Applied Soft Computing Journal 78 (2019) 685–696

Contents lists available at ScienceDirect

Applied Soft Computing Journal

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

A new approach of integrating piecewise linear representation and weighted support vector machine for forecasting stock turning points

Huimin Tang [a,e], Peiwu Dong [a], Yong Shi [b,c,d,e,*]

[a] School of Management and Economics, Beijing Institute of Technology, Beijing 100081, China
[b] School of Economics and Management, University of Chinese Academy of Sciences, Beijing 100190, China
[c] Key Laboratory of Big Data Mining and Knowledge Management, Chinese Academy of Sciences, Beijing, 100190, China
[d] Research Center on Fictitious Economy and Data Science, Chinese Academy of Sciences, Beijing, 100190, China
[e] College of Information Science and Technology, University of Nebraska at Omaha, Omaha, NE 68118, USA

HIGHLIGHTS

- A fitness function is proposed to select the threshold of PLR automatically.
- A turning point is considered as a period rather than a point.
- Samples are balanced by oversampling and undersampling.
- The relative strength index is used to determine trading signals.
- The proposed model significantly outperforms other models and is the most stable.

ARTICLE INFO

Article history:
Received 28 May 2018
Received in revised form 8 January 2019
Accepted 25 February 2019
Available online 1 March 2019

Keywords:
Turning points (TPs)
Piecewise linear representation (PLR)
Weighted support vector machine (WSVM)

ABSTRACT

Financial data forecasting is one of the most important areas in financial markets. In the stock market, if the stock falls or rises to a point and then rises or falls for a long time, these points are turning points (TPs). Everyone wants to buy or sell stocks at the TP to maximize profits. This paper integrates the piecewise linear representation (PLR) and the weighted support vector machine (WSVM) to forecast stock TPs and proposes several methods to enhance the performance of the PLR–WSVM model. Firstly, a fitness function is proposed to select the threshold of the PLR automatically. Secondly, an oversampling method suitable for the problem of forecasting stock TPs is proposed. The random undersampling combined with the oversampling is used to balance the number of samples. Thirdly, the relative strength index (RSI) is integrated to determine whether the predicted TP is a buying point or selling point. Twenty stocks are used to test the proposed model. The experimental results show that the proposed model significantly outperforms other models. The coefficient of variation of the revenues obtained by the proposed model is the lowest, indicating the proposed model is the most stable.

1. Introduction

Financial data forecasting is one of the most important areas in financial markets. The financial data are complex, nonlinear, high volatility, and noisy [1]. Machine learning algorithms have been successfully applied to predict financial data due to their excellent performance in nonlinear data [2–10]. To make a profit in the financial market, investors are more concerned with making trading decisions than forecasting daily prices [11].

In the supply chain field, companies want to find the optimal cycle length and the optimal inventory quantity [12,13]. A good lot-sizing policy can reduce inventory costs and increase profits [14,15]. Similarly, in the stock market, investors want to find optimal trading cycles and trade stocks at the optimal price [16]. A good trading policy can increase investment profits. If the stock falls or rises to a point and then rises or falls for a long time, these points are turning points (TPs). Everyone wants to buy or sell stocks at the TP to maximize profits. Therefore, it is extremely important to identify the TP of stocks correctly. The piecewise linear representation (PLR) is a method to split a series into several segments, and the maximum error of each segment does not exceed a threshold [17]. With the PLR, the financial time

* Corresponding author at: School of Economics and Management, University of Chinese Academy of Sciences, Beijing 100190, China.
E-mail address: yshi@unomaha.edu (Y. Shi).

https://doi.org/10.1016/j.asoc.2019.02.039
1568-4946/

series can be split into some rising and falling segments, and these split points are TPs.

The PLR has been successfully integrated into machine learning algorithms in recent years, but some problems still exist. The price fluctuations of different stocks are different, and the price fluctuations of stock in different periods are also different. How to properly choose the PLR threshold based on the different price fluctuations is still unresolved. The problem of forecasting stock TPs is a type of imbalance problem, in which the number of TP samples are significantly less than that of ordinary point (OP) samples. Unbalanced samples can decrease the performance of machine learning algorithms [18].

The main contributions of this paper include the following aspects:

Firstly, a fitness function is proposed to select the threshold of the PLR automatically. The threshold of the PLR should be different due to different price fluctuations. This paper proposes a fitness function that focuses on medium or long-term trends rather than short-term rebounds. In this case, the threshold of the PLR can be automatically selected by maximizing the fitness function.

Secondly, an oversampling method suitable for the problem of forecasting stock TPs is proposed. The number of TP samples increased by defining that a TP should be treated as a period rather than a point. The random undersampling [19] combined with the oversampling is used to balance the number of samples in this paper.

Thirdly, the relative strength index (RSI) is integrated to determine whether the predicted TP is a buying point (BP) or selling point (SP). The RSI is a technical curve based on the ratio of the rise and the fall in a certain period and can reflect the prosperity of the stock markets. A stock can be considered overbought or oversold according to its RSI [20]. Therefore, determining the trading signal based on the RSI is appropriate and effective.

Twenty stocks are used to test the proposed model. The experimental results show that the proposed model significantly outperforms other models. The coefficient of variation of the revenues obtained by the proposed model is the lowest, indicating the proposed model is the most stable.

The rest of this paper is organized as follows: Section 2 reviews the literature; Section 3 reviews the PLR and the WSVM; Section 4 describes the proposed model; Section 5 introduces the data and the investment strategy; Section 6 gives the experimental results and discussions; Section 7 presents the conclusions.

2. Literature review

2.1. Financial data forecasting

Financial data forecasting can be mainly divided into three aspects: the financial time series forecasting, the price trend forecasting, and the trading signal forecasting. The financial time series forecasting uses continuous variables as prediction targets. Machine learning algorithms, such as the artificial neural networks (ANN) and the support vector machines (SVM), have been widely used in financial time series forecasting and have better performance than traditional linear models such as autoregressive integrated moving average (ARIMA) [1–3]. Combining the nonlinear algorithm and linear algorithm also showed a good performance in financial time series forecasting [21,22]. The performance of different ANN structures in financial time series forecasting was compared in [9].

The financial time series forecasting typically uses the root mean square error (RMSE) as the performance metric, but it cannot give the accuracy of stocks' rises and falls [4]. Therefore, some researches focus on the price trend forecasting [5–8,10]. The price trend forecasting uses binary variables or multivariate variables as prediction targets. Technical indicators are used to forecast the stock's daily trends [4–7]. The ANN [8] has been used to forecast the trend of stocks during one day, five days and ten days. The random forest (RF) [10] has been used to forecast the trend of stocks during different time periods.

However, investors are more concerned with making trading decisions than forecasting daily prices [11]. The financial time series forecasting and the price trend forecasting mainly focus on daily forecasting. If investors simply buy and sell stocks according to the daily predicted trends, frequent transactions will lead to high transaction fees and low profits. Therefore, some researches focus on the trading signal forecasting [23–25]. The purpose of the trading signal forecasting is to establish a system that predicts when to buy and sell stocks to make a profit in the financial market. The ensemble artificial neural network (EANN) [26] has been used to predict TPs and earned about 5% more than the buy-and-hold strategy (BHS). [27] used the genetic network programming and earned about 5% more than the BHS as well. [28] proposed a trading system based on risk management and company assessment. [29] proposed a trading system based on technical analysis and sentiment analysis. Reinforcement learning is an algorithm based on maximizing reward and has been used to establish a system to trade stocks [16,30,31]. [32] used the league championship algorithm, network structure, and reinforcement learning to extract stock trading rules, which earned about 17% more than the BHS. Recently, the piecewise linear representation (PLR) was integrated into the ANN to predict the trading points of stocks [11,33]. The PLR is a method to split a series into several segments, and the maximum error of each segment does not exceed a threshold [17]. The financial time series can be split into different segments by the PLR, and these split points are TPs. The SVM and other algorithms were combined with the PLR to forecast trading signals as well [17,34–36].

