Expert Systems with Applications 163 (2021) 113761

Contents lists available at ScienceDirect

Expert Systems with Applications

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

A Q-learning agent for automated trading in equity stock markets

Jagdish Bhagwan Chakole [a,*], Mugdha S. Kolhe [a], Grishma D. Mahapurush [a], Anushka Yadav [b], Manish P. Kurhekar [a]

[a] Department of Computer Science and Engineering, Visvesvaraya National Institute of Technology, Nagpur, India
[b] Department of Computer Science and Engineering, Indian Institute of Information Technology, Nagpur, India

ARTICLE INFO

Article history:
Received 28 February 2020
Revised 22 June 2020
Accepted 12 July 2020
Available online 2 August 2020

Keywords:
Reinforcement Learning
Algorithmic trading
Stock market

ABSTRACT

Trading strategies play a vital role in Algorithmic trading, a computer program that takes and executes automated trading decisions in the stock market. The conventional wisdom is that the same trading strategy is not profitable for all stocks all the time. The selection of a trading strategy for the stock at a particular time instant is the major research problem in the stock market trading. An optimal dynamic trading strategy generated from the current pattern of the stock price trend can attempt to solve this problem. Reinforcement Learning can find this optimal dynamic trading strategy by interacting with the actual stock market as its environment. The representation of the state of the environment is crucial for performance. We have proposed two different ways to represent the discrete states of the environment. In this work, we trained the trading agent using the Q-learning algorithm of Reinforcement Learning to find optimal dynamic trading strategies. We experimented with the two proposed models on real stock market data from the Indian and American stock markets. The proposed models outperformed the Buy-and-Hold and Decision-Tree based trading strategy in terms of profitability.

1. Introduction

Trading in the stock market is an attractive field for people from various domains ranging from professionals to laymen in the hope of lucrative profits. It is not so easy to earn profit from trading in the stock market because of the uncertainty involved in the stock price trend. Traders' emotion is also one of the reasons for losses in stock market trading. The conventional wisdom is that the stock market trading is the game of GREED and FEAR. The trend and the price of stock changes so frequently that a human trader cannot react instantly every time (Chia, Lim, Ong, & Teh, 2015; Huang, Wang, Tao, & Li, 2014).

We can replace the human trader with a computer program. The program will trade as per trading logic as written by the programmer, and this process of automated trading is called Algorithmic Trading (AT) (Treleaven, Galas, & Lalchand, 2013; Yadav, 2015). AT eliminates human emotions, and also the time required in AT for making trading decisions, and executing the taken trading decision is far less compared to a human trader (Andriosopoulos, Doumpos, Pardalos, & Zopounidis, 2019; Meng & Khushi, 2019). Trading decisions can be Buy, Sell, or Hold (do nothing) stock of companies registered on the stock exchange. The stock exchange is the platform where traders can buy or sell the stock. Generally, stock exchanges are under the regulations of the Government in many countries.

The trader does the stock trading for profit, which depends on the timing of buying or selling. Rules which determine when to sell, buy, or hold the stock are called trading strategies. There are many trading strategies available, for example, Mean reversion (Miller, Muthuswamy, & Whaley, 1994). The same predefined trading strategy is not always profitable as it might be profitable in one of the trends out of uptrend, downtrend, or sideways trend but not in all trends. One of the significant research problems in stock trading is the selection of the best trading strategy from the pool of trading strategies at a specific instant of time. It is helpful to select a strategy if we have some predictive ideas about the future trend of the stock price.

The prediction of future trend or price of a stock is also a research problem. The stock price of any company in the stock market depends on diverse aspects like the financial condition of the company, the company's management, policy of the government, competition from other companies, natural disasters, and many more (Fischer, 2018). Many of these aspects are uncertain aspects like the government can hike tax on petrol to promote the use of electric vehicles. News about a company can cause a sudden rise or fall in the price of the stock of the company. As

* Corresponding author.
E-mail addresses: jagdish06@students.vnit.ac.in (J.B. Chakole), manishkurhekar@cse.vnit.ac.in (M.P. Kurhekar).

https://doi.org/10.1016/j.eswa.2020.113761
0957-4174/

the price of the stock depends on news, long-duration (Week, Month, or Year) prediction of price or trend of the stock is not reliable. Whereas the prediction methods can work well if the duration of prediction is small (Minute, Hour, or a Day) as chances of big impacting news related to the company are reduced (Zhu, Yang, Jiang, & Huang, 2018).

Trading strategies based on predictions are mostly static in nature (James et al., 2003; Fong, Si, & Tai, 2012; Hu, Feng, Zhang, Ngai, & Liu, 2015). In a static strategy, once a strategy is decided and deployed for trading, it remains unchanged for the entire trading period. Static trading strategies have a high risk as the trends in the stock market are uncertain and change very frequently. An ideal trading strategy should be dynamic, and it should be capable of changing its trading decision as per the changes in the stock trend. So, instead of using any predefined strategy, we should use a self-improving trading strategy from the current stock market behaviour (Environment). Reinforcement Learning (RL) can develop such a self-improving trading strategy. In RL, the learning base on the rewards received from the environment (Sutton & Barto, 2018; Moody, Wu, Liao, & Saffell, 1998; Gao & Chan, 2000; Du, Zhai, & Lv, 2016).

In this research paper, our objective is to develop a self-improving agent-based Algorithmic trading model (a computer program), that will find a dynamic trading strategy using Reinforcement Learning (RL) from the current stock market patterns. Once the strategy found, it will be deployed for stock trading and will be updated to incorporate the changes in the stock market patterns. In this study, we have used a model-free off-policy RL algorithm called as Q-learning (García-Galicia, Carsteanu, & Clempner, 2019; Lee, Park, Jangmin, Lee, & Hong, 2007). The proposed trading model will help algorithmic traders for short term trading. We experimented with the proposed model on various individual stocks from the Indian stock market and also tested on the index stocks of the Indian and the American stock market. In this paper, we propose an innovative way to use unsupervised learning method k-Means to represent the behaviour of the stock market (environment) using a finite number of states. Grouping historical information using unsupervised learning methods to represent the behaviour of the stock market (environment) using a finite number of states for trading using Reinforcement Learning (RL) is key to our proposed work. No work is available for the Indian Equity stock market using the Q-learning method of RL with a finite number of discrete states of the environment.

The RL agent to handle the financial portfolio is proposed by Garcia-Galicia et al. (2019). The RL agent to improve the trading strategy based on the box theory proposed by Zhu et al. (2018). The author Lee et al. (2007) proposed use of multiple RL agents for trading. The authors in Park, Sim, and Choi (2020) proposed deep Q-learning based trading strategy for a financial portfolio with continuos size state space. They experimented with their proposed model on US Portfolio and Korean Portfolio data. The authors in Calabuig, Falciani, and Sánchez-Pérez (2020) proposed a Reinforcement Learning model for financial markets using Lipschitz extensions for a reward function. They produce some more states artificially for better learning of the model. The authors in Wu et al. (2020) proposed a stock trading strategy using deep reinforcement learning methods and Gated Recurrent Unit used to capture significant features from a raw financial dataset. They claimed that the proposed model outperforms the DRL trading strategy and Turtle trading strategy.

All the models which use deep reinforcement learning have continuos and infinite state space to represents the behaviour of the environment. For a human trader, the state-action mapping of such a model is a black box, whereas, in our proposed models, this is not the case as we are using finite states. The author Pendharkar and Cusatis (2018) recommended discrete RL agents to do automated trading for a personal retirement portfolio. The authors in Yun, Lee, Kang, and Seok (2020) proposed a two-stage deep learning framework for the management of the portfolio. And used cost function to address absolute and relative return. Their model outperforms ordinary deep learning models. The authors in Alimoradi and Kashan (2018) proposed a method for extracting stock trading rules using League Championship Algorithm and backward Q-learning. They claimed that their proposed model outperforms Buy-and-Hold and GNP-RA methods.

The rest of the paper organised as follows: Section 2 describes methodologies used like Q-learning. Section 3 tells details of the proposed models and also provides background for the proposed models, and Section 4 discussed the experimental results. The conclusion the paper is described lastly in Section 5.

