FINANCE AND RISK ENGINEERING
FRE-GY.6123 FINANCIAL RISK MANAGEMENT
SPRING 2022

FINAL WRITTEN PAPER

Reinforcement Learning Strategy via Fundamental and Technical Indicators

Jiaqi Liang (jl12581)
Guojun Chen (gc2817)
Nicole Zhao (xz2616)

Supervised By
Professor James Adams

1 Introduction

Quantitative strategy is a significant topic in financial analysis, trading, and risk management. Factor-based models are widely applied and evolved over the years among all types of trading strategies. The factor models empower investors to understand and manage risk more efficiently[1]. By tilting toward exposure to different kinds of factors, investors could obtain greater diversification than asset classes provide[2]. For decades, the research on algorithms to compute factors and strategies to combine factors smartly to guide investment actions is always a blooming direction.

This paper mainly focuses on combining fundamental factors with technical indicators to generate trading rules via Q-learning. We made a few changes to the work introduced by Hu[3], and the experimental outcomes will further prove the profitability and generalization of their model.

1.1 Related Work

Reinforcement Learning (RL) have gained increasing popularity in financial market and industry. Classifier systems and Q-Learning are two typical RL models commonly used for generating trading rules. Classifier systems are rule-based systems that combine temporal difference learning or supervised learning with a genetic algorithm to solve classification and reinforcement learning problems. Q-Learning is a model-free reinforcement learning algorithm to learn the value of an action in a particular state. Classifier systems combined with Genetic algorithm are better tackling with high-dimensional features using Heuristic Algorithm, while Q-Learning are better with low-dimensional features by Exhaustion.

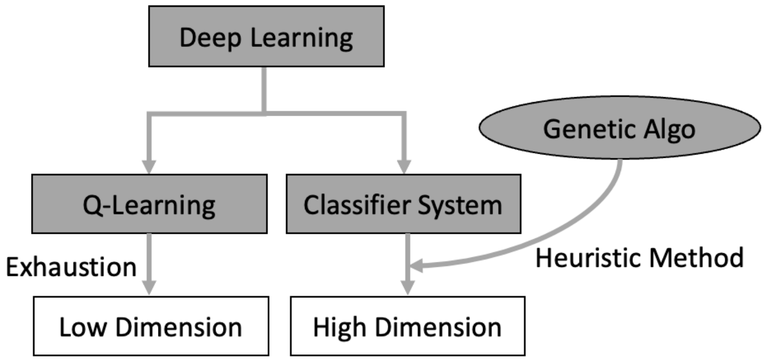


Figure 1: Relationship between Reinforcement Learning Technique

Genetic algorithm (GA), first introduced in the 1970s, is a mature and powerful technique to solve optimization and search problems. It was first used to generate trading rules by Allen & Karjalainen in 1999[3], and has been refined and developed since then. For example, Silva, Neves & Horta (2014) conducted portfolio management using a multi-objective GA with objectives return and risk[4]. Chen (2019) defined a group trading strategy portfolio (GTSP) and utilized the grouping GA to solve the problem[5]. Zhang and Khushi (2020) designed an automatic robotic trading system, where the trading rules are selected using GA to maximize Sharpe and Sterling Ratio [6].

Among versions of algorithms and models, eXtended Classifier Systems (XCS) is a particular approach that combines GA with the learning process of reinforcement learning (RL). Hu (2015) proposed this innovative method to identify long-term and short-term market trends. This study used daily trading data of three indexes (Shanghai Composite Index, Shanghai Industrial Index, and Shanghai Business Index) from 2001 to 2013. It turns out that compared to the buy-and-hold strategy, the novel hybrid approach generates excess returns with a higher Sortino ratio after the transaction cost[3]. Furthermore, more research on XCS is presented in the quantitative finance area. Karlsen and Moschoyiannis (2018) used the XCS model to evolve a set of 'control rules' for several Boolean network instances[7]. Besides, Uwano (2018) conducted data mining with the XCS model based on rules generated from Random Forests (RF)[8]. It shows that RF helps XCS acquire optimal solutions as knowledge by developing appropriately generalized rules.

Q-learning is a model-free RL algorithm and it can handle problems with stochastic transitions and rewards without requiring adaptations. Jagdish Chakole (2019) used Trend following deep Q-Learning strategy for stock trading and found the model outperformed forecasting-based methods in terms of profitability and risk[9]. HyungjunPark et al.(2020) took advantage of three features to outperform other strategies in real-world trading[10]. Muhammad Reza Alimoradi (2018) extracted various stock trading rules for various kinds of stock market conditions using backward Q-learning[11].

1.2 Relevance to the course

Our work is concerned with using intraday data and Q-Learning to generate trading rules. Melas [1] stated that factor models provide a good way to understand and diversify the portfolio risk.Besides, risk control is an important aspect of developing trading strategies. Strategy could be assessed and tuned via metrics such as volatility, Sharpe ratio (SR), and maximum drawdown. SR illustrates the strategy's premium return per unit risk. Additionally, maximum drawdown indicates maximum downside exposure of our portfolio and shows portfolio's sustainability in long-run.

1.3 Proposed Claim

In Hu's paper, two limitations may affect the persuasiveness of XCS's generalization ability. (1) The authors included commonly evaluated factors, for instance, moving average convergence divergence (MACD), which were invented by Gerald in 1979. Although it is suggested that such popular technical indicators still have substantial guidance on predicting risk premium for Chinese market [12], a similar profitability of simple technical analysis could not be detected in U.S. equity market. (2) The experiment of their study was restricted on three indexes from Shanghai Stock Exchange. Therefore, the result could have bias due to the distinct geographical characteristics. Since research have demonstrated that the Chinese market appears long-time inefficiency[13], a well-behaved strategy implemented by fancy algorithm may not perform as well on highly efficient market in U.S.

In order to increase the timeliness and practicality of the model and its outcomes, we construct dataset by retrieving data in recent years from stocks in S&P 500. Moreover, studies have paid attention to combining technical indicators and fundamental variables and found significant predictive ability in equity market[14]. Thus, we will combine technical factors that proved to be effective by institute investors and fundamental factors into the trading strategy. **We hypothesize that the trading rules generated by Q-Learning algorithm based on a hybrid of deliberately selected technical and fundamental factors will outperform that of classical momentum indicators on U.S. equity market.** If the hypothesis is proved correct by empirical analysis, we could reasonably conclude that the bundle of fundamental and statistical factors would provide great insights about market movement of equities.

2 Reinforcement Learning Theory

Different from ML, RL develops from the idea of "interacting with the environment", which means the RL training agent gains experience and improves performance by constantly interacting with an environment. More specifically, the basic framework of RL problem consists of agent (the learner and decision maker)and environment that is defined by the Markov Decision Process (MDP) which is denoted by $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$. $\mathcal{S}$ is the set of all possible states (state space), $\mathcal{A}$ is collection of all available actions (action space), and $\mathcal{R}$ decides the reward received after each transformation between states.The goal of agent is to implement the optimal policy π^* which directs to take actions and gets the maximum total amount of reward (expected return) in the whole trajectory starting from any state.

Another important concept is value function. Formally, value function v under policy π is defined by:

$$v_\pi(s) = \mathbb{E}_\pi[\sum_{k=0}^{\infty} d^k R_{t+k+1}]$$

This state-value function tells the value of state s. Similarly, the action-value function that tells the value of taking action a at state s is given by:

$$q_\pi(s,a) = \mathbb{E}[\sum_{k=0}^{\infty} d^k R_{t+k+1} | S_t = s]$$

A fundamental property of value function is the recursive relationship it satisfies.The value under policy π at state s can be expressed by summation of the value of all possible successor states weighted by the transformation probability:

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + dv_\pi(s')]$$

It is also called the Bellman equation for v_π. The value function is a powerful tool to measure the performance of policies, and a better policy is defined to be the one that has no less value at any state than the other one. From this well defined order we can find the optimal policy which takes the maximum value at every state. Thus the evaluation of a policy can be naturally transformed to the estimation of its value function v_π, and performance of actions can be directly illustrated as the difference of value in next state and value in present state. In the training process, the goal-seeking agent trades off be- tween exploration (taking actions it has not known the reward) and exploitation (taking actions with the greatest expected reward) to decide the next action, and gradually drives the value function at each state towards the maximum. The value function under optimal policy in the form of Bellman equation can be expressed by:

$$v_{\pi^*}(s) = \max_{a \in A(s)} \sum_{s',r} p(s', r|s, a)[r + dv_{\pi^*}(s')]$$

Equivalently, for the action-value function under optimal policy it is:

$$q_{\pi^*}(s,a) = \sum_{s',r} p(s', r|s, a)[r + dv \max_{a' \in A(s')} q_{\pi^*}(s', a')]$$

In the most of cases, people do not have the complete knowledge of the environment, as the state transformation probability $p(\cdot)$ is always unknown. The different approaches to approximate the solution to optimal Bellman equation forms the basis of a number of RL algorithms.

