Knowledge-Based Systems 000 (2016) 1–11

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Dynamic financial distress prediction with concept drift based on time weighting combined with Adaboost support vector machine ensemble

Jie Sun[a,*], Hamido Fujita[b], Peng Chen[c], Hui Li[d]

[a] Department of Accounting, Business School, Tianjin University of Finance and Economics, NO. 25 Zhujiang Road, Tianjin, 300222, PR China
[b] Faculty of Software and Information Science, Iwate Prefectural University, Iwate, Japan
[c] School of Economics and Management, Zhejiang Normal University, NO. 688 Yingbin Dadao Road, Jinhua, 321004, Zhejiang, PR China
[d] College of Tourism and Service Management, Nankai University, NO. 38 Tongyan Road, Tianjin, 300350, PR China

ARTICLE INFO

Article history:
Received 17 August 2016
Revised 18 December 2016
Accepted 19 December 2016
Available online xxx

Keywords:
Dynamic financial distress prediction
Concept drift
Time weighting
Adaboost
Support vector machine

ABSTRACT

Dynamic financial distress prediction (DFDP) is important for improving corporate financial risk management. However, earlier studies ignore the time weight of samples when constructing ensemble FDP models. This study proposes two new DFDP approaches based on time weighting and Adaboost support vector machine (SVM) ensemble. One is the double expert voting ensemble based on Adaboost-SVM and Timeboost-SVM (DEVE-AT), which externally combines the outputs of an error-based decision expert and a time-based decision expert. The other is Adaboost SVM internally integrated with time weighting (ADASVM-TW), which uses a novel error-time-based sample weight updating function in the Adaboost iteration. These two approaches consider time weighting of samples in constructing Adaboost-based SVM ensemble, and they are more suitable for DFDP in case of financial distress concept drift. Empirical experiment is carried out with sample data of 932 Chinese listed companies' 7 financial ratios, and time moving process is simulated by dividing the sample data into 13 batches with one year as time step. Experimental results show that both DEVE-AT and ADASVM-TW have significantly better DFDP performance than single SVM, batch-based ensemble with local weighted scheme, Adaboost-SVM and Timeboost-SVM, and they are more suitable for disposing concept drift of financial distress.

1. Introduction

Since the global financial crisis in 2008, the world's economic growth began to slow down. In the unstable economic environment, enterprises have to bear great pressure, and many of them fall into financial distress or even bankruptcy due to unscientific management. Meanwhile, the financial distress of enterprises will increase the risk of banking industry to some extent. When many enterprises are unable to repay the bank loan because of financial distress, the banks' bad debts would increase and this would furthermore bring huge hidden troubles to the whole financial system. In such a situation, it needs an efficient financial distress predict (FDP) system, which can help enterprises improve risk management and help banks make scientific credit decision.

However, most FDP research is based on static sample data, which is contradictory with the dynamic sample data flow in reality. Some studies have considered financial distress concept drift (FDCD) and focused on dynamic financial distress predict (DFDP) [1–4], but they only considered the method of time window or data batch combination, ignoring the idea that new samples and old samples should have different weights in DFDP modeling. This paper attempts to propose two new approaches for DFDP in condition of FDCD, based on time weighting combined with Adaboost support vector machine (SVM) ensemble, so as to enrich the methodology system of DFDP.

2. Literature review

2.1. Literature review on FDP

Many researches have focused on the subject of FDP. Altman [5] used MDA to identify companies into known categories and concluded that bankruptcy could be explained quite completely by a combination of five financial ratios. Ohlson [6] was the first to apply logistic regression (Logit) model to predicting financial distress. Frydman et al. [7] also tried decision tree (DT) to FDP. Afterward, neural network is the most widely used machine learning method in this field. For example, Chen and Du [8] and Wu et al. [9] respectively constructed the three-layer feed-forward back-propagation neural network model and the probabilistic

* Corresponding author.
E-mail address: sunjiehit@gmail.com (J. Sun).

http://dx.doi.org/10.1016/j.knosys.2016.12.019
0950-7051/

neural network model for FDP. Besides, Sun and Hui [10] and Sun and Li [11] respectively put forward a FDP method based on similarity weighted voting CBR and a systematic data mining method of decision tree. However, among various data mining methods, SVM is proved to be a relatively good method for FDP because it is based on the structural risk minimization principle rather than the empirical risk minimization principle [12–15].

Based on the single classifier FDP methods, ensemble methods are widely applied in the field of FDP. Some researches constructed ensemble classifiers for FDP using several classification algorithms on the same training set [16–18]. Other researches constructed ensemble classifiers for FDP using only one algorithm on different data subsets [19,20] or under different parameters [21–24]. Among them, Sun et al. [23] and Kim and Upneja [24] found that AdaBoost ensemble method showed good performance in the empirical research of FDP.

In recent years, DFDP became an important branch of FDP research. It focuses on how to update the FDP model dynamically when the new sample data batches gradually emerge and FDCD happens over time [25]. For FDP modeling based on a lateral data set composed of many distress and non-distress sample companies, Sun and Li [1] applied the instance selection mechanism based on time windowing, and Sun et al. [2] proposed the adaptive and dynamic ensemble of SVM based on data batch combination. For FDP modeling based on the longitudinal financial data stream of a certain company, the process of financial condition evaluation and the process of FDP can be dynamically integrated. For example, Sun et al. [3] attempted sequential floating forward selection with principal component neural network optimized by genetic algorithm, and Sun et al. [4] put forward an approach based on the entropy-based weighting, SVM and vertical sliding time window.

2.2. Literature review on concept drift

Schlimmer and Granger [26] defined concept drift as changes in the target concept, and these changes are induced by changes in the hidden context. Widmer and Kubat [27] further categorized concept drift into the real and the virtual, and considered the concept drift defined by Schlimmer and Granger [26] as the real concept drift. Furthermore, they considered the virtual concept drift as the change in the underlying data distribution when the target concept remains the same. In terms of the degree of virtual concept drift, it is further divided into sudden concept drift and gradual concept drift [28].

When concept drift happens, the model trained on the old data set may become ineffective for the new concept environment. To solve this problem, three types of methodology are proposed. The first is sample selection with time windowing. For example, Hulten [29] introduced a method of Concept-adapting Very-Fast Decision Tree learner, to build a time window to select samples and solve concept drift in data flowing. Sample selection with time windowing is the most simple approach to treat concept drift. However, it completely abandons the old samples outside the time window, and this leads to the limitation that some old samples that are informative are excluded. The second is sample weighting. For instance, Koychev [30] introduced a gradual forgetting algorithm that considered sample age for adaptation to concept drift. Klinkenberg [31] brought us two time weighting methods based on exponential function, namely global scheme and local scheme. Sample weighting can avoid the limitation of sample selection to some extent. But for such a special situation as concept repeat, certain sampling weighting scheme may discord with the importance of samples and its effectiveness shows instability in such a condition. The third is classifier ensemble. Kyosuke et al. [32] proposed an adaptive classifier ensemble method by adding online learners into the ensemble system. Wu et al. [33] proposed an online bagging strategy to construct classifier ensemble model that can update itself by considering inflow of new samples. Song et al. [34] constructed an ensemble model named as Dynamic Clustering Forest to classify textual data stream with concept drift. These classifier ensemble methods deal with concept drift by adding new base classifiers or updating their weights with emergence of new samples. But they do not consider the different roles of older samples and newer samples in the process of base classifier training. Since classifier ensemble has the advantages such as stability and better classification performance than single classifier, it is necessary to integrates sample weighting with classifier ensemble for disposing concept drift.