2. Methodology

This section briefly explains the methodology used in this work, including Markov Decision Process, Reinforcement Learning and its Q-learning method, Technical analysis of stocks.

2.1. Markov Decision Process

Markov Decision Process (MDP) uses to define a sequential decision-making problem having uncertainty. It defines a problem that learns by interacting with the environment to find the target state. In MDP all state follows Markov property which says that next state s_{t+1} only depends on present state s_t. Markov process is a chain of random states having Markov property. An MDP formulate the problem as a tuple (A, S, R, Z, γ), where

- A: Finite set of actions
- S: Set of Markov states.
- Reward function R is used to calculate the return.
  $R(s,a) = \mathbb{E}(R_{t+1}|s_t = s, a_t = a)$
- Z: Probability matrix for state transition,
  $Z^a_{ss\prime} = P(s_{t+1} = s\prime|s_t = s, A_t = a)$
- $\gamma \in (0,1)$, is a Discount factor.

A policy π govern the behaviour of an agent and it selects action for the provided state,

$$\pi(a|s) = P(A_t = a|S_t = s) \tag{1}$$

The rewards are of two types, immediate reward as defined by the function R and the return (or total accumulated rewards) X_t is the total discounted reward from the time instant t as formulated by Eq. (2).

$$X_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{2}$$

where, $\gamma \in (0,1)$, it uses to discount future rewards and to avoid infinite returns in cyclic Markov processes. The total accumulated rewards depend on the sequence of the action taken. The selection of proper reward function is important for better performance of the model.

The expected return gain at time instant t by beginning from state s, taking action a, and later following policy π is provided by state-action-value function $Q_\pi(s,a)$.

$$Q_\pi(s,a) = \mathbb{E}_\pi[X_t|s_t = s, a_t = a] \tag{3}$$

The Bellman equation can decompose the state-action-value function into immediate reward plus discounted future value of the state-action-value function of the successor state.

$$Q_\pi(s,a) = \mathbb{E}_\pi(R_{t+1} + \gamma Q_\pi(s_{t+1}, a_{t+1})|s_t = s, a_t = a) \tag{4}$$

The optimal state-action-value function $Q_*(s,a)$ is the maximum value state-action-value function over all policies

$$Q_*(s,a) = \max_{\pi} Q_\pi(s,a) \tag{5}$$

The optimal policy π_* is a policy which is the best among all the policies in terms of total accumulated rewards.

2.2. Reinforcement Learning

Typically Reinforcement Learning (RL) consists of an agent and the environment, as depicted in Fig. 1 and the agent's objective is to get maximum total accumulated rewards. A set of states can describe the environment, and the agent can take action on the environment from a set of actions at any time instant (for example Buy, Sell, or Hold is an action set for the trading agent). The agent desires an environment's particular state that the state is called a goal state. In the beginning, the environment may be in a particular state known as the initial state. The agent acts on the environment as per its policy π. The environment goes into the next state, and the agent gets a reward. This process started with the initial state and continued until the next state is the desired state.

The agent obtains its policy π from interaction with the environment. It is a mapping of state to action. The agent observes the environment using the state s_t. A state s_t is the status of the environment at time t. So a set of states S can represent the environment. The agent takes action a_t on the environment at instant t from the set of actions A and in response, it gets a delayed reward R_{t+1}. Also the state of the environment updates to s_{t+1} as shown in Fig. 1. Reward either positive or negative and defined by the reward function R as defined in Section 2.1 and total accumulated rewards called return is defined in Eq. (2), (Sutton & Barto, 2018; Gorse, 2011).

The policy is a mapping function for state to action, it selects optimal action for a state, and it is updated using rewards got from the environment for the action taken on the state. The optimal action is that action among all possible actions which gives the maximum reward. The policy is arbitrarily initialized at the beginning and is updated based on rewards received from the environment. So the RL agent's objective is to obtain an optimal policy. The policy which gives maximum total accumulated reward among all policies that policy termed as optimal policy. The agent selects the chain of actions to reach the desired state, so this is the sequential decision-making problem. An MDP defines this sequential decision-making problem.

2.3. Q-learning

It is a model-free off-policy RL method in which the agent aims to obtain the optimal state-action-value function $Q_*(s,a)$ by interacting with the environment. It maintains a state-action table Q[S, A] called as Q-table containing Q-values for every state-action pair.

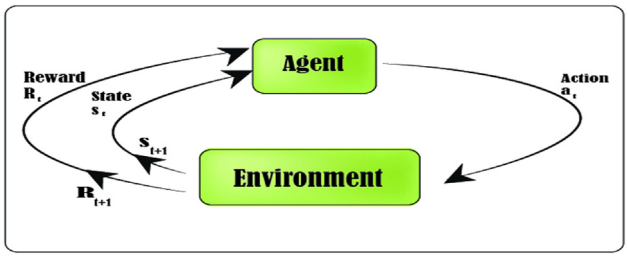


Fig. 1. Typical Reinforcement Learning system.

At the start, Q-values initialize to zeroes. Q-learning updates the Q-values using Temporal Difference method.

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha(R_{t+1} + \gamma \max_a Q(s_{t+1},a) - Q(s_t,a_t)) \tag{6}$$

where α is the learning rate and $\gamma \in [0,1]$. $Q(s_t,a_t)$ is the actual Q-value for state-action pair (s_t,a_t). The target Q-value for state-action pair (s_t,a_t) is $R_{t+1} + \gamma \max_a Q(s_{t+1},a)$ i.e. immediate reward plus discounted Q-value of next state. The table converges using iterative updates for each state-action pair. To efficiently converge the Q-table ϵ - greedy approach is used.

The ϵ - greedy approach: At the starting of the training, Q-values of the Q-table initialize to zeroes. It means all actions for a state have the same chance to be selected. So to converge the Q-table using iterative updates, exploration–exploitation trade-off is used. The exploration update the Q-value of random state-action pair (s_t,a_t) of the Q-table by randomly selecting the action. The exploitation selects greedy action (a_t) for the state (s_t) from the Q-table having maximum rewards.

So, to converge Q-table from an initial condition (all Q-values of the Q-table initialize to zeroes), at the initial level of iterative update exploration is needed and at a later time of the iterative updates, requires more exploitation. It can use probability ϵ, in which random action selected using a probability ϵ, and action is chosen from the Q-table using probability $1-\epsilon$. At the beginning, the value of ϵ is one, and it reduces with time, and once the table converges, it becomes nearly zero.

2.4. Technical analysis of stocks

Fundamental and technical analysis are the two tools commonly used for investment and trading decision making by traditional investors and traders. The investor generally invests for long term uses fundamental analysis. In fundamental analysis, the company's financial condition and future growth are checked by various parameters using financial reports of the company.

The traders commonly use technical analysis to know the near term trend of the stock price. Traders generally trade for a short duration spanning from a few minutes to a few months. Technical analysis provides technical indicators that commonly used to predict short term momentum or trend of the stock price. We used technical indicators Commodity Channel Index, Relative Strength Index, and (%R) Williams Percent Range (Kumar & Thenmozhi, 2006; Chakole & Kurhekar, 2019) in our proposed models. These indicators provide momentum for the current stock price trend.

3. Proposed system

This section describes our two proposed models in detail. The intuition to use the RL approach to find an optimal dynamic trading strategy is that the MDP formulate a Sequential decision-making problem, and RL solves MDP. Stock market trading can be thought of as a sequential decision-making problem as at every duration of the stock trading session a trader has to make a trading decision.

Our target market is the Equity stock market. We have performed experimentation on the Indian and the American Equity stock markets. The proposed models follow the rules of trading in the Equity stock market. We can take a short position (first sell and buy then) or can have a long position (first buy and later sell). The short position in the Equity stock market is possible using the Securities Lending & Borrowing (SLB) mechanism.

We experimented with our proposed models on daily data of the stock, and daily only one trading action is permitted, either a Buy, Sell, or Hold for simplicity. In Q-learning, representation of the state of the environment has significant importance. The trading agent can only sense the environment (the stock market)

through the state. We have proposed two different trading models based on the two different representations of a state of the environment.