2.1 Q-Learning Algorithm

Q-learning is a model-free RL algorithm and it can handle problems with stochastic transitions and rewards without requiring adaptations. For any finite Markov decision process (FMDP), Q-learning finds an optimal policy in the sense of maximizing the expected value of the total reward over any and all successive steps, starting from the current state. Q-learning can identify an optimal action-selection policy for any given FMDP, given infinite exploration time and a partly-random policy. "Q" refers to the function that the algorithm computes – the expected rewards for an action taken in a given state.

After Δt steps into the future the agent will decide some next step. The weight for this step is calculated as $\gamma^{\Delta t}$, where γ (the discount factor) is a number between 0 and 1 $(0 \leq \gamma \leq 1)$and has the effect of valuing rewards received earlier higher than those received later (reflecting the value of a "good start"). γ may also be interpreted as the probability to succeed (or survive) at every step Δt.

The algorithm, therefore, has a function that calculates the quality of a state–action combination:

$Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$

Before learning begins, Q is initialized to a possibly arbitrary fixed value (chosen by the programmer). Then, at each time t the agent selects an action a_t, observes a reward r_t, enters a new state s_{t+1} (that may depend on both the previous state s_t and the selected action), and Q is updated. The core of the algorithm is a Bellman equation as a simple value iteration update, using the weighted average of the old value and the new information:

$$Q^{new}(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\Bigg(\underbrace{\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}}}_{\text{new value (temporal difference target)}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \Bigg)}^{\text{temporal difference}}$$

where r_t is the reward received when moving from the state s_t to the state s_{t+1}, and α is the learning rate $(0 < \alpha \leq 1)$.

Note that $Q^{new}(s_t, a_t)$is the sum of three factors. The first one is $(1-\alpha)Q(s_t, a_t)$, the current value weighted by the learning rate. Values of the learning rate near to 1 make the changes in Q more rapid. The second one is $\alpha\, r_t$, the reward $r_t = r(s_t, a_t)$ to obtain if action a_t is taken when in state s_t (weighted by learning rate). The third one is $\alpha\gamma \max_a Q(s_{t+1}, a)$, the maximum reward that can be obtained from state s_{t+1}(weighted by learning rate and discount factor)

An episode of the algorithm ends when state s_{t+1} is a final or terminal state. However, Q-learning can also learn in non-episodic tasks (as a result of the property of convergent infinite series). If the discount factor is lower than 1, the action values are finite even if the problem can contain infinite loops. For all final states s_f, $Q(s_f, a)$ is never updated, but is set to the reward value r observed for state s_f. In most cases, $Q(s_f, a)$ can be taken to equal zero.

2.2 Q-Learning Implementation

From the pseudo code, we can see the only difference is the update rule for value function. In the process of Q-learning, generated actions in a trajectory does not follow the action selected in Q's updating. This implies that the learning data is derived from the policy that is "off" the policy being estimated, and this kind of algorithms represented by Q-learning is termed "off-policy". Q-learning algorithms directly learns the optimal value function thus it usually converges in faster speed, and the result of Q-learning may be affected by the large variance of training data and could be trapped in the local optimal control.

Algorithm 1 Q-Learning Algorithm

```
Require: State space $\mathcal{S}$, action space $\mathcal{A}$,discount rate $d$ and learning rate $\alpha$
  Initialize $Q(\mathcal{S}, \mathcal{A})$ with zeros
  repeat forever (for each episode):
      Initialize $S$
      Generate action $A$ under $\epsilon-$soft policy from present $V(A)$
      repeat forever (for each episode):
          Take action $A$, observe $R, S'$
          $Q(S, A) \leftarrow Q(S, A) + \alpha[R + d \max_a Q(S', a) - Q(S, A)]$
          $S \leftarrow S'$
          Choose $A$ from $S$ using $\epsilon$-soft based on the present $Q$
      until S is terminal
```

3 Methodology

Inspired by our main reference paper, we decide to implement Q-Learning strategy compared with XCS mentioned by Hu and apply some modifications to the algorithm. Our overall framework is described as follows:

- **Step1**: Retrieve 2021 full year 5-min historical high-frequency market and financial data of all S&P 500 stocks in 4 industries from EOD HISTORICAL DATA.
- **Step2**: Calculate intra-day technical indicators and quarterly fundamental indicators using Python and transform features into dummy variables.
- **Step3**: Apply Q-learning method to generate a bunch of effective trading rules.
- **Step4**: Backtest four quantitative strategies based on the rules in step2. Strategies are:

  (1) Use 6 intraday technical indicators only;

  (2) Combine 6 technical indicators with 7 classical momentum indicators;

  (3) Combine 6 technical indicators with 7 fundamental indicators;

  (4) Change the mapping method of Boolean Sequence Mapping.
- **Step5**: Analyze model's performance on each market sector using cumulative returns, volatility, Sharpe Ratio (SR), and maximum drawdown.

3.1 Q-Learning For Rules Generation

We use *Q-Learning* to generate the trading rules pool, where the rules that are pairs of certain patterns of sequential indicators and corresponding actions are contained in. The training process is composed of four steps, perception, action selection, feedback and learning. Figure 2 uses an example to show the steps of training rules in QL during a single loop.

First, encode the equity market environment into a sequence of 0/1 labels with length N, where N is the total number of indicators and 0/1 refers to the status for each of it. Then the

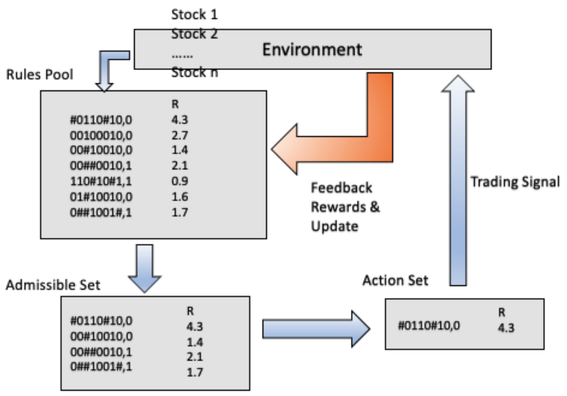


Figure 2: Flow Chart of Rules Generation

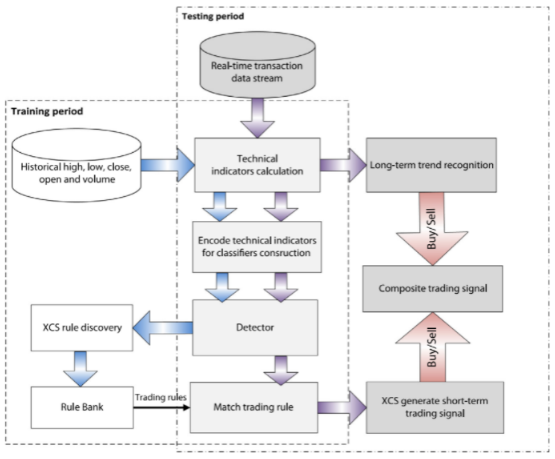

Testing period
Real-time transaction data stream
Training period
Historical high, low, close, open and volume
Technical indicators calculation
Long-term trend recognition
Buy/Sell
Encode technical indicators for classifiers consruction
Composite trading signal
XCS rule discovery
Detector
Buy/Sell
Rule Bank
Trading rules
Match trading rule
XCS generate short-term trading signal

Figure 3: Flow Chart of Trading System [3]

system will initialize randomly a bunch of trading rules into its memory pool $\mathcal{M}$. For example, in the sequence of "0010#1#,1", the '0/1' before the comma refers to the status of indicators, the '#' means not care the status of this indicator, and the '1' behind the comma gives the trading action 'BUY', vice versa.

For each step of the training process, the AI engine will search through every stock and compare their indicator binary sequences with rules in pool. The ones that match any of the rules will be stored in admissible set, which is denoted as $\mathcal{A}$. Next, the engine will determine the trading action for each stock in $\mathcal{A}$ via selecting the rule with the greatest reward. Additionally, in order to balance between exploitation and exploration, the engine will randomly select a new rule with a small probability, denoted by ϵ, that we give. After executing the selected actions one by one, the environment will step forward from t_k to t_{k+1}, and get gain/loss for each stock in holdings from the market. This feedback will be sent to the responding rule in memory pool and update the system. Rules that always have low or negative rewards will be eliminated from the pool.

3.2 Back-tested Trading System

Figure 3 is cited from Hu's paper, it displays a full process of the back-tested trading system. In order to obtain well balance between portfolio return and risk control, we will calculate its volatility, Sharpe ratios etc. as our risk metrics. We will compare the result with our benchmark, the buy-and-hold strategy.

4 Data Description

4.1 Intraday Data of S&P 500 Stocks

We selected all stocks from four heavily weighted sectors of the S&P 500 index from 2021-01-01 to 2021-12-31 as our training dataset. Here we chose Information Technology, Healthcare, Consumer Staples, and Financials to represent growth and value industries. Four industries contain 74, 65, 31, and 66 stocks respectively.

As the technical indicators we mentioned previously require intraday data, we retrieved the open, close, high, low, adjusted close price, and volume of every five minutes for each stock. Besides, we retrieved one-year fundamental data, including net debt, total assets, net income, total stockholder equity, total revenue, dividends paid, and total cash flow from operating activities, to calculate financial ratios as our fundamental indicators. Both fundamental and intraday price data were obtained from EOD HISTORICAL DATA.