3. Background theory

3.1. Time weighting

Time weighting has been used widely to solve concept drift. The basic assumption it relies on is that the new samples are always more important than the older ones. With time flowing, the importance of an old sample gradually decreases. Klinkenberg [31] introduced a time weighting method, with the basic opinion that the importance of data sample batches is monotonically decreasing with time flowing. Its function is shown in formula (1):

$$w_t = \exp(-\lambda t) \tag{1}$$

In this formula, t is the time variable that denotes the time point t time steps ago, and λ is the time weighting parameter. The larger λ is, the sooner an example becomes irrelevant. For $\lambda \to \infty$, the model is learned only on the newest examples, and for $\lambda = 0$, all examples share an equal weight.

3.2. Support vector machine

SVM is a statistical learning method [35]. For a classification sample data set expressed by $D = \{x_i, y_i\}_{i=1}^{N}$ where $y \in \{1, -1\}$, SVM learning is to construct the optimal separating hyper-plane with the biggest margin width between the two classes. It is an optimization problem expressed as formula (2).

$$\begin{aligned} &\min \quad \tfrac{1}{2} w^{\mathrm{T}} w + C \sum_{i=1}^{N} \xi_i \\ &\text{s.t.} \\ &\begin{cases} y_i [w^{\mathrm{T}} \Phi(x_i) + b] \geq 1 - \xi_i & (i = 1, 2, \ldots, N) \\ \xi_i \geq 0 & (i = 1, 2, \ldots, N) \end{cases} \end{aligned} \tag{2}$$

In the above formula, ξ_i are the slack variables, which are used to allow certain degree of training misclassification, $C \in R^+$ is a tuning parameter that weights the importance of classification error with the margin width, and w^{T} represents the weight vector. The kernel function $\Phi(x_i)$ is used to map the original feature space into a high-dimensional space, and there are mainly four types of kernel functions, i.e. linear kernel function, polynomial kernel function, radial basis kernel function (RBF), and sigmoid kernel function [36].

3.3. Batch-based ensemble with local weighted scheme (BE-LWS)

To solve concept drift problem, Klinkenberg proposed a local weighted scheme (LWS) [31], whose algorithm is illustrated in Table 1, in which *Batch T* is the most recent training data batch and *Batch* 1 is the oldest available training data batch.

In the above algorithm, it uses the most recent data batch to build a classifier, so as to test the suitability of the rest data batches for current modeling. This is based on the assumption that the distribution of the recent data batch is supposed to be the most similar to the future testing data set. Therefore, after

Please cite this article as: J. Sun et al., Dynamic financial distress prediction with concept drift based on time weighting combined with Adaboost support vector machine ensemble, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.12.019

Table 1
The algorithm of batch-based ensemble with local weighted scheme.

Algorithm Batch-based ensemble with local weighted scheme.
Input: Training data batches *Batch* 1, 2, ..., *T*; Testing set.
Do:
(1) Train *Classifier* 0 on *Batch T*.
(2) Test *Classifier* 0 on *Batch* 1, 2, ..., (*T*-1).
(3) Resort the data batches in terms of ascending testing accuracies.
(4) Rename the sorted data batches as *Batch* 1*, 2*, ..., *T** (*Batch T** = *Batch T*).
(5) Use *Batch* 1*, 2*, ..., *T** to built *Classifier* 1, 2, ..., *T*.
(6) Weight *Classifier* 1, 2, ..., *T* as $w_j = \exp(-\lambda t) = \exp[-\lambda(T-j)]$.
(7) Apply *Classifier* 1, 2, ..., *T* to the testing set and obtain the results $[r_1, r_2, \ldots, r_T]$.
Output: The final result denoted as $\mathrm{sig}[\sum_{j=1}^{T} r_j w_j]$.

testing the older data batches with that classifier, it resorts the batches and weights the classifiers trained on them with the time weighted function.

As for the complexity analysis, suppose the complexity of base classifier is *CPX_BC*. In BE-LWS algorithm, the complexity for the sorting process of *T* batches is $O(T^2)$ in maximum case, and the complexity for training *T* base classifiers is $O(T \times CPX\_BC)$. Therefore, the complexity for BE-LWS algorithm is $O(\max\{T^2, T \times CPX\_BC\})$.

3.4. Adaboost

Adaboost is an iterative machine learning algorithm to construct a classifier ensemble [37]. It uses only one classification algorithm to construct diverse weak base classifiers, which are trained on data sets selectively sampled from an initial training data set. Each sample is given a weight representing its possibility to be selected as a training sample, and all examples share an equal weight in the first round of iteration. In next iterations, the samples correctly classified by the base classifier of last iteration will get lower weights and the samples misclassified will get higher weights. Thus the base classifiers will focus more and more on the "difficult" data points that may locate near the classification margin and finally improve the classification accuracy. After certain times of iterations, the base weak classifiers are combined to be a strong classifier and the final classification result is output.

4. Time-weighting-Adaboost SVM ensemble model for DFDP

4.1. Motivation of time-weighting-Adaboost for DFDP

Although the Adaboost algorithm that has been widely applied in FDP can effectively improve the FDP performance by combining multiple weak classifiers, it has obvious shortcoming in DFDP. In the iterations of Adaboost, the "difficult" samples that are easily misclassified will be given higher weights, and the "easy" samples that are easily correctly classified will be given lower weights and even be abandoned. However, it neglects the different roles of old samples and new samples in DFDP modeling, and this may decrease the suitability of current FDP model for future prediction especially when gradual FDCD continuously exists. It is possible that some old "difficult" samples with high weights would not fit the data distribution of new samples, and this would lead to decrease of the model's performance for current classification. In the process of DFDP modeling, concept drift may exist, and the information a classifier can obtain from a sample may become gradually decreasing. Therefore, it is necessary to consider the factor of time in the process of sample weighting, and propose time-weighting-Adaboost algorithm for DFDP.

4.2. Double expert voting ensemble based on Adaboost-SVM and Timeboost-SVM (DEVE-AT)

4.2.1. Basic idea of DEVE-AT

To consider the factor of time in DFDP modeling, DEVE-AT is proposed to make decision based on both the outputs of Adaboost-SVM and Timeboost-SVM, which are two independent iteration systems with different decision preference. Instead of sharing an equal weight in the first round of iteration for Adaboost-SVM, the samples get an initial weights based on time weighting in the first round of iteration for Timeboost-SVM. To some extent, Adaboost-SVM is an error-based decision expert and Timeboost-SVM is a time-based decision expert, and DEVE-AT can integrate both advantages of Adaboost-SVM and Timeboost-SVM. They can both output fuzzy classification decision values due to the weighted voting mechanism for multiple base classifiers. This makes their outputs suitable to be further combined by weighted voting to make DFDP decision, and therefore the decision of possible failure can easily be avoided.

4.2.2. Framework and algorithm of DEVE-AT

The framework of DEVE-AT is shown in Fig. 1. Suppose $ITS^* = \{(x_i, y_i, t_i)\}$ $(i = 1, 2, \ldots, N)$ is an initial training set with time label for each sample. The time label t_i is an integer and this means that the corresponding sample is obtained in t_i time steps ago. For example, for the samples in the latest batch $t_i = 0$ and for the samples in the batch obtained five years ago $t_i = 5$. As Fig. 1 shows, Adaboost-SVM and Timeboost-SVM firstly run independently and then their outputs are combined to make the final decision.

For Adaboost-SVM, the algorithm is stated in Table 2.

In the above algorithm, the weights of base classifiers are calculated in terms of the resubstitution classification error rate e_u. When over-fitting happens in training process, SVM_u is good at classifying the training samples but could be bad at classifying other out-sample testing examples. In such situation, e_u is a very small value near zero and SVM_u will have a high voting weight in Adaboost-SVM ensemble, which will greatly decrease the final classification accuracy of classifier ensemble for testing examples. Therefore, it is necessary to introduce the threshold c to prevent over-fitting base classifiers from entering the ensemble system. On the other hand, to guarantee the acceptable accuracy of the Adaboost ensemble, it requires that the error rate of base learners should be at least smaller than 0.5.

For Timeboost-SVM, the algorithm is stated in Table 3.

