Reinforcement learning in Portfolio Management and its interpretation

Laurens Weijs

366515lw@eur.nl

Personal: *https://sites.google.com/view/laurensweijs*

Project: *https://laurenswe.github.io*

January 22, 2018

Master Thesis Quantitative Finance

Professor: Rutger-Jan Lange

Abstract

Since all machine learning methods commonly in use today are viewed as black boxes, the goal of this paper is to make one of these methods transparent in the context of Portfolio management. I interpret the strategies implied by reinforcement learning (RL) and relate them to the strategies implied by academic portfolio advice with the help of their classical portfolio (CP) management models. Because RL is actually approximate dynamic programming (DP), it is perfectly suited for the volatile DP environment of portfolio management compared to other machine learning methods in use. In terms of performance, this RL method is able to: 1) achieve the same average terminal wealth of 1.33, which is an increase of 33% in portfolio value over five years, as the CP model with a low risk-aversion 2) diminish the standard deviation of the terminal wealth by 30% from 0.35 to 0.25 3) have a lower turnover than the same CP model by three percent. This can mostly be explained by the conservative investing of the reinforcement learning method overall.

Erasmus School of Economics

1 Introduction

The correct long-term portfolio management decision is the most important decision for large institutional investors such as mutual funds or pension funds. At the same time, the right risk-return trade-off in highly volatile markets still is one of the least understood topics. With the arrival of sophisticated quantitative modeling techniques, the stock market became more predictable and long-term portfolio management was revived again [Cochrane, 1999]. Now with the arrival of machine learning models, a branch of artificial intelligence, which do not need any knowledge of financial markets or the sophisticated models in use, a new era of long-term portfolio management has begun [Schindler et al., 2017].

Because the large institutional investors do not want to put faith in a model which cannot be explained by the financial theory, this research will help them generate the same results of the machine learning methods while knowing exactly what the model does. The need to adapt for pension and mutual funds is high, several large hedge funds [Metz, 2016] already made the step to embrace artificial intelligence for their portfolio management decisions and with the low-interest environment of today's world, further diversifying the portfolios of pension funds is a necessary step in which machine learning can support [Noriega and Ballinas, 2016]. Although one might think that both pension and hedge funds have very different risk appetite, this opposes no problem for the proposed method.

In this paper, I investigate what the underlying dynamics are of a reinforcement learning method based on Sutton and Barto [1998] in an out-of-sample portfolio management context, and how the current portfolio management methods, called classical in this paper, can be improved with the knowledge of the reinforcement learning methods. These models are rather shallow compared to modern day deep models stated by Li [2017], where the Q-value functions are replaced by deep machine learning techniques such as neural networks. The emphasis of this paper lies mainly on understanding in which cases one of the two methods outperforms the other and where the possible improvements lie for both models.

As a benchmark, I take the classical portfolio (CP) management method as stated in Campbell and Viceira [2001], this method models stock returns with a Vector Autoregressive (VAR) model. Based on Monte Carlo simulations [Brandt et al., 2005] from this VAR model the average utility over all the future paths of stock returns for each portfolio weight is calculated. By the average utilities over time and over the portfolio weights, the weight in the next step is chosen by the maximum average utility.

The second method is the reinforcement learning (RL) method, in operations research often referred to as approximate DP. This method learns the optimal state-action function in order to make a decision on what portfolio weight to take each period. This state-action function represents the value of the next state given the action to take. With this function, the actor is able to choose the state with maximum utility and choose its respective action. The state-action function is estimated with a neural network which gradually learns the optimal state-action function. Respected papers within the reinforcement learning community with a finance context are Moody et al. [1998] and Du et al. [2016] however, they do not use a function approximation for the estimation of the mapping of states to actions and therefore their applications are limited in their use case.

Reinforcement learning methods are sometimes used in the context of finance, but not yet fully investigated in terms of their strategies in an out-of-sample context. Most research focuses on proving that they can be used to replace dynamic programming within a known environment like shown in Hens and Woehrmann [2007], which is therefore not usable in this context. Also Jiang and Liang [2017] and Jin and El-Saawy [2017] show as a proof of concept that deep reinforcement learning methods can work, but lack the further investigation of their inner workings. This out-of-sample context and the investigation of the strategies used by the reinforcement learning method is provided by this paper in a sufficiently bounded environment.

The reinforcement learning method that optimizes over the maximum reward instead of utility shows the most resemblance with the classical portfolio method for a risk aversion of $\gamma = 2$, which is equal to a low risk-aversion of the investor. It is able to achieve the same average terminal wealth during the whole observation period, namely 1.33, while having a lower standard deviation, 0.25 compared to 0.35, and lower turnover, 63.37 compared to 65.13. This is mainly due to the relative conservativeness of the RL method in investing. This conservativeness translated into action is that the RL rarely takes a full weight in one of the weights, while it does when it is completely sure given the historical returns. This holds true in the simulations and also in the real case.

Compared to other literature this paper compares RL methods to state-of-the-art econometric portfolio management techniques, and together with these methods tries to give more insights into the RL methods. One could conclude that given the sophisticated tools there is no more information in the dataset currently at hand. Especially because of Fama [1970] state that prices reflect all available information according to the efficient market hypothesis. One should consider larger datasets of more diverse information for the methods to gather from the datasets.

Especially because reinforcement learning methods are widely seen as a black box, it would certainly help to investigate the methods in a simulated and controlled environment. The results can help to understand the RL methods and possibly improve the classical portfolio management techniques. Therefore, I investigate the following questions:

How does the reinforcement learning method perform in a simulated environment with known parameters compared to classical portfolio management?

Also, Diris et al. [2015] recognize that the true data generating process is not known and therefore the VAR model is prone to misspecification and parameter estimation errors. This leads to the fact that dynamic portfolio is the same as the repeated myopic portfolio, only looking one step forward at each time and is not able to beat the naive portfolio diversification of equal weights. Reinforcement learning does not try to estimate the dynamics of assets but the dynamics of the state-action function. Therefore, it can be able to improve on estimation or even the naive diversification. This brings us to the next objective question of this paper:

How can reinforcement learning improve on classical portfolio management or the naive portfolio diversification out of sample?

The remainder of this paper is structured as follows: section 2 describes the data used in this paper together with the methods of simulation data; section 3 presents the environment of portfolio management, lays out the models of CP and RL, and presents the theoretical bridge between them in order to gain a better understanding of the methods; section 4 solves the portfolio management problem statement and shows that the RL method is able to improve on CP in terms of stability because it has a lower standard deviation and turnover; section 5 concludes that RL slightly outperforms the CP method while also its caveats are discussed, of which most noteworthy are the hyperparameters and hunger for feature-rich datasets.

2 Data

2.1 Historical data

This research is based on the monthly stock and bond market of the United States. Because I consider three asset classes to choose from, I gather these three classes from various data sources. Please see Table 1 for a description of the data and their source.

Table 1 – Data used in this research combined with the data source.

Stock class	**US Asset**	**source**
Short-term nominally risk-free T-bills (r_f)	Nominal 3-month T-Bill	FRED
Long-term nominal bonds (x_b)	Nominal 5-year T-Note	FRED
Equity (x_s)	Weighted average of NYSE, NASDAQ, and AMEX	CRSP

To convert the nominal values in Table 1 to real values one needs to perform the following operations: For the ex-post real T-bill (r_f) one needs to subtract the log inflation (retrieved from CRSP) from log return of the nominal 3 month T-bill. For the excess real log stock returns (x_b), the 50-year T-Note is subtracted by the nominal log return of the 3-month T-Bill, and for the excess real log stocks returns (x_s) the value-weighted average of the assets stated in Table 1 is also subtracted from the nominal log return of the 3-month T-bill. The excess returns are taken because one is assumed to lend against the risk-free rate when investing in this research.

The summary statistics stated in Table 2 of the real assets constructed above show typical market behavior. The safest asset, with the lowest volatility but also a low return, is the Ex-post T-Bill rate r_f. The stocks, on the other hand, are more volatile but on average have a higher return. Note that this table is about the returns and not about prices because returns exhibit more attractive statistical properties like stability.

Table 2 – Summary statistics of the three assets taken into consideration, the ex-post real T-bill returns, excess-bond returns, and the value-weighted stock returns. The dataset starts in February 1954 and ends in December 2016 and is notated in monthly returns.

	r_f	x_b	x_s
Avg.	0.0008	0.0013	0.0057
Std. dev.	0.0033	0.0146	0.0432
Min	-0.0108	-0.0687	-0.2305
Max	0.0193	0.0951	0.1594
AR(1)	0.4456	0.1193	0.0907

2.2 Simulated data

Based on these assets, simulations are made for four different scenarios based on the knowledge of the distribution of returns, Table 3 displays these different scenarios. Classical portfolio management assumes that by estimating a model of the returns they have found the true data generating process of the returns and thus operate in the domain of scenarios 1 and 2. This research, however, does not assume it knows the model of the underlying assets and therefore, operates in scenarios 3 and 4.