3.1. Proposed model 1: Cluster to represent a state

In this model, we assume that history tends to repeat itself. The current trading session pattern of the stock market can be similar to the pattern of a trading session in the past. Our objective is to find that past trading session and the most optimal trading action at that time and apply that trading action to the current trading session.

The common three patterns or trends on any trading session can be an uptrend, a downtrend or a sideways trend. There are some sub-patterns inside these three common patterns. So we can divide the historical trading session's data into n groups or clusters. Two trading sessions in a cluster have some common features. Similarly, two trading sessions in different clusters differ in some features.

The n groups or clusters are formed from historical trading sessions' data using the k-Means clustering algorithm. As shown in Fig. 2, the trading session data pre-processed before forming clusters. The pre-processing extract desired information from raw data. The stock market data is available in the $[Open, High, Low, Close, Volume]$ form for the daily trading session. We pre-processed this daily trading session data to represent the state of the environment, as tuple mentioned below.

$State = [O, H, L, C, V]$ 用一个向量表示状态

where

1. O = Percentage change in open price as compare to previous day close price = $\left(\frac{Open_t - Close_{t-1}}{Close_{t-1}}\right) * 100$
2. H = Percentage change in high price as compare to same day open price = $\left(\frac{High_t - Open_t}{Open_t}\right) * 100$
3. L = Percentage change in low price as compare to same day open price = $\left(\frac{Low_t - Open_t}{Open_t}\right) * 100$
4. C = Percentage change in close price as compare to same day open price = $\left(\frac{Close_t - Open_t}{Open_t}\right) * 100$
5. V = Percentage change in volume as compare to previous day volume = $\left(\frac{Volume_t - Volume_{t-1}}{Volume_{t-1}}\right) * 100$

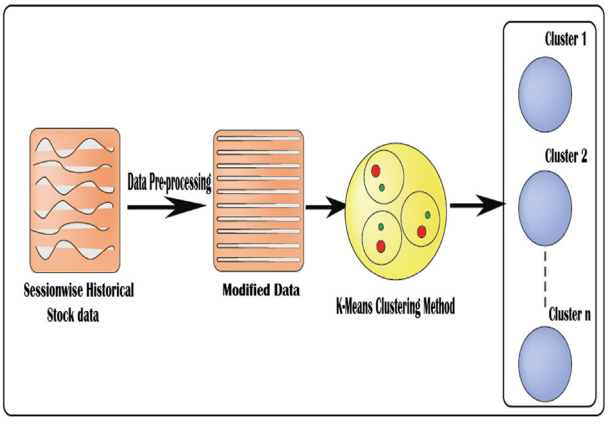


Fig. 2. Clusters formation from historical stock market data.

The pre-processed historical trading sessions data in the $[O, H, L, C, V]$ is provided to the k-Means clustering method to form n clusters or groups.

The trading agent is implemented using the Q-learning method of RL. It maintains its trading strategy using a state-action mapping table called as Q-table, which contains a Q-value for each state-action pair. It senses the environment (Stock market) using the state of the environment. The trading agent initially does not know its environment so; at the start, all Q-values are zero. In the proposed model 1, each cluster is a state of the environment. So the entire states are n, and the total actions are three (Buy, Sell, or Hold). So initially, the trading strategy, i.e. Q-table, is as shown in Fig. 3.

The trading agent senses the environment's state and performs trading action based on Q-table and state, in response, gets profit or loss as rewards, as shown in Fig. 3. The Q-value for each state-action pair of Q-table is updated using Eq. (6) based on rewards received for trading action. Iterative updates obtain optimal Q-table values. The optimal trading action for a state is selected using the ϵ-greedy approach discussed in Section 2.3. The optimal Q-table obtained from historical (Training) data.

The trading agent selects trading action based on the state of the environment from Q-table for each trading session. In this model, the state depends on the daily values of $[Open, High, Low, Close, Volume]$, state changes as the trading session (the day) changes. In proposed model 1, the trading agent at trading session t takes trading action at the beginning of the session t (at open price) depend on the state of the environment at previous trading session $t-1$.

3.2. Proposed model 2: Candlestick to represent a state

In this proposed model, we use candlestick to form the state of the environment. The candlestick represents stock price information for the trading session such as Open, High, Low, and Close price, as shown in Fig. 4.

The trading session length varies from one minute to one month, and so on. In this proposed model candlestick length is one day. Traditionally traders in the stock market use candlestick patterns to guess the future stock price trend.

Typically a candlestick has three parts that are body, upper shadow, and lower shadow. The body is the difference between the close and open price of the stock for the trading session. If the open price is small than the close price, it indicates an increase in the price, and such a candle is called a Bullish candle. Generally, the Green colour represents the Bullish candle. Whereas if the open price is higher than the close price, it indicates a decrease in the price, and such a candle is called a Bearish candle; generally, the Red colour represents the Bearish candle.

The upper shadow is the difference between the high price and the close price in bullish candlestick and the difference between the high price and the open price in bearish candlestick. Similarly, the lower shadow is the difference between the lower price and the open price in bullish candlestick and the difference between the lower price and the close price in bearish candlestick.

The price trend of the current trading session of a stock is related to the price trend of the previous trading session, and so on. The body portion of the candlestick represents the relation between the close price and the open price. It is capable of encapsulating the price trend of the stock; in this model, the state of the environment formed from the body portion of the candlestick.

We have formed six different categories from a day trading session based on the percentage change between the open and close price of the stock. These six categories are the six states of this model, as shown in Table 1. If the close price is higher than the

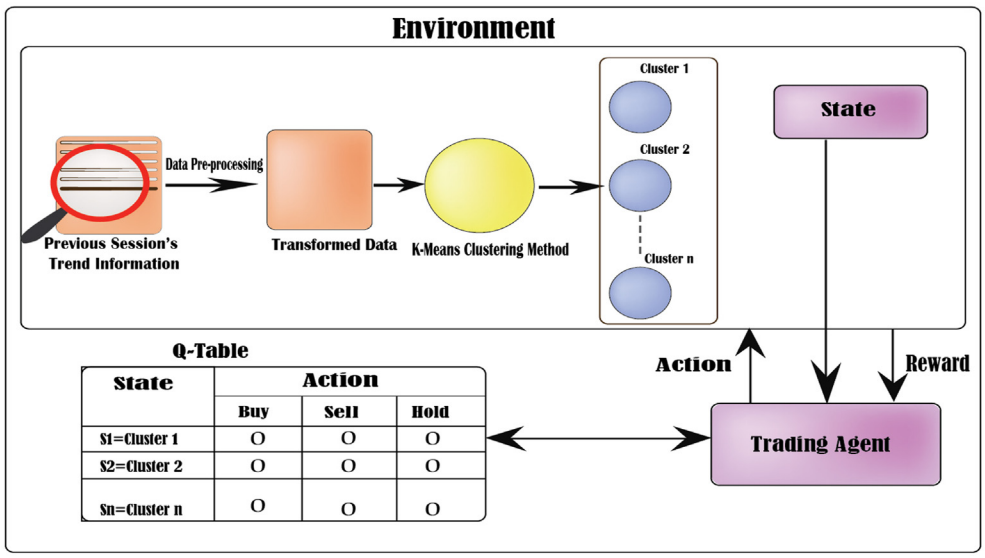


Fig. 3. Our Proposed Model 1 Trading System.

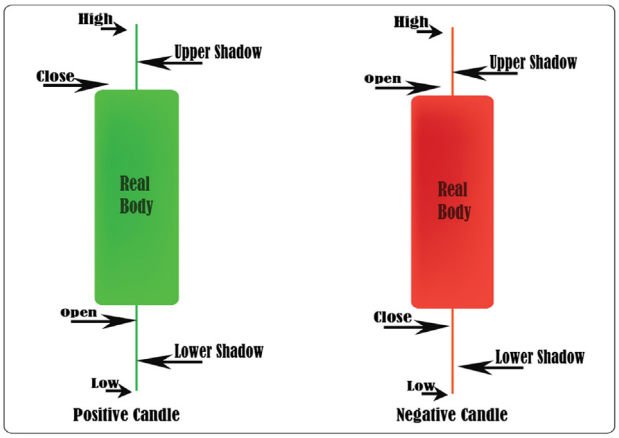


Fig. 4. A candlestick to represent variation in the stock price during a trading session.