Timeboost-SVM originates from Adaboost-SVM, but it abandons its assumption that misclassified samples are more important. Instead, Timeboost-SVM considers newer samples as the more important. In detail, the initial training set ITS^* are composed of several data batches with different time labels and the samples in the same batch share the same time label. This means that the samples in the newer batches have higher weights than those in

Please cite this article as: J. Sun et al., Dynamic financial distress prediction with concept drift based on time weighting combined with Adaboost support vector machine ensemble, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.12.019

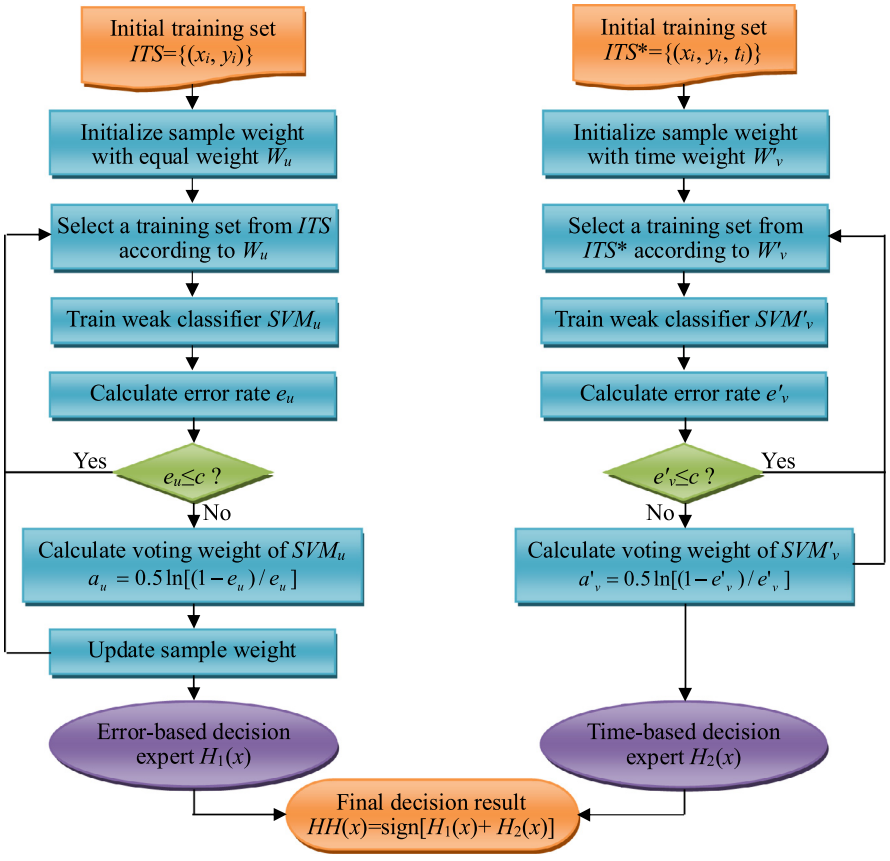

Fig. 1. The framework of DEVE-AT.

Table 2
The algorithm of Adaboost-SVM algorithm.

Algorithm Adaboost-SVM algorithm.
Input: Initial training set $ITS=\{(x_i, y_i)\}$ $(i = 1, 2, \ldots, N)$.
Initialization:
(1) Initialize sample weight distribution as $W_1 = \{w_1^1, w_1^2, \ldots, w_1^N\} = \{\frac{1}{N}, \frac{1}{N}, \ldots, \frac{1}{N}\}$.
(2) Initialize error threshold as c.
For $u = 1, 2, \ldots, U$
(1) Select a training set TS_u from ITS according to the weight distribution W_u.
(2) Train a base SVM classifier SVM_u, also represented as $h_u(x)$, based on TS_u.
(3) Resubstitute TS_u to SVM_u and calculate the error rate e_u.
(4) If $e_u \le c$ or $e_u \ge 0.5$, then go back to step 1 with the value of u unchanged.
(5) Calculate the voting weight of SVM_u as $a_u = 0.5\ln[(1-e_u)/e_u]$.
(6) Update sample weight $WW_{u+1}(i) = W_u(i) \times \exp[-\alpha_u y_u h_u(x_i)]$.
(7) Normalize sample weight as $W_{u+1}(i) = \frac{WW_{u+1}(i)}{\sum WW_{u+1}(i)}$
End
Output: Error-based decision expert $H_1(x) = \sum_{u=1}^{U} [h_u(x) \times \alpha_u]$

Table 3
The algorithm of Timeboost-SVM algorithm.

Algorithm Timeboost -SVM algorithm.
Input: Initial training set with time label, namely $ITS^* = \{(x_i, y_i, t_i)\}$ $(i = 1, 2, \ldots, N)$.
Initialization:
(1) Initialize sample weight distribution as $W_1' = \{w'^i_1\}(i = 1, 2, \ldots, N)$, $w'^i_1 = \frac{\exp(-\lambda t_i)}{\sum \exp(-\lambda t_i)}$.
(2) Initialize error threshold as c.
For $v = 1,2,\ldots,U$
(1) Select a training set TS'_v from ITS^* according to the weight distribution W'_v.
(2) Train a base SVM classifier SVM'_v, also represented as $h'_v(x)$, based on TS'_v.
(3) Resubstitute TS'_v to SVM'_v and calculate the error rate e'_v.
(4) If $e'_v \le c$ or $e'_v \ge 0.5$, then go back to step 1 with the value of v unchanged.
(5) Calculate the voting weight of SVM'_v as $a'_v = 0.5\ln[(1-e'_v)/e'_v]$.
End
Output: Time-based decision expert $H_2(x) = \sum_{v=1}^{V} [h'_v(x) \times \alpha'_v]$

Please cite this article as: J. Sun et al., Dynamic financial distress prediction with concept drift based on time weighting combined with Adaboost support vector machine ensemble, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.12.019

the older batches, but the samples in the same batch have the same weights. The training set TS'_v is selected by the roulette wheel selection algorithm, in which there is a random number generator. Therefore the training samples selected in different iteration usually have certain degree of diversity although the time-based sample weights are not updated in each iteration.

When both decision experts are constructed respectively based on Adaboost-SVM and Timeboost-SVM, they can be combined by voting to construct DEVE-AT, whose output is denoted as $HH(x)$.

$$HH(x) = \text{sign}[H_1(x) + H_2(x)] \tag{3}$$

In the above formula (3), $HH(x) = 1$ when $H_1(x) + H_2(x) \geq 0$ and $HH(x) = -1$ when $H_1(x) + H_2(x) < 0$. Hence, Timeboost is proposed by replacing the error-based weighting of Adaboost with time-based weighting, and DEVE-AT is the combination of Adaboost and Timeboost. It is a new attempt to apply DEVE-AT for DFDP modeling in condition of FDCD.

As for the complexity analysis, suppose *CPX_BC* still denotes the complexity of base classifier. The complexity of either Adaboost-SVM or Timeboost-SVM is O(U×*CPX_BC*). In DEVE-AT algorithm, both Adaboost-SVM and Timeboost-SVM should firstly run respectively and then their outputs are combined. Therefore, the time complexity for DEVE-AT algorithm is twice as much as Adaboost-SVM.

