Expert Systems with Applications 42 (2015) 3194–3204

Contents lists available at ScienceDirect

Expert Systems with Applications

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

Selection of Support Vector Machines based classifiers for credit risk domain

Paulius Danenas [a,b,*], Gintautas Garsva [a]

[a] Department of Informatics, Kaunas Faculty, Vilnius University, Muitines Str. 8, Kaunas, Lithuania
[b] Currently works in: Center of Information Systems Design Technologies, Department of Information Systems, Kaunas University of Technology, Studentu Str. 50-313a, Kaunas, Lithuania

ARTICLE INFO

Article history:
Available online 10 December 2014

Keywords:
Support Vector Machines
SVM
Particle swarm optimization
Credit risk
Default assessment
Classification

ABSTRACT

This paper describes an approach for credit risk evaluation based on linear Support Vector Machines classifiers, combined with external evaluation and sliding window testing, with focus on application on larger datasets. It presents a technique for optimal linear SVM classifier selection based on particle swarm optimization technique, providing significant amount of focus on imbalanced learning issue. It is compared to other classifiers in terms of accuracy and identification of each class. Experimental classification performance results, obtained using real world financial dataset from SEC EDGAR database, lead to conclusion that proposed technique is capable to produce results, comparable to other classifiers, such as logistic regression and RBF network, and thus be can be an appealing option for future development of real credit risk evaluation models.

1. Introduction

One of the most important research issues in financial domain is development of working credit risk evaluation and bankruptcy prediction models. Credit risk is one of frequently faced financial risks, which can be defined as the possibility that counterparty will fail to meet its obligations by agreed terms that will cost invested money for the lender. Minimization of such debts is critical for managing risk in financial institutions as Basel II capital accord defines new standards for capital adequacy in banks thus optimal capital allocation is essential for financial institutions. Thus proper, efficient and effective credit risk evaluation tools for and credit risk, such as highly discriminative credit scoring models, are obligatory for every financial institution. A credit score is primarily based on various financial, social, demographic and other data provided of the customers and about the customer, such as credit reports and information obtained from external evaluators and auditors such as major credit reporting agencies. Credit scores are often used to determine the amount of loan or interest rate that particular customer qualifies for.

Various machine learning techniques, such as artificial neural networks (abbr. ANN), have also gained a lot of attention from various researchers which are working in credit risk domain. ANN is understood as a computing model with a graph, that defines data structure for neural network, and interconnection pattern, which describes its architecture. This technique is well suited for developing accurate credit scoring systems and can perform competitively when compared to other classification techniques, such as logistic regression, MDA, decision trees. However, Support Vector Machines (abbr. SVM) technique has recently become one of the most widely research and applied techniques in this field. This technique offers several advantages compared to ANN such as absence of local minimas and relatively simple architecture. Many works in credit risk evaluation domain showed that they can show performance comparable to ANN or to outperform them (Danenas & Garsva, 2010; Kim & Ahn, 2012; Yu, Yao, Wang, & Lai, 2011).

Linear SVM are not widely explored in this domain, mainly because of its reduced flexibility, related to absence of kernel function and nonlinear mappings. However, state-of-the art linear SVM implementations show much faster performance than nonlinear SVM, leading to their suitability for large-scale SVM classification and regression problems. Nonlinear SVMs are not efficient on larger scale learning and also suffer from imbalanced learning problem (Batuwita & Palade, 2013). To the knowledge of the authors, this problem is not yet addressed in popular SVM packages (LibSVM, SVM^{Light}, etc.), except weight assignment to different classes (a variation of cost-sensitive learning). Large scale learning is important in our context as our research framework involves

* Corresponding author.
E-mail addresses: danpaulius@gmail.com (P. Danenas), gintautas.garsva@khf.vu.lt (G. Garsva).

http://dx.doi.org/10.1016/j.eswa.2014.12.001
0957-4174/

obtaining and preprocessing significant amounts of Extensible Business Reporting Language (abbr. XBRL)[1] documents from external datasources (we used SEC datasource as basis for our framework) after they are issued, as well as mining obtained data using automatic label identification and classifier training. Several linear SVM classifiers from LIBLINEAR package (Fan, Chang, Hsieh, Wang, & Lin, 2008) were chosen for development of classification functionality in our system. Promising results, obtained previously (Danenas & Garsva, 2010; Danenas, Garsva, & Gudas, 2011), as well as results in (Chang, Hsieh, Chang, Ringgaard, & Lin, 2010) motivate the research of linear SVM classifiers in parallel with nonlinear state-of-the-art SVM modeling techniques. In particular, our research seeks to explore the potential of this technique against medium or larger datasets (in this context, "larger" is defined as "having 2000 or more instances") in credit risk domain, while combining it with "sliding window" approach for training and testing. The classifier selection is optimized using swarm intelligence metaheuristic (particularly, particle swarm optimization); this approach has gained significant amount of attention from the research community because of its conceptual simplicity and ability to balance both exploration and exploitation (Thangaraj, Pant, Abraham, & Bouvry, 2011).

The remainder of the paper is organized as follows. In Section 2, the key points on relevant credit risk related research is presented. Section 3 briefly describes Support Vector Machines, the classification technique used to develop our approach; additionally, it presents linear SVM algorithms, used in this research, together with motivation to use them. Particle swarm optimization is described in Section 4, together with necessary improvements. Section 5 presents the whole research methodology, together with metrics used for evaluation, while Section 6 gives a brief description of the data used in experiment, describes the experiment configuration and discusses the obtained results. Finally, Section 7 highlights the conclusions and directions for future research.

2. Earlier works

The earliest works in research of credit risk date to 1968, when Altman applied multiple discriminant analysis (MDA) to develop his Z-Score model (Altman, 1968), using two different samples and obtaining accuracy of 96% and 79%, respectively. MDA was also applied by other researchers to develop their own models (Deakin, 1972; Taffler, 1982) or to improve and analyze existing ones (Grice & Dugan, 2001; Grice & Ingram, 2001). Another well-known early development (Springate, 1978) was also based on stepwise MDA and four ratios, resulting in accuracy rate of 92.5%; 83.3% and 88% accuracy rates later were reported after testing the developed model with other samples (Sands, Springate, & Var, 1983). (Ohlson, 1980) applied logit analysis reporting accuracy of 96.12%, 95.55% and 92.84% for prediction within one year, two years and one or two years respectively. While (Begley, Ming, & Watts, 1996) showed that Ohlson's model might perform better than Altman original and improved Z-Scores, their evaluation has also been criticized (Grice & Dugan, 2001). Zmijewski (1984) used two samples of 840 companies (40 of them were bankrupt companies) for training and prediction purpose, using probit and maximum likelihood techniques, and obtained 72% accuracy. Another known credit risk model (Shumway, 2001) was developed using hazard analysis and the same predictors as in original Altman model.

Different machine learning techniques became an object of interest for solutions in financial domain soon after they were discovered to show their potential in solving different problems. Artificial neural network based techniques (abbr. ANN) were the first techniques to be successfully applied in this field. An early survey (Vellido, Lisboa, & Vaughan, 1999) indicated that backpropagation neural networks (abbr. BPNN) were the most popular machine learning technique among researchers in credit risk domain during 1992–1998; this is also confirmed for both cases of finance and business domain in general (Wong, Lai, & Lam, 2000; Wong & Selvi, 1998). Recent research proposed a lot of state-of-the-art ANN-based hybrid models; fuzzy ANN with particle swarm optimization (abbr. PSO) for parameter selection (Huang, 2008), wavelet neural networks with differential evolution applied for their training (Chauhan, Ravi, & Karthik, 2009), knowledge-based artificial neural network (abbr. KBANN) with rule extraction from trained neural networks (Bae & Kim, 2011), neurofuzzy systems (Chen, Huang, & Lin, 2009), ensembles of ANN (Tsai & Wu, 2008; Yu, Wang, & Lai, 2008) are only a few examples. Other important techniques in the domain of credit risk evaluation and bankruptcy prediction include decision trees (Duman, Ekinci, & Tanrıverdi, 2012; Khandani, Kim, & Lo, 2010) and their ensemble variations, particularly random forests (Fantazzini & Figini, 2008; Kruppa, Schwarz, Arminger, & Ziegler, 2013) or other (Zhang, Zhou, Leung, & Zheng, 2010).

