Expert Systems With Applications 103 (2018) 1–13

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Trading financial indices with reinforcement learning agents

Parag C. Pendharkar*, Patrick Cusatis

School of Business Administration, Pennsylvania State University at Harrisburg, 777 West Harrisburg Pike, Middletown, PA 17057, United States

ARTICLE INFO

Article history:
Received 16 July 2017
Revised 22 February 2018
Accepted 23 February 2018
Available online 6 March 2018

Keywords:
Reinforcement learning
Multi-agent systems
Markov decision process
Portfolio management

ABSTRACT

Intelligent agents are often used in professional portfolio management. The use of intelligent agents in personal retirement portfolio management is not investigated in the past. In this research, we consider a two-asset personal retirement portfolio and propose several reinforcement learning agents for trading portfolio assets. In particular, we design an on-policy SARSA (λ) and an off-policy Q(λ) discrete state and discrete action agents that maximize either portfolio returns or differential Sharpe ratios. Additionally, we design a temporal-difference learning, TD(λ), agent that uses a linear valuation function in discrete state and continuous action settings. Using two different two-asset portfolios, the first asset being the S&P 500 Index and the second asset being either a broad bond market index or a 10-year U.S. Treasury note (T-note), we test the performance of different agents on different holdout (test) samples. The results of our experiments indicate that the high-learning frequency (i.e., adaptive learning) TD(λ) agent consistently beats both the single asset stock and bond cumulative returns by a significant margin.

1. Introduction

There are extensive studies in the finance literature that deal with trading strategies. Early studies involve the use of filter rules to determine when to buy or sell a stock and conclude that a buy and hold strategy dominates trading strategies based on filters (Alexander, 1961; Fama & Blume, 1966). Other studies focus on momentum and contrarian strategies. Momentum trading strategies assume that winners will continue to be winners, and losers will continue to be losers. Jegadeesh and Titman (1993, 2001) find that momentum trading produces abnormal trading profits. Contrarian strategies seek to exploit market overreactions by buying losers and selling winners. A special case of the contrarian strategy is called pairs trading. Pairs trading involves identifying pairs of stocks that are close substitutes and buying the losers and selling the winners of the pairs. Do and Faff (2010, 2012) and Gatev, Goetzmann, and Geert Rouwenhorst (2006) find significant gains from using a pairs trading strategy.

In addition to trading strategies, intelligent systems are often used in dynamic control problems including financial investing (Almahdi & Yang, 2017). These dynamic investment decisions are often modeled as a Markov process. Peck and Yang (2011) use a Markov decision process (MDP) to model investment returns. In their model, investors in each period observe prior investment decisions. Based on this history, the investors make their future investment decisions. Zhang (2001) presents an investing model that identifies an optimal selling rule for stocks based on a finite-state Markov process. MDP has been applied to a variety of applications in finance (Bauerle & Rieder, 2011).

The opponents of MDP argue that it is rare that complex modern financial markets are truly Markovian (Nevmyvaka, Feng, & Kearns, 2006). Thus, in financial securities' trading and investment decision making, it is rare to have prior knowledge of a stochastic transition model and any knowledge of the reward function. In such cases, using the traditional MDP model poses a few challenges because a partially-observable environment is treated as a fully-observable environment (Nevmyvaka et al., 2006). These challenges can be handled by breaking down the MDP model into two stages. In the first stage, an algorithm can be designed to learn the state transition model and a reward function. In the second stage, the MDP can be solved to learn policy mapping. Alternately, the policy can be learned directly through trial and error. A variety of algorithms exist to learn policies using trial-and-error, *model-free* approaches. These algorithms fall into a broad category known as reinforcement learning (RL) or adaptive or approximate (Kara & Dogan, 2018; Li & Womer, 2015; Ohno, Boh, Nakade, & Tamura, 2016) dynamic programming. Over the last two decades, a number of studies have applied RL to financial trade execution (Bertoluzzo & Corazza, 2012; Bertsimas & Lo, 1998; Moody & Saffell, 2001; Moody et al., 1998; Nevmyvaka et al., 2006).

There is a special class of problems broadly defined as multi-armed bandit (casino slot machine) problems. In these problems, a player may observe a current state (earnings) and select an action (an arm to pull among many arms at a slot machine), but the

* Corresponding author.
E-mail addresses: pxp19@psu.edu (P.C. Pendharkar), pjc101@psu.edu (P. Cusatis).

https://doi.org/10.1016/j.eswa.2018.02.032
0957-4174/

future reward of the action and its probability distribution are unknown to the player and are independent of the player's actions. We assume that our individual two-asset investment problem falls into the multi-arm bandit class of problems, where the probability distribution of the reward of an investor's action is assumed to be unknown. Algorithms designed to solve multi-arm bandit problems have to balance the exploration–exploitation tradeoff (Kuleshov & Precup, 2000). Exploitation occurs when an investor uses trading strategies that provided the largest gains in the past, and exploration occurs when new actions are attempted to learn previously unknown strategies to maximize gains. Kuleshov and Precup (2000) argue that simple heuristics, such as ε-greedy and Boltzmann exploration, outperform other theoretically sound algorithms by significant margins. For the multi-arm bandit class of problems, the performance of algorithms is sensitive to the choice of parameters that manage exploration and exploitation tradeoffs, and these parameters must be chosen carefully to obtain superior results (Kuleshov & Precup, 2000).