2.2. Nonlinear programming and optimization method

Nonlinear programming has a wide range of applications in real life. Interior point and sequential quadratic programming are two effective optimization algorithms to solve nonlinear programming problems [37–39]. [40] solved the single machine scheduling problem by maximizing the reward in reward-drive systems. [41] investigated a selective maintenance model with stochastic maintenance quality to find the optimal cost maintenance actions. [42] proposed a multi-product and multi-buyer model including the buyers' total cost and the vendor's total cost under penalty, green, and quality control policies and a vendor managed inventory with consignment stock agreement. The mixed-integer nonlinear programming problem is optimized by an outer approximation with equality relaxation and augmented penalty algorithm. [43] measured the return and risk of the portfolio based on the credibility theory, and discusses a class of portfolio adjusting problems for an existing portfolio. [44] proposed a general risk parity portfolio problem, which is convex and can be reduced to a quadratic programming (QP) problem in some cases.

QP is a special kind of nonlinear programming. The essence of SVM is a QP problem, and the optimization problem can be solved by its Lagrangian dual problem. The decomposition method breaks a large QP problem into a set of small QP problems and uses the selected variables per iteration, which can effectively solve the dual problem over large datasets [45–49]. The sequential minimal optimization (SMO) uses only two variables in each step and is an extreme case of the decomposition methods [49]. [50] transferred all evaluations of kernel function to GPU, which can accelerate the training of SVM for large and sparse problems. LibSVM [51] is a popular SVM implementation and uses the SMO-type decomposition method [52].

3. Review of the PLR and the WSVM

The PLR is a method to split a series into several segments [17]. Given a threshold δ, a series can be split into several segments, and the maximum error of each segment does not exceed the threshold δ. The PLR algorithm [53] is shown in Algorithm 1.

The SVM, first proposed by Vapnik [54], is a method based on structural risk minimization. With the principle of structural risk, the SVM can give the best generalizability. The advantage of SVM is that it can deal with small samples, nonlinear data and also high dimensional problems.

To solve nonlinear problems, the SVM switches the input vectors to a higher dimensional feature space. In the higher dimensional space, an optimal separation is found to maximize the interval of two classification instances. SVM is therefore also known as a methodology to obtain the maximum margin hyperplane between two classification instances. The vectors on the edge hyperplane are known as support vectors.

Given a set of points $M = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, $x_n \in R^k$. x_n is the input feature variable, which belongs to either of the two categories according to its label $y_n \in \{-1, +1\}$. For the nonlinearly separable problem, a high-dimensional maximum margin hyperplane can be represented as follows:

$$y = b + \sum w_i y_i K(x(i), x) \tag{1}$$

where x is the test example, $x(i)$ is the support vector, $K(\cdot)$ is the kernel function. The kernel function can map low dimensional feature variables to high dimensional feature variables. Common choices of the kernel functions include the polynomial kernel function $K(x_i, x_j) = (x_i x_j + 1)^d$ and the Gaussian radial basis function (RBF) $K(x_i, x_j) = \exp\left(-g(x_i - x_j)^2\right)$.

To obtain an optimal hyperplane, the above problem can be converted into the following QP problem:

$$\begin{aligned} &\text{Minimize } \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \\ &\text{Subject to } \begin{cases} y_i(w \cdot \phi(x_i) + b) + \xi_i \geq 1, i = 1, \ldots, n \\ \xi_i \geq 0, i = 1, \ldots, n \end{cases} \end{aligned} \tag{2}$$

where C is the penalty factor, ξ_i is the slack variable, $\cdot$ is the dot product, $\phi(\cdot)$ is the nonlinear map, w is the normal vector of the hyperplane, and b is the bias.

When each training sample x_i has a weight μ_i, the penalty factor C is replaced by $\mu_i C$ in the SVM model, and the SVM becomes the WSVM.

4. Proposed model

In the proposed model, the PLR is used to generate TPs and OPs, and the threshold of the PLR is automatically selected by a fitness function. The sample weights are calculated by the change rate of price between adjacent TPs. The oversampling and the undersampling are used to balance the number of samples. The WSVM is used to forecast the TPs, and the trading signals are determined by the RSI rule and the DODS. Fig. 1 shows the flowchart of the proposed model, and the detailed methods are introduced in the following subsections.

4.1. Input indicators

The stock indicators are shown in Table 1 [6,17,35]. KDJ is a momentum indicator and has been widely used to analyze the stock trends. There are three indicators, K, D, and J, in KDJ, and they are calculated as follows:

$$K(t) = \frac{2}{3} \times K(t-1) + \frac{1}{3} \times \frac{p_c(t) - L_n}{H_n - L_n} \times 100 \tag{3}$$

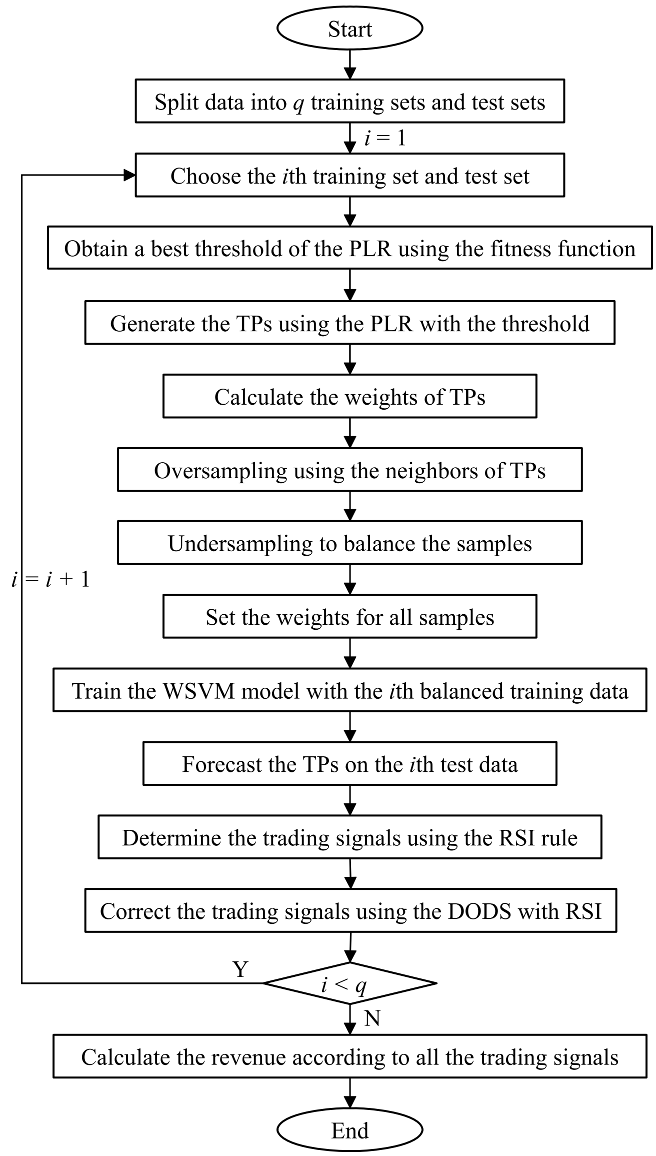

Fig. 1. The flowchart of the proposed model.

where L_n and H_n are the lowest price and the highest price among n days respectively.

$$D(t) = \frac{2}{3}D(t-1) + \frac{1}{3}K(t) \tag{4}$$

If there are no K and D values the previous day, they can be set to 50.

$$J(t) = 3K(t) - 2D(t) \tag{5}$$

Usually, when the K value is less than the D value, and the K line breaks through the D line, it is a buying signal. When the K value is greater than the D value, and the K line falls below the D line, it is a selling signal. Therefore, the different types of KDJ are used as another input indicator.

The days d chosen to calculate the $BIAS_d$ are 5, 10, 20, 30 and 60. The days d chosen to calculate the RSI_d are 6, 12 and 24. Therefore, there are a total of 23 indicators used as input variables.

Algorithm 1 Top-down algorithm

If the maximum error of the segment is higher than the threshold δ:

Split the segment into two segments from the position of the maximum error.

If the maximum error of the left segment is higher than the threshold δ:

  Split the left segment using Algorithm 1.

If the maximum error of the right segment is higher than the threshold δ:

  Split the right segment using Algorithm 1.

Table 1
Stock indicators and their formulas.