Support Vector Machines (abbr. SVM) are another type of learning machines, which are able to perform comparably to ANN, while overcoming their problems of architectural complexity and entrapment in local minimas. One of the most actively researched and discussed problems, related to SVM, is parameter selection for kernel function and cost/complexity parameter; it is pointed out in the relevant literature that it should be set by the expert. Yet, a lot of work has been done in order to simplify this problem using various heuristic techniques, such as genetic algorithm (Cao, Lu, Wang, & Wang, 2012; Wu, Tzeng, Goo, & Fang, 2007) or swarm intelligence (Yun, Cao, & Zhang, 2011; Zhou, Bai, Tian, & Zhang, 2008). A survey of SVM-based methods in credit risk domain (Danenas & Garsva, 2009) also indicated that evolutionary or swarm intelligence techniques for SVM parameter selection or fuzzy logic/rough sets integration usually helps to improve classifier performance.

Recent SVM technique, Least Squares SVM, abbr. as LS-SVM (Suykens & Vandewalle, 1999), gained a lot of attention from different researchers, as its applications identified the efficiency in performance, while at the same time simplifying SVM computing using to a set of linear equations. LS-SVM has been applied as standalone or part of hybrid technique in credit risk domain by several authors (Cao et al., 2012; Lai, Yu, Zhou, & Wang, 2006; Li, Song, & Li, 2012). Ensemble learning, another trend of soft computing, has also been widely researched in the context of credit risk, as different authors prove empirically the capability of classifier ensembles to obtain better classification performance by stabilizing the classification results by reflecting variation within a data set (Hsieh & Hung, 2010). Recent developments of SVM ensemble models include reliability-based and weight-based strategies (Zhou, Lai, & Yu, 2010), adaptive linear ANN (Yu, Yue, Wang, & Lai, 2010), bagging or boosting procedures (Ghodselahi, 2011; Wang & Ma, 2012).

However; while higher accuracy is mostly obtained using novel nonlinear SVM methods on small amounts of data, performance of such techniques often suffers on real-world larger datasets. Our main focus lies on research which is performed on such datasets. The amount of it is not large, which may be influenced by the limitations of availability of the necessary financial/bankruptcy data (although the number of open financial datasources seems to be rising). (Harris, 2015) used a dataset of over 20,000 entries from Barbados credit unions for model development to develop SVM linear and nonlinear classifier together with clustered SVM; the results indicated that performance of linear SVM did not significantly differ from SVM using RBF kernel; similar conclusion can be drawn from the results in (Niklis, Doumpos, & Zopounidis,

[1] https://www.xbrl.org/

2014). Other recent research (Zhang, Gao, & Shi, 2014) used a USA credit dataset of over 6000 instances and also reported results which indicate that application of nonlinear SVM kernel for generic SVM, fuzzy SVM and hybrid fuzzy SVM does not show significant increase in classification accuracy, compared to linear SVM (resulting in accuracy of ~75%). Other works, which apply their developments on large datasets, tend to use SVM with RBF kernel function, as the most popular choice among such research (Harris, 2013; Horta & Camanho, 2013).

3. Overview of SVM methods

3.1. Formulation of Support Vector Machines

Support Vector Machines (abbr. as SVM) is an efficient and effective pattern recognition technique, which is based on Vapnik–Chervonenkis *structural risk minimization* (abbr. as SRM) theory (Cortes & Vapnik, 1995; Vapnik, 2000). SVM learning is based on mapping the sample points into a high-dimensional feature space in order to search and obtain an optimal separating hyperplane, which maximizes the sum of the distances between two classes in this space. Given the set of l training instances $S = \{(x_1, y_1), \ldots, (x_l, y_l)\} \in (X \times Y)^l$ where $x_i \in \mathbb{R}^l, Y \in \{-1; 1\}$, we seek to obtain the optimal real-valued function $\varphi(\mathbf{x})$ in order to form hyperplane $sgn(\langle \varphi(\boldsymbol{x}) \cdot \boldsymbol{w} \rangle + b)$, that allows separation of points in S. Given distance w, there exist two parallel boundaries $\langle \varphi(\boldsymbol{x}) \cdot \boldsymbol{w} \rangle + b = \mp 1$, which can exactly separate two classes. The separating hyperplane is between them. The margin between these two boundaries is defined as $2/\|w\|$. In order to maximize this margin, the following problem must be solved

$$\min_{\boldsymbol{w},b} \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w}$$

$$s.t.\ y_i(\varphi(x_i)w_i + b) \geq 1, \quad i = \overline{1 \ldots l} \tag{3.1}$$

By solving this problem, we obtain optimal hyperplane $sgn((\boldsymbol{w}^* * x) + b^*)$, with $\boldsymbol{w}^*$ and b^* solutions of (3.1). This problem can be rewritten in dual quadratic programming problem form using the Lagrangian

$$\min_{\propto} \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \propto_i \propto_j (x_i \cdot x_j) - \sum_{i-1}^{l} \propto_i$$

$$s.t. \sum_{i-1}^{l} y_i \propto_i = 0, \quad \propto_i \geq 0, \quad i = \overline{1 \ldots l} \tag{3.2}$$

Here α is a vector of l Lagrange multipliers, where each α_i corresponds to a training example (x_i, y_i). To adopt this approach to linearly non-separable training sets, each instance (x_i, y_i) is associated with a slack variable $\xi_i \geqslant 0$; the constraint in (3.1) becomes $y_i(\varphi(x_i) \cdot w_i + b) + \xi_i \geq 1$ Thus the problem in (3.1) becomes

$$\min_{\boldsymbol{w},b} \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_{i-1}^{l}\xi_i$$

$$s.t.\ y_i(\varphi(x_i) \cdot w_i + b) + \xi_i \geq 1, \quad i = \overline{1 \ldots l} \tag{3.3}$$

where C is a parameter that determines the trade-off between the maximum margin and the minimum classification error. Kernel function $K(x_i, x_j) \equiv \varphi(x_i)^T\varphi(x_j)$can be used to map linearly non-separable instances into a higher (maybe infinite) dimensional space; the Lagrangian then becomes

$$\min_{\propto} \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \propto_i \propto_j K(x_i \cdot x_j) - \sum_{i-1}^{l} \propto_i$$

$$s.t. \sum_{i-1}^{l} y_i \propto_i = 0, \quad 0 \leq \propto_i \leq C, \quad i = \overline{1 \ldots l} \tag{3.4}$$

If $\alpha^* = \overline{(\alpha_1, \ldots, \alpha_l)}$, the optimal separating hyperplane becomes

$$Sgn\left(\sum_{i=1}^{l} y_i \alpha_i^* K(x_i, x_j) + b^*\right) \tag{3.5}$$

Note, that in this work only linear kernel functions are considered, i.e., $K(x_i, x_j) \equiv x_i^T x_j$.

3.2. Linear SVM classifiers

Linear SVM classifiers generate weight vector $\boldsymbol{w}$ as the model using a decision function $sgn(\boldsymbol{w}^T\boldsymbol{x})$ (i.e., an instance is predicted as "positive" if $\boldsymbol{w}^T\boldsymbol{x} > 0$). They benefit over SVM implementations, using kernel functions, in terms of training speed and reduced complexity, as the exclusion of kernel mapping simplifies training and reduces the amount of computation to train classifier. However, that lack of nonlinear mapping results in less flexibility and reduced performance, compared to kernel-based SVM classifiers. Yet, practical application of the latter may also be significantly impacted by selection of best kernel function or highly unbalanced training. SVM is one of the machine learning techniques, which usually do not perform well in case of unbalanced training without additional steps (Batuwita & Palade, 2013), although many solutions are developed to overcome this problem such as internally implemented class-weighting, cost-sensitive learning and evaluation (Elkan, 2001), internal classifier enhancements (Lessmann, 2004; Qi, Tian, Shi, & Yu, 2013), sampling techniques, such as bootstrap, undersampling, oversampling (Margineantu & Dietterich, 2000). Therefore, it may sometimes be appropriate to use linear classifiers in practice in order to achieve performance, which is desirable, or at least as near desirable. State-of-the-art linear SVM classifiers, which implement novel computational techniques and are optimized for large-scale learning, can be used as an alternative to achieve this objective.