Table 1
State representation for the proposed model 2 using candlestick.

%[(Close-Open)/Open]	State
(0,1)	UP
(1,2)	HUP
(> 2)	VHUP
(−1,0)	DOWN
(−2,−1)	HDOWN
(< −2)	VHDOWN

open price, it is positive price movement, and we call it UP, if greater than 1% it is called HUP, and if greater than 2% it is called VHUP. Similarly, if the close price is down by open price, it is an adverse price movement, and we call it DOWN, if less than 1%, it is called HDOWN, and if less than 2% it is called VHDOWN.

The proposed model 2 maintains and converges the Q-table as similar to the proposed model 1, as shown in Fig. 5. The main difference between these two models is in the formation of a state of the environment. This model uses candlestick's body portion to form a state. The number of states is six as shown in Table 1.

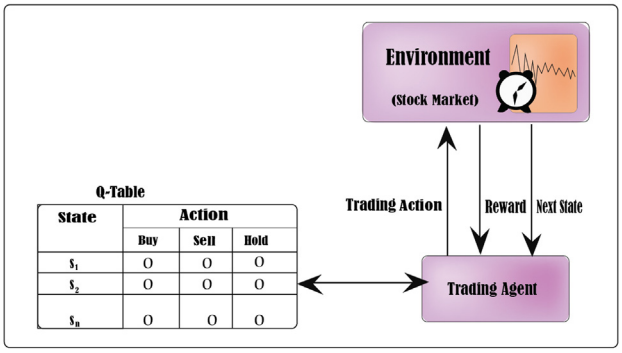


Fig. 5. Our Proposed Model 2 Trading System.

In this proposed model 2, the trading agent selects trading action based on the state of the environment from Q-table for each trading session, as shown in Fig. 5. As a state based on a daily candlestick, state changes as the trading session (the day) change. The trading agent at trading session t takes trading action at the end of the daily trading session t (at close price) based on the state of the environment at trading session t. The trading action is taken 5 min before the end of the trading session based on state formed from the beginning of the trading session to approximate the end of the same trading session (our close price will be stock price 5 min before the end of the trading day).

4. Results and discussion

This section provides the details of experiments conducted with the two proposed models, including dataset, performance evaluation metrics, and experimental results.

4.1. Experimental dataset

We experimented with the two proposed models on the daily data of the individual stocks and the index stocks shown in Table 2, and Table 3 respectively. Figs. 6 and 7 are the plots of the Indian

Table 2
The index stocks experimental datasets.

Name of stock	Period	Total period	Training period	Testing period
NASDAQ	From	2001-01-02	2001-01-02	2006-01-03
	To	2018-12-31	2005-12-30	2018-12-31
DJIA	From	2001-01-02	2001-01-02	2006-01-03
	To	2018-12-31	2005-12-30	2018-12-31
NIFTY	From	2007-09-17	2007-09-17	2011-07-28
	To	2018-12-31	2011-07-27	2018-12-31
SENSEX	From	2007-09-17	2007-09-17	2011-07-28
	To	2018-12-31	2011-07-27	2018-12-31

Table 3
Experimental datasets of six individual stocks.

Name of stock	Period	Total period	Training period	Testing period
Bank of Baroda	From	2012-01-01	2012-01-01	2017-09-29
	To	2019-08-31	2017-09-28	2019-08-31
Tata Motors	From	2012-01-01	2012-01-01	2017-09-29
	To	2019-08-31	2017-09-28	2019-08-31
Asian Paint	From	2012-01-01	2012-01-01	2017-09-29
	To	2019-08-31	2017-09-28	2019-08-31
Infosys	From	2012-01-01	2012-01-01	2017-09-29
	To	2019-08-31	2017-09-28	2019-08-31
PNB	From	2012-01-01	2012-01-01	2017-09-29
	To	2019-08-31	2017-09-28	2019-08-31
Suzlon	From	2012-01-01	2012-01-01	2017-09-29
	To	2019-08-31	2017-09-28	2019-08-31

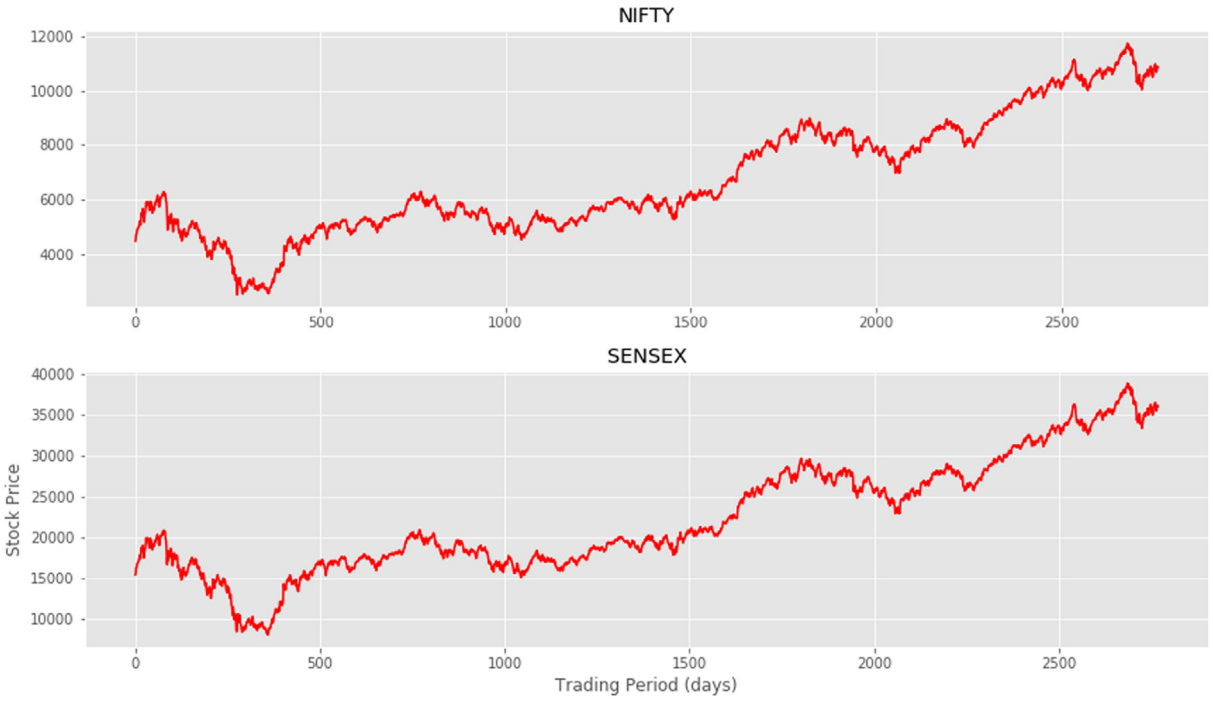


Fig. 6. Experimental dataset of the Indian Index stocks.

and the American experimental index stocks. Fig. 8 shows experimental dataset of six Indian individual stocks.

The index stocks NASDAQ and DJIA are from the American stock exchanges, and the index stocks SENSEX and NIFTY are from the Indian stock exchanges. The experimental dataset obtained from Yahoo finance. Our code and experimental dataset are available at the URL: https://sites.google.com/view/jagdishchakole/. We used Python programming to develop the proposed models.

4.2. Experimental scheme

The transaction charge is 0.10% of the trading amount on both sides, sell or buy. As risk management is necessary for trading in the stock market, one should not depend on only one method or logic for trading. We have used two risk management tools viz stop loss and Trend Following. The stop loss of 10% used. If the current trading position's loss is more than 10%, then that trading position should be closed. The stop loss value depends on trader domain knowledge.