Table 3 – Different assumptions involved in the asset allocation problem, the numbers state different scenarios and their respective assumptions.

Distribution returns	Constant parameters	Time-varying parameters
Known parameters	1	2
Unknown parameters	3	4

Corresponding to the different assumptions three econometric models are chosen to simulate the dynamics of the scenarios. Scenario 1 with the known and constant parameters is simulated by the constant expected return (CER) [1] model. This CER model simply states that the returns are simulated by a mean and the idiosyncratic error. Scenario 2 is simulated by the vector autoregressive (VAR) model which incorporates the previous return one period back into the equation. And at last scenario 4 is simulated by a Bayesian vector autoregressive (BVAR) model. This BVAR model states that the parameters of the returns over time are unknown and generated by a predefined distribution function. The mathematical notation of the models are shown respectively in the next equations,

[1]The CER model is introduced because it has a clear analytical solution, the myopic solution.

$$y_t = \mu + \varepsilon_t, \text{ with } \varepsilon_t \sim N(0, \sigma^2), \tag{1}$$

$$y_t = \hat{A} + \hat{B} y_{t-1} + \varepsilon_t, \text{ with } \varepsilon_t \sim N(0, \sigma^2), \tag{2}$$

$$y_t = A + B y_{t-1} + \varepsilon_t, \tag{3}$$

$$\text{with } \varepsilon_t \sim N(0, \sigma_i^2),\ A_i, B_i \sim N(\hat{A} \text{ or } \hat{B}, \frac{\sigma_i^2}{T}), \text{ and } \sigma_i^2 \sim iGamma2(SSE, T-1).$$

Where y_t denotes the vector of asset returns $(r, x_b, x_s)'$ and for each simulation i the uncertainty parameter of equation (3) has a different draw of the inverse Gamma distribution with as parameters the sum squared residual of the shrinkage model stated in equation (2) and $T - 1$. The parameters for the simulations are shown in Table 4.

Table 4 – Parameters of the simulation models in the control experiment, estimated on the historical data described in the beginning of the data section with the respective model. For example, the parameters of the CER simulation are retrieved by estimating a CER model on the whole sample of historical data.y_t is denoted as a vector of the three assets: $(r, x_b, x_s)'$. For the BVAR model no parameters are shown because these are the same as the parameters of the VAR model.

Model:	Simulation equation
CER	$y_{t+1} = \begin{pmatrix} 0.0008 \\ 0.0013 \\ 0.0057 \end{pmatrix} + \epsilon_{t+1},\ \text{with } \epsilon_{t+1} \sim N\left(0, \begin{pmatrix} 0.0000 \\ 0.0002 \\ 0.0019 \end{pmatrix}\right)$
VAR	$y_{t+1} = \begin{pmatrix} 0.0004 \\ 0.0012 \\ 0.0047 \end{pmatrix} + \begin{pmatrix} 0.46 & -0.01 & 0.01 \\ 0.45 & -0.06 & 0.11 \\ 0.10 & 0.07 & 0.36 \end{pmatrix} y_t + \epsilon_{t+1},\ \text{with } \epsilon_{t+1} \sim N\left(0, \begin{pmatrix} 0.003 \\ 0.014 \\ 0.043 \end{pmatrix}\right)$
BVAR[2]	$y_{t+1} = B_0 + B_1 y_t + \varepsilon_{t+1}, \varepsilon$, with $P(\Sigma \mid B, Y) = iWishart\left((Y - XB')'(Y - XB'), T\right)$ $P(B \mid \Sigma, Y) = N_{trunc}\left(\hat{B}, \Sigma \otimes (X'X)^{-1}\right)$

Note again that the simulated models show again typical market behavior for the different assets in consideration. With the safest asset, the risk-free rate (r_f), has a low mean and a near zero standard deviation in the CER and similar properties for the VAR and BVAR. This near zero property of the r_f makes sense because the risk-free rate is essentially risk free.

[2]The Bayesian VAR is simulated with the help of the Gibbs sampler and the two probability density functions given in the second and third row with a thinning of 100 and burn-in of 1000.

3 Methodology

3.1 Problem Statement

This research focuses on a world where an investor can choose between three assets: equity, long-term real T-Notes, and short-term real T-Bills. The problem stated for a long-term investor in portfolio management is a dynamic intertemporal weight optimization problem with uncertainty about the future states. The objective function of such an investor with intertemporal utility $U(\cdot)$ is given by:

$$\begin{aligned} &\max_{w_t,...,w_{t+K-1}} \mathbb{E}_t[U(W_{t+K})] \text{ s.t.} \\ W_{s+1} &= W_s(w_s' r_{s+1} + r_{f,s+1}) \quad \text{, for } s = t, \dots, t+K-1 \\ w_s'\iota &= 1 \quad \text{, with } \iota \text{ being a vector of ones.} \end{aligned} \tag{4}$$

The second equation is the budget restriction and the third equation restricts the sum of the weights to count up to one. Your wealth one period ahead can only change by a change in the value of the assets and your current holding in the specific assets. The other parameters in these equations are described as follows: K is the number of periods to optimize over in the future, W_s is the current wealth at time s, not to be confused with w_s, which is the weight per asset class at time s, r_{s+1} is a vector of excess returns on the asset classes one period in the future, $r_{f,s+1}$ is the risk-free asset at time $s+1$, and $U(\cdot)$ is the utility function, which in this research is the power utility function defined by $\frac{(\cdot)^{1-\gamma}}{1-\gamma}$, with γ being the risk aversion of the investor. The risk aversions taken into account in this research are $\gamma = 0, 2, 5, 10$, with 0 being completely risk neutral and 10 being risk averse. Risk-seeking behavior is not incorporated in this research because this is not a behavior typically performed by large institutional investors.

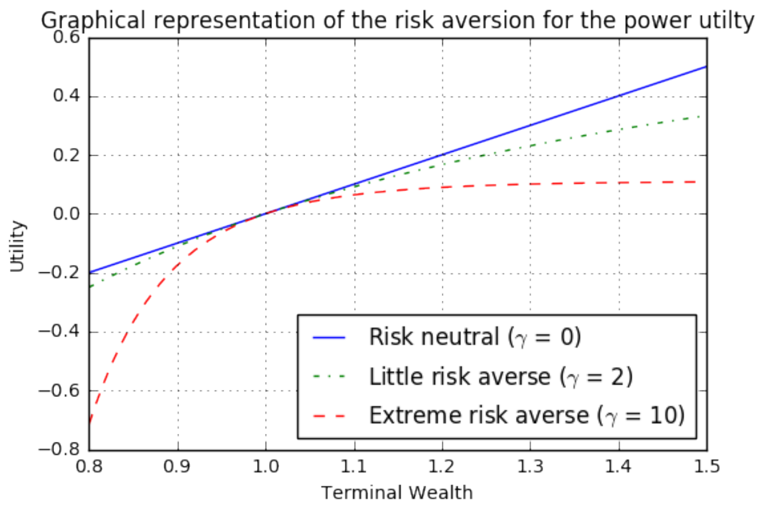


Figure 1 – Utility gained per unit of Terminal Wealth with the power utility for three values of γ.

For a range of terminal wealth, the respective utilities are shown in Figure 1. This figure shows that the marginal utility gained for each extra unit of terminal wealth is lower when you have a higher risk aversion. Especially in the case when the terminal wealth is lower than 1.0 the more risk-averse an investor is, the exponentially less utility it gains. This utility function is a behavior which closely resembles human biases against risk.