4.2 Indicator List

In our paper, we used both financial ratios and technical indicators that have proved to be effective in recent years. All factors are listed in Table 1 below.

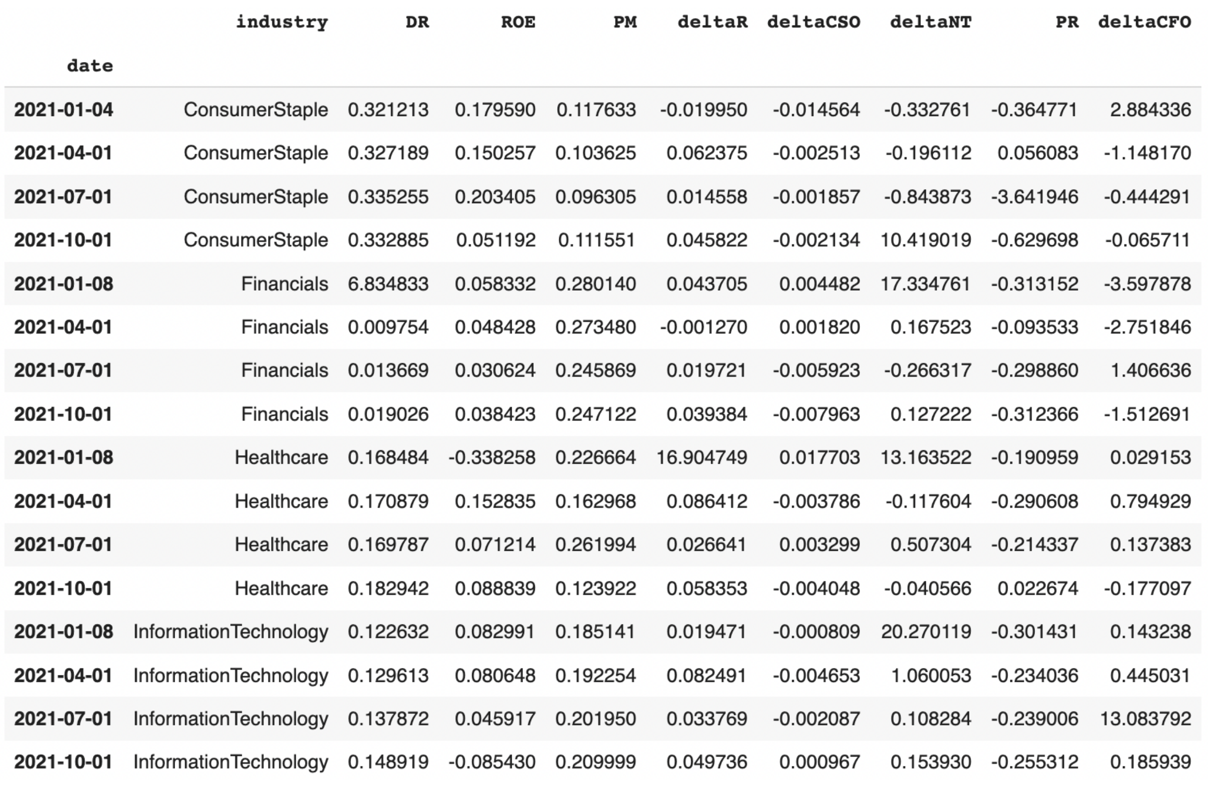
Table 1: Fundamental Financial Ratios & Technical Indicators

Name	Notation	Formula
Debt Ratio	DR	$\frac{Total\ Debt}{Total\ Asset}$
Return on Equity	RoE	$\frac{Net\ Income}{Total\ Equity}$
Profit Margin Ratio	PM	$\frac{Net\ Income}{Revenue}$
% Change of Revenue Growth	ΔR	$\frac{Revenue_t - Revenue_{t-1}}{Revenue_{t-1}}$
% Change of Net Income	ΔNT	$\frac{NI_t - NI_{t-1}}{NI_{t-1}}$
Payout Ratio	PR	$\frac{Dividend\ per\ share}{Earnings\ per\ share}$
Cash from Operating Activity	ΔCFO	$\frac{CFO_t - CFO_{t-1}}{CFO_{t-1}}$
High Frequency Skewness	HFS	$\frac{1}{T}\sum_{n=t}^{n=t-T+1} \frac{\sqrt{N}\sum_{j=1}^{N} r_{i,j,n}^3}{(\sum_{j=1}^{N} r_{i,j,n}^2)^{1.5}}$
Downside Volatility Ratio	DVR	$\frac{1}{T}\sum_{n=t}^{n=t-T+1} \frac{\sqrt{N}\sum_{j=1}^{N} r_{i,j,n}^2 \cdot I_{r_{i,j,n}<0}}{\sum_{j=1}^{N} r_{i,j,n}^2}$
Late Transaction Ratio	LTR	$\frac{1}{T}\sum_{n=t}^{n=t-T+1} \frac{Vol_{i,15:30-16:00,n}}{Vol_{i,j,n}}$
Volumn-Price Correlation	VPC	$\frac{1}{T}\sum_{n=t}^{n=t-T+1} corr(Close_{i,j,n}, \frac{Vol_{i,j,n}}{\sum_j Vol_{i,j,n}})$
Avg. Transaction Outflow Ratio	ASTOR	$\frac{1}{T}\sum_{n=t}^{n=t-T+1} \frac{\sum_{j=1}^{N} Amt_{i,j,n} \cdot I_{r_{i,j,n}<0}}{\sum_{j=1}^{N} Amt_{i,j,n}}$
Large Order Drives Gain	LODR	$\prod_{n=t}^{n=t-T+1} Prod(1 + r_{i,j,n} \cdot I_{\{j \in IdxSet\}}) - 1$

On the one hand, seven financial ratios refer to ones used by Silva, Neves & Horta[4], including return on equity, profit margin ratio, change of Net Income, change of Cash Flow from Operating activities, debt ratio, change of revenue growth, and payout ratio. These financial ratios analyze a company in terms of profitability, liquidity, debt, and growth. If the company's ratio is better than the average of its industries, then we assign "1" to this attribute, otherwise "0". Except for the debt ratio, all ratios are higher the better. Figure 4 summarizes quarterly averages for the four industries' fundamental ratios.

On the other hand, six technical intraday indicators were selected from the report of institute investors, which shows they have relatively high profitability power: high frequency skewness, downside volatility ratio, late transaction ratio, volume-price correlation, and average transaction outflow ratio, and large order drives gain. In formulas, i, j, and n represent the i-th stock, the j-th minute, and the n-th trading day, respectively. T is 5 for weekly stock selection. Similarly, if the company's ratio is better than the average of all stocks, we assign "1" to this attribute; otherwise, "0" is given. All technical indicators are discussed in detail below:

- *High Frequency Skewness* characterizes the rapid rise or fall of stock prices within a day. Assuming that there are two stocks with the same intraday gains, one's gain is accumulated from a sustained and stable slight rise. In contrast, the other's gain is due to a short-term sharp rise. The latter has a greater probability of having a profit reversal in the future. From a risk premium perspective, the latter stock has a high downside risk and higher risk premium. Thus, the smaller this indicator, the better.



date	industry	DR	ROE	PM	deltaR	deltaCSO	deltaNT	PR	deltaCFO
2021-01-04	ConsumerStaple	0.321213	0.179590	0.117633	-0.019950	-0.014564	-0.332761	-0.364771	2.884336
2021-04-01	ConsumerStaple	0.327189	0.150257	0.103625	0.062375	-0.002513	-0.196112	0.056083	-1.148170
2021-07-01	ConsumerStaple	0.335255	0.203405	0.096305	0.014558	-0.001857	-0.843873	-3.641946	-0.444291
2021-10-01	ConsumerStaple	0.332885	0.051192	0.111551	0.045822	-0.002134	10.419019	-0.629698	-0.065711
2021-01-08	Financials	6.834833	0.058332	0.280140	0.043705	0.004482	17.334761	-0.313152	-3.597878
2021-04-01	Financials	0.009754	0.048428	0.273480	-0.001270	0.001820	0.167523	-0.093533	-2.751846
2021-07-01	Financials	0.013669	0.030624	0.245869	0.019721	-0.005923	-0.266317	-0.298860	1.406636
2021-10-01	Financials	0.019026	0.038423	0.247122	0.039384	-0.007963	0.127222	-0.312366	-1.512691
2021-01-08	Healthcare	0.168484	-0.338258	0.226664	16.904749	0.017703	13.163522	-0.190959	0.029153
2021-04-01	Healthcare	0.170879	0.152835	0.162968	0.086412	-0.003786	-0.117604	-0.290608	0.794929
2021-07-01	Healthcare	0.169787	0.071214	0.261994	0.026641	0.003299	0.507304	-0.214337	0.137383
2021-10-01	Healthcare	0.182942	0.088839	0.123922	0.058353	-0.004048	-0.040566	0.022674	-0.177097
2021-01-08	InformationTechnology	0.122632	0.082991	0.185141	0.019471	-0.000809	20.270119	-0.301431	0.143238
2021-04-01	InformationTechnology	0.129613	0.080648	0.192254	0.082491	-0.004653	1.060053	-0.234036	0.445031
2021-07-01	InformationTechnology	0.137872	0.045917	0.201950	0.033769	-0.002087	0.108284	-0.239006	13.083792
2021-10-01	InformationTechnology	0.148919	-0.085430	0.209999	0.049736	0.000967	0.153930	-0.255312	0.185939

Figure 4: Fundamental Data Description

- *Downside Volatility Ratio* has the same logic as high-frequency skewness. Relevant literature shows that the downward trend of high-frequency stock returns measures the probability of a stock price plummeting. It means that a higher downside volatility ratio is better.