In this research, classifiers implemented in LIBLINEAR package (Fan et al., 2008) were considered for the research. LIBLINEAR includes a family of linear SVM and logistic regression classifiers for large-scale SVM classification. Five of these classifiers were used in the experiment; their formulations as optimization problems are given below:

- L2-regularized L1-loss SVC, formulated as

$$\min_{\boldsymbol{w}} \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_{i-1}^{l} \max(0; 1 - y_i\boldsymbol{w}^T x_i) \tag{3.6}$$

- L2-regularized L2-loss SVC, defined as

$$\min_{\boldsymbol{w}} \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_{i-1}^{l} (\max(0; 1 - y_i\boldsymbol{w}^T x_i))^2 \tag{3.7}$$

- L2-regularized logistic regression, expressed as

$$\min_{\boldsymbol{w}} \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_{i-1}^{l} \log\left(1 + e^{-y_i\boldsymbol{w}^T x_i}\right) \tag{3.8}$$

- L2-regularized L2-loss SVC, defined as

$$\min_{\boldsymbol{w}} \|w\|_1 + C\sum_{i-1}^{l} (\max(0; 1 - y_i\boldsymbol{w}^T x_i))^2 \tag{3.9}$$

- multi-class SVM by Crammer and Singer:

$$\min_{\boldsymbol{w},\xi} \frac{1}{2}\boldsymbol{w}_m^T\boldsymbol{w}_m + C\sum_{i-1}^{l}\xi_i \tag{3.10}$$

L1-SVM (3.6) and L2-SVM (3.7) implementations use coordinate descent optimization method, whereas logistic regression (3.8) and L2-SVM (3.9) are implemented using trust region Newton method; for more details of their implementation we refer to (Fan et al., 2008). Bias term b is obtained as complementary dimension of the vector $\boldsymbol{w}$ ($\boldsymbol{w} \equiv [\boldsymbol{w}^T, b]$), after specifying constant B for each instance s.t. $x_i^T \equiv [x_i^T, B]$ (Fan et al., 2008). The selection of this parameter, together with cost parameter C, is performed by experts; thus, it is purposeful to apply certain heuristic search procedures, such as grid search, or artificial intelligence techniques, particularly evolutionary techniques or swarm intelligence, as possible solutions for this task. The latter is further discussed in Section 4.

4. Particle swarm optimization for linear SVM classifier selection

Particle swarm optimization (Kennedy, Eberhart, & Shi, 2001) is one of social optimization approaches motivated by the coordinate movement of fish pools and bird flocks, and based on patterns of their synchronous cooperative behavior and regrouping after some influential change. The conceptual simplicity of this technique led to many modifications and practical applications. It is often reported to converge to optimal or near-optimal solutions, while incorporating efficient mechanisms to control search space exploration. In PSO, each possible solution is represented as a particle, while fitness value of this particle defines its position relatively to the searched object. Two optimal solutions – local optimal solution, found by the particle itself, and global optimal, found by the whole swarm, – are identified and updated in each iteration. The number of iterations can be fixed or the algorithm might be terminated if no further improvement is observed.

PSO is applied for various tasks. One of them is optimization of classifier performance by selecting optimal parameters for the classifiers, such as SVM (Kong, Cheng, Ding, & Chai, 2010; Li-xia, Yi-qi, & Liu, 2011; Lin, Ying, Chen, & Lee, 2008). In this paper, we explore heuristic search principle based on PSO for selection of linear SVM classifier from the set of classifiers described in Section 3.2 (further referred as PSO-LinSVM). Initial experiments on Australian and German credit datasets indicated promising results, as described in (Garsva & Danenas, 2014). Initially this approach was presented in previous work (Danenas & Garsva, 2012), however, the version of PSO-LinSVM algorithm in this paper is slightly improved for better exploration of hybrid search space. This is done by modifying velocity and position equations according to the influence of variables that are integer by their nature, rather than rounding them, as in our initial version of PSO-LinSVM.

In, PSO-LinSVM, each particle $p = \langle p_1; p_2; p_3 \rangle$ defines the algorithm used for classification, cost parameter C and parameter for bias term B. Main steps of PSO-LinSVM are described in Algorithm 1. Here $cl \in \{i | cl_{\min} \leqslant i \leqslant cl_{\max}, cl_{\min} \in Z, i \in Z, cl_{\max} \in Z\}$ is a set of inner integer encodings, corresponding to particular classifiers, $rangeC = [C_{\min}; C_{\max}]$ is a range of cost parameters which is considered ($C \geq 0$, according to predefined constraint in SVM formulation); $rangeBias = [b_{\min}; b_{\max}]$ is an interval to search for B parameter. *no_iterations* represent defines the number of iterations run in this algorithm; while it is not expected to converge to an *ideal* solution (i.e., to a classifier which is able to separate given data points perfectly), the main goal is a *satisfactory* solution, obtained after a certain number of iterations. Other options, such as certain threshold to stop search, can be established as well. Finally, c_1 and c_2 represent coefficients for cognitive and social components, as in original PSO algorithm.

Algorithm 1. PSO-LinSVM algorithm

PSO-LinSVM(c1, c2, rangeC, rangeBias, no_iterations)
Initialize a 3-dimensional swarm P
For each particle p in P
  Initialize $p = \langle p_1 \in [cl_{min;} cl_{max}]; p_2 \in [C_{min;} C_{max}];$
  $p_3 \in [b_{min;} b_{max}] \rangle$, $p_1 \in$, $p_2 \in$, $p_3 \in R$
While i < no_iterations
  if i = no_iterations return SVM(y_p);
  For each particle p in P
    Compute fitness value of x_p
    if $f(x_p) <$ *then set* x_p *as the global best position*
    if $f(y_p) < f(\hat{y})$ *then* $\hat{y} = y_p$
  For each particle p in P
    Calculate maximum allowed velocity V_{max} *for each dimension in p*
    Calculate next step velocity $v_p(i+1)$ *for each dimension in p*
    If $v_p(i+1) > V_{max}$ *then* $v_p(i+1) = V_{max}$
    Calculate particle position and fitness value
    If particle position $p_1 > cl_{max}$ *then it is set to* cl_{min}
    If particle position $p_2 > C_{max}$ *then it is set to* C_{min}
  Calculate global best position value $f(\hat{y}_t)$ *obtained at iteration t*
Next i
Output: *Optimal linear SVM classifier SVM*(y_p)

The main objective is to maximize fitness function defined as sum of *True Positive rate* (defined in Section 5.2) values for each class:

$$f_{fitness} = \sum_{i=1}^{N_C} TPR_i \tag{4.1}$$

where N_C is the number of classes, $TPR_i - TPR$ value for ith class (defined further in Section 5.2). Note that many similar works in this field choose to maximize accuracy in order to obtain a classifier with best accuracy performance. Although accuracy is considered as one of the important metrics in classification, it is possible to obtain high accuracy rate with highly imbalanced datasets, when most "majority" instances are classified correctly, although the discriminating function completely fails to identify "minority" instances. This is especially important in credit risk domain, as the inability to identify bankrupt companies would result in higher loss, when the misclassification, resulting in "good" company identified as bad, would result in unobtained profit. Therefore, function in (4.1) is designed to account respectively for both of these classes.

To summarize, the main differences, compared to the original PSO algorithm, are:

- Modifications in initialization procedure and velocity equation to match mixed search space requirements in this algorithm.
- The search space is constrained by hard constraints which cannot be violated. To deal with this problem, each particle is "teleported" to lower boundary after it gets to the upper boundary of the constrained space.