Indicator	Formula	Description
ATP	$\widetilde{p}(t) = TM(t)/TV(t)$	The average transaction price
ALT	$(p_h(t) - p_l(t))/p_l(t)$	The amplitude of the price movement
ITL	$\begin{cases} 1 & \text{if } p_c(t) > p_o(t) \\ 0 & \text{if } p_c(t) \le p_o(t) \end{cases}$	The index for the type of K-line
CATP	$(\widetilde{p}(t) - \widetilde{p}(t-1))/\widetilde{p}(t-1)$	The change rate of average transaction price to the previous trading day
CTM	$(m(t) - m(t+1))/m(t-1)$	The change rate of transaction money compared to the previous trading day
TR	$TV(t)/TS(t)$	The turnover rate
CTR	$(TR(t) - TR(t-1))/TR(t-1)$	The change rate of turnover rate
PCCP	$\begin{cases} \dfrac{2p_c(t) - p_l(t) - p_h(t)}{p_h(t) - p_l(t)} & \text{if } p_h(t) \ne p_l(t) \\ 1 & \text{if } p_h(t) = p_l(t) \end{cases}$	The position constant of the closing price
PCTV	$\begin{cases} \dfrac{2TV(t) - TV10_{\min}(t) - TV10_{\max}(t)}{TV10_{\max} - TV10_{\min}} & \text{if } TV10_{\max} \ne TV10_{\min} \\ 1 & \text{if } TV10_{\max} = TV10_{\min} \end{cases}$	The position constant of transaction volume on ten days
RDMA	$(MA_{10} - MA_{50})/MA_{50}$	The relative differences of MA between the short run and the long run
RMACD	$RDIFF - \left(\sum_{i=i-8}^{t} RDIFF(i)\right)/9$	The relative moving average convergence–divergence
BIAS_d	$(p_c(t) - MA_d)/MA_d$	The degree of price deviating from the average size
KDJ	Please see Eqs. (4)–(6)	The stochastic oscillator
ITS	$\begin{cases} 1 & \text{if } K(t) \le D(t) \text{ and } K(t) > K(t-1) \\ -1 & \text{if } K(t) \ge D(t) \text{ and } K(t) < K(t-1) \\ 0 & \text{otherwise} \end{cases}$	The index for the type of KDJ
RSI_d	$100 - 100/(1 + RS_d)$	The relative strength index

$p_o(t)$, $p_c(t)$, $p_h(t)$, $p_l(t)$ are the opening price, the closing price, the highest price, and the lowest price on day t respectively;
$TM(t)$, $TV(t)$, $TS(t)$ are the transaction money, the transaction volume, and the size of tradable shares on day t respectively;
$MA_d = \frac{1}{d}\sum_{i=t-d+1}^{t} p_c(i)$; $RDIFF = (MA_{12} - MA_{26})/MA_{26}$; $RS_d = \frac{\frac{1}{d}\sum_{i=t-d+1}^{t} max(0, p_c(i) - p_c(i-1))}{\frac{1}{d}\sum_{i=t-d+1}^{t} |min(0, p_c(i) - p_c(i-1))|}$.

4.2. Generate TPs using the PLR

The data set is split into q training sets and test sets sequentially, and the q is calculated as follows [17,35]:

$$q = \lceil (r - r_1)/r_2 \rceil \tag{6}$$

where r is the data set size, r_1 is the training set size, and r_2 is the test set size. The example of splitting data set is shown in Fig. 2.

For each training set, the PLR is used to obtain the TPs of stocks. The points with trough or peak are classified to TPs, and the other points are classified to OPs. The threshold δ has an important influence on the TPs generated by the PLR. As can be seen in Table 2, the smaller the threshold value, the more TPs the PLR generates.

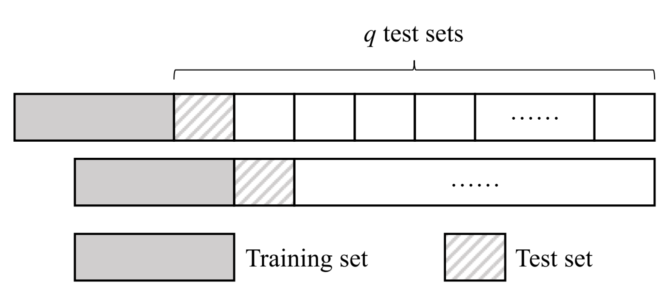

Fig. 2. The example of splitting data set sequentially.

As the threshold decreases, the number of TPs increases, but the PLR will more easily generate some TPs occurred within a

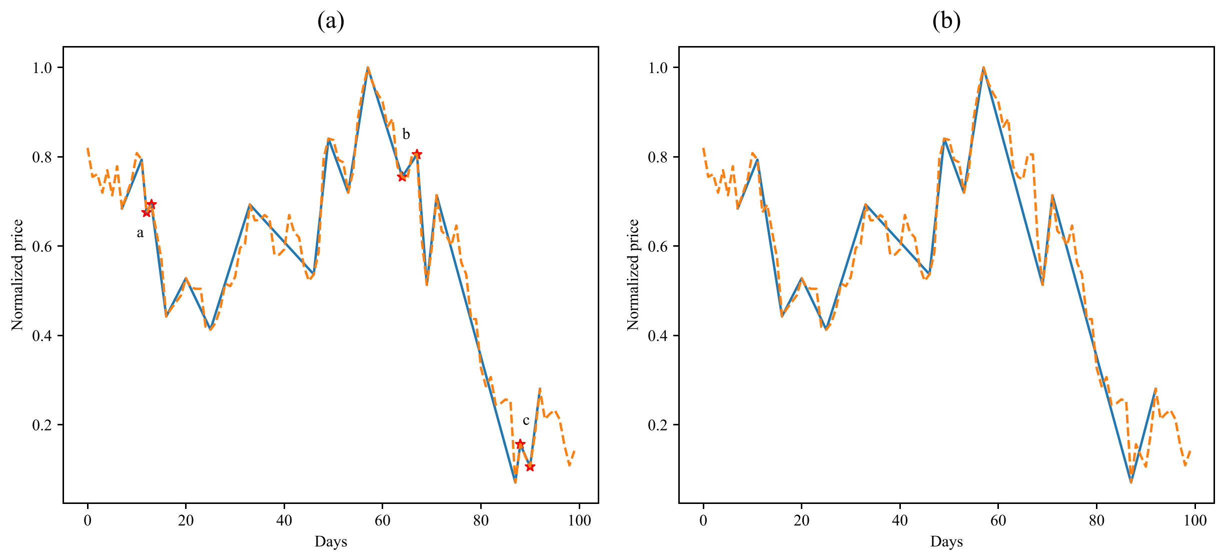

Fig. 3. The TPs generated by the PLR.

Table 2
Number of TPs under different thresholds.

	$\delta = 0.01$	$\delta = 0.05$	$\delta = 0.10$	$\delta = 0.15$	$\delta = 0.20$
Number of TPs	64	37	21	14	8
Percent of TPs	32%	18.5%	10.5%	7%	4%

short period. These points are only short-term rebound points rather than TPs. Fig. 3 shows an example. In Fig. 3(a), three pairs of TPs, a, b and c, occurred within a short period. The three pairs of TPs are still in a falling or rising trend and should not be TPs. Therefore, the more reasonable TPs are shown in Fig. 3(b) after eliminating the short-term rebound points. Additionally, because of the different price fluctuations of different stocks, it is unreasonable to set the same threshold δ for different stocks.

To overcome the above problems, this paper proposes a fitness function that focuses on medium or long-term trends rather than short-term rebounds:

$$r_{PLR} = revenue_{PLR} - \alpha_{PLR} \sum_{i=2}^{n} \max\left(\beta_{PLR} - (x_i - x_{i-1}), 0\right) \quad (7)$$

where $revenue_{PLR}$ is a revenue calculated according to the TPs generated by the PLR, α_{PLR} is a penalty factor, β_{PLR} is a threshold to control the period within which the TPs should not exist, and x_i is the day of the ith TP.

Because it can accurately buy at a low price and sell at a high price, the larger the number of TPs, the larger the first item of the formula ($revenue_{PLR}$). But with the second item, the trades happened within β_{PLR} days will be punished. The smaller the number of days between trades, the greater the penalty is. Therefore, the threshold δ of the PLR can be automatically selected by maximizing the fitness function r_{PLR}.

Table 3
The comparison of different models.