4.3. Adaboost SVM internally integrated with time weighting (ADASVM-TW)

4.3.1. Basic idea of ADASVM-TW

Just as mentioned above, the only deciding factor for Adaboost to update the sample weight is whether a sample is misclassified or not. This makes it unsuitable for DFDP in case of FDCD. For a very old sample that is misclassified repeatedly, its weight will keep increasing, which makes such an old sample have higher possibility to be chosen as a training sample. If this happens frequently in the Adaboost training process, the classifier ensemble's performance for current FDP will greatly decrease especially in case of FDCD. Although DEVE-AT relieves this problem to some extent, it is still the external combination of Adaboost-SVM and Timeboost-SVM and needs two rounds of iteration respectively. To make full use of the advantages of both Adaboost and time weighting in one round of iteration, we can internally integrate time weighting into Adaboost to propose a new sample weight updating mechanism for boosting SVM. It should weaken the role of old samples and strengthen the role of new samples in weak classifier training, but still remain those old samples that are still informative. The assumption of concept drift is that newer samples should be given higher weights than old samples in training sample selection, while the assumption of Adaboost is gradually increasing the weights of misclassified samples and reducing the weights of correctly classified samples in the next iteration of training. By combining these two assumptions, ADASVM-TW increases the weights of misclassified newer samples much more than misclassified older samples, and reduces the weights of correctly classified newer samples much less than correctly classified older samples.

4.3.2. Framework and algorithm of ADASVM-TW

The framework of ADASVM-TW is shown in Fig. 2, and its algorithm is stated in Table 4.

ADASVM-TW internally integrates error-based weighting with time weighting in the step of sample weight updating. When a sample is correctly classified, $y_u h_u(x_i) = 1$ and the formula for updating the sample weight can be restated as $WW_{u+1}(i) = W_u(i) \times \exp[-\alpha_u \exp(\lambda t_i)]$, in which $\exp[-\alpha_u \exp(\lambda t_i)] \in (0, 1)$. The value of $\exp[-\alpha_u \exp(\lambda t_i)]$ is bigger for newer samples with smaller t_i than for older samples with bigger t_i. Therefore, the weights of newer samples correctly classified will be reduced in less degree than the older samples correctly classified, as shown in Fig. 3(c). On the contrary, when a sample is misclassified, $y_u h_u(x_i) = -1$ and the formula for updating the sample weight can be restated as $WW_{u+1}(i) = W_u(i) \times \exp[-\alpha_u \exp(-\lambda t_i)]$,, in which $\exp[\alpha_u \exp(-\lambda t_i)] > 1$. The value of $\exp[\alpha_u \exp(-\lambda t_i)]$ is also bigger for newer samples with smaller t_i than for older samples with bigger t_i. Therefore, the weights of newer samples misclassified will be increased in more degree than older samples misclassified, as shown in Fig. 3(d).

As shown in Fig. 3, the sample weights in Adaboost-SVM are unrelated with time although they are reduced when correctly classified and increased when misclassified. While ADASVM-TW introduces a sample weighting mechanism related with time into the iterative process, and this is the main difference between ADASVM-TW and Adaboost-SVM. In other words, compared with older samples, the weights of newer samples are less reduced when they are correctly classified and are more increased when they are misclassified. After certain times of iterations, the informative newer samples that are misclassified have more probability to be remained in the training set, and theoretically this can make the base classifiers more suitable for FDCD and improve the accuracy of DFDP.

The complexity of ADASVM-TW algorithm is also O(*U*×*CPX_BC*), which is at the same complexity level as Adaboost-SVM. In other words, ADASVM-TW algorithm needs about half of the computational time of DEVE-AT.

5. Empirical experiment and analysis

5.1. Design of empirical experiment

The empirical experiment uses one year as the time step. The new samples of financial distress and non financial distress emerge year after year and they constitute sample data flow for DFDP. Sample data of successive thirteen years are used. FDP models are constructed based on the sample companies' financial data that are produced two years before financial distress year, namely T-2 data. Therefore, they can be used to predict whether an enterprise will run into financial distress two years later, and they are tested by the real world data collected two years after the modeling year. For example, let's suppose the simulated current year is 2003 and T-2 data of sample companies collected in and before 2003 are used for model construction. Then T-2 data of 2005 sample companies can be used as testing set and the performance of 2003 model can be really tested. Similarly, when the simulated current year moves to 2004, T-2 data of 2004 sample companies are added into the available data set for model construction and T-2 data of 2006 sample companies are instead used as testing set. In this way, time moving is simulated as shown in Fig. 4.

The performance of DEVE-AT and ADASVM-TW are compared with single SVM and BE-LWS (see Section 3.3). Model training and testing experiments are carried out with the software of Matlab 2010R.

5.2. Data collection and preprocessing

We choose financial distress samples as the Chinese listed companies with negative net profit in successive two years or with net assets per share lower than stock book value in 2000–2012. Corresponding non financial distress companies are collected in terms of the same industry and similar asset size. Both financial distress and non financial distress companies are listed in China Shanghai Stock Exchange and Shenzhen Stock Exchange.

We selected 44 initial financial ratios, which reflect short-term liability, long-term liability, profitability, operating capacity,

Please cite this article as: J. Sun et al., Dynamic financial distress prediction with concept drift based on time weighting combined with Adaboost support vector machine ensemble, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.12.019

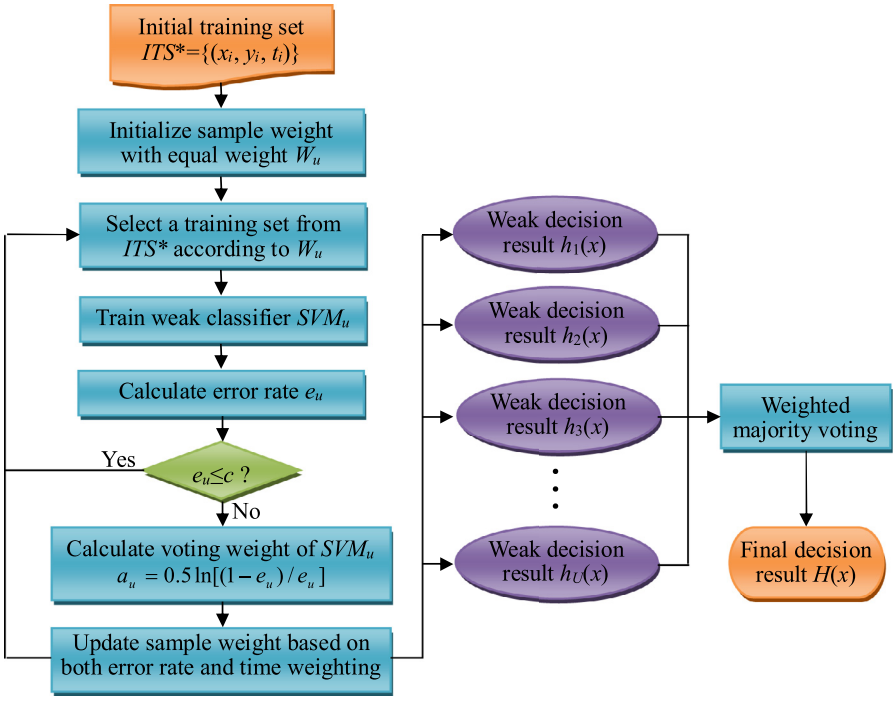

Fig. 2. The framework of ADASVM-TW.

Table 4
The algorithm of ADASVM-TW algorithm.

Algorithm ADASVM-TW algorithm.
Input: Initial training set with time label, namely $ITS^* = \{(x_i, y_i, t_i)\}$ $(i = 1, 2, ..., N)$.
Initialization:
(1) Initialize sample weight distribution as $W_1 = \{w_1^1, w_1^2, \ldots, w_1^N\} = \{\frac{1}{N}, \frac{1}{N}, \ldots, \frac{1}{N}\}$.
(2) Initialize error threshold as c.
For $u = 1, 2, ..., U$
(1) Select a training set TS_u from ITS^* according to the weight distribution W_u.
(2) Train a base SVM classifier SVM_u, also represented as $h_u(x)$, based on TS_u.
(3) Resubstitute TS_u to SVM_u and calculate the error rate e_u.
(4) If $e_u \le c$ or $e_u \ge 0.5$, then go back to step 1 with the value of u unchanged.
(5) Calculate the voting weight of SVM_u as $a_u = 0.5 \ln[(1 - e_u)/e_u]$.
(6) Update sample weight $WW_{u+1}(i) = W_u(i) \times \exp\{-\alpha_u y_u h_u(x_i) \exp[\lambda y_u h_u(x_i) t_i]\}$.
(7) Normalize sample weight $W_{u+1}(i) = \frac{WW_{u+1}(i)}{\sum WW_{u+1}(i)}$.
End
Output: ADASVM-TW ensemble $H(x) = \sum_{u=1}^{U} [h_u(x) \times \alpha_u]$.

growth, risk level, and cash flow capacity. We collect all sample companies' T-2 financial ratio data from China Stock Market and Accounting Research Database and use them as initial experimental data set. After deleting the sample companies with missing values, we use the three-times standard deviation method to further delete noisy samples. To select significant discriminative financial ratios, mean comparison is carried out between financial distress samples and non-distress samples. Then the final financial ratios for FDP modeling are selected out by stepwise multiple discriminant analysis.