5. Research and evaluation methodology

5.1. Development and testing principles

This research relies on our previously used principles (Danenas & Garsva, 2010; Danenas et al., 2011); however, it is extended with sliding window approach which is useful for testing classifier performance for several following periods. The full methodology of the experiment is as follows:

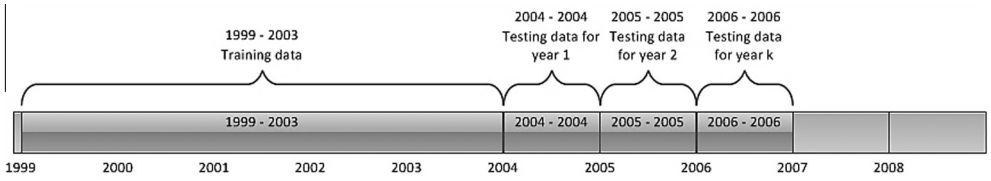

Fig. 1. Illustration of sliding window testing approach.

1. Evaluate each financial entry by using expert evaluation methods (discriminant analysis, other) and compute bankruptcy classes.
2. Eliminate instances, which resulted in null (empty) evaluations in Step 1. Lack of data or division by zero may be of the main reasons for such problems.
3. Remove attributes from the dataset which have more missing values than specified threshold. 30% was considered in the experimental research, although lower or higher threshold may be set, depending on the availability of the data.
4. Impute data by filling missing values with specified value of particular attribute. Mean value of the attribute was used in this experiment, although other options may be chosen as well. One of such options is the mean value for the particular attribute, according to the dimensionality of the dataset (company, group, sector, etc.); such imputation was also applied in our previous research.
5. Perform the following steps for each $m \in [1; n\text{-}k]$ (n is the total number of periods, k is the number of periods which are used for forecasting):
   (a) Apply feature selection procedure in order to select the most relevant attributes and reduce number of dataset dimensions.
   (b) Perform classifier parameter selection manually or using heuristic procedures.
   (c) Train classifier using data from first m periods.
   (d) Apply hold-out testing using data from period $p \in [m+1; m+k]$, $p \in \mathbb{Z}$.

Training and testing steps are illustrated in Fig. 1. The process starts with training dataset of single year data (in this case, 1999–2000); training dataset iteratively increases as it is complemented with data from next year for the next training step.

Feature selection step is important for 3 reasons:

1. To reduce data dimensionality of attribute space, thus forming a new subset of attributes and reducing the complexity of the model (the aspects of quality and complexity).
2. To obtain a set of statistically significant attributes to develop a new classifier based on other evaluator (the aspect of importance).
3. To remove ratios which correlate between themselves, in order to ensure that all attributes used in the model are statistically significant (the aspect of significance). This is especially important in this methodology, as different financial ratios are often composed of the same variables and can be related to each other.

The output of each iteration in experimental stage is the trained classifier and the list of selected attributes.

5.2. Evaluation of results

Metrics, commonly used in machine learning for evaluation in similar problems, such as accuracy, TP (True Positive) and F-Measure rates, were selected to evaluate classification performance.

Accuracy is defined as a proportion of correct predictions to total predictions as

$$accuracy = \frac{\#TruePositives + \#TrueNegatives}{\#Total\ number\ of\ instances} \tag{5.1}$$

where *#TruePositives* and *#TrueNegatives* correspondingly are the numbers of "positive" and "negative" cases correctly identified. *True Positive rate* (TPR) or *Sensitivity* is a ratio of *#TruePositives* and total number of total positive instances:

$$TPR_i = \frac{\#TruePositives_i}{\#TruePositives_i + \#FalseNegatives_i} \tag{5.2}$$

Here $\#FalseNegatives_i$ is the number of "positive" cases in ith class incorrectly classified as "negative". $\#TruePositives_i$ and $\#TrueNegatives_i$, naturally, represent the numbers of correctly identified "positive" and "negative" cases of this class.

Finally, *F-Measure* is defined as harmonic mean of *precision* and *recall (True Positive)* measures; similarly to TPR measure, it can be defined for each class in the training dataset. It is preferred to accuracy for analysis of the classification performance case of unbalanced learning:

$$F_i = \frac{2 \times precision_i \times recall_i}{precision_i + recall_i} \tag{5.3}$$

where *precision* is defined as $\#TruePositives_i/(\#TruePositives_i + \#FalsePositives_i)$. $\#FalsePositives_i$ is the number of "negative" cases in ith class, incorrectly classified as "positive".

6. Experimental results

6.1. Data used in the experiment

The experiments were made by using data of period 1999–2007 from EDGAR database, manufacturing sector. Table 1 presents main characteristics of dataset, including number of instances in each of formed classes.

The initial dataset used in the experiment consists of yearly financial records with 51 financial ratios used in financial analysis, computed using original primary financial data from balance and income statement data. These ratios are listed in Table 2.

Table 1
Main characteristics of data used in experiment.

Year	# Entries labeled as		Total entries	# Of selected attributes
	Not risky (NR)	Risky (R)		
1999	1312	537	1849	12
2000	1869	589	2458	15
2001	1753	672	2425	15
2002	1709	777	2486	13
2003	1770	723	2493	14
2004	1920	637	2557	13
2005	1964	660	2624	14
2006	1636	429	2065	14
2007	1545	393	1938	14
2008	483	109	592	14
Total	15961	5527	21487	

Table 2
Financial ratios used in research.

General efficiency ratios	Accounts receivables effectiveness ratio
Sales to inventory ratio	Asset turnover ratio
Accounts payable to sales ratio	Capital turnover
Collection period ratio	*Fixed asset turnover ratios*
Profitability ratios	Fixed assets turnover
Profitability of own capital	Current assets turnover
Constant capital turnover	Cost of goods for each sale unit ratio
Return on assets	Receivables turnover
Constant capital profitability	Liabilities turnover
Gross margin profit ratio	Total operational costs for each sale unit ratio
Sales income ratio	Current ratio
Return on capital employed	Quick ratio
Return on equity capital	*Short-term solvency*
Return on investment	Liquidity ratio for constant capital
General profitability ratio	Short-time payment ratio
Net profitability	Quick liquidity ratio
Operating profitability	Short term liabilities ratio
Ownership and debt ratios	*Long-term solvency*
Capital share	Current liabilities to inventory ratio
Financial debt to equity minus goodwill ratio	Total liabilities to net worth ratio
Financial debt to equity ratio	Liabilities and equity ratio
Interest coverage ratio	Liabilities to equity without goodwill and intangibles ratio
Debt ratio	Debt coverage in cash ratio
Net capital to total assets ratio	Long term liabilities coverage in fixed assets
Adjusted capital ratio	Total liabilities ratio
Turnover ratios	Long-term liabilities to equity ratio
Depreciation ratio	Long-term liabilities to total assets ratio
Fixed assets return ratio	Short-term and long-term liabilities ratio
Fixed assets turnover ratio	

Entries were evaluated and labeled using Zmijewski's score (Zmijewski, 1984), as one of the most widely applied techniques in the domain of credit risk evaluation. It was selected according to the origin of the data (the data comes from USA companies). This score is defined as follows:

$$Z = -4.336 - 4.513 * (Net\ Revenue/Total\ Assets) + 5.679 * (Total\ Debt/Total\ Assets) + 0.004 * (Current\ Assets/Current\ Liabilities) \quad (6.1)$$

The company is considered as tend to bankrupt if $Z > 0$. Thus two groups of companies – companies which are "healthy" or "non-risky" (possibly are not going to bankrupt) and "risky" (which might become bankrupt) – were formed.