Stock	Basic (PLR–WSVM)	Basic+RSI rule	Basic+RSI rule+balanced sample	Proposed model	IPLR–WSVM	PLR–ANN	BHS
600736	−22.37	−3.37	−1.76	57.73	148.04	−11.44	−11.70
600197	7.75	−9.41	6.33	16.79	−3.85	11.95	−24.05
600211	31.43	58.62	48.29	80.40	0.40	40.62	−13.20
600694	17.46	36.27	34.82	33.53	24.49	6.70	−19.81
600351	61.16	32.04	28.82	46.27	−2.51	24.72	−34.94
600488	−2.11	−6.69	41.96	46.74	−5.27	43.26	−27.35
600054	10.37	−8.34	36.94	32.65	5.88	−0.24	−16.03
600019	−22.16	−18.37	−8.23	−3.62	−17.71	−3.02	−31.84
600058	−16.20	50.09	73.56	106.04	173.14	−22.95	−14.46
600682	14.81	22.74	13.69	19.86	12.94	32.95	−4.05
600597	5.85	3.20	30.72	23.30	56.79	9.57	−3.82
600066	69.71	55.28	44.38	−3.85	2.95	44.44	−9.26
600881	−11.00	8.71	15.95	8.46	−35.37	28.74	−3.76
600228	7.58	99.92	24.16	87.96	116.65	−20.52	−1.75
600697	−12.31	−8.42	10.27	−0.96	−12.28	−12.79	1.90
600107	−3.29	18.85	−36.53	−22.24	48.06	79.43	16.27
600053	−43.32	58.72	72.89	95.39	−31.03	7.53	9.36
600051	45.01	−35.79	−14.17	23.76	−42.65	66.08	16.43
600163	20.99	−1.70	117.32	98.40	44.33	23.04	11.18
600167	−37.64	21.60	58.15	128.20	−34.97	−29.40	19.47
Average	6.08	18.70	29.88	**43.74**	22.40	15.93	−7.06
σ	29.10	32.49	34.06	41.61	58.97	28.81	15.80
CV	4.78	1.74	1.14	**0.95**	2.63	1.81	2.24

Table 4
The comparison between automatically selecting δ and using a constant δ.

Stock	$\delta = 0.01$	$\delta = 0.05$	$\delta = 0.10$	$\delta = 0.15$	$\delta = 0.20$	Automatically
600736	2.08	−22.91	106.52	92.28	−6.15	57.73
600197	−1.97	−2.34	19.46	25.14	11.03	16.79
600211	62.81	107.50	62.63	75.37	44.54	80.40
600694	18.50	23.49	2.40	22.55	17.05	33.53
600351	33.77	6.55	33.85	16.01	49.81	46.27
600488	24.26	52.95	−3.21	−3.21	32.80	46.74
600054	17.16	32.65	20.69	32.65	−5.98	32.65
600019	−4.48	−18.78	−21.23	−6.29	−5.94	−3.62
600058	122.50	96.47	69.39	27.62	116.68	106.04
600682	18.33	18.33	17.68	4.32	18.33	19.86
600597	19.67	27.00	25.00	30.47	6.14	23.30
600066	−10.03	−7.89	1.89	−3.85	2.89	−3.85
600881	34.89	35.75	4.69	−0.14	22.63	8.46
600228	4.08	67.44	81.08	101.66	69.36	87.96
600697	12.29	12.29	9.00	−0.32	−5.87	−0.96
600107	−22.24	−22.24	−22.24	−0.59	−6.83	−22.24
600053	53.30	61.02	57.65	34.21	37.83	95.39
600051	17.03	11.68	−25.96	22.60	−20.92	23.76
600163	52.66	91.50	80.86	94.03	58.40	98.40
600167	107.82	132.54	80.91	112.30	92.04	128.20
Average	28.12	35.15	30.05	33.84	26.39	**43.74**
σ	35.89	44.10	38.49	38.08	35.55	41.61
CV	1.28	1.25	1.28	1.13	1.35	**0.95**

Table 5
The comparison of different sample weights.

Stock	None	[0.1, 1]	[0.5, 1]	[0.5, 2]	[1, 2]	[1.5, 2]	[1.5, 3]
600736	57.73	103.16	63.02	63.39	57.73	63.02	63.02
600197	13.64	11.03	11.03	11.03	16.79	16.79	16.79
600211	80.40	66.72	65.71	76.11	80.40	80.4	69.11
600694	34.92	26.41	34.92	34.92	33.53	33.53	33.53
600351	46.27	−16.69	46.27	11.14	46.27	31.35	31.35
600488	42.23	43.55	43.55	43.55	46.74	45.40	45.40
600054	32.65	32.65	32.65	32.65	32.65	32.65	32.65
600019	−4.48	−4.48	−3.62	−3.62	−3.62	5.25	5.25
600058	106.04	80.01	80.37	80.37	106.04	106.04	78.96
600682	19.86	17.68	19.86	19.86	19.86	19.86	19.86
600597	18.23	27.00	22.87	27.00	23.30	23.30	23.30
600066	2.89	−3.85	−3.85	−3.85	−3.85	2.89	2.89
600881	8.46	27.16	9.56	9.56	8.46	8.46	8.46
600228	61.96	70.59	61.96	87.96	87.96	87.96	87.96
600697	−8.70	−5.87	−0.32	−0.32	−0.96	−0.96	−0.96
600107	−22.24	−22.24	−22.24	−22.24	−22.24	−22.24	−22.24
600053	63.58	52.73	59.66	47.41	95.39	86.28	86.28
600051	23.76	17.03	9.93	9.93	23.76	20.31	20.31
600163	98.40	62.81	107.91	98.16	98.40	98.40	98.40
600167	128.20	118.07	157.55	122.80	128.20	128.20	128.20
Average	40.19	35.17	39.84	37.29	**43.74**	43.34	41.42
σ	39.43	38.14	41.8	38.36	41.61	40.54	38.36
CV	0.98	1.08	1.05	1.03	0.95	0.94	**0.93**

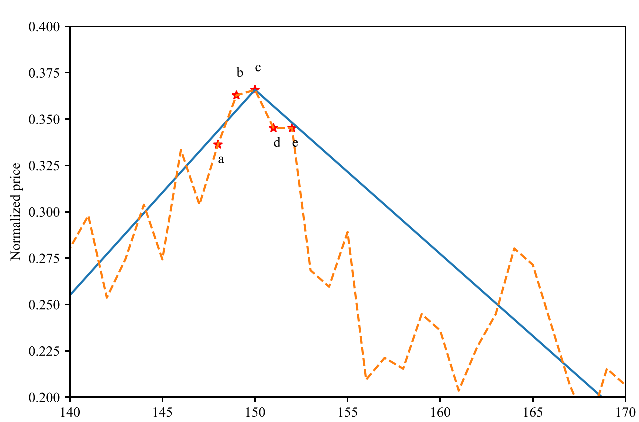

Fig. 4. Example for the neighbor window.

Different samples, TPs and OPs, are differently weighted as follows [35]:

$$\mu_t^{(tr)} = \begin{cases} \left|p_c\left(ns_t\right) - p_c\left(t\right)\right| / p_c\left(t\right) & \text{if } t \text{ is a TP} \\ \lambda * \min\limits_{s_t} \mu_{s_t} & \text{if } t \text{ is a OP} \end{cases} \tag{8}$$

where $p_c\left(\cdot\right)$ is the closing price, s_t is the TP, ns_t is the next TP, and λ is a scaled factor. The sample weights are normalized into [1, 2] by the following equation:

$$\mu_i^{(tr)} = 1 + \frac{\mu_i^{(tr)} - \mu_{\min}^{(tr)}}{\mu_{\max}^{(tr)} - \mu_{\min}^{(tr)}} \tag{9}$$

4.3. Forecast TPs using the WSVM

Generally, unbalanced samples will decrease the accuracy of a classifier [18]. As can be seen in Table 2, the number of OPs is larger than that of TPs. Since the fitness function is used to avoid generating TPs occurred within a short period, the percentage of

Algorithm 2 DODS with RSI

for each trading day t:

if t is a BP, the downward trend is alleviated on the day $t+1$ $(cr(t+1) > \alpha * cr(t))$, and the RSI_{12} is below 50 on the day $t+1$:

Cancel the trading on the day t and set the day $t+1$ as a BP

elseif t is an SP, the upward trend is alleviated in the day $t+1$ $(cr(t+1) < \beta * cr(t))$, and the RSI_{12} is above 50 in the day $t+1$:

Cancel the trading on the day t and set the day $t+1$ as an SP

else:

Cancel the trading on the day t and set the day $t+1$ as an OP

TPs is usually between 7.5% and 17.5%. Therefore, the oversampling and the undersampling are used to balance the number of samples.