As a result, the final data set for FDP modeling consists of 932 samples (namely 466 pairs of distress and non-distress companies) with 7 financial ratios (shown in Table 5), and they are divided into 13 batches according to different years.

5.3. Concept drift test

The main idea of concept drift test method based on error rate [38] can be summarized as follows. Firstly, divide the initial data set into several data subsets B_1–B_T according to time and use them to construct classifiers C_1–C_T. Secondly, use each classifier to test each data subset and save the cross testing performance. Finally, check whether the cross testing performance is worse when the training data subset is farther away from the testing data subsets. If yes, there is concept drift phenomenon. In terms of this method, we test whether FDCD exists between 2005–2012 using SVM as the classifier, and the results of cross testing accuracies are listed in Table 6.

On the whole, the cross testing performance is better when the nearer the training and testing data subsets are, although this is not always the true. With the testing subset going farther away from the training subset, the testing performance becomes worse overall. Therefore, the above experimental data set shows some degree of concept drift phenomenon and is suitable for this study.

5.4. Parameter setting

There are mainly three parameters to be set in the above algorithms, namely the number of Adaboost iterations, error threshold c, and time weighting parameter λ.

Please cite this article as: J. Sun et al., Dynamic financial distress prediction with concept drift based on time weighting combined with Adaboost support vector machine ensemble, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.12.019

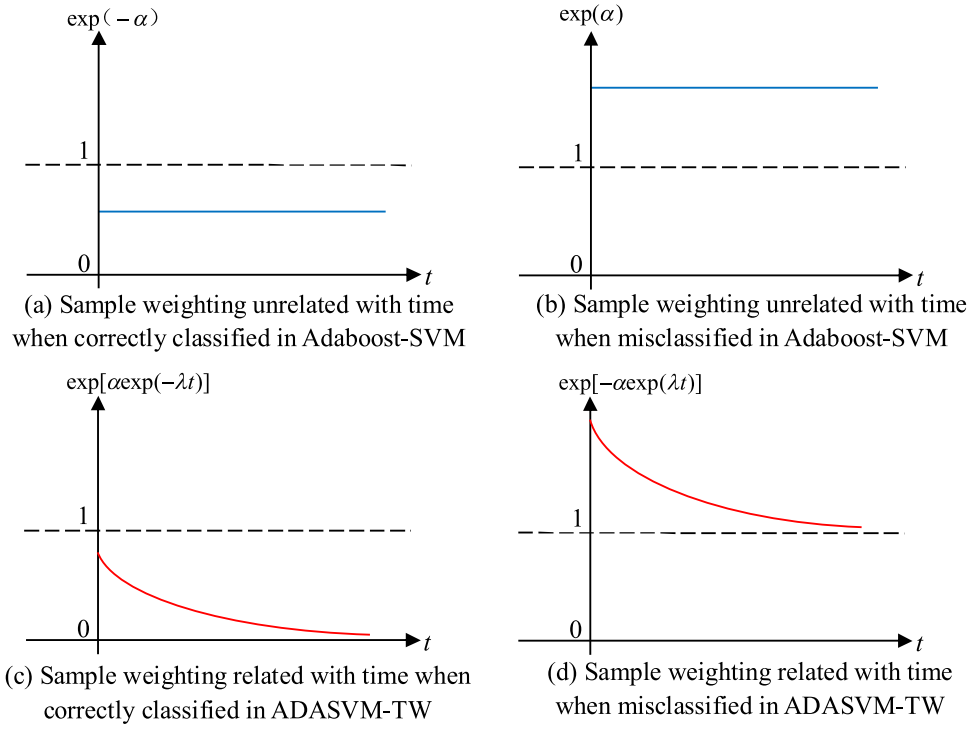

Fig. 3. The sample weight adjustment function of ADASVM-TW.

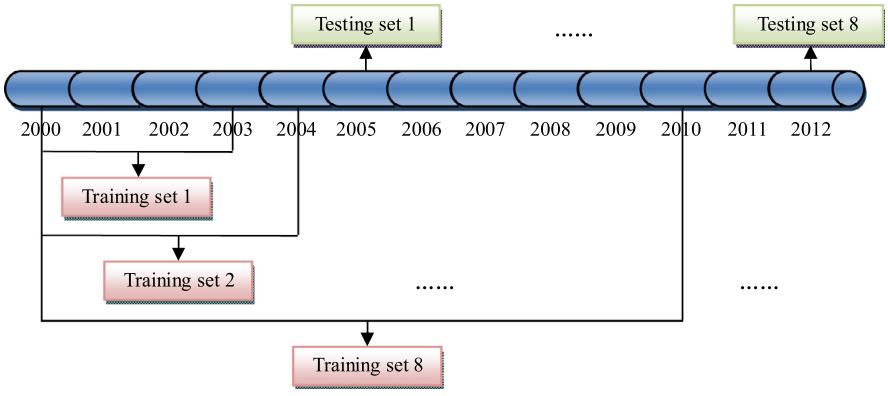

Fig. 4. Simulation of time moving in the experiment.

Table 5
Financial ratios selected for FDP modeling.

No.	Financial ratio	Category
1	Working capital ratio	Short-term liability
2	Cash flow ratio (Net cash flow from operating activities/current liabilities)	Short-term liability
3	Net cash flow from investment activities per share	Cash flow capacity
4	Business gross profit ratio	Profitability
5	Net profit to total assets	Profitability
6	Net profit rate to current assets	Profitability
7	Current assets turnover	Operating capacity

5.4.1. Setting the number of Adaboost iterations

Theoretically speaking, the performance of Adaboost ensemble improves with increasing of the number of Adaboost iterations. The training accuracy tends to 100% when the iteration number tends to infinity. However, on one hand too many times of iterations would lead to more computational time. On the other hand, sample weight would focus on certain part of difficult samples and finally make the training subset selected in later iterations remain unchanged, which would decrease the performance of Adaboost ensemble for testing samples. These two problems will make it unsuitable for realistic applications. Therefore, we train Adaboost-SVM ensembles on 2005–2010 data batches and validate them on 2012 data batch with different numbers of Adaboost iterations for 30 times. Table 7 shows the average accuracy for certain iteration number. It is obvious that Adaboost-SVM obtains ideal performances when the iteration number equals to 20 or 30. When the iteration number is over 50, the performance of Adaboost-SVM decreases obviously. Hence the number of Adaboost

Please cite this article as: J. Sun et al., Dynamic financial distress prediction with concept drift based on time weighting combined with Adaboost support vector machine ensemble, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.12.019

Table 6
Results of concept drift test.