6.2. Experiment configuration

The code and algorithms for the experiments were implemented using WEKA[2] machine learning framework with LIBLINEAR 1.7. The test was run using 5 classifiers described in Section 3.2. Cost parameter C and bias b for these algorithms were chosen experimentally, by using direct search in range of $C \in [0; 100]$ and $b \in [0; 1]$. For each formed dataset, feature selection procedure was applied using WEKA's correlation-based feature subset evaluator (Hall, 1998). This evaluator aims to select the subset that has maximal correlation to the class, while the features have minimal intercorrelation. The optimality of feature subset is defined by a merit measure

$$Merit_S = \frac{k\overline{r_{cf}}}{\sqrt{k + k(k-1)\overline{r_{ff}}}} \quad (6.2)$$

where $\overline{r_{cf}}$ is the average correlation between features and class, and $\overline{r_{ff}}$ is the average correlation between features (using Pearson's coefficient). According to Hall (1998), this technique is preferable for its low computational demand – CFS requires $m \times ((n^2 - n)/2)$ operations for computing the pairwise feature correlation matrix, where m is the number of instances and n is the initial number of features, and the feature selection search requires $(n^2 - n)/2$ operations (worst case).

The number of periods for testing (k, as indicated in Section 5.1) was set to 3. The main objective was to identify best classifier in terms of accuracy during all three testing periods, i.e., the condition for evaluation is formally defined as

$$\max_{C,B} \sum_{i=1}^{k} accuracy_i \quad (6.3)$$

where acc_i is the value of classification accuracy obtained at ith testing period. It is important to note that classifiers, which performed best during all testing periods, were preferred to better performance in any certain single period. In the case of imbalanced learning, maximum sum of TPR values would be preferred, considering that the classifier, which can identify all classes at some level of accuracy, is more preferable than classifier which can identify, for example, instances of one dominating class:

$$\max_{cl} \sum_{i=1}^{k} \sum_{j=1}^{N_C} TPR_{ij} \quad (6.4)$$

$$s.t. 0 \le \sum_{i=1}^{N_C} TPR_i \le N_C$$

Here TPR_{ij} is TPR value for jth class obtained while testing classifier testing at ith testing period, N_C – the number of classes.

PSO-LinSVM was configured to run with *rangeC* = [1; 100] and *rangeBias* = [−5; 10], using 15 iterations and a swarm of 20 particles. In order to compare performance to other classifiers, multinomial logistic regression model with a ridge estimator (le Cessie & van Houwelingen, 1992) and *RBFNetwork* (normalized Gaussian radial basis function network which uses *k-means* clustering algorithm to provide the basis functions) classifiers from WEKA package were used in the experiment, considering their performance with larger amounts of data. Logistic regression classifier was run using default parameters, while the number of clusters for RBF network classifier was set to 32 in order to increase its classification performance.

6.3. Experimental results

Feature selection results are given in Table 3. It can be seen that several ratios were selected to include in all of these models: accounts payable to sales, short-time payment ratio, quick liquidity and current liquidity ratios, long-term liabilities to equity, quick ratio and returns on assets. Therefore, short-time ratios related to debt payment and liquidity, as well as return on assets and sales tend to be seen as the most important in developed models. Profit ratio was indicated by feature selection procedure as important for the first three periods; depreciation ratio was selected for successive periods, starting with year 2004. Several ownership, financial performance and solvency ratios were also identified as important in developed models, although their influence changed during modeling, as the number of instances in the dataset increased.

Table 4 depicts classification performance of linear SVM with direct search. Classification accuracy together with True Positive rate and F-Measure rates for each class are given, where classifiers were selected according to the principles described in Section 6.1. Tables 5–7 give these results for PSO-LinSVM, logistic regression

[2] www.cs.waikato.ac.nz/ml/weka/

Table 3
Financial ratios selected after feature selection procedure.

	2000	2001	2002	2003	2004	2005	2006	2007
Accounts payable to sales ratio	☑	☑	☑	☑	☑	☑	☑	☑
Gross margin profit ratio	☑	☑	☑					
Short-time payment ratio	☑	☑	☑	☑	☑	☑	☑	☑
Capital share	☑	☑	☑	☑				
Debt ratio		☑		☑				
Depreciation coefficient					☑	☑	☑	☑
Quick liquidity ratio	☑	☑	☑	☑	☑	☑	☑	☑
Long-term liabilities to total assets ratio	☑	☑	☑	☑	☑	☑	☑	☑
Long-term liabilities to equity ratio			☑		☑	☑	☑	☑
Fixed assets turnover					☑			
Current ratio	☑	☑	☑	☑	☑	☑	☑	☑
Interest coverage ratio		☑	☑			☑	☑	☑
Adjusted capital ratio					☑	☑	☑	☑
Quick ratio	☑	☑	☑	☑	☑	☑	☑	☑
Financial debt to equity minus goodwill ratio	☑							
Liabilities and equity ratio	☑		☑	☑	☑	☑	☑	☑
Return on assets	☑	☑	☑	☑	☑	☑	☑	☑
Total liabilities to net worth ratio	☑			☑				
Total operational costs for each sale unit ratio	☑	☑	☑	☑	☑	☑	☑	☑

Table 4
Linear SVM experimental results.

Training period			2000	2001	2002	2003	2004	2005	2006	2007	Avg
Best performing classifier			CS-SVM	L1-LSVM (dual)	L1-LSVM (dual)	CS-SVM	L1-LSVM (dual)	L1-LSVM (dual)	L2-LSVM (primal)	L2-LSVM (dual)	
C			20	20	20	15	20	15	15	5	
Bias			0.7	1.0	0.7	1.0	0.4	0.7	0.7	1.0	
	Accuracy		96.702	96.344	95.471	95.504	91.604	93.085	92.008	92.295	94.127
Year 1	TP	NR	0.973	0.974	0.970	0.965	0.974	0.977	0.971	0.981	0.973
		R	0.951	0.940	0.917	0.925	0.745	0.756	0.724	0.675	0.829
	FMeas	NR	0.977	0.973	0.968	0.970	0.945	0.957	0.951	0.954	0.962
		R	0.941	0.942	0.922	0.911	0.818	0.820	0.789	0.770	0.864
	Accuracy		96.183	94.233	95.348	96.785	92.940	91.445	91.960	–	94.128
Year 2	TP	NR	0.966	0.966	0.972	0.983	0.977	0.966	0.977	–	0.972
		R	0.953	0.938	0.898	0.923	0.749	0.716	0.675	–	0.836
	FMeas	NR	0.972	0.970	0.969	0.979	0.956	0.947	0.952	–	0.964
		R	0.940	0.928	0.906	0.936	0.816	0.775	0.762	–	0.866
	Accuracy		96.032	96.286	96.710	97.389	91.291	92.127	–	–	94.973
Year 3	TP	NR	0.962	0.970	0.987	0.987	0.964	0.981	–	–	0.975
		R	0.956	0.940	0.908	0.923	0.716	0.667	–	–	0.852
	FMeas	NR	0.972	0.975	0.978	0.984	0.946	0.953	–	–	0.968
		R	0.933	0.927	0.933	0.936	0.772	0.764	–	–	0.878

and RBF network. Average values for each performance variable at *k*th testing step were also calculated in order to compare average classification performance of each classifiers; they are presented at the right of each table. Average values for each dataset used in training (1999–2000, 1999–2001, etc.) were also calculated as average of *k* values for overall accuracy, TPR for "healthy" and TPR for "risky" class measures. They are depicted graphically in Figs. 1 and 2.

The accuracy is above 90%, which can be considered as a comparatively good result. Best results were obtained while training classifier sequentially with data from first five years (starting with year 1999) as classification accuracy remained over 95%. Later it decreased, although the number of instances increased. No single classifier dominated among those which showed best results – Crammer–Singer multiclass SVM showed best performance twice, L1 dual linear SVM – four times and L2 linear SVM, both primal and dual – once per each classifier for the last two cases. The obtained TPR values for both "risky" (R) and "non-risky" (NR) classes can be considered as a good result (both were over 0.9 in first four periods, and over 0.7 in next periods). This indicates that instances for both of these classes were classified successfully; high F-Measure values also prove this. Unbalanced learning techniques may seem not necessary here, although their integration may considered for performance improvement. Parameters *C* and *bias* varied; the experiment showed that *bias* parameter had significant influence and the performance might depend on proper selection of this parameter.