The oversampling method typically generates virtual samples to balance the data set [55]. However, in the problem of forecasting TPs, TPs are labeled according to financial experts or algorithms. Therefore, this paper considers a TP should be a period rather than a point. In this case, the neighbors of TPs generated by the PLR should also be labeled as TPs. A neighbor window nw_{tp} ($nw_{tp} \geq 0$) is defined to control the neighbors that should be labeled as TPs. With the neighbor window nw_{tp}, the number of TPs will be expanded $2nw_{tp}$ times. The weights of the neighbor TPs are the same as their central TP. Further, the characteristic of a TP's neighbors is similar to that of the TP, and it may be hard to classify whether the neighbors are the TPs or not. Therefore, a neighbor window nw_{op} ($nw_{op} \geq 0$) is defined to control the neighbors of TPs that should not be labeled as OPs. For example, as can be seen in Fig. 4, point c is the TP obtained by the PLR. Given $nw_{tp} = 1$ and $nw_{op} = 1$, the points b and c are labeled as TPs, and the points a and e are discarded and are not labeled as OPs.

Note that in some cases, some points generated by the PLR are not TPs. For example, in Fig. 5, an uptrend segment may follow two downtrend segments, so the split points a and b are considered as OPs.

After adjusting samples according to the nw_{tp} and the nw_{op}, assuming the number of TPs generated by the PLR is N_{ptp}, the number of OPs generated by the PLR is N_{pop}, and the number of other OPs is N_{oop}, then the OPs can be selected as follows:

$$\begin{cases} N_{pop}\,\text{OPs} + \text{random select}\left(N_{ptp} - N_{pop}\right)\text{OPs} \\ \qquad\qquad\qquad\qquad \text{if}\, N_{pop} + N_{oop} > N_{ptp} \\ \text{keep}\left(N_{pop} + N_{oop}\right)\text{OPs} \quad \text{else} \end{cases} \tag{10}$$

The balanced samples and their weights are used to train the WSVM.

4.4. Determine trading signals using the RSI rule and the DODS

The WSVM forecasts the TPs, and the next step is to judge whether they are BPs or SPs. RSI is a technical curve based on the ratio of the rise and the fall in a certain period and can reflect the prosperity of the stock markets. The more the stock price rises, the larger the RSI, and vice versa. When the RSI is around 50, the stock is in a stable trend; When the RSI is above 70, the stock is overbought; When the RSI is below 30, the stock is oversold [20]. In this paper, the RSI is used to judge stock trends and determine trading signals. The points with an RSI of around 50 are in a stable trend, and it is difficult to determine whether they are BPs or SPs. Therefore, these points are discarded, and other points can be determined as follows:

$$\begin{cases} \text{If}\, RSI_{24}(t) < 40,\ \text{then}\, t\, \text{is a BP.} \\ \text{If}\, RSI_{24}(t) > 60,\ \text{then}\, t\, \text{is a SP.} \end{cases} \tag{11}$$

The delay-one-day strategy (DODS) [35] is a method to prevent the loss of prediction errors. With the DODS, the trading is delayed by one day to determine if the up or down trend will continue. Because the trading day is delayed according to the DODS, the RSI of the next day must be examined to make the right transaction. The RSI is further integrated into the DODS to adjust the transaction, and the new DODS is shown in Algorithm 2, where $cr(t)$ is the change rate and calculated by the equation $cr(t) = (P_c(t) - P_c(t-1))/P_c(t-1)$, α and β represent the degree of alleviation.

5. Data and investment strategy

The same stocks and investment strategy used in [35] are used to evaluate the proposed model. Twenty random stocks from the Shanghai Stock Exchange in China are collected from January 4, 2010 to August 18, 2011. Their stock codes are 600736, 600197, 600211, 600694, 600351, 600488, 600054, 600019, 600058, 600682, 600597, 600066, 600881, 600228, 600697, 600107, 600053, 600051, 600163, 600167. During the period, nine stocks lost more than 10%, including 600736, 600197, 600211, 600694, 600351, 600488, 600054, 600019, 600058; six stocks' revenue is between −10% and 10%, including 600682, 600597, 600066, 600881, 600228, 600697; five stocks earned more than 10%, including 600107, 600053, 600051, 600163, 600167.

Algorithm 3 shows the investment strategy where b_s, b_m, v_m, c_b, c_s are the balance number of shares, the balance money, the

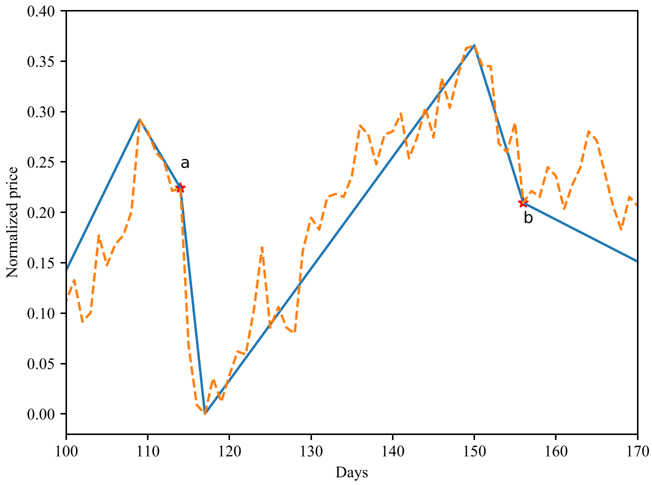

Fig. 5. Example for the OP generated by the PLR.

total investment money, the transaction costs of buying, and the transaction costs of selling respectively.

Algorithm 3 The investment strategy

Initial values: $b_m = 0, b_s = 0, v_m = 0$

If t is a BP and $b_s < 0$:

$$b_m = b_m + b_s \times p(t) \times (1 + c_s)$$

$$b_s = 0$$

If t is a BP and $b_s = 0$:

$$b_s = 1 / \left(p(t) \times (1 + c_b)\right)$$

$$v_m = v_m + \max(0, 1 - b_m)$$

$$b_m = b_m + \max(0, 1 - b_m) - 1$$

If t is a SP and $b_s > 0$:

$$b_m = b_m + b_s \times p(t) \times (1 - c_s)$$

$$b_s = 0$$

If t is a SP and $b_s = 0$:

$$b_s = -\max(1, b_m) / \left(p(t) \times (1 + c_b)\right)$$

$$v_m = v_m + \max(0, 1 - b_m)$$

$$b_m = \max(1, b_m) \times (2 - c_s)$$

6. Experimental results

Scikit-learn [56] is used to implement the WSVM model in this paper. The parameters used in this paper is as follows:

$r_1 = 200, r_2 = 10, \lambda = 1, \alpha = 0.5, \beta = 0.5, c_b = 0.0035$
$\alpha_{PLR} = 0.05, \beta_{PLR} = 5, nw_{op} = 1, nw_{tp} = 1, c_s = 0.0035$

Additionally, the radial basis function (RBF) is used as the kernel function of the WSVM. The choice of the penalty parameter C and the RBF parameter g of WSVM has an important influence on the performance of WSVM. Here, a simple grid search process, $C \in [1, 10, 100, 200, 500]$ and $g \in [0.0001, 0.001, 0.01, 0.1]$, is used to find good parameters (C, g) on the training set. The grid search shows the best parameters are $(200, 0.1)$, and the following experiments are all based on the parameters.

Twenty stocks are invested based on the algorithm 3. At the end of the test period, all the shares are sold, and all the short positions are closed to calculate the investment revenues.

Fig. 6 shows the TPs obtained by the proposed model in all twenty stocks. The red marks $\triangle$ are buying signals, and the green marks ∇ are selling signals. For example, in stock 600211, the three peak areas were found accurately, and selling signals were given. The first trough area was missed, but three other trough areas were found. In stock 600228, the trough area around the 130th day was found, and buying signals were given. Other stocks are similar. The proposed model has the ability to find the TPs and give investment advice.

Revenue is a reasonable indicator to evaluate different models, and coefficient of variation (CV) is another comprehensive indicator to evaluate the models. The smaller the CV, the better the model. Table 3 shows a comparison of the proposed model, PLR–WSVM, IPLR–WSVM [35], PLR–ANN, and BHS. The combination of the basic PLR–WSVM and the proposed methods are given in Table 3 as well.