Training								
Testing	2012	2011	2010	2009	2008	2007	2006	2005
2012	82.81	78.13	76.56	67.19	75.00	68.75	59.38	56.25
2011	84.78	93.48	86.96	71.74	95.65	73.91	63.04	65.22
2010	78.95	73.68	78.95	64.47	73.68	73.68	64.47	59.21
2009	61.11	64.81	61.11	75.93	70.37	74.07	72.22	64.81
2008	76.09	69.57	73.91	63.04	76.09	56.52	60.87	63.04
2007	71.15	76.92	71.15	77.88	82.69	83.65	78.85	73.08
2006	73.39	74.19	69.35	76.61	78.23	76.61	77.42	75.00
2005	64.52	70.97	56.45	77.42	74.19	83.87	80.65	77.42

Table 7
The average accuracies for different numbers of Adaboost iterations.

Iteration number	10	20	30	40	50	60	70
Average accuracy (%)	87.66	91.24	91.73	90.07	84.32	80.43	81.24

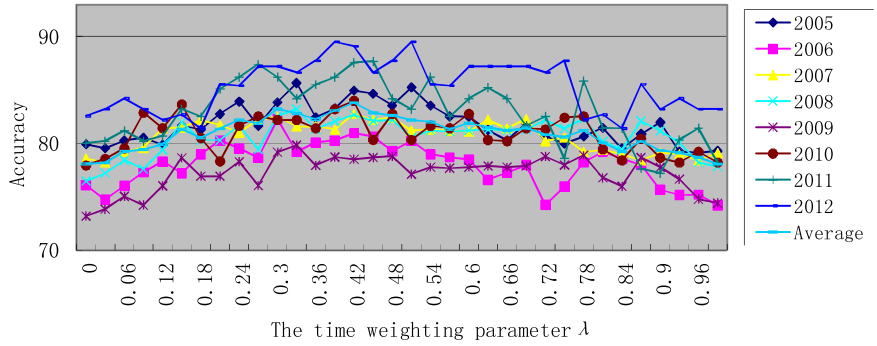

Fig. 5. Testing accuracies with different time weighting parameter values.

iteration is set as 20 in the following experiments since it is more efficient and has almost the same performance as 30 times of iteration. It is concluded from this experiment that the relatively ideal range for the Adaboost iteration number is [10,50]. Since Breiman indicated the number of base classifier should increase as the number of categories increased [39], it is advised that the Adaboost iteration number should be set as [10, 30] for binary classification data sets and be set as [30, 50] for data sets with three or more than three classes.

5.4.2. Setting the error threshold

Early empirical research on FDP of Chinese listed companies shows that single SVM has the accuracy between 68% and 92% [40]. At the same time, the results of our pre-experiment show that the highest accuracy of SVM is around 90%. Therefore, the error threshold is set as 10% in the following experiments.

5.4.3. Setting time weighting parameter

Pre-experiment of BE-LWS for DFDP from 2005 to 2012 are carried out as the time weighting parameter λ varies from 0 to 0.99 with the step of 0.03. For each value of the time weighting parameter, the testing accuracies from 2005 to 2012 are recorded and graphed in Fig. 5. It is shown that the accuracies from 2005 to 2012 rise or drop mostly in the same direction with the variation of λ. When $\lambda = 0.42$, the average testing accuracy is the highest and the testing accuracy of each year is also the highest or relatively high. Overall, when $\lambda > 0.7$, either the average or each year's testing accuracy begins to drop greatly. Therefore, the time weighting parameter λ is set as 0.42 in the following experiments.

5.5. Experimental result and analysis

By simulating the time moving process, the three DFDP models of BE-LWS, DEVE-AT, and ADASVM-TW are respectively trained and tested with the real data of Chinese listed companies. Since there is certain degree of randomness in the sample selection process of DEVE-AT and ADASVM-TW, we ran their algorithms for 30 times and calculated the average accuracy for each year. The experimental results of testing accuracies are listed in Table 8, in which the last column means the mean of the testing accuracies from 2005 to 2012. To visually illustrate the performance of the four methods, the testing accuracies from 2005 to 2012 are graphed as curves in Fig. 6, and the average testing accuracies are charted as columns in Fig. 7.

5.5.1. Analysis on overall performance for DFDP

Fig. 5 shows that both DEVE-AT and ADASVM-TW curves are above the other four cures, indicating that DEVE-AT and ADASVM-TW have better overall performance than single SVM, BE-LWS, Adaboost-SVM and Timeboost-SVM. Besides, Adaboost-SVM, Timeboost-SVM, DEVE-AT and ADASVM-TW curves have fewer waves than single SVM and BE-LWS cures, showing more stable DFDP performance with time moving on. In terms of the eight-year average testing accuracies in Table 8 and Fig. 7, ADASVM-TW is the highest and DEVE-AT follows it. In detail, ADASVM-TW is higher than single SVM, BE-LWS, Adaboost-SVM and Timeboost-SVM respectively by 7.83%, 5.88%, 1.31% and 1.15%, and DEVE-AT is higher than single SVM, BE-LWS, Adaboost-SVM and Timeboost-SVM respectively by 7.57% and 5.62%, 1.05% and 0.89%. Overall, Adaboost-SVM and Timeboost-SVM have almost the same DFDP performance, and they perform evidently better than single SVM

Please cite this article as: J. Sun et al., Dynamic financial distress prediction with concept drift based on time weighting combined with Adaboost support vector machine ensemble, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.12.019

Table 8
The experimental results of testing accuracies (%).

	2005	2006	2007	2008	2009	2010	2011	2012	Mean
Single SVM	81.67	77.87	80.39	77.27	75.00	79.73	81.82	84.38	79.77
BE-LWS	86.67	81.97	83.38	71.44	69.23	84.93	88.63	87.50	81.72
Adaboost-SVM	86.06	84.73	85.08	85.84	83.52	83.75	89.51	91.83	86.29
Timeboost-SVM	86.36	84.95	83.77	86.67	85.00	84.62	89.34	90.91	86.45
DEVE-AT	85.75	84.87	85.09	87.57	85.74	85.56	90.50	93.61	87.34
ADASVM-TW	86.24	85.04	85.32	87.51	85.45	86.16	91.21	93.88	87.60

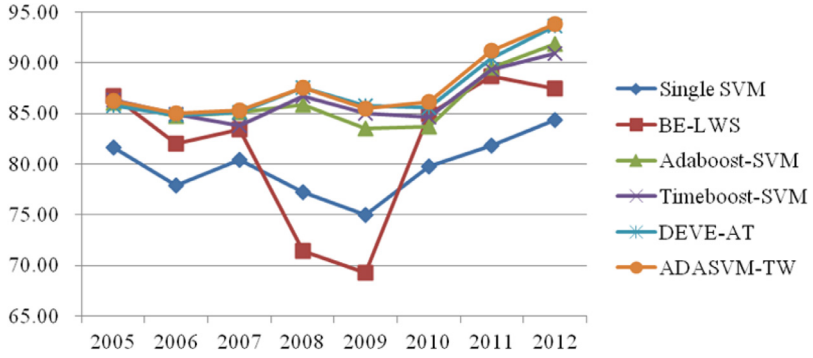

Fig. 6. Curves of the testing accuracies from 2005 to 2012.

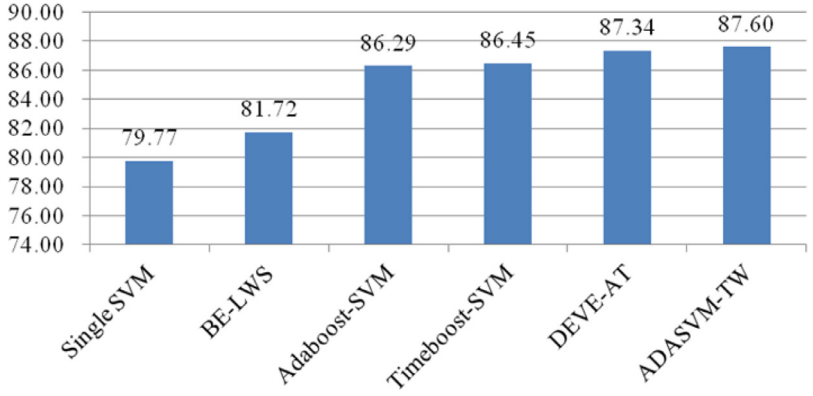

Fig. 7. Columns of the average testing accuracies from 2005 to 2012.

and BE-LWS. Compared with single SVM, BE-LWS has higher average testing accuracy and is overall more suitable for DFDP.