- *Late Transaction Ratio* measures the weekly average ratio of trading volume after 15:30 to the trading volume of the whole day. Generally speaking, the intraday trading volume of stocks presents a "U" or "W"-shaped trend, i.e. the trading volume is higher at the opening and closing stages than other trading period. The distribution of trading volume at each time point can reflect the behavioral characteristics of investors and contain additional information. This indicator has a good stock selection effect may be due to:

  (1) high speculation in late trading and prone to price manipulation;

  (2) retail investors are unwilling to bear intraday fluctuations and are more inclined to in late trading, institutional investors tend to trade in the early hours.

  In a word, we expect stocks with smaller late transaction ratio performs stronger.

- *Volume – Price Correlation* is the weekly average correlation between the 5-min closing price of a stock and the stock's 5-min trading volume ratio. Empirical shows that stock with a divergence in volume and price will perform better in the future.

- *Average Transaction Outflow Ratio* equals the transaction amount when the stock falls to total intraday transaction amount. This indicator represents bottom fishing that finds stocks to buy that are making new lows hoping that they will recover and zip to new highs. We expect this indicator to correlate with the stock's future returns positively.

- *Large Order Drives Gain* is calculated as weekly earnings from large orders. Here, we defined large order as the largest 30% of the intraday transaction amount, and Idxset represents the serial number of these large orders on day j. When $r_{i,j,n} < 0$, I is 1; otherwise I is 0. Practice finds that the larger price change driven by large orders, the worse the performance of stock in the future. In other words, a smaller large order drives gain is better.

5 Empirical Analysis

We credit our work to ntrang086[15], whose GitHub project "Q-learning Trading" inspired us a lot. We defined a *Q-learner* class to implement the q-learning method, and several AI trading agents *StrategyLearner* to conduct automatic simulated transaction based on trading rules trained by Q-learner. We trained the strategy by repeatedly feeding historical data to the Q-learner embedded in it for a bunch of epoches. Generally, that is the framework of our codes. By selecting multiple combinations of indicators and mapping methods, we successfully constructed four strategies, backtesting and comparing their performance results in the following sections.

5.1 Pre-defined Parameters & Data Pre-processing

Some basic parameters of our model were listed below:

Table 2: A Brief List Pre-defined Parameters

Parameter	**Value**	**Explanation**
Number of Stock	236	Number of stock used for RL
Size of dataset	59524	Total lines of full dataset
Minimum epoch	50	Minimum epoches required for training. When the actually training loops is more than this value, the strategy would check if the cumulative rewards of epoches are converged. The training process will stop if increase of rewards is slower to a threshold point.
Shape of Q table	$2^{13} \times 2$	Number of state space $\times$ action space
Initial value	10000	Initial stock value for trading strategy
Training period	[2021-01-01, 2021-09-30]	Training dataset
Testing period	[2021-10-01, 2021-12-31]	Testing dataset
Transaction fee	0.2%	Transaction fee equals transaction fee rate times order value
Risk free rate	0.05%	3-month Treasury rate

(Feature signs) Before putting into training pool, we conducted pre-processing operations to get clean and standardized data. It should be noticed that we carefully assign the sign so that all the features are positively related to the daily returns.

(Mapping method) We transformed the feature sequence to boolean pattern by mapping function. The value that larger than the average will be assigned 1 otherwise 0.

(Statistical Description) See from the Figure 5 and Table 3, we found the Financials sector had highest mean return in our data range, while consumer staples had lowest mean returns. Negative skewness and high kurtosis shows that stock returns in these four sectors are NOT normally distributed.

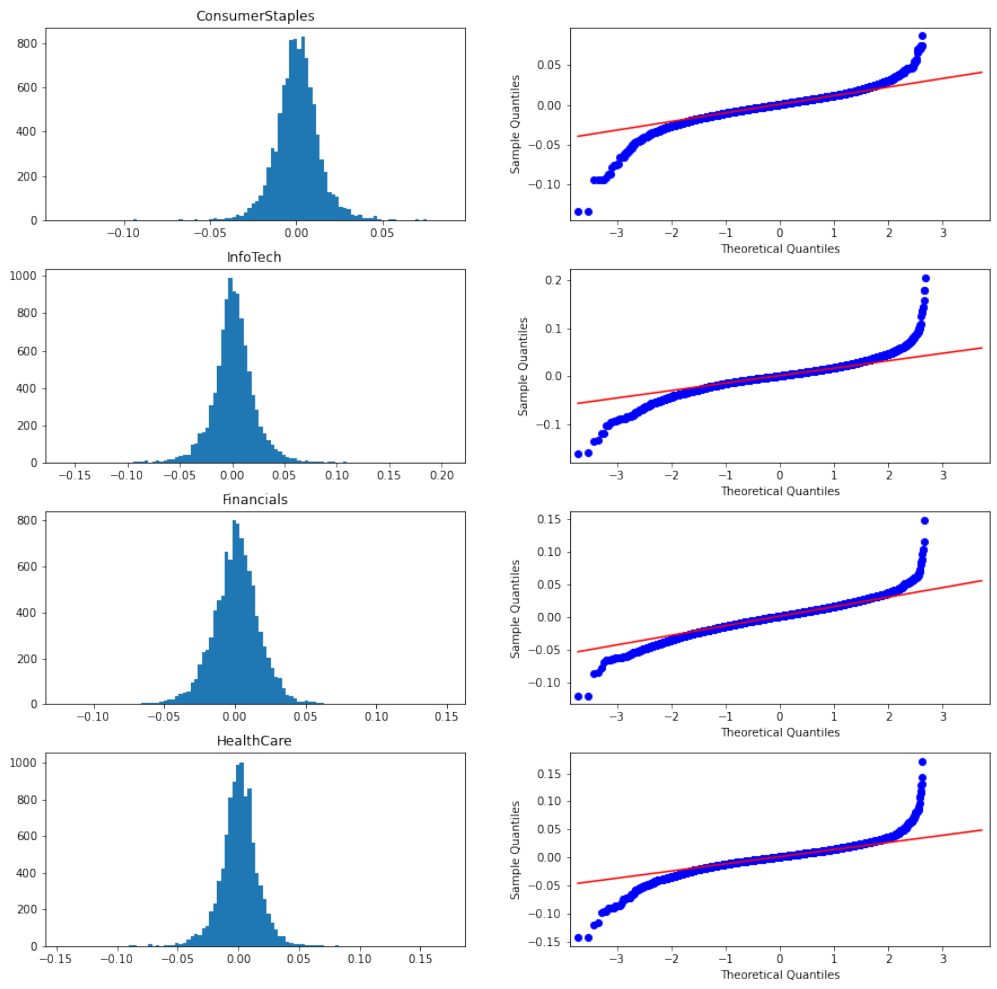


Figure 5: QQ Plot in different sector

Table 3: Statistics Summary of Returns

	Consumer Staples	Financials	Health Care	Information Technology
count	7918	16224	15985	18873
mean	0.000465	0.001340	0.000952	0.000875
std	0.013433	0.017574	0.018214	0.022972
min	-0.134464	-0.325650	-0.659729	-0.752313
25%	-0.006699	-0.008626	-0.007616	-0.009309
50%	0.000373	0.001366	0.001086	0.001107
75%	0.007680	0.011389	0.009792	0.011854
max	0.108847	0.186455	0.382825	0.245952
skewness	-0.159443	-0.234585	-2.399534	-4.745328
kurtosis	5.712014	10.137884	127.803107	148.035729

5.2 Strategy1: Using Effective Intraday Technical Indicators Only

First, we only used the six technical indicators mentioned in Data Description section. Figure 6 shows the learning curve of Strategy1. Though training for more than 50 epoches and each epoch contains 3,000 lines of training data, the cumulative returns did not have much improvement. On the contrary, the learning curve continuously fluctuates up and down, and breakthroughs downwards zero line from time to time. At this point, it's quite straightforward to say that this strategy does not make much sense. To illustrate it directly, we backtested this strategy on CME stock. According to Figure 8, even during the in-sample testing, the AI agent totally did not catch up with the right time to trade.