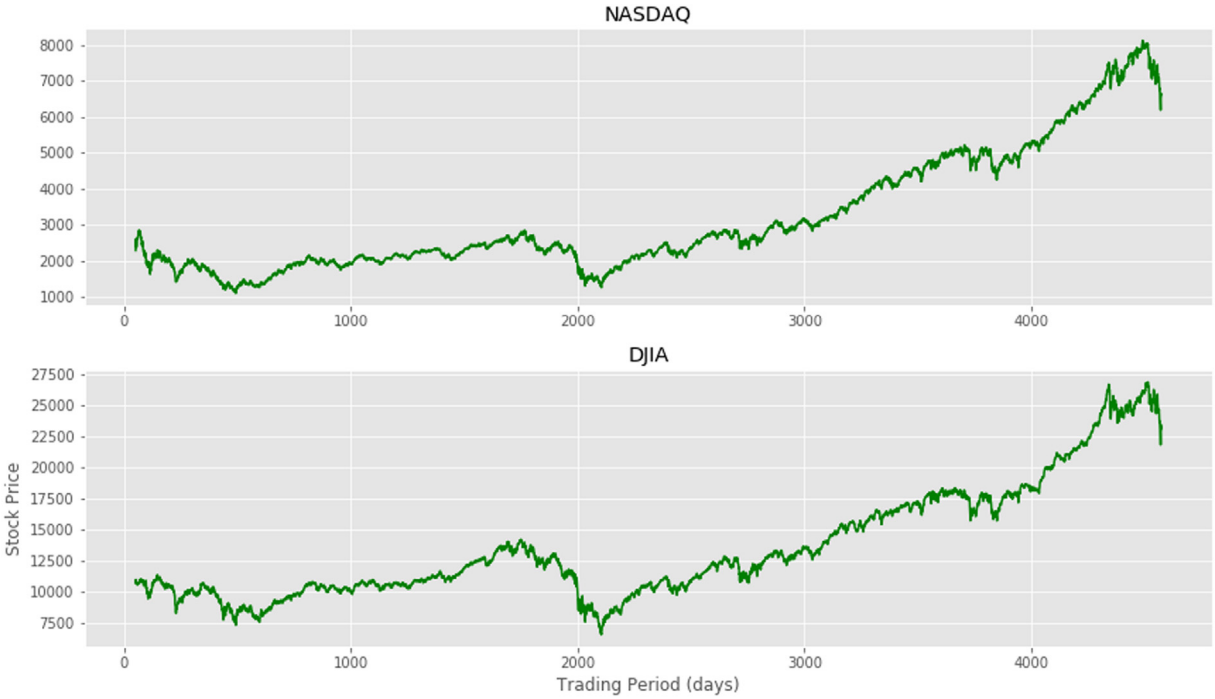


Fig. 7. Experimental dataset of the American Index stocks.

The other risk management tool is Trend Following, in which trading action can take according to the current stock price trend. Momentum technical indicators used to know the current Trend. We used three technical momentum indicators CCI, Relative Strength Index (RSI), and % R (Dash & Dash, 2016). These indicators provide Buy, Sell, and Hold signals. The trading agent in our proposed models performs trading action suggested by Q-table if that trading action is matching with the majority of trading actions aimed by these three indicators.

4.3. Performance evaluation metrics

We compared the two proposed models with two benchmark trading strategies viz the Buy-and-Hold trading strategy, and the trading strategy using the Decision Tree method. Also compared the performance of the two proposed models with each other. Performance evaluation metrics used for the performance comparisons are Average annual return, Accumulated return, Average daily return, Maximum Drawdown, Standard Deviation, and Sharpe ratio (Chakole & Kurhekar, 2019).

Percentage Accumulated Return (%AR) define as % change of start amount of the investment to the end amount of the investment.

$$\%AR = \frac{Amount\ at\ last - Amount\ at\ start}{Amount\ at\ start} * 100 \tag{7}$$

Average annual return is the average of total Accumulated return over the total number of years. Average daily return is the average of total Accumulated return over the total number of trading days. The optimal value of all these three returns should be large. Maximum Drawdown is a tool to measure the risk of the trading strategy. Drawdown is the difference between the historically high value to the current value of the strategy. Maximum Drawdown is the maximum value among all the Drawdowns. Standard Deviation is a tool to know the volatility of the trading strategy. Sharpe ratio measures the risk associated with the return of the trading strategy. Higher the Sharpe ratio lower is the risk.

4.4. Experimental results

The two proposed models are trained on the training data and tested on the testing data. Tables 2 and 3 shows the partition of experimental datasets in train and test data.

4.4.1. Results of proposed model 1: Cluster to represent a state

The proposed model 1 experimented on individual stocks of three different trends viz Sideways, Uptrend, and Downtrend. We have two individual stocks of each trend, so in total on six individual stocks. We experimented on three different sizes of clusters (n) = 3, 6, and 9. The minimum cluster size is three, as the stock price trend has three trends.

The comparison of the performance of all four models on six individual stocks are recorded in Table 4. The performance of the proposed model 1 with different cluster size (n = 3, 6, and 9) on six individual stocks with three different trends shown in Table 4 and plotted in Fig. 9. The finding of this experimentation on proposed model 1 summarised below:

- The % accumulated return increases with an increase in the number of clusters for the Uptrend stocks.
- The % accumulated return is independent of the number of clusters for the sideways stocks.
- The % accumulated return decreases with an increase in the number of clusters for the Downtrend stocks.

The comparison of the performance of the four different trading strategies in terms of percentage accumulated return on six individuals stocks on the test dataset shown in Fig. 10. Cluster size n = 6 considered for comparison with the other three models for