Traditionally, researchers have used RL agents for professional portfolio management (Almahdi & Yang, 2017) and strategic foreign exchange asset allocations (Dempster & Leemans, 2006). The current research focuses on the application of RL agents for individual retirement portfolio management. Professional portfolio management (e.g., managing active mutual funds or hedge fund portfolios) involves different sets of tools and strategies. Professional portfolio managers have access to real-time information, can make trades at significant discounts and can make frequent trades at a fraction of a second (Bertsimas & Lo, 1998). Individual retirement portfolio management is a different problem because individuals are often restricted from making frequent trades by their brokerages. Additionally, information access and the number of assets in an individual portfolio are also limited. We assume a simple retirement portfolio containing two asset classes, one containing S&P 500 Index fund/ETF and another asset class containing either AGG Bond Index or 10 year US Treasury note, and illustrate that RL agents can be successfully used by individuals to manage their retirement portfolios. We propose and use a model-free RL agent to learn retirement portfolio trading strategies using major stock and bond indices/U.S. Treasury note (T-note) data. We assume that the first security in the retirement portfolio is a stock market security, which is an exchange trade fund (ETF) or an index mutual fund (IMF) that represents the return on the S&P 500 Index (S&P 500). The second security is a bond market security, which is either an ETF or IMF that mimics the return on the Barclays Capital U.S. Aggregate Bond Index (AGG) or a 10-year U.S. T-note. The investment objective is to determine a trading strategy that maximizes either portfolio returns or differential Sharpe ratios over long investment periods of 10 years or longer. We assume that the two-asset portfolio is reallocated exactly once per fixed-time trading period using information on the last trading day of the trading period. On this day, an investor uses the information on returns of the S&P 500 ETF and the bond assets (AGG ETF or T-note) to determine the portfolio allocation for the next trading period. Fixed trading periods in our research may be quarterly, semi-annual or annual. In their decision-making, an investor only considers whether previous trading period returns of the S&P 500 and the bond asset (AGG or T-note) are positive or negative, and does not consider the magnitude of these returns. We use the S&P 500 and the AGG (introduced in 1973) data for the years 1976–2016. For our experiments with T-note data, we use 46 years of data from years beginning in 1970 and ending in 2016. The use of T-notes allows us to obtain more finely grained return data (quarterly and semi-annual) over a longer time horizon, since, unlike the AGG, quarterly data is available. Thus, our experiments with the S&P 500 Index and the AGG portfolio assume annual trading. In contrast, our experiments with the S&P 500 and the T-note portfolio allow us to use three different trading periods: quarterly, semi-annual and annual.

We divide our datasets into two parts—training and holdout/test datasets. We learn trading strategies using the training dataset and apply them to the holdout dataset to monitor their performance. Since we use a portfolio of only two assets, the yearly return on our portfolio is a convex combination of the returns on the two assets. Thus, finding a trading strategy that uses two indices and beats the cumulative performances of both indices over a decade is no trivial task. If such a trading strategy can be found then it will be efficient on the risk-return efficient frontier since it will provide a higher return at a lower level of risk. Our RL algorithms use an $\varepsilon-$greedy strategy, and we experiment with different performance parameters to manage the exploration–exploitation tradeoff.

The rest of our paper is organized as follows: In the next section, we review some of the studies that applied RL for trading financial securities. In Section 3, we define the stochastic decision process model that applies to our research problem and introduce different reinforcement learning agents used in our research. In Section 4, we describe our data and report the results of our experiments. We provide our conclusions and propose directions for future work in Section 5.

2. Reinforcement learning applications for stock trade executions

RL is a type of learning that is used for sequential decision-making problems (Sutton & Barto, 1998). An RL agent recognizes different states and takes an action where it receives a feedback (reward) and then it learns to adjust its actions to maximize its future rewards. RL algorithms have been previously used in quantitative finance for optimal execution of trades (Almahdi & Yang, 2017). Part of the interest in the application of RL algorithms in finance is due to the application of automated agents that process real time high frequency microstructure data (millisecond time scale) to execute trades. Bertsimas and Lo (1998) study an application of RL (also known as adaptive dynamic programming) for trading large equity blocks over a fixed finite number of time periods so that the expected cost of executing trades is minimized. Bertsimas and Lo (1998) look at equity trading from institutional investors' point of view because these investors execute high volume trades over the course of several days. They define best execution as a strategy that unfolds over several days. Because current trading affects current equity prices which in turn affect future trading costs, trading strategies must adapt to changing market conditions. Naïve strategies, such as equally dividing the sale (or purchase) of a block of shares over a fixed time interval or selling (or purchasing) all shares at once on the first day, are generally not optimal, because equity purchases constantly impact equity prices. In their Monte Carlo experiments, Bertsimas and Lo (1998) find that the RL algorithm strategy saved between 25% and 40% in execution costs when compared to the naïve strategy of trading in equal-size lots. The primary limitation of this study is the assumption that the volume of each buy trade is still large enough to increase the price of the traded security, excluding any random noise in security prices. This assumption of an institutional investor influencing security prices implicitly assumes that the security markets are small because for large markets security prices are beyond the control of an individual investor (Hildenbrand and Kirman, 1988).

Nevmyvaka et al. (2006) use an RL algorithm for optimizing trade execution using 1.5 years of millisecond time-scale limit order data from companies that trade on the NASDAQ. In limit orders, buyers and sellers specify prices at which they will buy or sell a security. In these cases, trade-offs may arise due to speed of