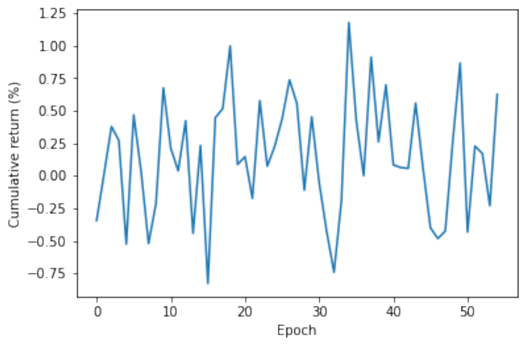


Figure 6: Strategy1 Learning Curve

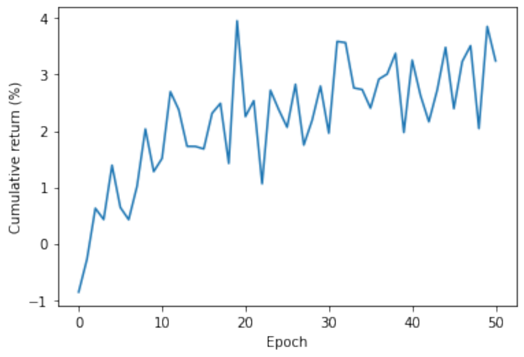


Figure 7: Strategy2 Learning Curve

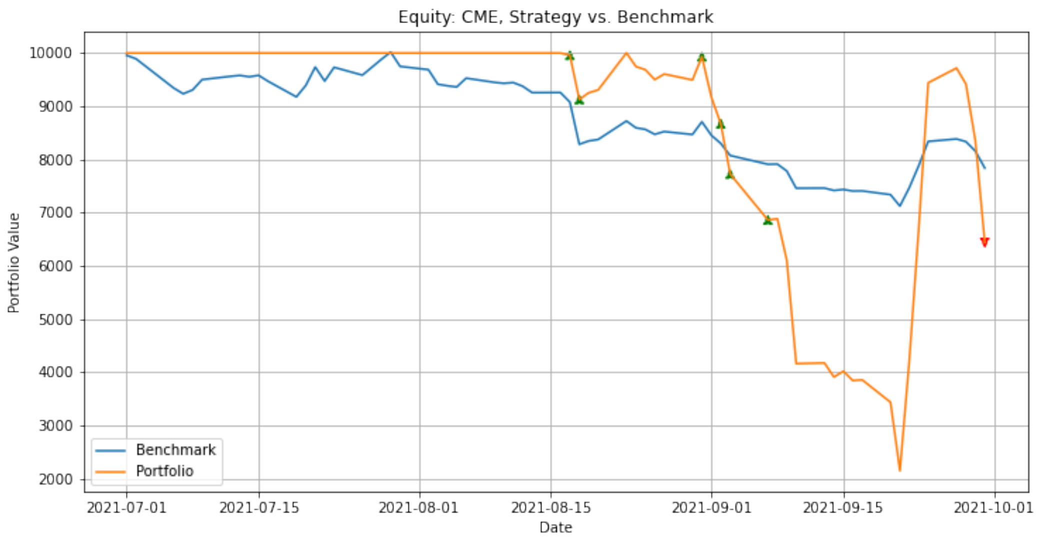


Figure 8: Strategy1 Backtesting Result

Conclusion1

It is quite obvious that the testing result of Strategy1 is not satisfying. Intuitively, six features could grow a state space with dimension $2^6 \times 3$, which is only 192 distinct pairs of (state, action). In contrast to the enormous information and changes of market, it is such a small amount that this state space certainly could not represent all the complex status of equities. Therefore, more indicators need to be considered to develop a more informed and predictable model.

5.3 Strategy2: Combining With Classical Momentum Indicators

Apart from the six effective intraday indicators, we included seven other commonly-used classical indicators, such as smooth moving average, momentum, Bollinger bands,and first 4 alphas of 101 real-life quantitative trading alphas presented by Zura Kakushadze[16].

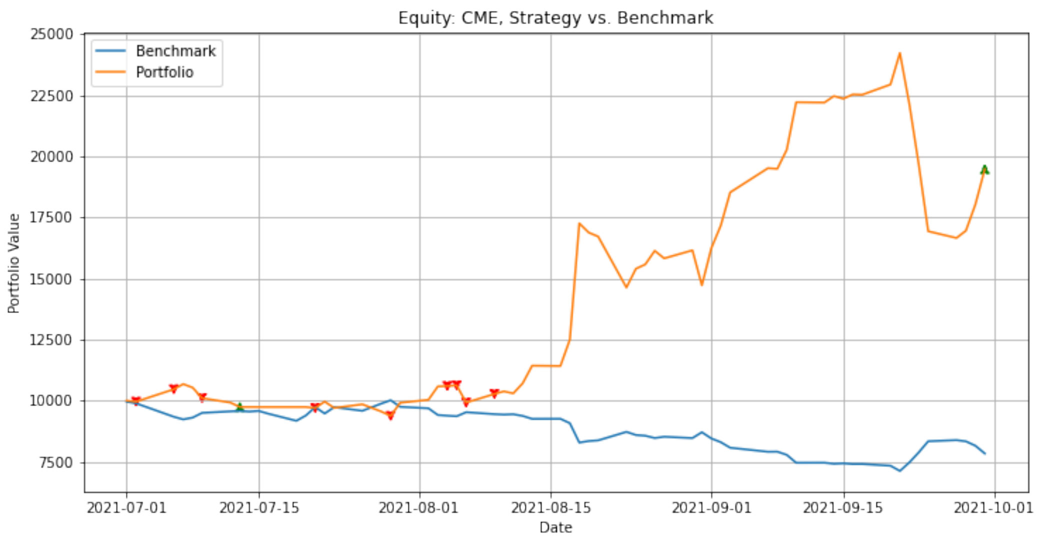


Figure 9: Strategy2 In-Sample Backtesting Result

From Figure 7, we could witness a clear improving process of the strategy, and it earned much more cumulative rewards than Strategy1. Applying the Strategy2 on the same stock CME, as shown in Figure 9, it shows a 'joyful' result with more than 150% returns in just two months. However, when it comes to the out-sample testing, the situation took a turn for the worse. According to Figure 10, the strategy2 made a series of wrong decisions and lost almost 20% of its initial cash in two months. This strategy, though a bit better than Strategy1, is far from a robust, well-performed model.

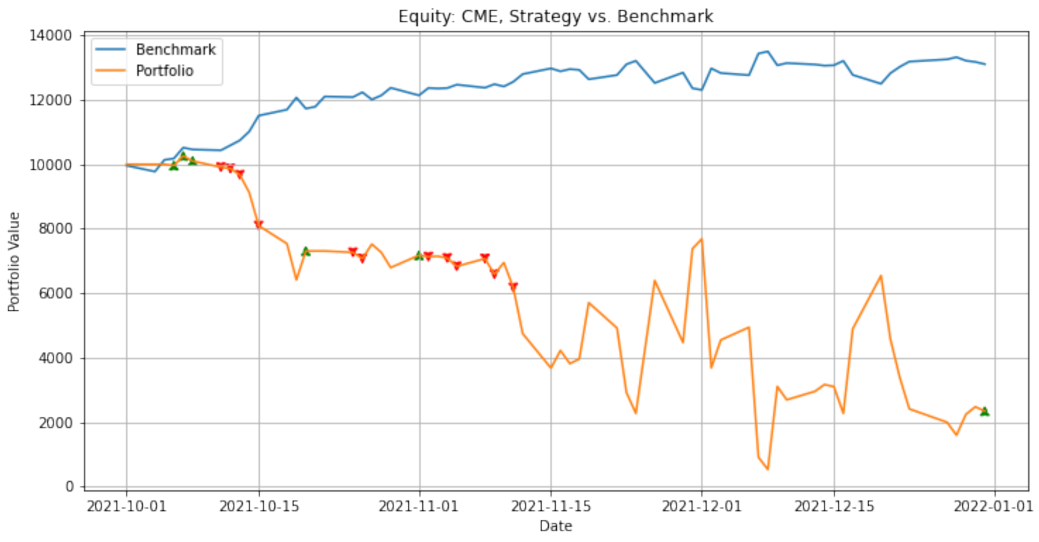


Figure 10: Strategy2 Out-Sample Backtesting Result

Conclusion2

During this process, we refined our model by adding in more features and enlarging the state space. The strategy2 got good profit during in-sample testing, whereas suffered more than 20%

drawndown during out-sample testing. That implies us that (1) the model is probably overfitting; (2) model trained tightly via technical indicators may not have much generalization capacity.

5.4 Strategy3: Combining With Fundamental Indicators

For this step, we combined the six effective features with seven fundamental indicators that were mentioned in previous section.

The Q-learner of Strategy3 took more than 80 epoches to finish training. According to what Figure 13 and Figure 14 is showing, the Strategy3 is also overfitting, well-behaved on trained set but badly-acted on testing set. The maximum drawdown on both in-sample and out-sample testing is more than 60%.