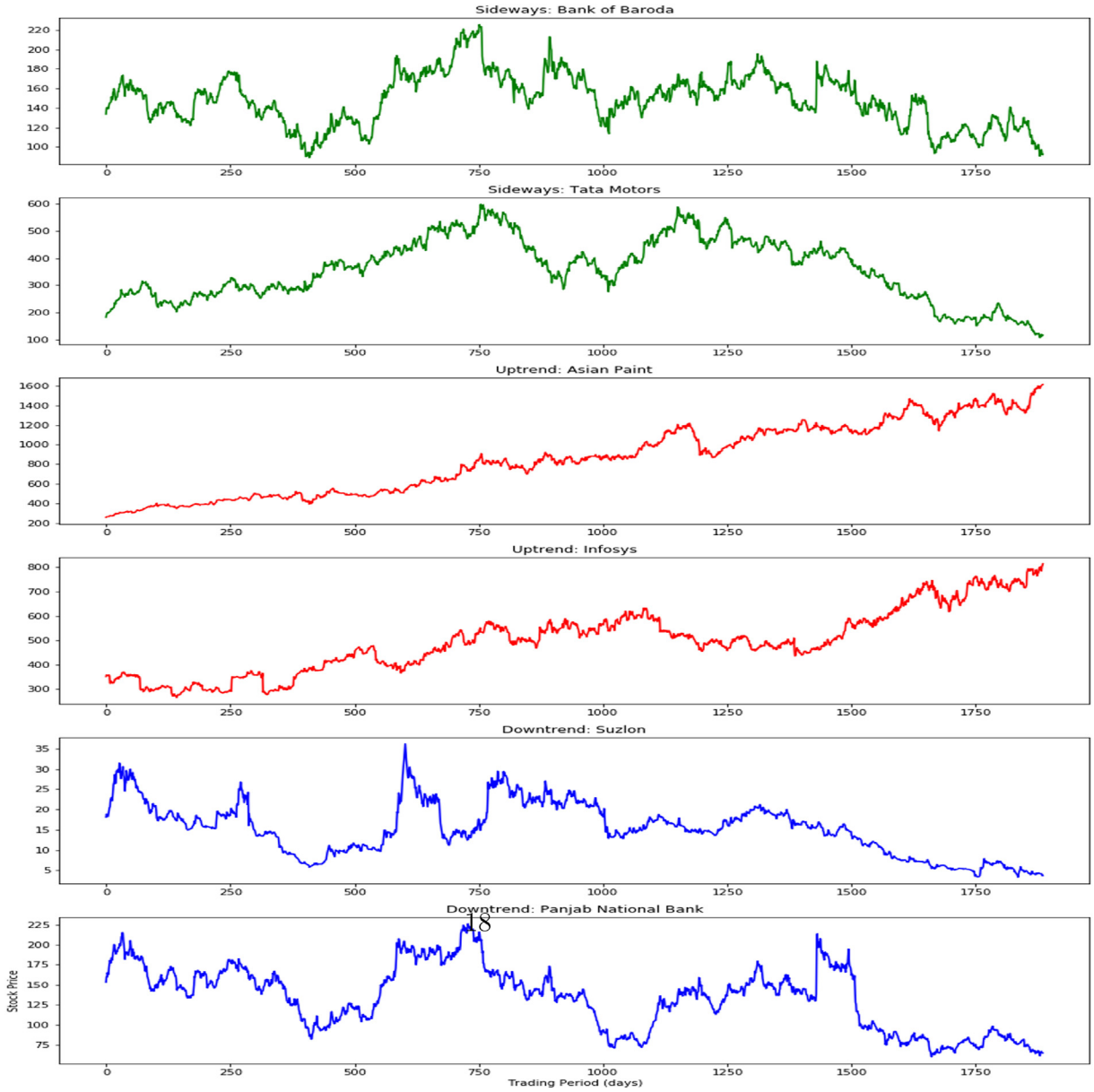


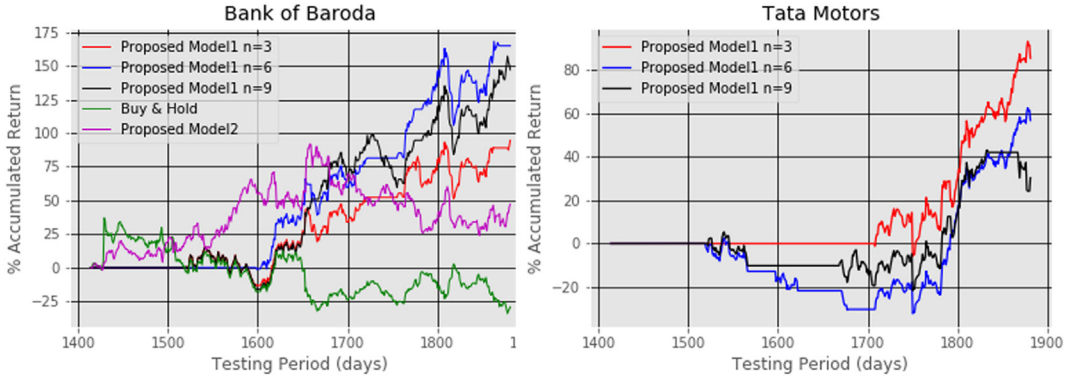
Fig. 8. Experimental dataset of six Individual stocks.

Table 4
Performance of all the trading strategies in terms of % accumulated return on six individual stocks with three different trends.

S.N.	Stock name	Buy-and-Hold	Decision tree	Proposed Model 1			Proposed Model 2	Trend
				$n = 3$	$n = 6$	$n = 9$		
1.	Bank of Baroda	−29.28	43.74	94.54	164.82	106.98	47.24	Sideways
2.	Tata Motors	−70.02	19.25	85.46	56.77	30.34	31.68	Sideways
3.	Asian Paint	41.33	48.16	60.66	80.00	86.30	58.50	Uptrend
4.	Infosys	74.38	44.32	75.50	108.95	111.00	150.92	Uptrend
5.	Suzlon	−73.07	−2.34	136.27	23.63	26.04	11.24	Downtrend
6.	PNB	−47.96	−6.37	58.32	42.37	52.37	13.10	Downtrend

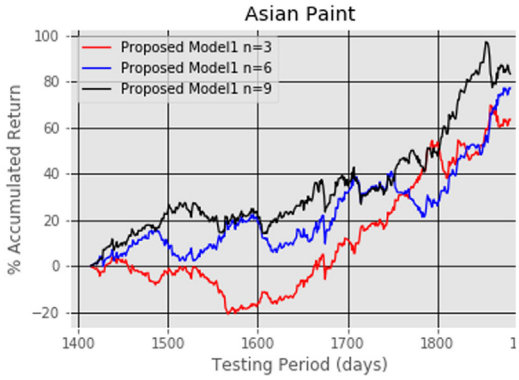
simplicity. The proposed model 1 outperforms the two benchmark models trading strategy using the Decision Tree and Buy-and-Hold model on all the six individuals test datasets. It also performed better than proposed model 2 for the majority of the test datasets of individual stocks except for Infosys stock.

The proposed model 1 also experimented with the four index stocks. Our experimental results reveal that cluster size $n = 3$ is best suitable for all index stocks, so we used a cluster size $n = 3$ for all index stocks. The comparison of the performance of all four models on four index stocks recorded in Table 5.

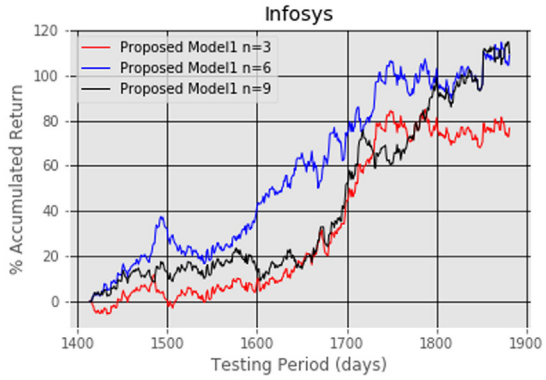


(a) Sideways: Bank of Baroda

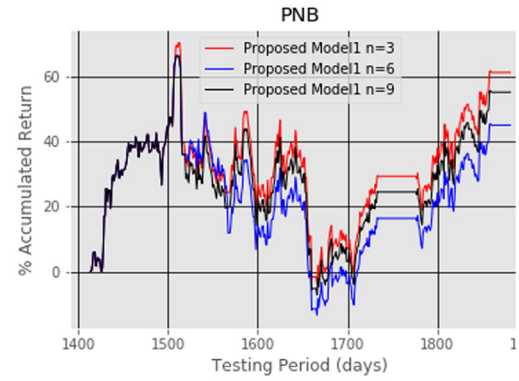
(b) Sideways: Tata Motors



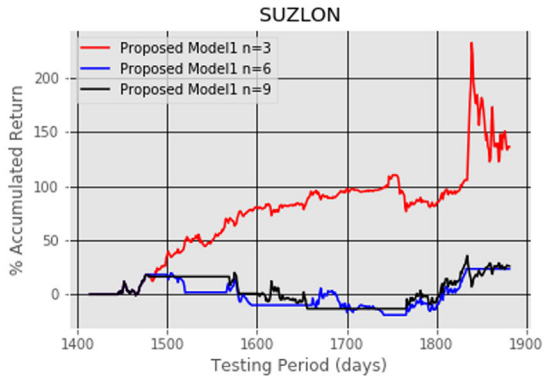
(c) Uptrend: Asian Paint



(d) Uptrend: Infosys



(e) Downtrend: Punjab National Bank



(f) Downtrend: Suzlon

Fig. 9. Performance of the proposed model 1 with different cluster (n = 3, 6, and 9) size on the test dataset of six individual stocks with three different trends in terms of % accumulated Return.

The performance of all four models using percentage accumulated return on the four index stocks NASDAQ, DJIA, SENSEX, and NIFTY plotted in Fig. 11–14 respectively.

Performance comparison using percentage accumulated return of all trading models on four index stocks on the test dataset shown in Fig. 15. The proposed model 1 outperforms the two benchmark models trading strategy using the Decision Tree and Buy-and-Hold model on four index stocks test datasets. It also performed better than proposed model 2 for the majority of the test datasets of index stocks except for NASDAQ index stock.

Table 5 concludes that on index stocks, the proposed model1's accumulated return, average daily return, and the average annual return is better than two benchmark models. The maximum drawdown of the proposed model 1 is better than other models on three index stock except for index stock NASDAQ where Decision Tree's maximum drawdown is best. The Standard Deviation and Sharpe Ratio of the proposed model 1 is reasonably acceptable compared to other models.