The previous research uses the linear fitting to determine trading signals [35]. The average revenue obtained by the PLR–WSVM using the RSI rule outperforms that of the PLR–WSVM using the linear fitting method. The RSI reflects the ratio of the rise and the fall in a certain period, and it can determine the trading signals effectively. The balanced sample further enhances the performance of the PLR–WSVM model. After combining all the methods, the proposed model significantly outperforms other models. Furthermore, the CV of the proposed model is the lowest, indicating that the proposed model is the most stable.

Fig. 7 shows some examples of selecting the thresholds in different stocks during different periods. The first image in Fig. 7 displays the fitness curve of the stock 600736 in the 2nd training set, the second image in Fig. 7 displays the fitness curve of the stock 600351 in the 7th training set, and so on. As can be seen, with a fitness function, the threshold δ of PLR can be automatically selected according to the different price fluctuations. The detailed comparison between automatically selecting δ and using a constant δ is shown in Table 4. The average revenue obtained by automatically selecting the threshold is higher than that obtained by a constant threshold, and the CV obtained by automatically selecting the threshold is the lowest.

The large weights of TPs make the model care more about TPs, which solves the problem of unbalanced samples to a certain extent. But the oversampling and the undersampling can solve this problem more effectively. As can be seen in Table 5, with balanced samples, the performance of the SVM is close to that of the WSVM. When using inappropriate weights, the performance of the model may even decrease.

The effect of the different RSI thresholds used in the RSI rules on the proposed model is shown in Fig. 8. The structure (a, b) means that a point is a BP when its RSI $< a$, a point is a SP when its RSI $> b$, and a point is discarded when its RSI is between a and b. When the RSI is around 50, the stock price is in a stable trend, and trading signals are prone to misjudgment. After discarding the points where the RSI is within the range of $(44, 56)$, the trading signals are determined effectively. But if the RSI threshold is set too large, some important TPs will be discarded, reducing the performance of the proposed model.

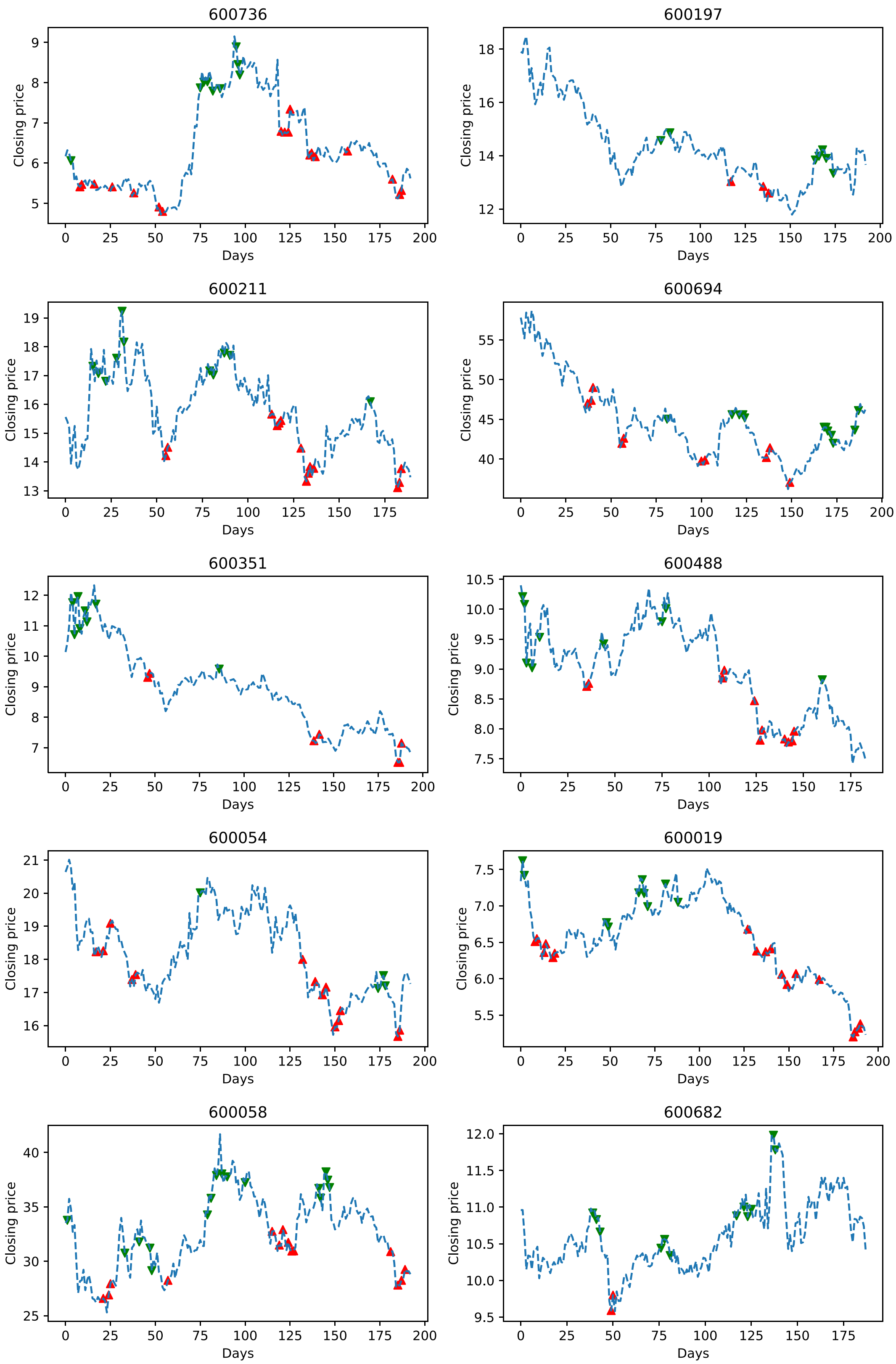

Fig. 6. The TPs obtained by the proposed model.

7. Conclusion

This paper has proposed a new approach of integrating piecewise linear representation and weighted support vector machine to forecast the stock turning points. A fitness function that focuses on medium or long-term trends rather than short-term rebounds has been proposed to select the threshold of the PLR automatically. An oversampling method has been proposed which is based