This dynamic problem can be transformed into the following Bellman equation to show the recursive nature of this problem statement:

$$\begin{aligned} V_{t+K}(W_t,\theta) &= \max_{w_t,...,w_{t+K-1}} \mathbb{E}_t[U(W_{t+K})], & (5)\\ &= \max_{w_t} \mathbb{E}_t\left[\max_{w_t+1,...,w_{t+K-1}} \mathbb{E}_{t+1}[U(W_{t+K})]\right], \\ &= \max_{w_t} \mathbb{E}_t\left[V_{t+1}(W_t(w_t' r_{t+1} + r_{f,t+1},\theta))\right], & (6)\\ \text{with terminal condition:} \quad & V_t(W_t) = U(W_t). \end{aligned}$$

Here θ represents the vector of parameters of the model from the asset classes stated in the different scenarios of Table 3. When the parameters are assumed to be unknown it is therefore not possible to estimate the expectation of the right-hand side of equation (6) immediately. If however the parameters are assumed to be known, one could calculate the expectation by the data generating process. When we write the budget constraint of equation (4), in terms of the starting wealth and terminal wealth, we get:

$$W_{t+K} = W_t \prod_{s=t}^{t+K-1} (w_s' r_{s+1} + r_{f,s+1}). \tag{7}$$

When we substitute equation (7) into the Bellman equation, work out W_t from the expectation, and use the power utility function as utility we get the following equation:

$$V_{t+K}(W_t,\theta) = \max_{w_t} \mathbb{E}_t\left[\frac{\left(W_t(w_t' r_{t+1} + r_{f,t+1})\right)^{1-\gamma}}{1-\gamma} \max_{w_{t+1}\cdots w_{t+K-1}} \mathbb{E}_{t+1}\left[\left(\prod_{s=t+1}^{t+K-1}(w_s' r_{s+1} + r_{f,s+1})\right)^{1-\gamma}\right]\right]. \tag{8}$$

From this point, we can solve the Bellman equation, which is a necessary condition for optimality, in four different ways with differing assumptions about the distribution of returns as described in Table 3. I state here that the classical portfolio management way is the solution method performed by Diris et al. [2015] with the assumption that the distribution is known and covers scenarios 1 and 2 of this table. The proposed way that is raised by the reinforcement learning literature has no assumptions about the distribution of returns and cover scenario 3 and 4. For scenario 3 it is assumed that the model parameters are fixed over time while unknown, this will not be taken into consideration in this research. Therefore, three cases will be simulated in this research and as an extension, the methods will also be compared with historical data. The benchmark methods chosen in this paper are the $1/N$ method stated in DeMiguel et al. [2009], three strategies which fully invest in a single asset, and the perfect foresight method. The latter one is constructed by $\max(x_b, x_s)$, which is the maximum of the bonds or the return of the stocks for each point in time.

3.2 Benchmark: classical portfolio management

For the classical way, I assume one knows a model of the returns and is, therefore, able to calculate the estimation of the wealth in the next period, which is $\mathbb{E}_{t+1}[U(W_{t+K})]$. This model is first estimated from the true sample of the historical returns of asset prices and the mathematical estimation is then retrieved from this estimated model of returns. The industry workhorse for the estimation is a VAR model, which is formulated as, $y_t = A + By_{t-1} + \varepsilon_t$. A is a vector of intercepts, B is an $(n \times n)$ of the slopes of the equation, and ε is a vector of idiosyncratic errors. With this estimated VAR model and Bayesian statistics, the expectation of the terminal wealth is calculated. According to Bayesian statistics, we approximate the expectation of any distribution with the simulated sample averages,

$$\mathbb{E}[f(\theta)] \approx \frac{1}{N}\sum_{i=1}^{N} f(\theta_i). \tag{9}$$

This is exactly what we do with the help of the Monte Carlo Markov chain (MCMC) algorithm, an extensive reversible Markov chain (MC) is sampled from the estimated VAR model and is assumed to be a true representation of the underlying dynamics of the asset returns. This Markov process (MP) is visualized in Figure 2, where each transition is chosen at random by the stated transition probability. In Finance literature, one will generally encounter not much information about the simulation method of the estimated returns, MCMC. Because the RL method basically constructs an MDP to reflect its knowledge on the underlying MDP of the assets it is worth it to investigate in the simulation method for the classical way.

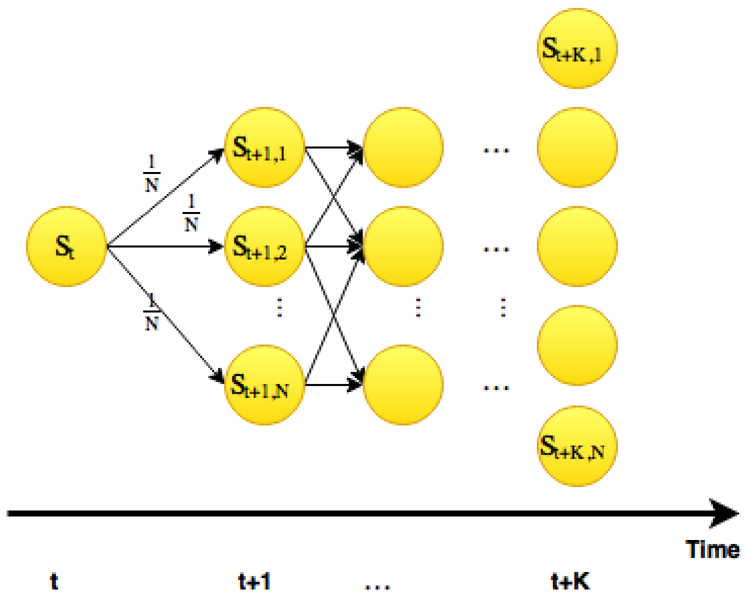


Figure 2 – Graphical representation of the Markov process of the asset prices, transitions are given in transition probabilities and determined by nature.

Given that we can construct the Markov process[3] of the asset returns, one can also construct a Markov decision process (MDP). This makes that the problem statement can be transformed into an MDP. Figure 3 shows the graphical representation of this decision process. In each node, the value of the objective function is shown and an action should be executed at each time period, this action is the weight to take in the specific asset classes. After the weights have been chosen, nature decides, based on the Markov process of the prices in Figure 2, what the price in the next period will be and therefore the new value of the objective function.

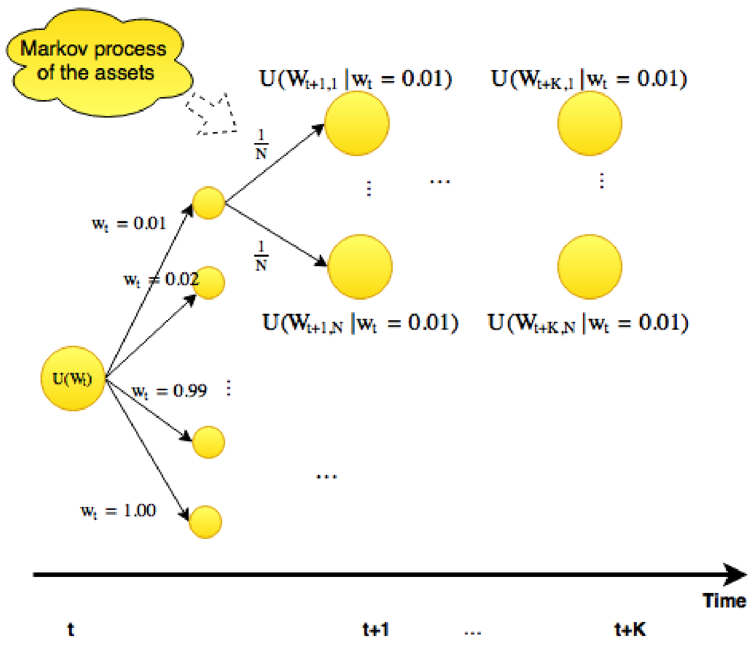


Figure 3 – Graphical representation of a Markov decision process representing the portfolio problem statement of equation (4). Small nodes are action nodes, where the weights should be determined and large nodes represent the state of the objective function.

Now that the objective function has been transformed into a fully deterministic MDP, we can solve the problem by backward induction. Working backward we calculate the expectation of utility for each weight at time $t + K - 1$ by taking the average over the utilities acquired by each w_{t+K-1}. By maximizing each period over the expected utility and starting at the end of the period we maximize the Bellman equation stated in equation (6). In the subsections to follow several extensions are presented, like the predictability of returns, Bayesian interference, and transaction costs.

Predictability of returns

Diris et al. [2015] not only implement the base case stated in the section before, they also take the Bayesian estimation of the probability distribution of the assets into account and predictability in the returns. The latter one now implies that instead of $\mathbb{E}[f(\theta)], \mathbb{E}[f(\theta|z_t)]$ needs to be estimated. One can approximate the conditional expectation by the fitted values of the across-path regression,

[3] For the ease of consistency in literature I write here Markov processes instead of Markov chains, while they are equivalent in discrete state spaces.

that is the fitted values of the regression of the simulated utilities on the state variables. One is not restricted to the data stated in Section 2, to improve the predictability of further returns the VAR model can be expanded by several predictors. A good starting point of choosing the right predictors would be Pesaran and Timmermann [1995].