We performed students' t-test on % daily return of the proposed models and the benchmark methods to check whether the results are statistically different. We accomplished this test between

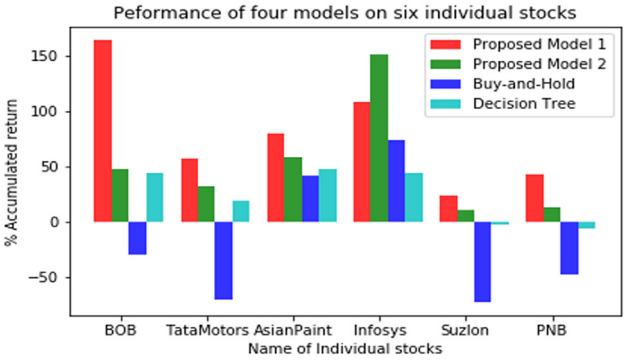


Fig. 10. Performance of the four different trading strategies on six individuals stocks on the test set in terms of % accumulated return.

Buy-and-Hold method and the proposed model 1. And, between Decision-tree method and the proposed model 1. and proposed model 1 and the proposed model 2, The t-values have recorded the Table 6. All absolute value of all t-values is higher than the level of significance 1.96. So, it concludes that the results are statistically significant.

4.4.2. Results of proposed model 2: Candlestick to represent a state

All the four models, including the proposed model 2 trained and tested on the same stocks with equal durations for a fair comparison. The Table 4 and Fig. 9 concludes that the performance of the proposed model 2 outperforms the Buy-and-Hold trading strategy as well as the decision tree trading strategy in terms of % accumulated return on the individual stocks.

Similarly Table 5 and Fig. 15 concludes that the performance of the proposed model 2 outperforms the Buy-and-Hold trading strategy as well as the decision tree trading strategy in terms of % accumulated return on the index stocks.

The proposed model 2 performs better than proposed model 1 for a few cases otherwise for the majority of cases the proposed model 1 outperforms proposed model 2 on the individual and index stocks as reported in Tables 4 and 5.

One of the findings from the experimentation results is that the performance of the proposed model 2 is not affected by the stock price trends. Table 5 concludes that the percentage accumulated return of the proposed model 2 is 256.49%, 58.38%, and 24.22% higher compared to the Decision Tree model, Buy-and-Hold model, and the proposed model 1 on the test dataset of index stocks NASDAQ respectively. It also concludes that the percentage accumulated return of the proposed model 2 is 133.49%, and 9.62% higher compared to the Decision Tree model, and Buy-and-Hold model on the test dataset of index stocks DJIA respectively. Similarly, 18.45% and 6.34% for index stocks NIFTY. Equally, 20.70% and 21.68% for index stocks SENSEX.

5. Conclusion

The generation of the optimal dynamic trading strategy is a crucial task for the stock traders. We tried to find an optimal dynamic trading strategy using the Q-learning algorithm of Reinforcement Learning. The performance of a trading agent using the Q-learning method firmly based on the representation of the states of the environment. We proposed two models based on the representation of the states of the environment.

In proposed model 1, we recommend an innovative way to form the states of the environment using unsupervised learning method k-Means which represent the behaviour of the stock market (environment) using a finite number of states (clusters). This grouping historical information using unsupervised learning methods to describe the behaviour of the stock market (environment) using a limited number of states for trading using Reinforcement Learning is key to our proposed work.

In proposed model 1, clusters are formed to represent the state; we also vary the number of clusters and test on the stock with dif-

Table 5
Comparison of trading performance on index stocks test dataset of four stock trading strategies.

Name of stock	Performance evaluation metrics	Decision tree	Buy-and-Hold	Proposed Model 1	Proposed Model 2
NASDAQ	Average Annual Return (%)	6.61	14.88	18.98	23.57
	Accumulated Return (%)	85.85	193.23	246.36	306.05
	Average Daily Return (%)	0.026	0.059	0.075	0.093
	Maximum Drawdown (%)	39.68	55.63	48.24	34.83
	Standard Deviation (%)	1.21	1.31	2.14	2.09
	Sharpe Ratio (%)	1.24	1.82	0.40	0.52
DJIA	Average Annual Return (%)	4.10	8.72	13.02	9.57
	Accumulated Return (%)	53.20	113.31	169.02	124.22
	Average Daily Return (%)	0.016	0.034	0.052	0.038
	Maximum Drawdown (%)	52.80	53.77	25.54	36.06
	Standard deviation (%)	1.06	1.12	1.96	2.05
	Sharpe Ratio (%)	1.01	1.51	0.107	−0.005
NIFTY	Average Annual Return (%)	12.03	13.22	19.79	14.06
	Accumulated Return (%)	86.57	96.43	144.40	102.55
	Average Daily Return (%)	0.048	0.052	0.078	0.056
	Maximum Drawdown (%)	22.44	22.52	21.50	27.01
	Standard Deviation (%)	0.88	0.95	2.51	2.50
	Sharpe Ratio (%)	1.84	1.87	−0.024	−0.204
SENSEX	Average Annual Return (%)	13.70	13.58	32.44	16.54
	Accumulated Return (%)	98.92	98.12	234.07	119.40
	Average Daily Return (%)	0.054	0.054	0.128	0.065
	Maximum Drawdown (%)	25.49	22.67	15.72	27.01
	Standard Deviation (%)	0.89	0.94	2.514	2.511
	Sharpe Ratio (%)	1.99	1.90	0.26	−0.13

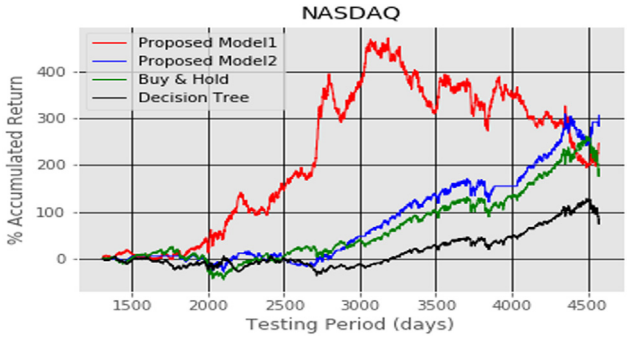


Fig. 11. The performance comparison of all four methods using % accumulated return on the test dataset of index stock NASDAQ.

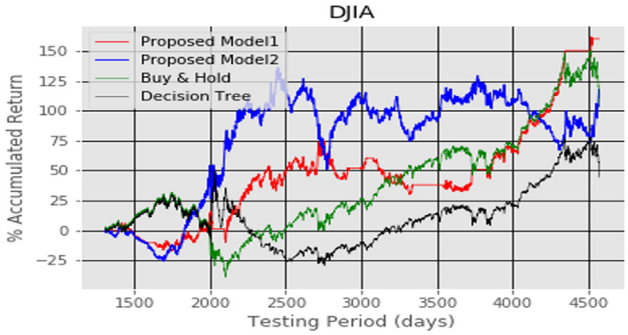


Fig. 12. The performance comparison of all four methods using % accumulated return on the test dataset of index stock DJIA.

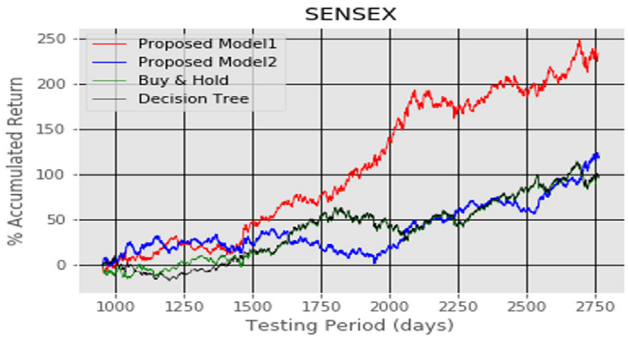


Fig. 13. The performance comparison of all four methods using % accumulated return on the test dataset of index stock SENSEX.