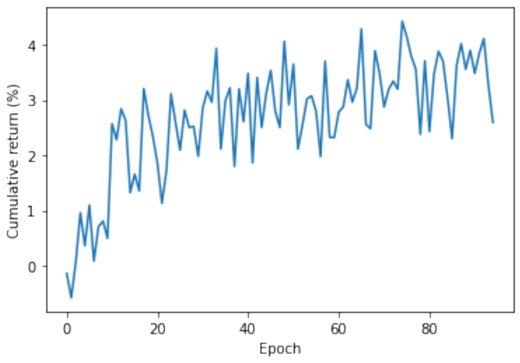


Figure 11: Strategy3 Learning Curve

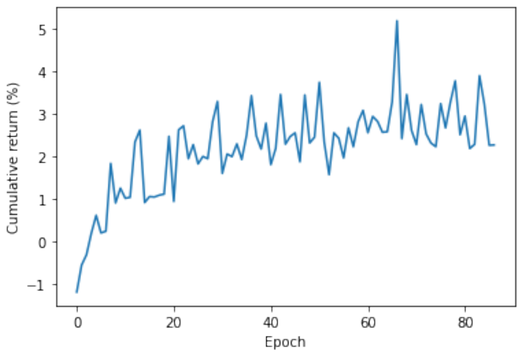


Figure 12: Strategy4 Learning Curve

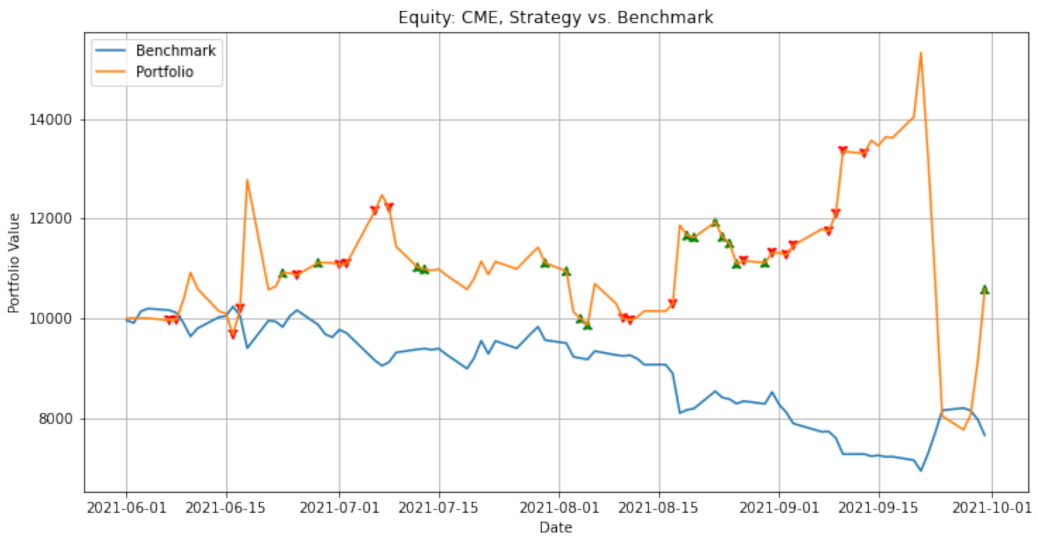


Figure 13: Strategy3 In-Sample Backtesting Result

During this period, we also realized that the lack of processing outliners could introduce large bias to the strategy. For example, as we neglected the influence of corporate activities on equity such as stock split of NVDA on 2021-07-20, the stock price on that day suddenly change to one fourth of it. This caused confusion to AI agent, misleading it to send short orders under the feature pattern before the splitting day. It may seem 'profitable' before that date, but actually

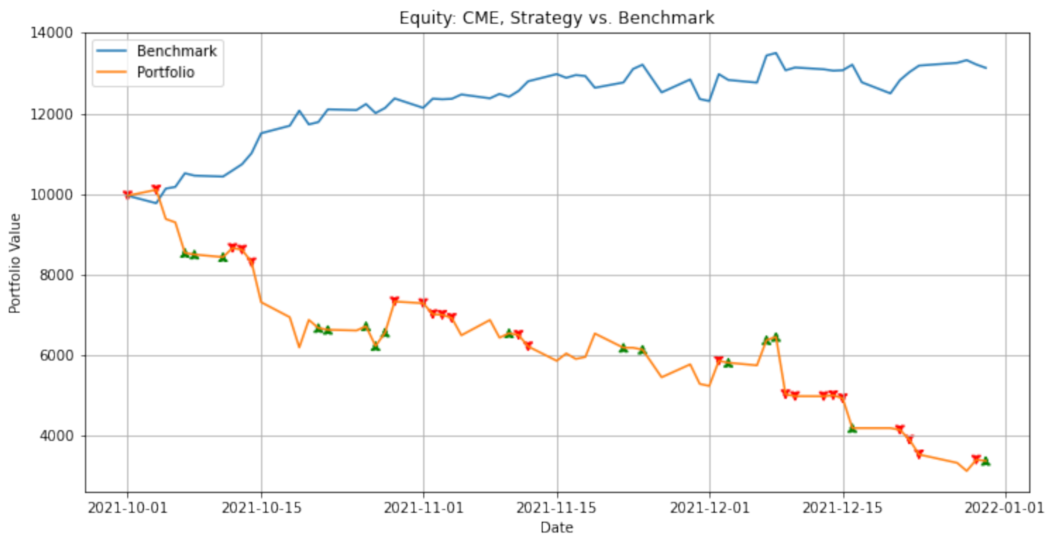


Figure 14: Strategy3 Out-Sample Backtesting Result

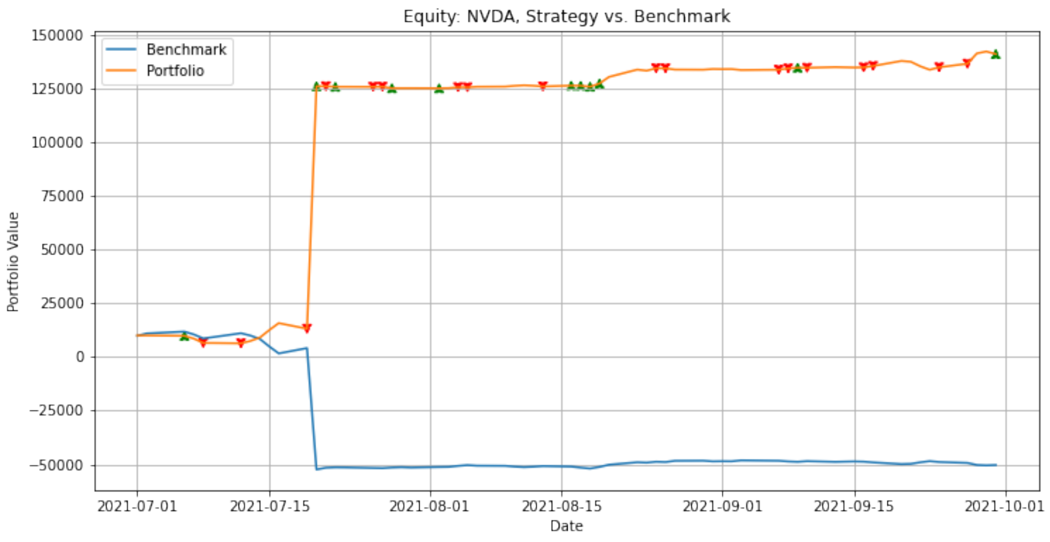


Figure 15: Backtesting Result on NVDA (Special Case)

it is not true. In order to fix this problem, we searched for the timestamp that the stocks changed suddenly and adjusted their prices and volume to be continuous.

Conclusion3

We tried a combination of intraday technical indicators with fundamental ratios to train strategy3. The performance of strategy did not change a lot. It easily got overfitting, and its portfolio value had large volatility, which is clearly against the hypothesis we made. Before drawing final conclusion, we were looking forward to seeking other ways to improve the model.

5.5 Strategy4: Changing the Way of Boolean Sequence Mapping

After trying different selection of indicators, the strategy still was not good enough, so we checked the way we transform features into 0/1 pattern. We realized that simply choosing the overall average as the unique feature's threshold for every stock is too trivial to effectively separate worth-investing stocks with others. Inspired by the thought of moving average indicator, we re-computed each stock's feature threshold as the moving average of the past 5 days, then mapped the feature sequence to boolean sequence according to the thresholds.

Figure 16 and Figure 17 indicate that for the same stock backtested in previous strategies, the Strategy4 has better profitability and generalization ability. It gained nearly 20% returns and performed well during the first month of out-sample testing. However, the model started to lose effect in the second month. In fact, we applied the four strategies on every stock, where they obtained great results in some stocks, while lost a lot of money in other stocks. We attached several the backtesting plots for various symbols in the end of this section.