To test whether one method has significantly higher average testing accuracy than the other, paired mean comparison is carried out by *t*-test for each pair of the sixmethods and the results are listed in Table 9. It is shown that ADASVM-TW significantly outperforms DEVE-AT, Timeboost-SVM, Adaboost-SVM, BE-LWS and Single SVM respectively at the significance level of 10%, 5%, 1%, 5% and 1%. Although DEVE-AT is significantly outperformed by ADASVM-TW, it still outperforms Timeboost-SVM, Adaboost-SVM, BE-LWS and Single SVM respectively at the significance level of 5%, 5%, 10% and 1%. Besides, either Timeboost-SVM or Adaboost-SVM outperforms BE-LWS and Single SVM respectively at the significance level of 10% and 1%. However, there is no significant difference between Timeboost-SVM and Adaboost-SVM and between BE-LWS and Single SVM, although the former has higher average testing accuracy than the latter.

Above all, among the four methods, both DEVE-AT and ADASVM-TW are better tools than Timeboost-SVM, Adaboost-SVM, BE-LWS and Single SVM for DFDP in case of FDCD. Among DEVE-AT and ADASVM-TW, ADASVM-TW is more preferred. Because it not only has better performance than DEVE-AT, but also it only needs half of the training time for DEVE-AT.

5.5.2. Analysis on concept drift disposal

It has been mentioned in Section 3.3, BE-LWS algorithm resorts the training batches according to whether their data distribution is similar to the newest training batch, and then allocates time weight to the corresponding classifiers trained on them. When the data distribution of the testing batch differs greatly from the newest training batch, BE-LWS shall theoretically have very bad performance. To some extent, BE-LWS provides us a good tool to recognize the happening of sudden concept drift by observing whether its accuracy drops suddenly and greatly with time going on. As shown in Table 8 and Fig. 6, the testing accuracies of BE-LWS for 2008 and 2009 drop suddenly and are even lower than those of single SVM. The possible reason for this phenomenon is that the data distribution of 2008 batch and 2009 batch is very different from 2006 batch and 2007 batch, which makes the *Classifier* 0 trained on 2006 or 2007 batch unsuitable for the new financial distress concept. Because *Classifier* 0 is used for determining the time sequences of other training batches, the base classifiers similar to *Classifier* 0 and unsuitable for the new concept are highly weighted, which finally leads to the failure of FDP when sudden FDCD happens in 2008 and 2009. However, the testing accuracies of DEVE-AT and ADASVM-TW from 2005–2012 keep relatively high and stable, showing their good performance even

Please cite this article as: J. Sun et al., Dynamic financial distress prediction with concept drift based on time weighting combined with Adaboost support vector machine ensemble, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.12.019

Table 9
Results of mean comparison between each pair of the four methods.

Pair	Mean difference	*t* statistic	Significance
ADASVM-TW vs. DEVE-AT	0.26500	2.239	0.060*
ADASVM-TW vs. Timeboost-SVM	1.14875	3.146	0.016**
ADASVM-TW vs. Adaboost-SVM	1.31125	4.059	0.005***
ADASVM-TW vs. BE-LWS	5.88250	2.513	0.040**
ADASVM-TW vs. Single SVM	7.83500	9.349	0.000***
DEVE-AT vs. Timeboost-SVM	0.88375	2.546	0.038**
DEVE-AT vs. Adaboost-SVM	1.04625	3.024	0.019**
DEVE-AT vs. BE-LWS	5.61750	2.301	0.055*
DEVE-AT vs. Single SVM	7.57000	8.420	0.000***
Timeboost-SVM vs. Adaboost-SVM	0.16250	0.490	0.639
Timeboost-SVM vs. BE-LWS	4.73375	1.972	0.089*
Timeboost-SVM vs. Single SVM	6.68625	8.186	0.000***
Adaboost-SVM vs. BE-LWS	4.57125	2.058	0.079*
Adaboost-SVM vs. Single SVM	6.52375	9.825	0.000***
BE-LWS vs. Single SVM	1.95250	1.118	0.300

* Represents the significance level of 10%.
** Represents the significance level of 5%.
*** Represents the significance level of 1%.

in case of sudden FDCD. On the whole, DEVE-AT and ADASVM-TW can dispose both gradual and sudden concept drift better than single SVM, BE-LWS, Adaboost-SVM and Timeboost-SVM. Although BE-LWS can dispose gradual concept drift better than single SVM, it is the most awkward when disposing sudden concept drift.

6. Conclusion

In the current unstable economic environment, effective FDP models are important for both enterprise financial risk management and bank credit risk management. Although many literatures focused on FDP problem in the past decades, DFDP research in consideration of concept drift is just rising and needs further research. This study proposes two new approaches, namely DEVE-AT and ADASVM-TW, for DFDP in condition of FDCD, based on time weighting combined with Adaboost SVM ensemble, so as to enrich the methodology system of DFDP.

DEVE-AT makes DFDP decision by combining both the outputs of Adaboost-SVM and Timeboost-SVM. The traditional Adaboost-SVM is purely an error-based decision expert, which considers misclassified samples as more important than the correctly classified samples. However, the Timeboost-SVM is a purely time-based decision expert, which considers newer samples as more important than older samples and does not concern about whether a sample is misclassified or not. Therefore, DEVE-AT can integrate both advantages of Adaboost-SVM and Timeboost-SVM to some extent, and are suitable for DFDP.

ADASVM-TW is the internal integration of Adaboost-SVM and time weighting. It uses a novel sample weight updating function in the Adaboost iteration by considering both factors of sample misclassification and time weighting. Through this way, the weights of the newer samples will be decreased in less degree than the older ones when correctly classified, and the weights of the newer samples will be increased in more degree than older ones when misclassified. Therefore, ADASVM-TW is more suitable for handling FDCD and can improve the accuracy of DFDP.

In the empirical experiment, the time moving process is simulated with one year as time step, and the performance of DEVE-AT and ADASVM-TW is compared with single SVM and BE-LWS. T-2 data of 932 Chinese listed companies' 7 financial ratios are divided into 13 batches, based on which 30 times of experiments are carried out. Experimental results indicate that ADASVM-TW achieves the highest average testing accuracy with DEVE-AT following it, and their performance is significantly better than single SVM, BE-LWS, Adaboost-SVM and Timeboost-SVM at least at level of 10%. Besides, DEVE-AT and ADASVM-TW can dispose either gradual or sudden FDCD well, followed by Adaboost-SVM and Timeboost-SVM, while BE-LWS is not good at disposing sudden FDCD.

Although this study focuses on non-stationary ensemble approaches for DFDP with non incremental SVM as the base classifiers, it is worth to explore the application of dynamic SVM algorithms for DFDP, e.g. the incremental SVM proposed by Laskov et al. [41] and the SVM for nonlinear time series adopted by Mukherjee et al. [42], and compare them with dynamic SVM ensemble approaches. Such experimental analysis is not provided in this study, which is also the limitation of this paper.

Acknowledgment

This research is partially supported by the National Natural Science Foundation of China [Grant numbers 71371171 and 71571167]. The authors gratefully thank anonymous referees for their useful comments and editors for their work.