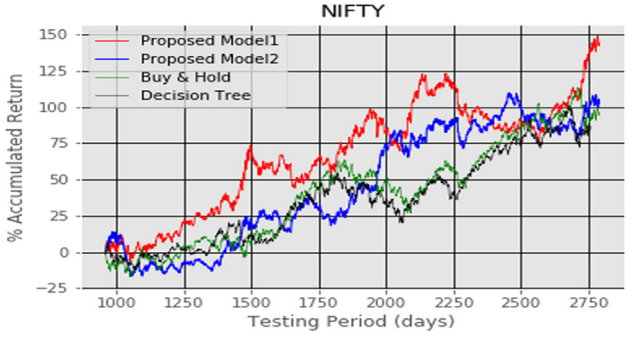


Fig. 14. The performance comparison of all four methods using % accumulated return on the test dataset of index stock NIFTY.

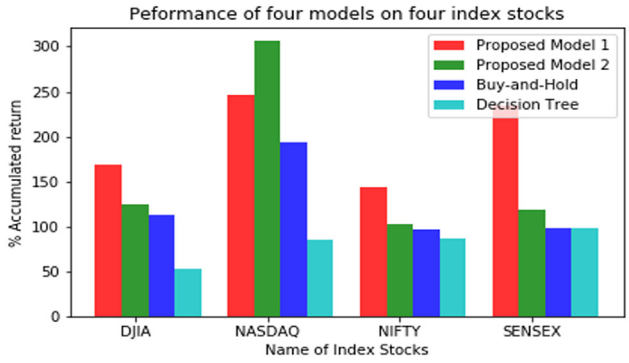


Fig. 15. Performance comparison using percentage accumulated return of all trading models on four index stocks on the test dataset.

Table 6
Result of the Student's t-Test (t-values) between different trading models.

Name of stock	Buy-and-Hold and Proposed Model 1	Decision tree and Proposed Model 1	Proposed Model 1 and Proposed Model 2
NASDAQ	53.7	76	46.3
DJIA	6.2	39.8	−27.6
NIFTY	21.8	27.5	16.5
SENSEX	34.36	34.0	35.0

ferent trends. The experiment result concludes that the increase in the number of the cluster is favourable for uptrend stocks; it does not show any effect on stocks having a sideways trend. In contrast, the increase in the number of the cluster is unfavourable for downtrend stocks.

The proposed model 2, used the candlestick to form the states of the environment. The experimental results of this model conclude that its performance is independent of the stock price trend. Our both the proposed models outperformed two benchmark models viz DecisionTree and Buy-and-Hold based trading strategies in terms of return on investment. The proposed model 1 performed better than the proposed model 2 in the majority of the experiments.

The transaction cost is essential in trading, and we included the transaction cost in our computations, The transaction cost depends on the number of trades as a higher number of total trades will result in lower performance. We limit the total number of trade using momentum indicators. One of the findings is that the initial values of the Q-table play a significant role in the convergence of the final values of the Q-table. All Q-values should initialize to the same values (e.g. zeroes) for the fair condition for all actions.

This study is based on the Indian and the American Equity stock market, which permits both long buy and short sell trading using the Securities Lending & Borrowing (SLB) mechanism. The experiment performed on daily data. In future, these models can use for other frequency datasets such as hourly dataset.

CRediT authorship contribution statement

Jagdish Bhagwan Chakole: Conceptualization, Methodology, Software, Writing - original draft. **Mugdha S. Kolhe:** Software, Validation. **Grishma D. Mahapurush:** Software, Validation. **Anushka Yadav:** Software, Visualization. **Manish P. Kurhekar:** Supervision, Writing - review & editing, Project administration, Resources.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Alimoradi, M. R., & Kashan, A. H. (2018). A league championship algorithm equipped with network structure and backward q-learning for extracting stock trading rules. *Applied Soft Computing, 68*, 478–493.

Andriosopoulos, D., Doumpos, M., Pardalos, P. M., & Zopounidis, C. (2019). Computational approaches and data analytics in financial services: A literature review. *Journal of the Operational Research Society*, 1–19.

Calabuig, J., Falciani, H., & Sánchez-Pérez, E. A. (2020). Dreaming machine learning: Lipschitz extensions for reinforcement learning on financial markets. *Neurocomputing*..

Chakole, J., & Kurhekar, M. (2019). Trend following deep q-learning strategy for stock trading. *Expert Systems*, e12514.

Chia, R. C. J., Lim, S. Y., Ong, P. K., & Teh, S. F. (2015). Pre and post chinese new year holiday effects: Evidence from hong kong stock market. *The Singapore Economic, 60*, 1550023.

Dash, R., & Dash, P. K. (2016). A hybrid stock trading framework integrating technical analysis with machine learning techniques. *The Journal of Finance and Data Science, 2*, 42–57.

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

Fischer, T. G. (2018). *Reinforcement learning in financial markets-a survey*. Technical Report FAU Discussion Papers in Economics..

Fong, S., Si, Y.-W., & Tai, J. (2012). Trend following algorithms in automated derivatives market trading. *Expert Systems with Applications, 39*, 11378–11390.

Gao, X., & Chan, L. (2000). An algorithm for trading and portfolio management using q-learning and sharpe ratio maximization. In *Proceedings of the international conference on neural information processing* (pp. 832–837).

García-Galicia, M., Carsteanu, A. A., & Clempner, J. B. (2019). Continuous-time reinforcement learning approach for portfolio management with time penalization. *Expert Systems with Applications, 129*, 27–36.

Gorse, D. (2011). *Application of stochastic recurrent reinforcement learning to index trading*. ESANN..

Hu, Y., Feng, B., Zhang, X., Ngai, E., & Liu, M. (2015). Stock trading rule discovery with an evolutionary trend following model. *Expert Systems with Applications, 42*, 212–222.

Huang, Q., Wang, T., Tao, D., & Li, X. (2014). Biclustering learning of trading rules. *IEEE Transactions on Cybernetics, 45*, 2287–2298.

James, J. et al. (2003). Simple trend-following strategies in currency trading. *Quantitative Finance, 3*, C75–C77.

Kumar, M., & Thenmozhi, M. (2006). Forecasting stock index movement: A comparison of support vector machines and random forest. In *Indian institute of capital markets 9th capital markets conference paper*. .

Lee, J. W., Park, J., Jangmin, O., Lee, J., & Hong, E. (2007). A multiagent approach to *q*-learning for daily stock trading. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, 37*, 864–877.

Meng, T. L., & Khushi, M. (2019). Reinforcement learning in financial markets. *Data, 4*, 110.

Miller, M. H., Muthuswamy, J., & Whaley, R. E. (1994). Mean reversion of standard & poor's 500 index basis changes: Arbitrage-induced or statistical illusion? *The Journal of Finance, 49*, 479–513.

Moody, J., Wu, L., Liao, Y., & Saffell, M. (1998). Performance functions and reinforcement learning for trading systems and portfolios. *Journal of Forecasting, 17*, 441–470.

Park, H., Sim, M. K., & Choi, D. G. (2020). An intelligent financial portfolio trading strategy using deep q-learning. *Expert Systems with Applications 158*, 113573.

Pendharkar, P. C., & Cusatis, P. (2018). Trading financial indices with reinforcement learning agents. *Expert Systems with Applications, 103*, 1–13.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT Press.

Treleaven, P., Galas, M., & Lalchand, V. (2013). Algorithmic trading review. *Communications of the ACM, 56*, 76–85.

Wu, X., Chen, H., Wang, J., Troiano, L., Loia, V., & Fujita, H. (2020). Adaptive stock trading strategies with deep reinforcement learning methods. *Information Sciences*.

Yadav, Y. (2015). How algorithmic trading undermines efficiency in capital markets. *Vanderbilt Law Review, 68*, 1607.

Yun, H., Lee, M., Kang, Y. S., & Seok, J. (2020). Portfolio management via two-stage deep learning with a joint cost. *Expert Systems with Applications, 143* 113041.

Zhu, Y., Yang, H., Jiang, J., & Huang, Q. (2018). An adaptive box-normalization stock index trading strategy based on reinforcement learning. In *International conference on neural information processing* (pp. 335–346). Springer.