Bayesian inference

When considering a Bayesian inference, the top right cell of the scenarios in Table 3, one can still make use of the MCMC algorithm to transform the assets into a known Markov process. Now each path that is sampled by the MCMC has different parameters of the probability distribution. This will in most cases result in a higher variance in the values of the Markov process.

Transaction costs

By adding transaction costs according to Gârleanu and Pedersen [2013] the Bellman equation in equation (8) changes to,

$$\max_{w_t} \mathbb{E}_t \left[\frac{\left(W_t(w_t' r_{t+1} + r_{f,t+1} - \Delta w_t TC)\right)^{1-\gamma}}{1-\gamma} \max_{w_{t+1}\cdots w_{t+K-1}} \mathbb{E}_{t+1}\left[\left(\prod_{s=t+1}^{t+K-1} (w_s' r_{s+1} + r_{f,s+1} - \Delta w_s TC) \right)^{1-\gamma} \right] \right].$$

The term added to the equation is the transaction costs $\Delta w_t TC$. Δw_t is the difference between the weights at time t and $t-1$, and TC is the constant transaction costs per unit of asset. It is harder to simulate with the numerical solution proposed in Diris et al. [2015], therefore I take the closed-form solution from Gârleanu and Pedersen [2013]. The optimal weight is the weighted average of the current weight and the aim portfolio (the weighted average of the current and future expected Markowitz portfolios),

$$w_t = \left(1 - \frac{a_{opt}}{TC}\right) w_{t-1} + \frac{a_{opt}}{TC} aim_t.$$

Here a_{opt} is the optimal weighting scheme further specified in Gârleanu and Pedersen [2013].

3.3 Reinforcement learning in portfolio management

In reality, the underlying Markov process of the asset returns is unknown and also volatile in terms of its model parameters. In the CP method, one assumes a single model, mostly a VAR model, and let the parameters be fixed for the whole estimation period. Because of these characteristics of the real asset returns I make use of a model-free RL method. With this method, no model of the returns needs to be assumed and is also dynamically altered after each point in time. Note that you need to take the natural logarithm of the objective function and the budget constraint in order to be able to solve the problem by RL. This is needed so we can rewrite the objective function in the summation of intermediate functions/rewards, in order for the RL agent to learn from each step in time. The main difference between the CP method and the RL method is that the latter solves the problem from the beginning, adjusting its model and parameters for each new prediction. While the CP method estimates a model on the returns and based on simulations from that model it determines the best possible action. The main problem one faces with RL is that one maps historical asset returns, which could be more than just the present asset returns, to so-called Q-values. The function for these Q-values will be explained later on but introduces another layer of abstraction to the problem statement at hand. A reinforcement learning agent can be represented as in Figure 4. At each point in time, the agent takes an action based on its current knowledge of the problem statement. Based on its action the environment will give a reward and the new state of the environment to the agent. In this environment of the asset market, the state (s) represents the current level of the assets, the action (a) the weights of the agent in the different assets, and the reward (R) the log utility gained from moving from state s_t to s_{t+1} with action a.

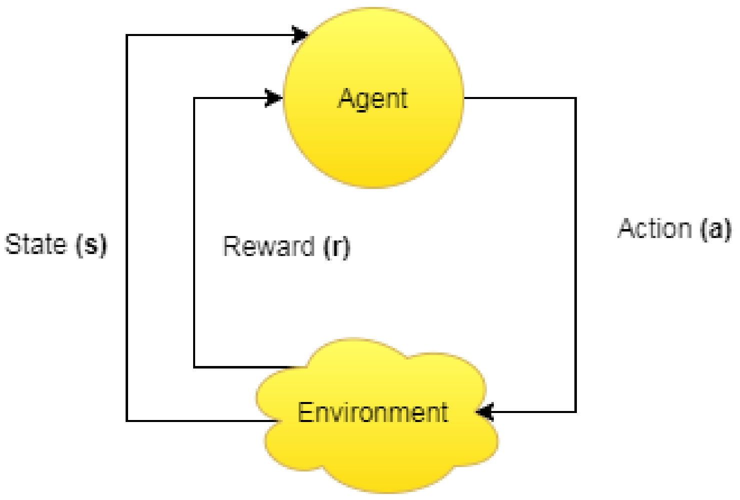


Figure 4 – High-level description of a reinforcement learning agent and how it interacts with the environment.

Consider the following numerical example how this notation fits in portfolio management, take action a to be the weights in the assets at timestamp t, for example, 0.5 in the stocks and 0.5 in the bonds. With this action, the environment returns the reward r respective of the weights a and state s. Take for example that at timestamp $t+1$ the returns over the stocks and bonds compared to timestamp t are +0.10 and -0.05, the stocks have gained ten percent in value and the bonds have shrunken five percent in value. These returns will be the state to be presented to the agent with eventually past returns, the reward to the investor will be $(0.5 \times +0.10) + (0.5 \times -0.05)$. This is the reward the investor would get if one sells the assets immediately the next period.

The starting point, given that we assume the structure of Figure 4 with an self-learning agent learning from its environment based on its actions, is the Bellman equation from equation (6). In order for the method to learn it needs to assign actions to rewards without knowing what the environment does by constructing a mapping from actions to rewards given the state of the environment. Therefore we decouple the Bellman equation of the CP method into the state function and state-action function. These are constructed by taking the value function conditional on a certain state and the value function conditional on a certain state-action pair and are defined by,

$$V_t(s) = \mathbb{E}_t[U(W_{t+K})|S_t = s],$$
$$Q_t(s,a) = \mathbb{E}_t[U(W_{t+K})|S_t = s, A_t = a].$$

The value function is actually the same as the intermediate Bellman equation of equation (5). This value function can be rewritten to an immediate reward R_{t+1}, directly resulting from a specific action performed in a given state, plus the value function of its successor state. In other words, this value function is the gain or loss for the portfolio of returns plus the recursive Bellman equation to the next period in time. This new Bellman equation is shown in equation (10) and the corresponding Bellman equation of the state-action function in equation (11). In these equations, the immediate reward is also substituted by the utility function resulting from an action, such that we have an analytical term for the immediate return R_{t+1}. The in-between steps for these equations are as follows:

$$\begin{aligned} V_t(s) &= \mathbb{E}_t[R_{t+1} + V_{t+1}(S_{t+1})|S_t = s], \qquad (10)\\ &= \mathbb{E}_t\left[\log\left(\frac{\left(W_t(w'_{t,optimal} r_{t+1,S_t} + r_{f,t+1,S_t})\right)^{1-\gamma}}{1-\gamma}\right) + \log(V_{t+1}(S_{t+1}))\middle| S_t = s\right],\end{aligned}$$

$$\begin{aligned} Q_t(s,a) &= \mathbb{E}_t[R_{t+1} + Q_{t+1}(S_{t+1}, A_{t+1})|S_t = s, A_t = a], \qquad (11)\\ &= \mathbb{E}_t\left[\log\left(\frac{\left(W_t(w'_{t,A_t} r_{t+1,S_t} + r_{f,t+1,S_t})\right)^{1-\gamma}}{1-\gamma}\right) + \log(Q_{t+1}(S_{t+1}, A_{t+1}))\middle| S_t = s, A_t = a\right].\end{aligned}$$

In summary, RL assumes that the expectations of assets (and therefore its respective MP) are unknown. By construction, the value function only exists out of the intermediate value functions and can be decoupled by conditioning on states and actions by equations (10) and (11). If one is in the last period and therefore knows the reward given a certain action, one can recursively calculate the previous state-action functions by backward induction. This could be done for both the functions, but we are more interested in which action to take given a certain state. This method is an approximate dynamic programming solution and can solve scenario 1 and 2 from Table 3. Currently, we did not adjust anything about the expected immediate reward and therefore operates in the same domain as standard CP methods.

Unknown Markov decision process / Q-learning

Now I assume that the MDP made with the help of the econometric model is not known. As a result, the right side of equation (11) (= $\log(Q_{t+1}(S_{t+1}, A_{t+1}))$), can not be estimated at all times. Now we estimate $Q(s,a)$ at specific states in order to be able to solve for the optimal state-action function at the end of the period. We would, therefore, learn the optimal Q-values, in the discrete case a matrix representing the knowledge of the MDP describing the asset dynamics. First, the Q-matrix initialized at zero and updated iteratively with the reward function and the next Q-value in the matrix. Because the reward function is retrieved from the environment at each state and action pair you can iterate with respect to the adjusted greedy policy (ε-greedy heuristic) to update the values of the Q-matrix. This heuristic chooses the next action based on the maximal Q-function and chooses a random action by the parameter ε. This updating rule follows directly from equation (11) and is stated as:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha \left[R + \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t) \right]. \tag{12}$$

Here α is the learning rate, the rate to which extent the value is updated with the estimated value of the Q-function. The action a, denoted here as the action leading to the highest state-action function in the next period chosen greedily. In order to improve exploration of the RL method, sometimes with a chance of ε the action will be random. By iteratively updating $Q(s, a)$ it converges to the optimal state-action function $Q_*(s, a)$ [Melo, 2001] within a deterministic framework, but also within a stochastic framework as stated by Jaakkola et al. [1994]. From the optimal Q-table, one can calculate by backward induction the optimal policy/weights for the assets. In the subsections to follow several extensions to the basic RL model are stated: transaction costs, function approximation and the clipping of rewards. This last extension is especially important in the case of financial data.