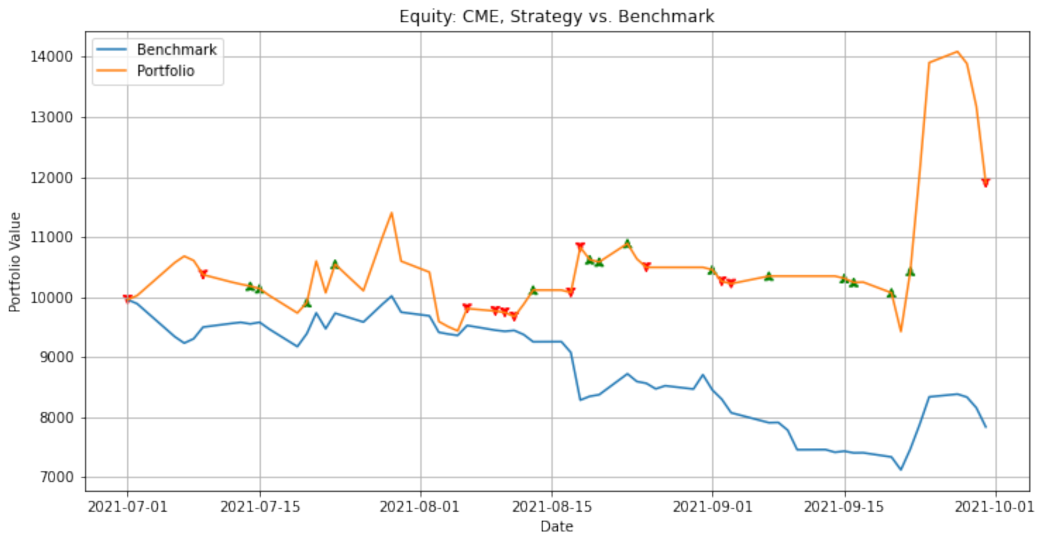


Figure 16: Strategy4 In-Sample Backtesting Result

The Q-table

We laid out the Q-table containing trained rewards embedded in Strategy4 as in Figure 18. It is a 8192×3 array table, where the rows represent the decimal number of corresponding Boolean array, and the three columns represent available actions [SHORT, HOLD, LONG].

We could see how the Q-learner makes transaction decisions based on Boolean feature sequences. For example, the pattern of "000000000000#" would always trigger a "HOLD" decision, since "0000000000000" and "0000000000001" all have the maximum reward under this action.

Another thing also drew our attention that many feature patterns remain zero value under all the actions, which indicated this Q-table is a sparse matrix and was not trained enough. Imagine that once the computer agent encounter a stock with pattern "1111111111111", it has no choice but need to randomly select an action to take! This reminds us that (1) always remember to take precaution measure to prevent AI making investment "randomly; (2) the training set did not contain enough data. In contrast to Q-table with shape of 8192×3, the whole data set with 60k lines could not satisfy the training demand. A full reference of Q-table could be found at **"Q.txt"** in Data folder.

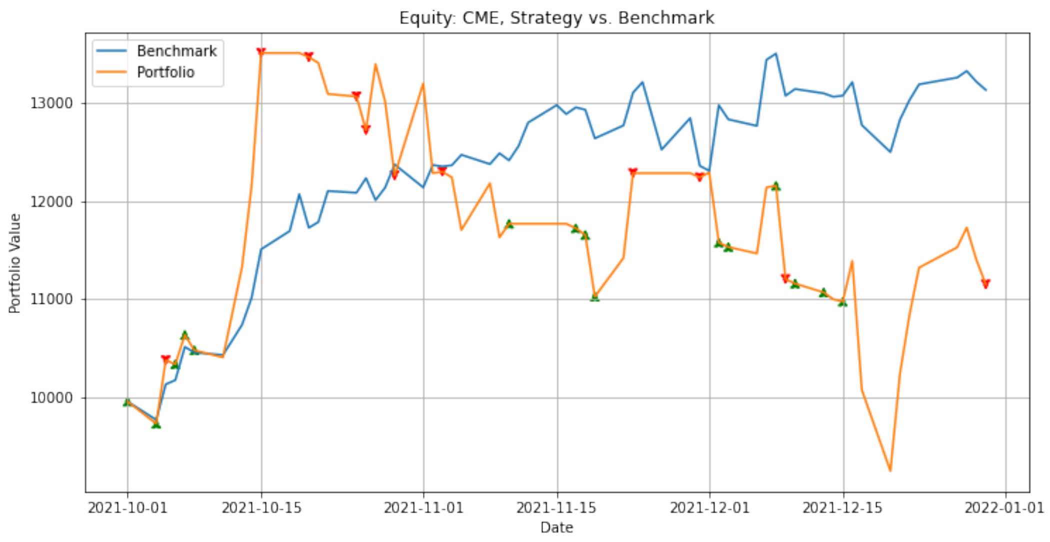
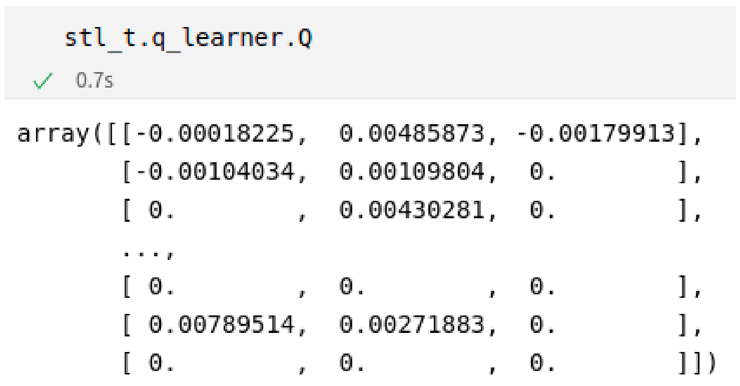


Figure 17: Strategy4 Out-Sample Backtesting Result



```
stl_t.q_learner.Q
✓ 0.7s

array([[-0.00018225,  0.00485873, -0.00179913],
       [-0.00104034,  0.00109804,  0.        ],
       [ 0.        ,  0.00430281,  0.        ],
       ...,
       [ 0.        ,  0.        ,  0.        ],
       [ 0.00789514,  0.00271883,  0.        ],
       [ 0.        ,  0.        ,  0.        ]])
```

Figure 18: Q-table of Strategy4

Conclusion4

In the last version of strategy, we still applied the technical and fundamental factors, but changed the transformation methods on indicators sequence. The resulting strategy showed a bit improvement on out-sample performance in the first month. Nevertheless, it could not beat the benchmark strategy after the whole period. **Based on what we got from all the backtesting results, we could not prove our hypothesis that a hybrid of deliberately constructed technical and fundamental factors would beat the classical momentum indicators on U.S. equity market.** For more details please refer to the explanations in Future Work section.

5.6 Model's Performance on Different Market Sectors

We are also interested in exploring whether the model's prediction capability varies among stocks from different market sectors. We grouped all the stocks by four sectors: *Consumer Staples*, *Information Technology*, *Financials*, *Health Care*, and calculated the average value of statistics of Strategy4's backtesting results.

Table 4 shows our model's in-sample performance from 2021-07-01 to 2021-09-30. Overall, our model has better performance than the buy-and-hold strategy in four industries, with higher Sharpe ratios, cumulative returns, and portfolio values. However, our model exhibits greater volatility than the buy-and-hold strategy in all four sectors. Significantly, the standard deviation of the daily return of our model in the tech industry is about 30 times the benchmark (1.50 vs. 0.06). Consumer Staple and Healthcare perform better average daily return than benchmarks, while Technology and Financials perform worse than benchmarks.

Table 4: In-Sample Performance From 2021-07-01 to 2021-09-30

	Consumer Staples	Information Technology	Financials	Health Care
sharpe_ratio	0.882323	0.762926	0.856663	0.902191
sharpe_ratio_bm	-0.936616	0.127627	0.397975	0.257672
cum_ret	0.104157	0.430156	0.314804	0.285114
cum_ret_bm	-0.044950	-0.060411	0.032064	0.065535
std_daily_ret	0.042768	1.499192	0.172305	0.321569
std_daily_ret_bm	0.011134	0.055007	0.023257	0.035223
avg_daily_ret	0.001841	-0.105492	-0.005786	0.005109
avg_daily_ret_bm	-0.000678	-0.002379	0.000785	0.001117
portvals	11030	14287	13123	12792
portvals_bm	9532	9362	10287	10594

Table 5 summarizes our model's out-sample performance from 2021-10-01 to 2021-12-31. As we can see, our model fails on the test set. No single metric is better than the benchmark. In our model, the tech industry shows abnormal volatility, with a standard deviation of daily return as high as 39%, compared to the benchmark, only 0.07%. Ultimately, the returns of Tech and Healthcare, which represent the growth industry, are both about -50%, which is far worse than the respective benchmark returns of about 10%. In contrast, the performance of the value industry is relatively stable, especially in the financial sector. Its final portfolio value is 9688, which is slightly lower than the original value of 10000, and not far away from the value of the benchmark of 10705.

