possibility of managing an investment portfolio by using an algorithm based on Q-learning, even though how it works is mainly based on iteration, with a large enough database for training, the algorithm can prove to be useful in active investment management and at least as an investment selection or analysis tool, that can be run before taking an investment decision.

In conclusion, we can state that the future of wealth management will be certainly marked by the use of reinforcement learning-based algorithms that will take the role of the active investment manager. The results also show the impressive way the algorithm works in a crisis context, it should be taken into account that the analyzed sample contains the aftermath of the COVID-19 crisis that had brought about a correction of the markets. Even though the results are far from the benchmark set by the optimized portfolios, the results are still better than other real-time management methods, such as the "turtle" and the Simple Moving Average-based algorithm. The problem posed by the application of the algorithms is the fact that they need to be implemented on a market that has a high degree of liquidity, this was resolved in the paper by applying them to the NYSE and NASDAQ market which are considered the most liquid in the world, in this way their application on emerging markets such as those in Eastern Europe can be problematic due to the lower degree of liquidity when comparing to the US markets.

References

Almahdi, S., & Yang, S. Y. (2017). An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown. *Expert Systems With Applications, 87*, 267–279. https://doi.org/10.1016/j.eswa.2017.06.023

Bertoluzzo, F., & Corazza, M. (2012).Testing different Reinforcement Learning configurations for financial trading: Introduction and applications. *Procedia Economics and Finance, 3*, 68–77. https://doi.org/10.1016/S2212-5671(12)00122-0

Brock, W., Lakonishok, J., & Lebaron, B. (1992). Simple technical trading rules and the stochastic properties of stock returns. *The Journal of Finance, 47*, 1731–1764. https://doi.org/10.2307/2328994

Cajas, D. (n.d). Riskfolio library. https://github.com/dcajasn/Riskfolio-Lib

Casqueiro, P. X., & Rodrigues, A. J. L. (2006). Neuro-dynamic trading methods. *European Journal of Operational Research, 175*, 1400–1412. https://doi.org/10.1016/j.ejor.2005.02.015

Chen, C. H., & Yu, H. Y. (2017). A series-based group stock portfolio optimization approach using the grouping genetic algorithm with symbolic aggregate approximations. *Knowledge-Based Systems, 125*, 146–163. https://doi.org/10.1016/j.knosys.2017.03.018

Chen, T., & Chen, F. (2016). An intelligent pattern recognition model for supporting investment decisions in the stock market. *Information Sciences*, 261–274. https://doi.org/10.1016/j.ins.2016.01.079

Chourmouziadis, K., & Chatzoglou, P. D. (2016). An intelligent short-term stock trading fuzzy system for assisting investors in portfolio management. *Expert Systems With Applications, 43*, 298–311. https://doi.org/10.1016/j.eswa.2015.07.063

Consigli, G., & Dempster, M. A. H. (1998). Dynamic stochastic programming for asset–liability management. *Annals of Operations Research, 81*, 131–161. https://doi.org/10.2139/ssrn.34780

Cumming, D., Johan, S., & Li, D. (2011). Exchange trading rules and stock market liquidity. *Journal of Financial Economics, 99*, 651–671. https://doi.org/10.1016/j.jfineco.2010.10.001

Dempster, M. A. H., & Leemans, V. (2006). An automated FX trading system using adaptive reinforcement learning. *Expert Systems with Applications, 30*, 543–552. https://doi.org/10.1016/j.eswa.2005.10.012

Eilers, D., Dunis, C. L., Mettenheim, H. J., & Breitner, M. H. (2014). Intelligent trading of seasonal effects: A decision support algorithm based on reinforcement learning. *Decision Support Systems, 64*, 100–108. https://doi.org/10.1016/j.dss.2014.04.011

Golub, B., Holmer, M., Mckendall, R., Polhlman, L., & Zenios, S. A. (1995). A stochastic programming model for money management. *European Journal of Operations Research, 85*, 282–296. https://doi.org/10.1016/0377-2217(94)00038-E

Grinold, R. C. & Khan, R. N. (2000). *Active portfolio management: A quantitative approach for producing superior returns and controlling risk (*2nd edition). McGraw Hill. https://doi.org/10.1093/rfs/13.4.1153

Hilpisch, Y. (2015). *Derivatives Analytics with Python*, (pp. 156-232), Wiley Finance.

Hilpisch, Y. (2015). *Python for Finance: Analyze Big Financial Data* (pp. 86-143), O'Reilly Media.

Hilpisch, Y. (2018). *Python for Finance – Mastering Data – Driven Finance* (pp. 78-341), O'Reilly Media.

Israelsen, C. L., et al. (2005). A refinement to the Sharpe ratio and information ratio. *Journal of Asset Management, 5*, 423–427. https://doi.org/10.1057/palgrave.jam.2240158

Jangmin, O., Lee, J., Lee, J. W., & Zhang, B. T. (2006). Adaptive stock trading with dynamic asset allocation using reinforcement learning. *Information Sciences, 176*(15), 2121-2147. https://doi.org/10.1016/j.ins.2005.10.009

Jansen, S. (2018). *Hands-On Machine Learning for Algorithmic Trading: Design and implement investment strategies based on smart algorithms that learn from data using Python,* (pp. 123-345), Packt Publishing.

Jeong, G., & Kim, H. Y. (2019). Improving financial trading decisions using deep Qlearning: Predicting the number of shares, action strategies, and transfer learning. *Expert System with Applications, 117*, 125–138. https://doi.org/10.1016/j.eswa.2018.09.036

Kouwenberg, R. (2001). Scenario generation and stochastic programming models for asset liability management. *European Journal of Operational Research, 134*, 279–292. https://doi.org/10.1016/S0377-2217(00)00261-7

Mansini, R., Ogryczak W., & Speranza, M.G. (2003). On lp solvable models for portfolio selection. *Informatica, 14*, 37–62. https://doi.org/10.15388/Informatica.2003.003

Markowitz, H. M. (1952). Portfolio Selection. *The Journal of Finance*, *7(1)*, 77–91. https://doi.org/10.2307/2975974

Markowitz, H. M. (1956). The Optimization of a Quadratic Function Subject to Linear Constraints. *Naval Research Logistics Quarterly, 3(1–2)*, 111–133. https://doi.org/10.1002/nav.3800030110

Moody, J., & Saffell, M. (2001). Learning to trade via direct reinforcement. *IEEE Transactions on Neural Networks 12(4)*, 875-89. https://doi.org/10.1109/72.935097

Moody, J., Saffell, M., Liao, Y., & Wu, L. (1998) Reinforcement Learning for Trading Systems and Portfolios: Immediate vs Future Rewards. *Refenes AP.N., Burgess A.N., Moody J.E. (eds) Decision Technologies for Computational Finance. Advances in Computational Management Science* (vol. 2). Springer. https://doi.org/10.1007/978-1-4615-5625-1_10

Neuneier, R. (1995). Optimal Asset Allocation using Adaptive Dynamic Programming. *NIPS*. https://doi.org/10.5555/2998828.2998962

Neuneier, R. (1997). Enhancing Q-Learning for Optimal Asset Allocation. *NIPS*. https://doi.org/10.5555/3008904.3009035

Zhang, X., Hu, Y., Xie, K., Zhang, W., Su, L., & Liu, M. (2015). An evolutionary trend reversion model for stock trading rule discovery. *Knowledge-Based Systems, 79,* 27–35. https://doi.org/10.1016/j.knosys.2014.08.010

Zolkepli, H. (n.d.). Stock Prediction Models. https://github.com/huseinzol05/Stock-Prediction-Models