Reference

[1] J. Sun, H. Li, Dynamic financial distress prediction using instance selection for the disposal of concept drift, Expert Syst. Appl. 38 (2011) 2566–2576.
[2] J. Sun, H. Li, H. Adeli, Concept drift-oriented adaptive and dynamic support vector machine ensemble with time window in corporate financial risk prediction, IEEE Trans. Syst. Man Cybern. Part A Syst. 43 (4) (2013) 801–813.
[3] J. Sun, K. He, H. Li, SFFS-PC-NN optimized by genetic algorithm for dynamic prediction of financial distress with longitudinal data streams, Knowl. Based Syst. 24 (2011) 1013–1023.
[4] J. Sun, H. Li, P.-C. Chang, K.-Y. He, The dynamic financial distress prediction method of EBW-VSTW-SVM, Enterp. Inf. Syst. 10 (2016) 611–638.
[5] E.I. Altman, Financial ratios discriminant analysis and the prediction of corporate bankruptcy, J. Financ. 4 (1968) 589–609.
[6] J. Ohlson, Financial ratio and the probabilistic prediction of bankruptcy, J. Account. Res. 18 (1980) 109–131.
[7] H. Frydman, E.I. Altman, D.L. Kao, Introducing recursive partitioning for financial classification: the case of financial distress, J. Financ. 1 (1985) 269–292.
[8] W.-S. Chen, Y.-K. Du, Using neural networks and data mining techniques for the financial distress prediction model, Expert Syst. Appl. 36 (2009) 4075–4086.
[9] D. Wu, L. Liang, Z. Yang, Analyzing the financial distress of Chinese public companies using probabilistic neural networks and multivariate discriminate analysis, Socio-Econ. Plan. Sci. 42 (3) (2008) 206–220.
[10] J. Sun, X.-F. Hui, Financial distress prediction based on similarity weighted voting CBR, Lect. Notes Artif. Intell. 4093 (2006) 947–958.
[11] J. Sun, H. Li, Data mining method for listed companies' financial distress prediction, Knowl. Based Syst. 21 (2008) 1–5.
[12] J.H. Min, Y.-C. Lee, Bankruptcy prediction using support vector machine with optimal choice of kernel function parameters, Expert Syst. Appl. 28 (2005) 128–134.
[13] X.-F. Hui, J. Sun, An application of support vector machine to companies' financial distress prediction, Lect. Notes Artif. Intell. 3885 (2006) 274–282.
[14] T. Gesel, B. Baesens, D. Martens, From linear to non-linear kernel based classifiers for bankruptcy prediction, Neurocomputing 73 (16-18) (2010) 2955–2970.

Please cite this article as: J. Sun et al., Dynamic financial distress prediction with concept drift based on time weighting combined with Adaboost support vector machine ensemble, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.12.019

[15] H. Li, J. Sun, Predicting business failure using support vector machines with straightforward wrapper: a re-sampling study, Expert Syst. Appl. 38 (10) (2011) 12747–12756.
[16] H. Jo, I. Han, Integration of case-based forecasting, neural network, and discriminant analysis for bankruptcy prediction, Expert Syst. Appl. 11 (4) (1996) 415–422.
[17] J. Sun, H. Li, Listed companies' financial distress prediction based on weighted majority voting combination of multiple classifiers, Expert Syst. Appl. 35 (2008) 818–827.
[18] V. Ravi, H. Kurniawan, P. Thai, P. Kumar, Soft computing system for bank performance prediction, Appl. Soft Comput. 8 (1) (2008) 305–315.
[19] E. Alfaro, N. García, M. Gámez, D. Elizondo, Bankruptcy forecasting: an empirical comparison of AdaBoost and neural networks, Decis. Support Syst. 45 (2008) 110–122.
[20] M. Kim, D. Kang, Ensemble with neural networks for bankruptcy prediction, Expert Syst. Appl. 37 (2010) 3373–3379.
[21] C.-F. Tsai, J. Wu, Using neural network ensembles for bankruptcy prediction and credit scoring, Expert Syst. Appl. 34 (2008) 2639–2649.
[22] J. Sun, H. Li, Financial distress prediction using support vector machines: ensemble vs. individual, Appl. Soft Comput. 12 (8) (2012) 2254–2265.
[23] J. Sun, M. Jia, H. Li, AdaBoost ensemble for financial distress prediction: an empirical comparison with data from Chinese listed companies, Expert Syst. Appl. 38 (2011) 9305–9312.
[24] S.Y. Kim, A. Upneja, Predicting restaurant financial distress using decision tree and AdaBoosted decision tree models, Econ. Model. 36 (2014) 354–362.
[25] J. Sun, H. Li, Q.-H. Huang, K.-Y. He, Predicting financial distress and corporate failure: a review from the state-of-the-art definitions, modeling, sampling, and featuring approaches, Knowl. Based Syst. 57 (2014) 41–56.
[26] J.C. Schlimmer, R.H. Granger, Incremental learning from noisy data, Mach. Learn. 1 (3) (1986) 317–354.
[27] G. Widmer, M. Kubat, Effective learning in dynamic environments by explicit context tracking, Lect. Notes Comput. Sci. 667 (1993) 227–243.
[28] S.J. Delanya, P. Cunninghamb, A. Tsymbalb, L. Coyle, A case-based technique for tracking concept drift in spam filtering, Knowl. Based Syst. 18 (2005) 187–195.
[29] G. Hulten, L. Spencer, P. Domingos, Mining time-changing data streams„ in: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2001, pp. 97–106.
[30] I. Koychev, Gradual forgetting for adaptation to concept drift, in: Proceedings of the ECAI 2000 Workshop Current Issues in Spatio-Temporal Reasoning, 2000, pp. 101–106.
[31] R. Klinkenberg, Learning drifting concepts: example selection vs. example weighting, Intell. Data Anal. 8 (2004) 281–300.
[32] N. Kyosuke, Y. Koichiro, O. Takashi, ACE: adaptive classifier-ensemble system for concept-drifting environments, in: Proceedings of the Sixth International Workshop on Multiple Classifier Systems. 176–185, 2005.
[33] Y. Wu, Y. Liu, G. Gao, Z.D. Mao, W.S. Ma, T. He, An adaptive ensemble classifiers for concept drifting stream, in: Proceedings of the 2009 Computer Intelligence and Data Mining, 2009, pp. 69–75.
[34] G. Song, Y. Ye, H. Zhang, X. Xu, R.Y.K. Lau, F. Liu, Dynamic clustering forest: an ensemble framework to efficiently classify textual data stream with concept drift, Inf. Sci. 357 (2016) 125–143.
[35] V.N. Vapnik, Statistical Learning Theory, Springer-Verlag, New York, 1998.
[36] N. Cristianini, J. Shawe-Taylor, An Introduction to Support Vector Machines and other Kernel-based Learning Methods, Cambridge University Press, England, 2000.
[37] Y. Freund, R.E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, J. Comput. Syst. Sci. 55 (1) (1997) 119–139.
[38] X. Luo, D. Wang, S. Feng, G. Yu, An algorithm for classifying data stream with recurrent concept drift, J. Comput. Res. Develop. 4 (2009) 400–405.
[39] L. Breiman, Randomizing outputs to increase prediction accuracy, Mach. Learn. 40 (3) (2000) 229–242.
[40] Z.-W. Ni, Y.-J. Xue, L.-P. Ni, H.-W. Xiao, Research of multiple kernel SVM based on manifold learning in financial distress prediction, Syst. Eng. Theory Pract. 34 (10) (2014) 2666–2674.
[41] P. Laskov, C. Gehl, S. Krüger, K.-R. Müller, Incremental support vector learning: analysis, implementation and applications, J. Mach. Learn. Res. 7 (3) (2006) 1909–1936.
[42] S. Mukherjee, E. Osuna, F. Girosi, Nonlinear prediction of chaotic time series using support vector machines, in: Proceedings of THE 1997 IEEE Workshops on Neutral Network for Signal Processing, 1997, pp. 511–520.

Please cite this article as: J. Sun et al., Dynamic financial distress prediction with concept drift based on time weighting combined with Adaboost support vector machine ensemble, Knowledge-Based Systems (2016), http://dx.doi.org/10.1016/j.knosys.2016.12.019