Table 5 indicates that PSO-LinSVM application for classification resulted in less stable performance, than using direct search. However, it is important to note that the latter classifier development is more conformant to "training–testing-validation" paradigm, that is, the results presented here are results obtained with validation

Table 5
PSO–LinSVM experimental results.

Training period			2000	2001	2002	2003	2004	2005	2006	2007	Avg
Classifier			L2-RLR (primal)	L2-SVM (dual)	L2-SVM (dual)	L2-RLR	L2-SVM (primal)	L2-SVM (dual)	L2-SVM (dual)	L2-SVM (dual)	
C parameter			46.5068	9.4532	20.0452	76.0741	1.0000	32.1152	40.2581	20.4178	
Bias parameter			−3.5519	9.5337	3.5257	−0.6641	5.2068	6.7547	2.2369	1.3727	
	Accuracy		95.218	95.46	87.655	94.253	91.679	93.52	86.12	90.372	91.785
Year 1	TP	NR	0.984	0.977	0.979	0.976	0.967	0.968	0.857	0.990	0.962
		R	0.869	0.905	0.626	0.841	0.769	0.812	0.878	0.523	0.778
	F-Measure	NR	0.967	0.967	0.918	0.962	0.945	0.959	0.908	0.944	0.946
		R	0.91	0.926	0.747	0.879	0.824	0.839	0.719	0.667	0.814
	Accuracy		93.853	95.311	89.367	94.743	92.94	91.744	86.486	–	92.063
Year 2	TP	NR	0.98	0.972	0.984	0.985	0.969	0.955	0.865	–	0.959
		R	0.847	0.906	0.62	0.836	0.777	0.771	0.862	–	0.803
	F-Measure	NR	0.956	0.967	0.933	0.965	0.956	0.949	0.913	–	0.948
		R	0.896	0.918	0.744	0.89	0.821	0.791	0.701	–	0.823
	Accuracy		93.908	95.387	90.053	96.373	91.073	92.736	–	–	93.255
Year 3	TP	NR	0.967	0.976	0.993	0.99	0.954	0.969	–	–	0.975
		R	0.87	0.889	0.629	0.863	0.74	0.743	–	–	0.789
	F-Measure	NR	0.957	0.969	0.937	0.977	0.945	0.956	–	–	0.957
		R	0.893	0.906	0.762	0.908	0.771	0.790	–	–	0.838

Table 6
Logistic regression classifier experimental results.

Training period			2000	2001	2002	2003	2004	2005	2006	2007	Avg
	Accuracy		92.636	92.165	92.920	90.614	90.106	91.159	89.637	90.196	91.179
Year 1	TP	NR	0.986	0.983	0.984	0.983	0.985	0.974	0.969	0.979	0.980
		R	0.737	0.762	0.808	0.718	0.647	0.726	0.618	0.601	0.702
	F-Measure	NR	0.953	0.948	0.95	0.937	0.937	0.943	0.937	0.937	0.943
		R	0.827	0.843	0.877	0.816	0.765	0.805	0.712	0.624	0.784
	Accuracy		91.588	91.392	93.141	92.217	91.654	92.785	89.525	–	91.757
Year 2	TP	NR	0.985	0.981	0.978	0.983	0.995	0.977	0.962	–	0.980
		R	0.737	0.766	0.817	0.739	0.682	0.741	0.631	–	0.730
	F-Measure	NR	0.944	0.94	0.953	0.95	0.947	0.955	0.936	–	0.946
		R	0.829	0.848	0.874	0.826	0.804	0.81	0.71	–	0.814
	Accuracy		90.225	90.734	94.173	93.217	91.719	91.796	–	–	91.977
Year 3	TP	NR	0.975	0.973	0.981	0.994	0.996	0.968	–	–	0.981
		R	0.743	0.747	0.823	0.747	0.615	0.72	–	–	0.733
	F-Measure	NR	0.932	0.937	0.962	0.956	0.95	0.95	–	–	0.948
		R	0.826	0.824	0.876	0.847	0.755	0.781	–	–	0.818

Table 7
RBF network classifier experimental results.

Training period			2000	2001	2002	2003	2004	2005	2006	2007	Avg
	Accuracy		90.684	91.423	91.311	90.293	90.106	89.825	91.671	90.661	90.747
Year 1	TP	NR	0.96	0.957	0.966	0.962	0.954	0.957	0.968	0.957	0.960
		R	0.737	0.804	0.797	0.759	0.743	0.724	0.723	0.71	0.750
	F-Measure	NR	0.94	0.942	0.939	0.934	0.935	0.934	0.948	0.942	0.939
		R	0.791	0.839	0.851	0.819	0.789	0.782	0.783	0.755	0.801
	Accuracy		90.062	89.743	91.135	91.514	90.435	91.622	91.744	–	90.894
Year 2	TP	NR	0.956	0.943	0.959	0.961	0.957	0.969	0.964	–	0.958
		R	0.756	0.797	0.795	0.776	0.747	0.716	0.733	–	0.760
	F-Measure	NR	0.933	0.927	0.939	0.944	0.937	0.948	0.949	–	0.940
		R	0.808	0.829	0.839	0.82	0.797	0.78	0.783	–	0.808
	Accuracy		89.099	90.132	92.648	92.073	91.671	90.403	–	–	91.004
Year 3	TP	NR	0.952	0.946	0.968	0.976	0.972	0.957	–	–	0.962
		R	0.757	0.793	0.801	0.756	0.706	0.697	–	–	0.752
	F-Measure	NR	0.923	0.932	0.952	0.949	0.949	0.941	–	–	0.941
		R	0.813	0.823	0.844	0.828	0.779	0.747	–	–	0.806

set, as testing is performed using cross-validation procedure inside of PSO-LinSVM), whereas previous results are selected according mainly to testing results. In this case, results given in Table 4, should be viewed more as "ideal" results, while the results of the next three classifiers more adequately conform to "real-world learning" performance results.

Although there were several cases, when PSO-LinSVM resulted in significantly worse performance (with testing accuracy >5%

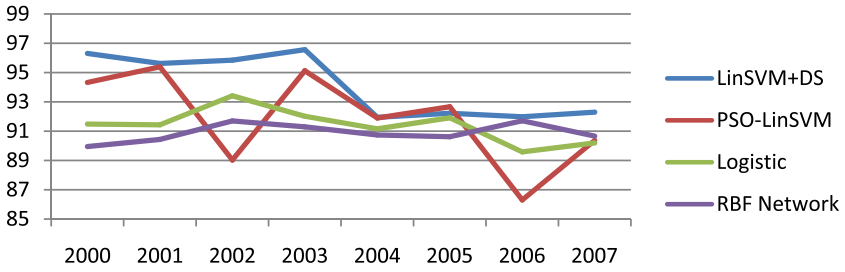

Fig. 2. Three-period average prediction accuracy of tested classifiers.

worse than direct search); these differences were not as significant in other cases. Although, there were cases when PSO-LinSVM resulted in higher accuracy.

Logistic classifier resulted in more stable classification results than PSO-LinSVM, yet, it did not outperform it. While PSO-LinSVM resulted in low performance results for 1999–2002 and 1999–2006 datasets, it outperformed logistic regression in other periods. RBF network classifier performed similarly to logistic regression, although its performance increased using larger datasets. Results in tables also indicate that PSO-LinSVM also achieved better average accuracy and TPR results or each testing period than logistic and RBF network classifiers.

Average prediction accuracy (Fig. 2) indicates that PSO-LinSVM mostly outperformed logistic and neural classifiers in terms of average accuracy, although they did not outperform linear SVM with direct search. All three classifiers also showed stable performance while recognizing instances, labeled as "healthy".