Transactions costs

Adding transaction costs to the RL method would result in the immediate reward (R_{t+1}) in equation (11) to be reduced by the transaction costs. Assuming the transactions costs are a percentage (TC) of the amount of stock traded in the previous period the total reduction in immediate reward will be, $\Delta w_{t+1} TC$. Here Δw_{t+1} represents the change in stock holding between period t and $t+1$. All together this will result in the following immediate reward at time $t+1$:

$$R_{t+1} = \log \left(\frac{\left(W_t (w'_{t,A_t} r_{t+1,S_t} + r_{f,t+1,S_t}) \right)^{1-\gamma}}{1-\gamma} - \Delta w_{t+1} TC \right).$$

When we now replace the immediate reward in the state action function from equation (11) with the newly introduced immediate reward with transaction costs we will get the next equation,

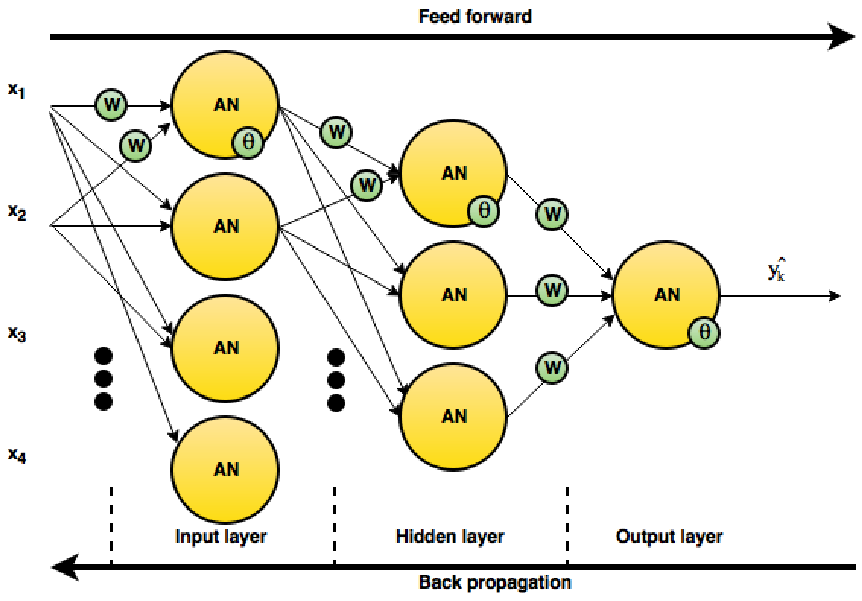
$$\mathbb{E}_t \left[\log \left(\frac{\left(W_t (w'_{t,A_t} r_{t+1,S_t} + r_{f,t+1,S_t}) \right)^{1-\gamma}}{1-\gamma} - \Delta w_{t+1} TC \right) + \log(Q_{t+1}(S_{t+1}, A_{t+1})) \middle| S_t = s, A_t = a \right].$$

Function approximation

Because the states, the returns at timestamp t, are hard to represent in discrete time due to their continuous nature one could introduce a function approximation. Function approximation makes use of a function which maps the current state to Q-values. So instead of constructing a large Q-table with all the possible states the system of returns could be in, one uses a single function to map these values to actions. This function could be anything, for example, a linear combination of the assets, an econometric model, or a neural network [Svozil et al., 1997]. This function approximation is an approximation of the real state-action function and therefore denoted by: $\hat{Q}(s, a, \theta) \approx Q_t(s, a)$.

A neural network as a function approximation is efficient because this function could generalize from states already encountered to new states and therefore reduce the memory usage and computation time significantly compared to the case of a single large Q-table. A representation of a neural network is shown in Figure 5. Although the function approximation is not known to be convergent, to the true state-action function, in practice it tends to oscillate around the state-action function closely.

Figure 5 – Graphical representation of a Neural Network.



To optimize the parameters of θ[4] from the function approximation I make use of the stochastic gradient descent method for neural networks[5]. For generality, neural networks are used so that we do not have to optimize the functional form of the function approximator because one can form many different functions with neural networks. Keep in mind that neural networks can represent any abstract relationship between any variables of interest, from linear to nonlinear, from low-dimensional to high-dimensional relationships. The loss function that would be needed and is the sole instrument needed to update the model parameters of the neural network is given by, $Loss = \sum_a \left(R + \max_{a_{t+1}}(Q(s_{t+1}, a_{t+1})) - Q(s_t, a_t)\right)$.

[4]The values for the model parameter θ for this research are shown in the Appendix.
[5]The package this paper uses for neural networks is Tensorflow for Python.

Clipping of rewards

One should take notice of the clipping of the rewards, for the RL to properly function. In order for the model to be flexible with multiple economic regimes, and therefore probably different underlying data generating functions, the rewards should be rescaled to the range $[-1, 1]$ [Hasselt et al., 2016]. This is especially the case in this dataset because returns are mostly positive for the bonds and the stocks. If the rewards are mainly positive it results in an almost continuous growth of the Q-values and, therefore, it will never converge. Furthermore, Ng et al. [1999] show that this is a necessary condition to preserve the optimal value for the MDP.

4 Results

This section is split up into three parts: the results of the different models in the three simulated cases, the different models applied to the historical data and the interpretation of the strategies in terms of portfolio advice.[6] For reference, another strategy has been added to the tables, the perfect foresight strategy this is the maximum possible terminal wealth one could get in the 60 periods is taken into account.

4.1 Simulated data

In this section, the results are shown of the models run on simulated data specified in equations (1), (2), and (3) for the benchmark methods (the model-independent strategies and the CP method) and the RL method. The CP methods are run for several risk aversions and the RL presented is the best of several combinations of hyperparameters, further specified in the Appendix.

Table 5 shows the results for the model-independent strategies, where notice should be given to the perfect foresight. This model takes a full stance in the bonds or the stocks while having information on the next period. This model, therefore, serves as the perfect timing benchmark, although it is unrealistic. Also, it should be mentioned that the terminal wealth for the VAR and BVAR simulated values are relatively low and will result in a losing portfolio with most strategies.

Table 5 – This table shows the average weight in the stock x_s ($\overline{w_s}$), the average terminal wealth ($\overline{TW}$), and the standard deviation of terminal wealth ($\sigma(TW)$) for three model-free strategies and one reference strategy for three different samples of the return models stated in equations (1), (2), and (3).

Method:	CER		VAR		BVAR	
	$\overline{TW}$	σ_{TW}	$\overline{TW}$	σ_{TW}	$\overline{TW}$	σ_{TW}
$1/N$ each asset	1.17	0.00	1.04	0.11	0.92	0.13
Full risk-free rate (3M T-Bill)	1.05	0.00	1.00	0.04	1.00	0.03
Full bond (5Y T-Note)	1.08	0.00	0.98	0.12	0.99	0.10
Full stock (WA NYSE,NASDAQ, and AMEX)	1.40	0.02	1.19	0.27	0.82	0.28
Perfect foresight	1.40	0.02	3.31	0.36	2.61	0.65

For the constructed models stated in this paper, the simulated values are principally executed as a bare minimum for the RL method. The results of the CP and RL methods for the simulated values are shown in Table 6. A notice should be given to the results for the simulated returns by the CER model, in this simulated environment, the stock returns are for 95% strictly greater than the bond returns. In this situation, the main drawback of the current RL model is exposed. Because all the input is randomized, instead of chronological order to improve the training of the model by reducing the chance of overfitting, the cases where a few of the returns for the bonds were higher than the stocks were in the last training sets and, therefore, have great influence on the prediction. This, however, does not break the confidence in the model but should be taken into account when constructing the model. Also, no model is able to perform well in the case where the returns are

[6]For the code and the data used please visit laurenswe.github.io.

simulated by a BVAR model. This BVAR model simply introduces too much uncertainty to the returns that the methods are not able to find any good estimation models for the returns. While the returns simulated by the VAR model shows most resemblance with the historical data discussed further on.