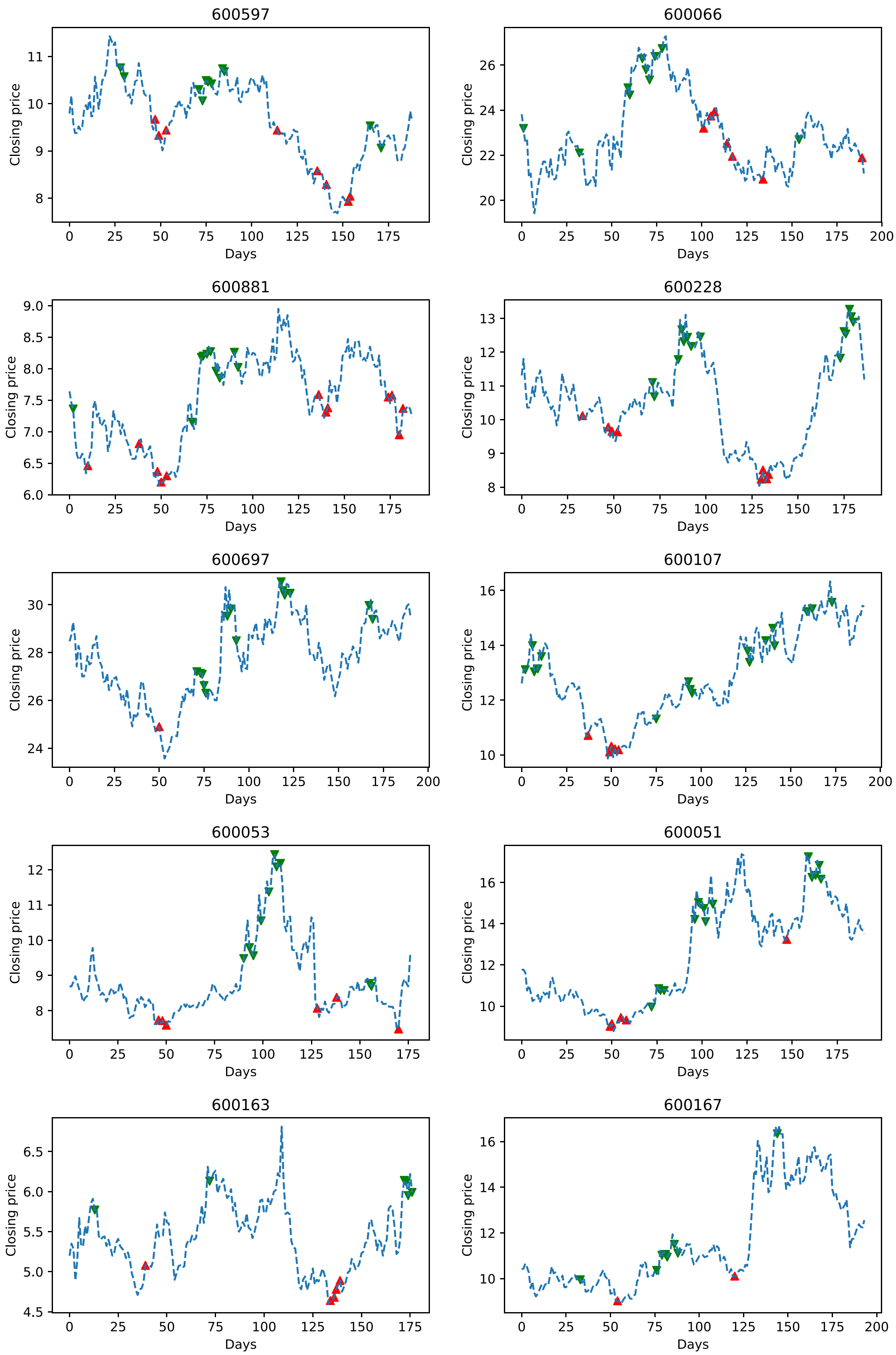

Fig. 6. (*continued*).

on the idea that a TP should be treated as a period rather than a point. The random undersampling combined with the oversampling has been used to balance the number of samples. The RSI has been integrated to determine the trading signals.

The experimental results showed that the revenues obtained by the proposed model are significantly higher than that obtained by other models. The CV of the revenues obtained by the proposed model also significantly outperforms that obtained by

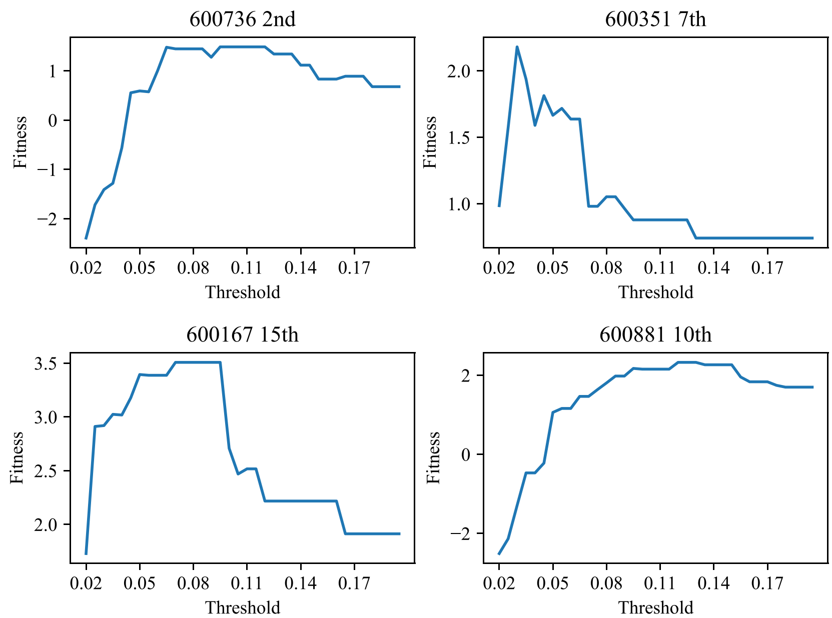

Fig. 7. Threshold selection in different stocks during different periods.

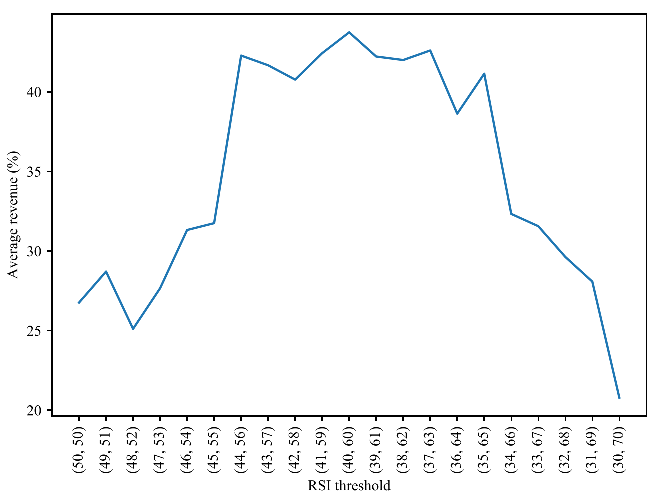

Fig. 8. The comparison of different RSI thresholds.

other models, indicating the proposed model is the most stable. The proposed model showed the ability to discover TPs and earn high and stable revenues in stock markets.

Even though high and stable revenues can be earned using the proposed model, some important TPs are still not predicted by the proposed model. In the future, the proposed model can be combined with trading systems based on reinforcement learning or other algorithms to forecast TPs. The integrated model should forecast TPs without the important missing and significant errors. The strategy used to determine trading signals can also be well-designed and improved. In addition, the long-term TPs may be affected by a company's multi-level chains and organizational strategies. It is recommended for interested readers to study the relationship between the long-term TPs and the performance rate of organizational strategies [57,58].

Acknowledgments

This work was partially supported by key grants from National Natural Science Foundation of China (Nos. 91546201, 71873015) as well as a grant from the International Graduate Exchange Program of Beijing Institute of Technology, China.

References

[1] L. Yu, S. Wang, K.K. Lai, A neural-network-based nonlinear metamodeling approach to financial time series forecasting, Appl. Soft Comput. 9 (2009) 563–574.
[2] C.L. Dunis, M. Williams, Modelling and Trading the EUR/USD Exchange Rate: Do Neural Network Models Perform Better? Working Paper Center for International Banking Economics & Finance, 2002.
[3] M.K. Okasha, Using support vector machines in financial time series forecasting, 4 (2014) 28–39.
[4] C.J. Huang, D.X. Yang, Y.T. Chuang, Application of wrapper approach and composite classifier to the stock trend prediction, Expert Syst. Appl. 34 (2008) 2870–2878.
[5] M.C. Lee, Using support vector machine with a hybrid feature selection method to the stock trend prediction, Expert Syst. Appl. 36 (2009) 10896–10904.
[6] I. Marković, M. Stojanović, J. Stanković, M. Stanković, Stock market trend prediction using AHP and weighted kernel LS-SVM, Soft Comput. (2016) 1–12.
[7] J. Patel, S. Shah, P. Thakkar, K. Kotecha, Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques, Expert Syst. Appl. 42 (2015) 259–268.
[8] W. Lertyingyod, N. Benjamas, Stock price trend prediction using artificial neural network techniques: Case study: Thailand stock exchange, in: Computer Science and Engineering Conference, 2017, pp. 1–6.
[9] B. Wu, T. Duan, A performance comparison of neural networks in forecasting stock price trend, Int. J. Comput. Intell. Syst. 10 (2017) 336.
[10] L. Khaidem, S. Saha, S.R. Dey, Predicting the direction of stock market prices using random forest, 2016, arXiv preprint arXiv:1605.00003.
[11] P.C. Chang, C.Y. Fan, C.H. Liu, Integrating a piecewise linear representation method and a neural network model for stock trading points prediction, IEEE Trans. Syst. Man Cybern. C 39 (2008) 80–92.
[12] S.H.R. Pasandideh, S.T.A. Niaki, A. Gharaei, Optimization of a multiproduct economic production quantity problem with stochastic constraints using sequential quadratic programming, Knowl.-Based Syst. 84 (2015) 98–107.
[13] A. Gharaei, S.H.R. Pasandideh, Modeling and optimization of four-level integrated supply chain with the aim of determining the optimum stockpile and period length: sequential quadratic programming, J. Ind. Prod. Eng. 34 (2017) 529–541.
[14] A. Gharaei, S.H.R. Pasandideh, S.T.A. Niaki, An optimal integrated lot sizing policy of inventory in a bi-objective multi-level supply chain with stochastic constraints and imperfect products, J. Ind. Prod. Eng. 35 (2017) 6–20.
[15] S.A.H. Shekarabi, A. Gharaei, M. Karimi, Modelling and optimal lot-sizing of integrated multi-level multi-wholesaler supply chains under the shortage and limited warehouse space: generalised outer approximation, Int. J. Syst. Sci. Oper. Logistics (2018) 1–21.