While PSO-LinSVM resulted in low performance results 1999–2002 and 1999–2006 datasets, it performed better in other periods (1999–2004 and 1999–2005) particularly. It also outperformed other classifiers in terms of average TPR ratio, showing that it was also capable to identify "risky" companies better than other of the tested classifiers. As indicated in Fig. 3, performance of identification of "risky" companies for logistic and neural classifiers started to decrease in at 2003 while PSO-LinSVM classifier showed comparatively stable results. Its performance decrease can be explained, as the classifier was trained also to focus on identify "minority" classes instead of classifying most of *all instances* which often resulted in large rate of correct classification of instances labelled with "majority" label ("healthy" companies, in this case). This can be easily recognized in case of 1999–2006 data, which resulted in comparatively low average accuracy, and TPR for "healthy" companies, but also in significantly better identification of "risky' companies. Therefore, it can be concluded that PSO-LinSVM was a better option for identification of "risky" companies in larger datasets where the number of instances representing such companies is significantly smaller than the number of instances representing "healthy" companies. This is often the case in real world bankruptcy related data.

7. Conclusions and future work

In this study we explore an approach for machine learning driven credit risk evaluation using linear Support Vector Machines, combined with sliding window approach for testing. It is considered that usage of higher dimensional data might improve performance as well as integrate additional knowledge. This technique is oriented at larger-scale learning, when nonlinear mappings are not necessary, not efficient to use or may result in highly increased computation. Such approach is also supported by the rising amount of available financial data, which is also standardized using specialized formats and frameworks, such as XBRL. Inspired by the idea of particle swarm optimization, we designed selection of the optimal classifier based on its core idea; performance of identifying individual classes is preferred to performance defined by overall accuracy in order to minimize the negative influence of imbalanced learning in favor of identification of companies labeled as "risky". Therefore, this approach seeks to maximize the identification of possible bankruptcies as much as possible.

Experimental research indicated high average classification accuracy (over 90%) although linear classifier was deployed. Obtained results are comparable to other classifiers, such as logistic regression and RBF network. Although performance of

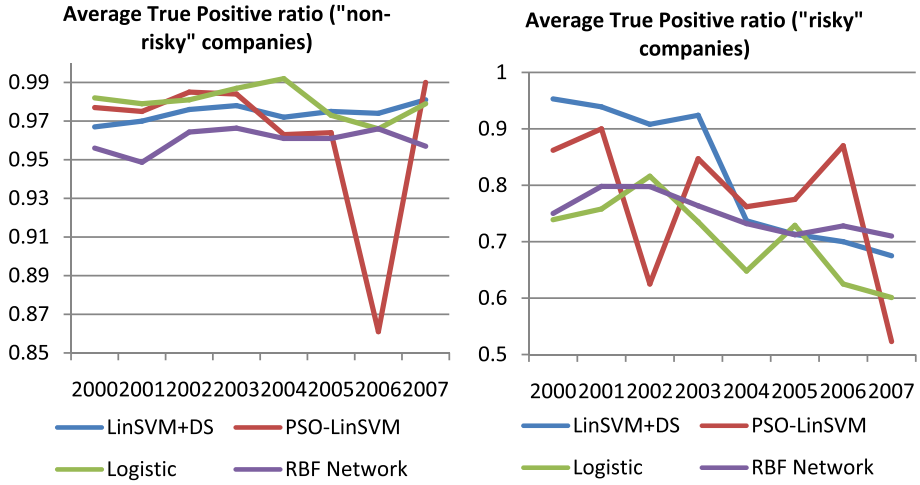

Fig. 3. Three-period average results of identification of different classes for tested classifiers.

PSO-LinSVM was less stable than of its competitors, it still indicates its potential in future research.

Therefore, future research would involve further PSO-LinSVM improvements, such as research on its stability, improved adoption for imbalanced learning and combination with other classifiers. Novel, state-of-the-art feature selection and instance selection techniques may be other fields which can be beneficial to our approach. Another important direction for further research is adoption for multiclass learning – the approach is designed to cope with multiple classes, although, additional testing and, possibly, necessary refinements must be made in order to ensure its proper functioning. In the future, we seek to explore it further using other types of expert evaluations; we also consider exploiting this approach to improve existing discriminant models by developing new ones on their basis, with additional bankruptcy data available.

References

Altman, E. I. (1968). Financial ratios, discriminant analysis and the prediction of corporate bankruptcy. *The Journal of Finance, 23*(4), 589–609.

Bae, J. K., & Kim, J. (2011). Combining models from neural networks and inductive learning algorithms. *Expert Systems with Applications, 38*(5), 4839–4850.

Batuwita, R., & Palade, V. (2013). Class imbalance learning methods for support vector machines. In H. He & Y. Ma (Eds.), *Imbalanced learning: Foundations, algorithms, and applications*. Hoboken, USA: John Wiley & Sons Inc.

Begley, J., Ming, J., & Watts, S. G. (1996). Bankruptcy classification errors in the 1980s: An empirical analysis of Altman's and Ohlson's models. *Review of Accounting Studies, 1*(4), 267–284.

Cao, J., Lu, H., Wang, W., & Wang, J. (2012). A novel five-category loan-risk evaluation model using multiclass LS-SVM by PSO. *International Journal of Information Technology & Decision Making, 11*(4), 857–874.

Chang, Y.-W., Hsieh, C.-J., Chang, K.-W., Ringgaard, M., & Lin, C.-J. (2010). Training and testing low-degree polynomial data mappings via linear SVM. *The Journal of Machine Learning Research, 11*, 1471–1490.

Chauhan, N., Ravi, V., & Karthik, Chandra. D. (2009). Differential evolution trained wavelet neural networks: Application to bankruptcy prediction in banks. *Expert Systems with Applications, 36*(4), 7659–7665.

Chen, H.-J., Huang, S. Y., & Lin, C.-S. (2009). Alternative diagnosis of corporate bankruptcy: A neuro fuzzy approach. *Expert Systems with Applications, 36*(4), 7710–7720.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning, 20*(3), 273–297.

Danenas, P., & Garsva, G. (2009). Support vector machines and their application in credit risk evaluation process. *Transformations in Business & Economics, 8*(3, Suppl. B), 57–69.

Danenas, P., & Garsva, G. (2010). Credit risk evaluation using SVM-based classifier. *Lecture Notes in Business Information Processing, 57*, 7–12.

Danenas, P., & Garsva, G. (2012). Credit risk evaluation modeling using evolutionary linear SVM classifiers and sliding window approach. *Procedia Computer Science, 9*, 1324–1333.

Danenas, P., Garsva, G., & Gudas, S. (2011). Credit risk evaluation model development using support vector based classifiers. *Procedia Computer Science, 4*, 1699–1707.

Deakin, E. (1972). A discriminant analysis of predictors of business failure. *Journal of Accounting Research, 10*, 167–179.

Duman, E., Ekinci, Y., & Tanrıverdi, A. (2012). Comparing alternative classifiers for database marketing: The case of imbalanced datasets. *Expert Systems with Applications, 39*(1), 48–53.

Elkan, Ch. (2001). The foundations of cost-sensitive learning. In *Proceedings of the 17th international joint conference on artificial intelligence*, Vol. 2, pp. 973–978.

Fan, R., Chang, K., Hsieh, C., Wang, X., & Lin, C. (2008). LIBLINEAR: A library for large linear classification. *The Journal of Machine Learning Research, 9*, 1871–1874.

Fantazzini, D., & Figini, S. (2008). Random survival forests models for SME credit risk measurement. *Methodology and Computing in Applied Probability, 11*(1), 29–45.

Garsva, G., & Danenas, P. (2014). Particle swarm optimization for linear support vector machines based classifier selection. *Nonlinear Analysis: Modelling and Control, 19*(1), 26–42.

Ghodselahi, A. (2011). A hybrid support vector machine ensemble model for credit scoring. *International Journal of Computer Applications, 17*(5), 1–5.

Grice, J. S., & Dugan, M. T. (2001). The limitations of bankruptcy prediction models: some cautions for the researcher. *Review of Quantitative Finance and Accounting, 17*(2), 151–166.

Grice, J. S., & Ingram, R. W. (2001). Tests of the generalizability of Altman's bankruptcy prediction model. *Journal of Business Research, 54*, 53–61.