Table 6 – This table shows the average weight in the stock x_s ($\overline{w_s}$), the average terminal wealth ($\overline{TW}$), the standard deviation of terminal wealth ($\sigma(TW)$), the average Turnover ($\overline{TO}$), and the average realized utility ($\overline{RU}$) for the five models. Three models are the classical portfolio management methods (CP) with different risk aversion γ, and the reinforcement methods denoted by (RL) followed by the method of function approximation. In this table the results are shown of runs on **simulated** data.

Method:	CER					VAR					BVAR				
	$\overline{w}_s$	$\overline{TW}$	σ(TW)	$\overline{TO}$	$\overline{RU}$	$\overline{w}_s$	$\overline{TW}$	σ(TW)	$\overline{TO}$	$\overline{RU}$	$\overline{w}_s$	$\overline{TW}$	σ(TW)	$\overline{TO}$	$\overline{RU}$
CP (γ = 2)	1.00	1.40	0.02	59.72	-0.71	0.43	1.07	0.21	65.49	-0.62	0.45	0.92	0.21	70.93	-1.15
CP (γ = 5)	1.00	1.40	0.02	59.75	-0.06	0.38	1.06	0.18	65.82	-0.27	0.30	0.94	0.18	69.32	-0.47
CP (γ = 10)	1.00	1.40	0.02	59.71	-0.01	0.28	1.03	0.15	65.99	-0.25	0.21	0.95	0.15	68.48	-0.66
RL-NN	**0.95**	**1.38**	**0.03**	**58.09**	-	**0.19**	**1.07**	**0.19**	**64.50**	-	**0.54**	**0.91**	**0.24**	**70.24**	-

Note: Realized Utility for the Reinforcement Learning methods is non-existing due to the absence of gamma when considering a reward function as just the wealth increase per period.

4.2 Historical data

Table 7 shows the results for the model-free benchmarks, the perfect foresight benchmark and the model-dependent methods in case of the historical data for benchmark purposes, like stated in Table 1. Graphically the benchmark models are shown in Figure 6. With a perfect timing of the market returns, the Perfect Foresight model in the figure, one could significantly gain in terms of terminal wealth within this dataset.

Table 7 – This table show the the average weight in the stock x_s ($\overline{w_s}$), the average terminal wealth ($\overline{TW}$), the standard deviation of terminal wealth ($\sigma(TW)$), the average Turnover ($\overline{TO}$), and the average realized utility ($\overline{RU}$) for the five modeled methods and the three model-free methods. Three models are the classical portfolio management methods (CP) with different risk aversion γ, and the reinforcement methods denoted by (RL) followed by the method of function approximation. In this table the results are shown of runs on **real** historical data.

Method:	$\overline{w}_s$	$\overline{TW}$
$1/N$ each asset	1.21	0.14
Full risk-free rate	1.06	0.14
Full bond	1.14	0.11
Full stock	1.48	0.42
Perfect foresight	3.77	0.83

	$\overline{w}_s$	$\overline{TW}$	σ(TW)	$\overline{TO}$	$\overline{RU}$
CP (γ = 0)	0.61	1.32	0.35	65.89	1.32
CP (γ = 2)	0.64	1.34	0.35	65.13	-0.79
CP (γ = 5)	0.61	1.31	0.33	66.16	-0.15
CP (γ = 10)	0.48	1.26	0.25	69.98	-0.06
RL-NN	**0.57**	**1.33**	**0.25**	**63.37**	-

From Table 7 one can conclude that the RL-NN model combines the best results of a risk-neutral and risk-averse investor. RL-NN is able to achieve a comparable average terminal wealth of 1.33 with the risk-neutral investor and a low standard deviation of 0.25 compared to the highly risk-averse investor. Together with these striking results, the turnover is slightly lower than the other CP models, this was not programmed in the model itself but a result of its conservative strategy of rarely investing fully in the assets. Although the RL-NN model shows a relatively good performance compared to the CP models its timing is far from optimal compared to the average terminal wealth of the perfect foresight of 3.77, of course, this result is still a long way off from the infeasible perfect foresight results.

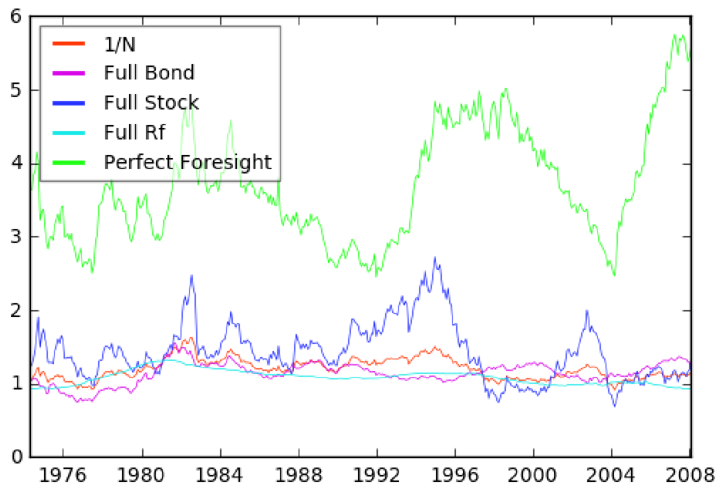


Figure 6 – Time series of the terminal wealth for the four benchmark strategies and the perfect foresight strategy.

The time series of the terminal wealth for selected methods is shown in Figure 7. Two things can be concluded from this figure; one, the RL-NN method is less volatile compared to the CP method and two, the methods are only able to outperform the full stocks strategy at the end of the dataset by effectively timing the portfolio weights. This suggests that both models need much data to train on or that both methods are lucky in this period due to the low returns in the stocks.

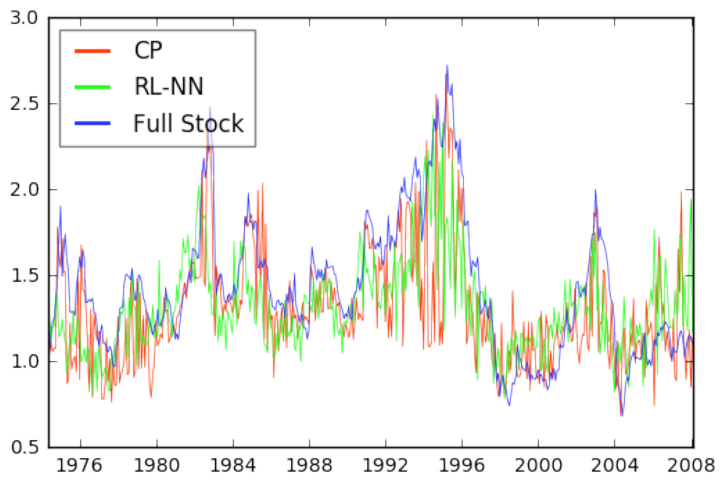
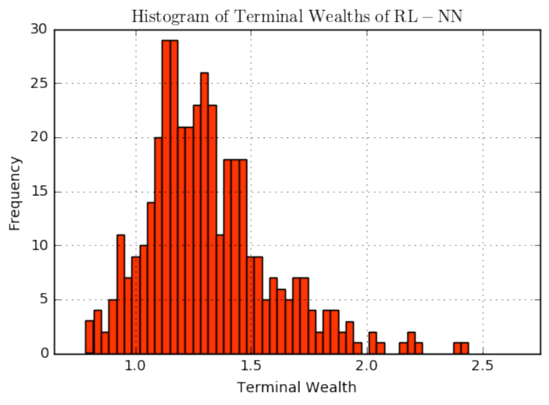
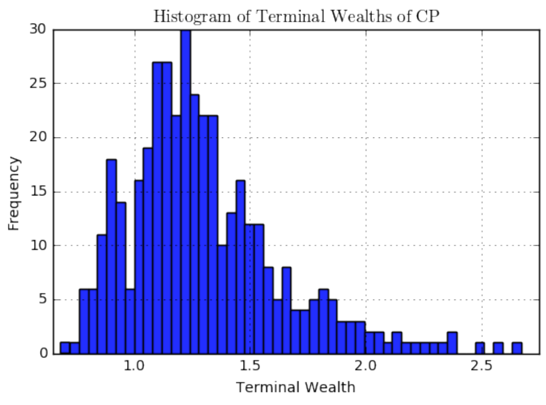


Figure 7 – Time series of the terminal wealth of three methods (RL with NN, full investment in the stock, and classical portfolio management with $\gamma = 2$).



(a) Histogram of the terminal wealths of the RL method.



(b) Histogram of the terminal wealths of the CP method.

Figure 8 – Histograms of terminal wealths of the long-term portfolio techniques.