Table 5: Out-Sample Performance From 2021-10-01 to 2021-12-31

	Consumer Staples	Information Technology	Financials	Health Care
sharpe_ratio	-0.765541	0.088652	0.245125	0.118770
sharpe_ratio_bm	1.996796	1.483485	0.778543	1.573909
cum_ret	-0.279154	-0.549977	-0.029059	-0.486492
cum_ret_bm	0.104533	0.169962	0.074350	0.088943
std_daily_ret	0.357300	39.097547	0.473415	0.767376
std_daily_ret_bm	0.011476	0.071266	0.025154	0.113315
avg_daily_ret	-0.008787	-4.851879	-0.036474	0.016891
avg_daily_ret_bm	0.001585	0.003574	0.001610	0.007883
portvals	7207	4498	9688	5112
portvals_bm	11023	11644	10705	10832

6 Future Work

Based on what we got from all the backtesting results, we could not prove our hypothesis that a hybrid of deliberately constructed technical and fundamental factors would beat the classical momentum indicators on U.S. equity market. Although we did not compute the 'best' model that perfectly support our assumptions, we did manage to refine our model to make it steadily better and better. Furthermore, we give some potential methods to improve our work in the following aspect:

1. Changing mapping method of features. In our strategy we used simple way to mapping the features into dummy variables, i.e. if above industry average or moving average we consider it as 1 otherwise 0. It is a rather simple way to distinguish and labelling features and may not be very accurate. Therefore, we can generate more sophisticated rules for mapping features into dummy variables.
2. Tuning parameters through grid research and cross validation. In our model, we used predetermined parameters for RL. Parameters are essential input for RL and would affect performance of the strategy. And there are several trade-offs we need to consider. For example, the minimum epoch to take needs to be carefaully defined to trade-off between overfitting and underfitting. Moreover, choosing parameters like random action rate and decay rate (their multiplication is the probability of selecting action randomly for each loop) needs considering the trade-off between exploitation & exploration.
3. Extending dataset corresponding to dimension. The dataset is limited in this case. We only have 60 thousand data for RL and the dimension for rule generation is $2^{13} \times 2$ (13 features). Therefore, compared with features, size of our dataset is rather limited and not enough for RL and in the future we may expand our dataset to 2-3 years. Larger dataset combined with different economic regime are more likely to generate stable and accurate results.
4. Enhancing trading details. Other manipulations such as increasing the trading frequency, classifying the equities based on their industry and developing strategies on distinct categories, setting stop-loss/stop-earning point can be used in our strategy for improving model performances.

As time limited, we couldn't realized the above operations to improve our model.

7 Team Contribution

Based on our respective backgrounds and strengths, we divide work as follows: Nicole is responsible for collecting data and coding on rule generation given her strong mathematical and data science skills; Guojun is in charge on backtesting and measuring risk to leverage his knowledge in risk management; and Jiaqi takes care of analysing results and writing report. However, it does not mean that each of us only does our own work. Such a due division only stands for who is responsible for which piece, but the other two are always there to help whenever the person in charge needs it.

References

[1] D. Melas, "Factor allocation model: Integrating factor models and strategies into the asset allocation process," *The Journal of Portfolio Management*, vol. 47, no. 5, pp. 51–57, 2021.

[2] M. Kritzman, "The role of factors in asset allocation," *The Journal of Portfolio Management*, vol. 47, no. 5, pp. 58–64, 2021.

[3] Y. Hu, B. Feng, X. Zhang, E. Ngai, and M. Liu, "Stock trading rule discovery with an evolutionary trend following model," *Expert Systems with Applications*, vol. 42, no. 1, pp. 212–222, 2015.

[4] A. Silva, R. Neves, and N. Horta, "A hybrid approach to portfolio composition based on fundamental and technical indicators," *Expert Systems with Applications*, vol. 42, no. 4, pp. 2036–2048, 2015.

[5] C.-H. Chen, Y.-H. Chen, J. C.-W. Lin, and M.-E. Wu, "An effective approach for obtaining a group trading strategy portfolio using grouping genetic algorithm," *IEEE Access*, vol. 7, pp. 7313–7325, 2019.

[6] Z. Zhang and M. Khushi, "Ga-mssr: Genetic algorithm maximizing sharpe and sterling ratio method for robotrading," in *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020, pp. 1–8.

[7] M. R. Karlsen and S. Moschoyiannis, "Evolution of control with learning classifier systems," *Applied Network Science*, vol. 3, no. 1, pp. 1–36, 2018.

[8] F. Uwano, K. Dobashi, K. Takadama, and T. Kovacs, "Generalizing rules by random forest-based learning classifier systems for high-dimensional data mining," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2018, pp. 1465–1472.

[9] J. Chakole and K. Manish, "Trend following deep q-learning strategy for stock trading," *Expert Systems with Applications*, vol. 34, 2019.

[10] H. Park, M. K. Sim, and D. G. Choi, "An intelligent financial portfolio trading strategy using deep q-learning," *Expert Systems with Applications*, vol. 158, 2020.

[11] M. R. Alimoradi and A. H. Kashan, "A league championship algorithm equipped with network structure and backward q-learning for extracting stock trading rules," *Applied Soft Computing*, vol. 68, pp. 478–493, 2018.

[12] M. Sun and P. Glabadanidis, "Can technical indicators predict the chinese equity risk premium?" *International Review of Finance*, vol. 22, no. 1, pp. 114–142, 2022.

[13] L. Gallo, "Efficient market hypothesis and economic reforms in china," *ScienceOpen Preprints*, 2021.

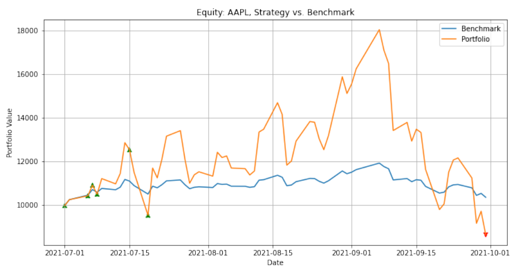
[14] C. J. Neely, D. E. Rapach, J. Tu, and G. Zhou, "Forecasting the equity risk premium: the role of technical indicators," *Management science*, vol. 60, no. 7, pp. 1772–1791, 2014.

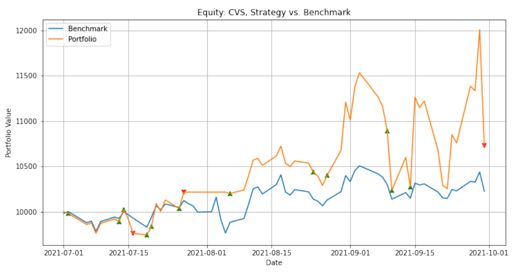
[15] T. Nguyen, "Ntrang086/q_learning_trading: Create a learning agent using q-learning," https://github.com/ntrang086/q_learning_trading, accessed: 2022-04-15.

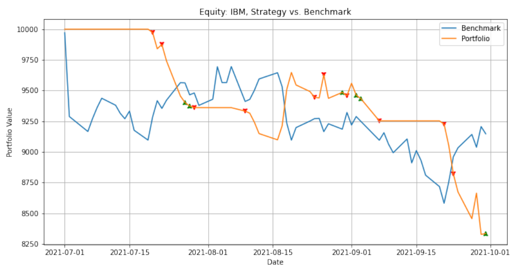
[16] K. Gobeli, "Jangob/101alphas," https://github.com/jaNGOB/101Alphas, accessed: 2022-04-15.

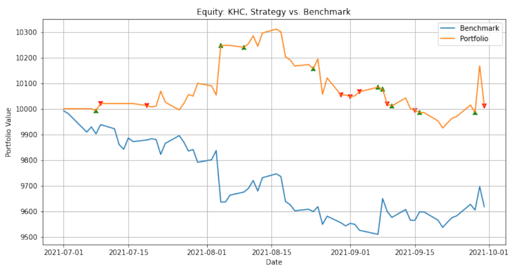
8 Appendix

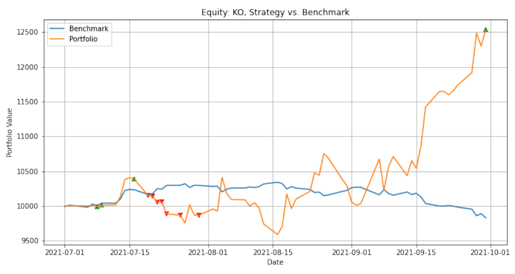
The following plots are created from Strategy4 backtesting results of ['AAPL', 'CVS', 'IBM', 'KHC', 'KO', 'PFE'].

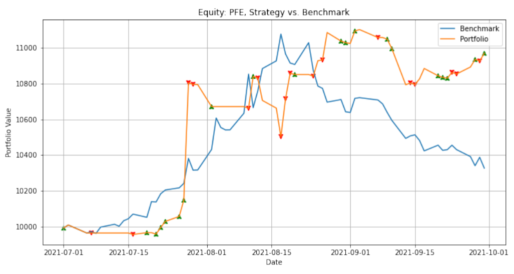

Equity: AAPL, Strategy vs. Benchmark
Benchmark
Portfolio
Portfolio Value
Date


Equity: CVS, Strategy vs. Benchmark
Benchmark
Portfolio
Portfolio Value
Date


Equity: IBM, Strategy vs. Benchmark
Benchmark
Portfolio
Portfolio Value
Date


Equity: KHC, Strategy vs. Benchmark
Benchmark
Portfolio
Portfolio Value
Date


Equity: KO, Strategy vs. Benchmark
Benchmark
Portfolio
Portfolio Value
Date


Equity: PFE, Strategy vs. Benchmark
Benchmark
Portfolio
Portfolio Value
Date