Hall, M.A. (1998). *Correlation-based feature subset selection for machine learning*. (Ph.D. thesis). Hamilton, New Zealand.

Harris, T. (2013). Quantitative credit risk assessment using support vector machines: Broad versus narrow default definitions. *Expert Systems with Applications, 40*(11), 4404–4413.

Harris, T. (2015). Credit scoring using the clustered support vector machine. *Expert Systems with Applications, 42*(2), 741–750.

Horta, I. M., & Camanho, A. S. (2013). Company failure prediction in the construction industry. *Expert Systems with Applications, 40*(16), 6253–6257.

Hsieh, N.-C., & Hung, L.-P. (2010). A data driven ensemble classifier for credit scoring analysis. *Expert Systems with Applications, 37*(1), 534–545.

Huang, F. (2008). A particle swarm optimized fuzzy neural network for credit risk evaluation. In *Proc. of 2008 second international conference on genetic and evolutionary computing*, pp. 153–157.

Kennedy, J., Eberhart, R. C., & Shi, Y. (2001). *Swarm intelligence*. Morgan Kaufmann Publishers.

Khandani, A. E., Kim, A. J., & Lo, A. W. (2010). Consumer credit-risk models via machine-learning algorithms. *Journal of Banking & Finance, 34*(11), 2767–2787.

Kim, K., & Ahn, H. (2012). A corporate credit rating model using multi-class support vector machines with an ordinal pairwise partitioning approach. *Computers & Operations Research, 39*(8), 1800–1811.

Kong, W., Cheng, W., Ding, J., & Chai, T. (2010). A reliable and efficient hybrid PSO for parameters optimization of LS-SVM in production rate prediction. In *2010 international symposium on computational intelligence and design*, pp. 140–143.

Kruppa, J., Schwarz, A., Arminger, G., & Ziegler, A. (2013). Consumer credit risk: Individual probability estimates using machine learning. *Expert Systems with Applications, 40*(13), 5125–5131.

Lai, K. K., Yu, L., Zhou, L., & Wang, S. (2006). Credit risk evaluation with least square support vector machine. In G.-Y. Wang, J. F. Peters, A. Skowron, & Y. Yao (Eds.), *Rough sets and knowledge technology*. *Lecture Notes in Computer Science* (Vol. 4062, pp. 490–495). Berlin Heidelberg: Springer.

le Cessie, S., & van Houwelingen, J. C. (1992). Ridge estimators in logistic regression. *Applied Statistics, 41*(1), 191–201.

Lessmann, S. (2004). Solving imbalanced classification problems with support vector machines. In *Proceedings of the international conference on artificial intelligence, IC-AI '04*, Las Vegas, Nevada, USA, pp. 214–220.

Li, B., Song, S., & Li, K. (2012). Improved conjugate gradient implementation for least squares support vector machines. *Pattern Recognition Letters, 33*(2), 121–125.

Lin, S.-W., Ying, K.-C., Chen, S.-C., & Lee, Z.-J. (2008). Particle swarm optimization for parameter determination and feature selection of support vector machines. *Expert Systems with Applications, 35*(4), 1817–1824.

Li-xia, L., Yi-qi, Z., & Liu, X. (2011). Tax forecasting theory and model based on SVM optimized by PSO. *Expert Systems with Applications, 38*, 116–120.

Margineantu, D. D., & Dietterich, Th. G. (2000). Bootstrap methods for the cost-sensitive evaluation of classifiers. *Proceedings of the Seventeenth International Conference on Machine Learning*, 583–590.

Niklis, D., Doumpos, M., & Zopounidis, C. (2014). Combining market and accounting-based models for credit scoring using a classification scheme based on support vector machines. *Applied Mathematics and Computation, 234*, 69–81.

Ohlson, J. (1980). Financial ratios and the probabilistic prediction of bankruptcy. *Journal of Accounting Research, 18*(1), 109–131.

Qi, Zh., Tian, Y., Shi, Y., & Yu, X. (2013). Cost-Sensitive support vector machine for semi-supervised learning. *Procedia Computer Science, 18*, 1684–1689.

Sands, E. G., Springate, G. L. V., & Var, T. (1983). Predicting business failures. *CGA Magazine*, 24–27.

Shumway, T. (2001). Forecasting bankruptcy more accurately: A simple hazard model. *Journal of Business, 74*(1), 101–124.

Springate G. L. V. (1978). Predicting the possibility of failure in a Canadian firm, Unpublished M.B.A. Research Project, Simon Fraser University.

Suykens, J. A. K., & Vandewalle, J. (1999). Least squares support vector machine classifiers. *Neural Processing Letters, 9*(3), 293–300.

Taffler, R. (1982). Forecasting company failure in the UK using discriminant analysis and financial ratio data. *Journal of the Royal Statistical Society. Series A (General), 145*, 342–358.

Thangaraj, R., Pant, M., Abraham, A., & Bouvry, P. (2011). Particle swarm optimization: Hybridization perspectives and experimental illustrations. *Applied Mathematics and Computation, 217*(12), 5208–5226.

Tsai, C., & Wu, J. (2008). Using neural network ensembles for bankruptcy prediction and credit scoring. *Expert Systems with Applications, 34*(4), 2639–2649.

Vapnik, V. N. (2000). *The nature of statistical learning theory* (2nd ed.). New York: Springer.

Vellido, A., Lisboa, P. J. G., & Vaughan, B. (1999). Neural networks in business: A survey of applications (1992–1998). *Expert Systems with Applications, 17*, 51–70.

Wang, G., & Ma, J. (2012). A hybrid ensemble approach for enterprise credit risk assessment based on Support Vector Machine. *Expert Systems with Applications, 39*(5), 5325–5331.

Wong, B. K., Lai, V. S., & Lam, J. (2000). A bibliography of neural network business applications research: 1994–1998. *Computers & Operations Research, 27*, 1045–1076.

Wong, B., & Selvi, Y. (1998). Neural network applications in finance: A review and analysis of literature (1990–1996). *Information & Management, 34*, 129–139.

Wu, C., Tzeng, G., Goo, Y., & Fang, W. (2007). A real-valued genetic algorithm to optimize the parameters of support vector machine for predicting bankruptcy. *Expert Systems with Applications, 32*(2), 397–408.

Yu, L., Wang, S., & Lai, K. K. (2008). Credit risk assessment with a multistage neural network ensemble learning approach. *Expert Systems with Applications, 34*(2), 1434–1444.

Yu, L., Yao, X., Wang, S., & Lai, K. K. (2011). Credit risk evaluation using a weighted least squares SVM classifier with design of experiment for parameter selection. *Expert Systems with Applications, 38*, 15392–15399.

Yu, L., Yue, W., Wang, S., & Lai, K. K. (2010). Support vector machine based multiagent ensemble learning for credit risk evaluation. *Expert Systems with Applications, 37*(2), 1351–1360.

Yun, L., Cao, Q.-Y., & Zhang, H. (2011). Application of the PSO–SVM model for credit scoring. *Seventh International Conference on Computational Intelligence and Security*, 47–51.

Zhang, Z., Gao, G., & Shi, Y. (2014). Credit risk evaluation using multi-criteria optimization classifier with kernel, fuzzification and penalty factors. *European Journal of Operational Research, 237*, 335–348.

Zhang, D., Zhou, X., Leung, S. C. H., & Zheng, J. (2010). Vertical bagging decision trees model for credit scoring. *Expert Systems with Applications, 37*(12), 7838–7843.

Zhou, J., Bai, T., Tian, J., & Zhang, A. (2008). The study of SVM optimized by culture particle swarm optimization on predicting financial distress. *IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, 1689–1693.

Zhou, L., Lai, K. K., & Yu, L. (2010). Least squares support vector machines ensemble models for credit scoring. *Expert Systems with Applications, 37*(1), 127–133.

Zmijewski, M. (1984). Methodological issues related to the estimation of financial distress prediction models. *Journal of Accounting Research, 22*, 59–82.