The empirical cumulative distribution functions (CDF) are constructed for the CP and RL method from histograms of Figures 8b and 8a and are shown in Figure 9. At first sight one cannot detect any large difference in the histograms except for the large tail on the right of the CP method, therefore, the CDFs are created. One can not detect stochastic dominance when a certain CDF is smaller than or equal to other CDF for all fractions. Although it does not differ much, the loss of the CP model, with a higher density for the region with low terminal wealth, is compromised by several high terminal wealth values. If, for example, RL were to have a smaller CDF as a whole and therefore stochastically dominate then it would always have a higher expected terminal wealth. This is only the case for 72.4% of the dataset for terminal wealth values under the threshold of 1.43. Therefore, RL second-order stochastically dominates for a threshold of terminal wealth equal to 1.43.

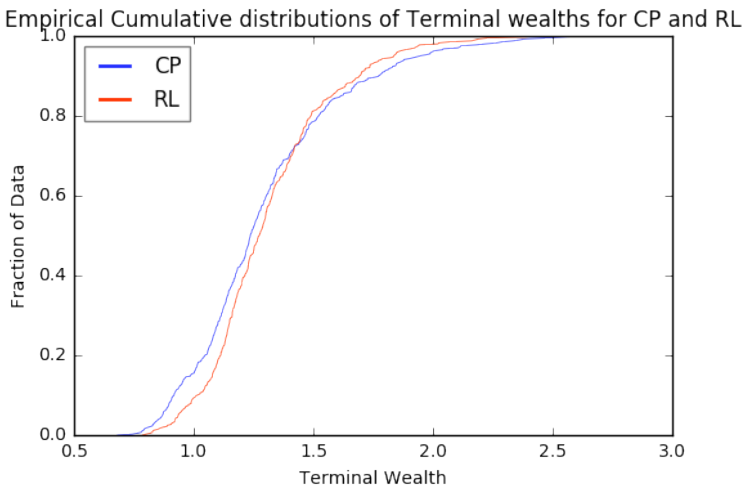
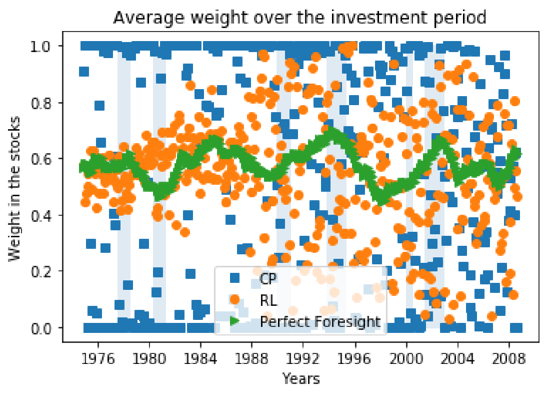
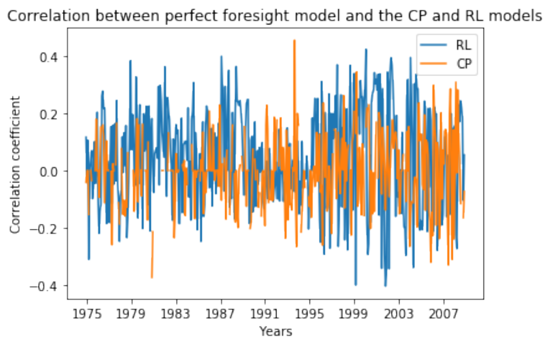


Figure 9 – Empirical CDF for terminal wealths of the CP with $\gamma = 2$ and the RL-NN method.

When one looks into the average weights which construct the terminal wealth values of Figure 7, then we can see in Figure 10a the average weights of three models: perfect foresight, CP, and RL. The perfect foresight does not differ much relative to the previous weight because the perfect strategy differs only by the latest introduced time period. For CP and RL, the weights differ much more from period to period. This is because of the re-instantiation of the models each period. One can see the stated conservativeness of the RL-NN method compared to the CP. The CP method regularly takes a full stance in one of the assets for the full period of five years. This is not something a practitioner would prefer to do. Another observation is that the RL method grows in the standard deviation of the average weights over time. This could be explained by the fact that the RL method has learned more from distinctive events, for example, business cycles. When comparing the full duration of 60 periods of the weights instead of the average weights one cannot conclude whether a method is strictly better in terms of being able to time the market. Figure 10b shows that on overall the correlation with the perfect foresight for both models is quite low, between -0.3 and 0.3, and similar for both models. Otherwise, 60% of the time the correlation between the perfect timing and the RL method is higher than the correlation of the perfect timing and the CP method. Besides that, on average, the correlation for the RL method is 0.04 compared to the -0.001 of the CP method, which is not remarkably high.



(a) Average weight for the investment period of 60 months for the following methods: Perfect foresight, CP with $\gamma = 2$, and RL-NN.



(b) Time series of the correlation coefficients for the CP and RL method with the perfect foresight.

Figure 10 – Graphical representations of the weights characteristics for the CP and RL models corresponding to the time series of Terminal Wealths shown in Figure 7.

5 Conclusion and Discussion

Compared to the classical portfolio management models the reinforcement learning model shows great resemblance with a slightly risk-averse investor (with a γ of 2) in terms of average terminal wealth while having a lower standard deviation and lower turnover.

The RL model rarely takes a position fully in the stocks or in the bonds, and therefore is able to lose less compared to the CP model when a drop in the stock prices happens unexpectedly. Also, the RL model is much faster in responding to certain events while the CP model keeps high weights in the stocks while the downfall of the stock index is started. However, when the RL takes a full stance on either side, one could argue that the method has more profound reasons for a strong up or down market movement.

In the simulated environment of the CER, it is expected from the RL method to always take a full stance in the stocks, however, sometimes during the whole runtime the weight in stocks is only two-thirds. This is probably due to the five percent of the dataset in which the returns of the bonds are higher than the returns on the stocks. A possible explanation for this is that the randomizing of input has such a large influence on the decision of the weights. This can again be shown by the case where the data is simulated by the CER model, in this dataset in 5% of the time the bond returns are higher than the stock returns. When a few of this 5% of the data points is a point in the last couple of training rounds of the RL model, it will result in weights of 0.78 for the whole period to estimate. This makes the method more fault-proof due to the randomizing of the input. This could be adjusted by giving the model the input chronologically with an experience replay. Experience replay returns a random input from the past, which would improve the accuracy of the model but not overfit on the most recent input. Experience replay is a technique used in order for the RL method to not overfit on the most recent input while not ignoring its past. Wawrzyński and Tanwani [2013] consider this method for reinforcement learning.

Neural networks should not be too complex in this case study, otherwise, it does not generalize anything. But will only replicate previously seen state-action and reward combinations. In the context of finance, this is a bad practice, because it rarely happens that the past is a good predictor of the future. Instead of choosing a smaller neural network, one can also choose to insert dropout in between the hidden layers [Srivastava et al., 2014]. Although it is counterintuitive to drop several neurons, the results show that it greatly reduces overfitting of the neural network.

More iterations of the neural network to train improve the results, while there is a tipping point. This tipping point is when the network is overfitting the data. Ideally, in the case of finance, one would like to have no epochs, because all returns and their historical returns are unique and non-repetitive. This could be achieved for assets which have a higher frequency, for example, digital currencies or individual stocks traded through the stock exchange. It would be very informative to further investigate assets with various frequencies and more assets than the three stated in this research.

Figure 7 strongly indicates that there is not much more information in the dataset at hand due to the state-of-the-art RL methods performing almost equally well in comparison to the CP method while being unable to beat the full stock model-free method in terms of terminal wealth. This should not come as a big surprise because the data used are limited. However, this is a conscious choice because there would be an unlimited amount of other factors influencing the problem statement if comparing these methods with more data. I would suggest other kinds of data than solely stock returns. One needs to search for data that is not already incorporated in the expected returns thanks to the efficient market hypothesis. For example, sentiment data for the specific asset gathered from Twitter or sentiment analysis bureaus.

Machine learning methods work well in the case where plentiful data are available and therefore also much information is available in the data. When the data are scarce it is not possible to get informative data. Implementing a naive machine learning method when only considering simple time series would fall short. It also introduces new challenges, together with the extremely tedious job of the finetuning of the many hyperparameters with the machine learning methods. For the hyperparameters selection, a genetic algorithm could be used with the inverse of the loss after training as the fitness function.

Instead of using an expanding window, which proved to be more useful than a moving window by Diris et al. [2015], it would also be informative to look at the moving window cases for the reinforcement learning methods. This, however, has the disadvantage of fewer data which is very necessary for this kind of machine learning methods.