[16] Z. Tan, Q. Chai, P.Y.K. Cheng, Stock trading with cycles: A financial application of ANFIS and reinforcement learning, Expert Syst. Appl. 38 (2011) 4741–4755.
[17] L. Luo, X. Chen, Integrating piecewise linear representation and weighted support vector machine for stock trading signal prediction, Appl. Soft Comput. 13 (2013) 806–816.
[18] S. Sukhanov, A. Merentitis, C. Debes, J. Hahn, A.M. Zoubir, Bootstrap-based SVM aggregation for class imbalance problems, 2015.
[19] G.E.A.P.A. Batista, R.C. Prati, M.C. Monard, A study of the behavior of several methods for balancing machine learning training data, ACM Sigkdd Explor. Newslett. 6 (2004) 20–29.
[20] R. Bhargavi, S. Gumparthi, R. Anith, Relative strength index for developing effective trading strategies in constructing optimal portfolio, Int. J. Appl. Eng. Res. 12 (2017) 8926–8936.
[21] G.P. Zhang, Time series forecasting using a hybrid ARIMA and neural network model, Neurocomputing 50 (2003) 159–175.
[22] R. Adhikari, R.K. Agrawal, A combination of artificial neural network and random walk models for financial time series forecasting, Neural Comput. Appl. 24 (2014) 1441–1449.
[23] T. Chavarnakul, D. Enke, A hybrid stock trading system for intelligent technical analysis-based equivolume charting, Neurocomputing 72 (2009) 3517–3528.
[24] P. Sevastianov, L. Dymova, Synthesis of fuzzy logic and Dempster–Shafer theory for the simulation of the decision-making process in stock trading systems, Math. Comput. Simul. 80 (2010) 506–521.
[25] D. Bao, Z. Yang, Intelligent stock trading system by turning point confirming and probabilistic reasoning, Expert Syst. Appl. 34 (2008) 620–627.
[26] X. Li, Z. Deng, J. Luo, Trading strategy design in financial investment through a turning points prediction scheme, Expert Syst. Appl. 36 (2009) 7818–7826.
[27] S. Mabu, K. Hirasawa, M. Obayashi, T. Kuremoto, Enhanced decision making mechanism of rule-based genetic network programming for creating stock trading signals, Expert Syst. Appl. 40 (2013) 6311–6320.
[28] V.P. Hai, E.W. Cooper, T. Cao, K. Kamei, Hybrid Kansei-SOM model using risk management and company assessment for stock trading, Inform. Sci. 256 (2014) 8–24.
[29] J.L. Wu, L.C. Yu, P.C. Chang, An Intelligent Stock Trading System using Comprehensive Features, Elsevier, Science Publishers B. V., 2014.
[30] J. Zhang, D. Maringer, Using a genetic algorithm to improve recurrent reinforcement learning for equity trading, Comput. Econ. 47 (2016) 1–17.
[31] H.H. Bahar, M.H.F. Zarandi, A. Esfahanipour, Generating ternary stock trading signals using fuzzy genetic network programming, in: Fuzzy Information Processing Society, 2017, pp. 1–6.
[32] M.R. Alimoradi, A.H. Kashan, A league championship algorithm equipped with network structure and backward Q-learning for extracting stock trading rules, Appl. Soft Comput. 68 (2018).
[33] Y.K. Kwon, H.D. Sun, A hybrid system integrating a piecewise linear representation and a neural network for stock prediction, Strategic Technol. Int. Forum 2 (2011) 796–799.
[34] X. Chen, Z.J. He, Prediction of stock trading signal based on support vector machine, in: International Conference on Intelligent Computation Technology and Automation, 2016, pp. 651–654.
[35] L. Luo, S. You, Y. Xu, H. Peng, Improving the integration of piece wise linear representation and weighted support vector machine for stock trading signal prediction, Appl. Soft Comput. 56 (2017) 199–216.
[36] L.I. Feng, G. Feng, K. Peng, Integrating piecewise linear representation and Gaussian process classification for stock turning points prediction, J. Comput. Appl. (2015).
[37] A. Gharaei, S.H.R. Pasandideh, A.A. Khamseh, Inventory model in a four-echelon integrated supply chain: Modeling and optimization, J. Model. Manage. 12 (2017) 739–762.
[38] A. Gharaei, S.H.R. Pasandideh, Four-echelon integrated supply chain model with stochastic constraints under shortage condition: Sequential quadratic programming, Ind. Eng. Manage. Syst. 16 (2017) 316–329.
[39] A. Gharaei, S.H.R. Pasandideh, Modeling and optimization the four-level integrated supply chain: Sequential quadratic programming, Int. J. Comput. Sci. Inf. Secur. 14 (2016) 650–669.
[40] A. Gharaei, B. Naderi, M. Mohammadi, Optimization of rewards in single machine scheduling in the rewards-driven systems, Manage. Sci. Lett. 5 (2015) 629–638.
[41] C. Duan, D. Chao, A. Gharaei, J. Wu, B. Wang, Selective maintenance scheduling under stochastic maintenance quality with multiple maintenance actions, Int. J. Prod. Res. (2018) 1–19.
[42] A. Gharaei, M. Karimi, S.A.H. Shekarabi, An integrated multi-product, multi-buyer supply chain under penalty, green, and quality control polices and a vendor managed inventory with consignment stock agreement: The outer approximation with equality relaxation and augmented penalty algorithm, Appl. Math. Model. 69 (2019) 223–254.
[43] X. Zhang, W.G. Zhang, R. Cai, Portfolio adjusting optimization under credibility measures, J. Comput. Appl. Math. 234 (2010) 1458–1465.
[44] Y. Feng, D.P. Palomar, SCRIP: Successive convex optimization methods for risk parity portfolio design, IEEE Trans. Signal Process. 63 (2015) 5285–5300.
[45] E.E. Osuna, R. Freund, F. Girosi, Support vector machines: Training and applications, A. I. Memo (1602) C. B. C. L. Paper 144 (1997) 1308–1316.
[46] T. Joachims, Making large-scale support vector machine learning practical, in: Advances in Kernel Methods, 1999.
[47] S.S. Keerthi, G.E. Gilbert, Convergence of a generalized SMO algorithm for SVM classifier design, Mach. Learn. 46 (2002) 351–360.
[48] C.W. Hsu, C.J. Lin, A simple decomposition method for support vector machines, Mach. Learn. 46 (2002) 291–314.
[49] J.C. Platt, Fast training of support vector machines using sequential minimal optimization, 1999.
[50] K. Sopyła, P. Drozda, P. Górecki, SVM with CUDA accelerated kernels for big sparse problems, in: International Conference on Artificial Intelligence & Soft Computing, 2012.
[51] C.C. Chang, C.J. Lin, LIBSVM: A library for support vector machines, 2 (2011) 1–27.
[52] R.E. Fan, P.H. Chen, C.J. Lin, Working set selection using second order information for training support vector machines, 2005.
[53] E. Keogh, S. Chu, D. Hart, M. Pazzani, An online algorithm for segmenting time series, Icdm (2001) 289–296.
[54] V. Vapnik, The nature of statistical learning theory, 1995, pp. 988–999.
[55] J. Wang, Y. Yao, H. Zhou, M. Leng, X. Chen, A new over-sampling technique based on SVM for imbalanced diseases data, in: International Conference on Mechatronic Sciences, Electric Engineering and Computer, 2014, pp. 1224–1228.
[56] F. Pedregosa, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, Scikit-learn: Machine learning in python, J. Mach. Learn. Res. 12 (2016) 2825–2830.
[57] M.A. Sobhanallahi, A. Gharaei, M. Pilbala, Provide a new method to determine effectiveness or performance rate of organization strategies based on Freeman model and using improved dimensional analysis method, in: International Conference on Industrial Engineering, 2016.
[58] M.A. Sobhanallahi, A. Gharaei, M. Pilbala, Provide a practical approach for measuring the performance rate of organizational strategies, in: International Conference on Industrial Engineering, 2016.