Actually, the RL is a myopic solution to the problem statement in Section 3 because the function approximation maps current states to what action to do next based on the maximum immediate reward it will receive next period instead of looking at what reward it will receive over the rest of the investment period. Although Diris et al. [2015] show that there is almost no difference between the myopic and dynamic solution of long-term portfolio management in that research it would be informative to rewrite the optimization function to also conduct a correct long-term strategy.

Another possible field of investigation would be the choice of predictors in both the underlying model for the assets for the CP method and the predictors in the function approximation for the RL method. A good starting point would be Pesaran and Timmermann [1995], this paper investigates whether single variables are good predictors for the US stock market of the late 20th century. Because it also concludes that different models hold different information and it would be informative to combine several forecasts. This reasoning in its turn also supports the idea of a neural network as function approximator because, like stated before, neural networks can represent any abstract relationship between the predictors.

References

M.W. Brandt, A. Goyal, P. Santa-Clara, and J.R. Stroud. A simulation approach to dynamic portfolio choice with an application to learning about return predictability. *The Review of Financial Studies 1*, 18(3), 2005. doi: "10.1093/rfs/hhi019". URL "`http://www.nber.org/papers/w10934`".

J.Y. Campbell and L.M. Viceira. *Strategic Asset Allocation: Portfolio Choice for Long-Term Investors*. Clarendon Lectures in Economics, 2001. URL "`https://faculty.fuqua.duke.edu/~charvey/Teaching/BA453_2006/Campbell_Viceira.pdf`".

J.H. Cochrane. New facts in finance. 1999. URL "`http://www.nber.org/papers/w7169`".

V. DeMiguel, L. Garlappi, and R. Uppal. Optimal versus naive diversification: How inefficient is the 1/n portfolio strategy? *The Review of Financial Studies*, 22, 2009. doi: "10.1093/rfs/hhm075". URL "`http://faculty.london.edu/avmiguel/DeMiguel-Garlappi-Uppal-RFS.pdf`".

B. Diris, F. Palm, and P. Schotman. Long-term strategic asset allocation: An out-of-sample evaluation. *Management Science*, 61(9):2185–2202, 2015. doi: "http://dx.doi.org/10.1287/mnsc.2014.1924". URL "`http://arno.uvt.nl/show.cgi?fid=93151`".

X. Du, J. Zhai, and K. Lv. Algorithm trading using q-learning and recurrent reinforcement learning. *CS229*, 2016. URL "`http://cs229.stanford.edu/proj2009/LvDuZhai.pdf`".

E.F. Fama. Efficient capital markets: A review of theory and empirical work. *The Journal of Finance*, 25, 1970. doi: "10.1111/j.1540-6261.1970.tb00518.x". URL "`http://www.jstor.org/stable/2325486`".

N. Gârleanu and L.H. Pedersen. Dynamic trading with predictable returns and transaction costs. *The Journal of Finance*, 67, 2013. doi: "10.1111/jofi.12080". URL "`http://docs.lhpedersen.com/DynamicTrading.pdf`".

H. Hasselt, A. Guez, M. Hessel, V. Mnih, and D. Silver. Learning values across many orders of magnitude. *Google Deepmind*, 2016. URL "`https://papers.nips.cc/paper/6076-learning-values-across-many-orders-of-magnitude.pdf`".

T. Hens and P. Woehrmann. Strategic asset allocation and market timing: A reinforcement learning approach. *Computational Economics*, 29(3):369–381, 2007. doi: "10.1007/s10614-006-9064-0". URL "`https://link.springer.com/content/pdf/10.1007/s10614-006-9064-0.pdf`".

T. Jaakkola, M.I. Jordan, and S.P. Singh. Convergence of stochastic iterative dynamic programming algorithms. *Neural Computation*, 1994. URL "`https://web.eecs.umich.edu/~baveja/Papers/Neural-Computation-94.pdf`".

Z. Jiang and J. Liang. Cryptocurrency portfolio management with deep reinforcement learning. 2017. URL "`http://arxiv.org/abs/1612.01277v5`".

O. Jin and H. El-Saawy. Portfolio management using reinforcement learning. 2017. URL "`http://cs229.stanford.edu/proj2016/report/JinElSaawy-PortfolioManagementusingReinforcementLearning-report.pdf`".

Y. Li. Deep reinforcement learning: an overview. 2017. URL "`https://arxiv.org/pdf/1701.07274.pdf`".

F.S. Melo. Convergence of q-learning: a simple proof. 2001. URL "`http://users.isr.ist.utl.pt/~mtjspaan/readingGroup/ProofQlearning.pdf`".

C. Metz. The rise of the artificially intelligent hedge fund, 2016. URL "`https://www.wired.com/2016/01/the-rise-of-the-artificially-intelligent-hedge-fund/`". Accessed: 2017-21-10.

J. Moody, L. Wu, Y. Liao, and M. Saffell. Performance functions and reinforcement learning for trading systems and portfolios. *Journal of Forecasting*, 17:441–470, 1998. doi: ”10.1.1.87.8437”. URL "http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.87.8437&rep=rep1&type=pdf".

A. Ng, D. Harada, and S. Russell. Policy invariance under reward transformations: Theory and applications to reward shaping. *Data Bibliometrics*, 1999. URL "http://www.robotics.stanford.edu/~ang/papers/shaping-icml99.pdf".

C. Noriega and O. Ballinas. Global pension funds: Best practices in the pension funds investment process. *PricewaterhouseCoopers Luxembourg*, 2016. URL "https://www.pwc.lu/en/asset-management/docs/pwc-awm-global-pension-funds.pdf".

M. Pesaran and A. Timmermann. Predictability of stock returns: Robustness and economic significance. 1995. URL "http://www.jstor.org/stable/2329349".

J. Schindler, A. de Souza Moraes, and S. Li. Artificial intelligence and machine learning in financial services. 2017. URL "http://www.fsb.org/wp-content/uploads/P011117.pdf".

N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15, 2014. URL "http://jmlr.org/papers/v15/srivastava14a.html".

R.S. Sutton and A.G. Barto. *Introduction to Reinforcement learning*. MIT press, 1st edition, 1998. URL "http://neuro.bstu.by/ai/RL-3.pdf".

D. Svozil, V. Kvasnicka, and J. Pospíchal. Introduction to multi-layer feed-forward neural networks. *Chemometrics and Intelligent Laboratory Systems*, 39:43–62, 1997. doi: ”10.1016/S0169-7439(97)00061-0”. URL "http://www.sciencedirect.com/science/article/pii/S0169743997000610".

P. Wawrzyński and A.K. Tanwani. Autonomous reinforcement learning with experience replay. *Neural Networks*, 2013. URL "http://www.sciencedirect.com/science/article/pii/S0893608012002936".

Appendix

The exact parameters and their description of the neural network which functions as a function approximator for the RL method and the variables for the CP method.

Table 8 – Hyperparameters for the different models stated in this research paper.

Hyperparameters	**Value**	**Description**
	Classical Portfolio Methods	
Periods (K)	60	The amount of months to take into account in the future.
Window	Expanding	The model should have a certain data points as history to estimate its Vector Autoregressive(VAR) Model on, due to the proven dominance over an expanding window over a moving window is chosen [Diris et al., 2015].
Simulations	400	From the VAR model simulated paths of future returns are simulated in order to determine the path with the best utility.
	Reinforcement learning Methods	
Number of lags	10	As input for the function approximation the last 10 lags are given to estimate the next state action value one period ahead of time.
The number of assets	3	For each asset(risk-free rate, bond, and stock) the number of lags is provided for the function approximation.
Number of actions	10	The amount of actions the agent could take is 10, this is a grid of possible weights in the stocks from ranging from 0 to 1.
Structure Neural Network	(20,45,10)	The Neural Network used as function approximation is structured with one input layer, one hidden layer, and one output layer. The number of nodes per layer is stated in the brackets.
Activation function	softmax	At the end of the neural network, in the output layer the values are transformed with a softmax activation function in order to convert the values of the network into probabilities for the best state.
Learning rate (Neural Network)	0.1	The learning rate for the Gradient Descent Optimizer which updates the neural network.
Learning rate (Q value)	0.01	The learning rate to update the new Q value to, to train the function approximation on.
Epochs	20	The number of times the dataset is given (randomized) to the Reinforcement Learning method in order to train on the past, this is kept low because normally in Finance the same past returns and future returns are rarely the same.
Epsilon (ε)	0.1	A random real number is chosen between 0 and 1, if it is smaller than 0.1 the action to take by the reinforcement learning method is also randomized to improve exploration otherwise the maximum of the output of the function approximation is taken.