A Modified Support Vector Machine model for Credit Scoring

Xiaoyong Liu [a,b,c]
[a] *Department of Computer Science, Guangdong Polytechnic Normal University,*
Guangzhou, Guangdong, 510665, China
[b] *National Science Library, Chinese Academy of Sciences, Beijing, 100190, China*
[c] *Graduate University of Chinese Academy of Sciences, Beijing 100049,China*
liugucas@gmail.com

Hui Fu
Department of Computer Science, Guangdong Polytechnic Normal University,
Guangzhou, Guangdong, 510665, China
lindafh819@126.com

Weiwei Lin
School of Computer Science and Engineering, South China University of Technology,
Guangzhou, Guangdong, 510640, China

Received: 11-03-2010
Accepted: 05-10-2010

Abstract

This paper presents a novel quantitative credit scoring model based on support vector machine (SVM) with adaptive genetic algorithm, gr-GA-SVM. In this study, two real world credit datasets in the University of California Irvine Machine Learning Repository are selected for the numerical experiments. SVM, GA-SVM and gr-GA-SVM, are employed to predict the accuracy of credit scoring in two datasets. Numerical results indicate that gr-GA-SVM is more accurate and efficient than SVM and GA-SVM.

Keywords: credit scoring, Support vector machine, Genetic algorithm, Radial Basis Kernel

1. Introduction

Recently, credit scoring has become a very important task because credit cards are now widely used by customers all over the world. Credit scoring is a method of evaluating the credit risk of loan applications. Using historical data and statistical techniques, credit scoring tries to isolate the effects of various applicant characteristics on delinquencies and defaults. The method produces a “score” that a bank can use to rank its loan applicants or borrowers in terms of risk. To build a scoring model, or "scorecard", developers analyze historical data on the performance of previously made loans to determine which borrower characteristics are useful in predicting whether the loan performed well. A well-designed model should give a higher percentage of high scores to borrowers whose loans will perform well and a higher percentage of low scores to borrowers whose loans won’t perform well. Even a good scoring system won’t predict with certainty any individual loan’s performance, but it should give a fairly accurate prediction of the likelihood that a loan applicant with certain characteristics will default. To build a good scoring model, developers need sufficient historical data, which reflect loan performance in periods of both good and bad economic conditions.[1][2]

In the past, banks used credit reports, personal histories and judgment to make credit decisions. But over the past 25 years, credit scoring has become widely used in issuing credit cards and in other types of

Published by Atlantis Press

consumer lending, such as auto loans and home equity loans. Although some models have been developed to estimate the default probabilities of large firms, they have been based on the performance of corporate bonds of publicly traded companies. It is not at all clear that these models would accurately predict the default performance of bank loans to these or other companies. To develop a more accurate loan scoring model for larger businesses, a necessary first step would be the collection of a vast array of data on many different types of businesses along with the performance of loans made to these businesses; the data would have to include a large number of bad, as well as good, loans. Since the typical default rate on business loans is in the range of 1 percent to 3 percent annually, banks would have to pool their data. Such data-collection efforts are currently under way. But the fact that loans to large businesses vary in so many dimensions will make the development of a credit scoring model for these types of loans very difficult.

In credit scoring models, there are two methods, statistical methods and machine learning methods. Several statistical methods are used to develop credit scoring systems, including linear probability models, logistic models and probabilistic models. They are standard statistical techniques for estimating the probability of default based on historical data on loan performance and characteristics of the borrower. These techniques differ in that the linear probability model assumes there is a linear relationship between the probability of default and the factors; the logistic model assumes that the probability of default is logistically distributed; and the probabilistic model assumes that the probability of default has a cumulative normal distribution. [1]

Several financial decision-making methods based on machine learning (examples of machine learning techniques used to solve the above financial decision-making problems are Atiya[3]; Huang, Chen, Hsu, Chen, & Wu[4]; Lee, Chiu, Chou, & Lu[5]) use the multi-layer perceptron (MLP) as classifier. Other tested classifiers are the Decision Tree and the Support Vector Machine [6] [7][8]. We want to stress that these studies show that the machine learning based systems are better than the traditional (statistical) methods for bankruptcy prediction and credit scoring problems (Huang et al.[4]; Ong, Huang, & Tzeng[9]; Vellido, Lisboa, & Vaughan[10]; Wong & Selvi[11]). In Tsai and Wu[12] the authors compare a single MLP classifier with multiple classifiers and diversified multiple classifiers on three datasets. However, they conclude that there is no an exact winner.

The structure of this paper is as following, section 2 introduces related credit scoring models and the novel models-gr-GA-SVM, which is presented in this study. Section 3 give results of different models in two real credit datasets from University of California Irvine Machine Learning Repository. Finally, conclusions are presented in Section 4.

2. Methods and Materials

2.1. *Methods*

2.1.1 *SVM*

Support vector machines[13][14][15](SVM) are a set of related supervised learning methods used for classification and regression. A support vector machine constructs a hyper plane or set of hyper planes in a high-dimensional space, which can be used for classification, regression or other tasks. Intuitively, a good separation is achieved by the hyper plane that has the largest distance to the nearest training data points of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier.

In order to extend the SVM methodology to handle data that is not fully linearly separable, we relax the constraints slightly to allow for misclassified points, the formulation is following (1.1) and (1.2). This is done by introducing a positive slack variable ξ_i, $i = 1,2,\cdots L$:

$$x_i \cdot w + b \geq +1 - \xi_i \quad (y_i = +1) \qquad (1.1)$$

$$x_i \cdot w + b \leq -1 + \xi_i \quad (y_i = -1) \qquad (1.2)$$

Which can be combined into

$$y_i(x_i \cdot w + b) - 1 + \xi_i \geq 0$$

where $\xi_i \geq 0$

In this soft margin SVM, data points on the incorrect side of the margin boundary have a penalty that increases with the distance from it. As we are trying to reduce the number of misclassifications, a sensible way to adapt our objective function from previously, is to find:

$$\min \frac{1}{2}\left\|w^2\right\| + C\sum_{i=1}^{L}\xi_i$$

$$s.t.\quad y_i(x_i \cdot w + b) - 1 + \xi_i \geq 0$$

Where the parameter C controls the trade-off between the slack variable penalty and the size of the margin. Reformulating as a Lagrange, which as before we need to minimize with respect to w, b and ξ_i and maximize with respect to α (where $\alpha_i \geq 0$, $u_i \geq 0$)

$$L_P \equiv \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{L}\xi_i$$

$$-\sum_{i=1}^{L}\alpha_i\left[y_i(x_i \cdot w + b) - 1 + \xi_i\right] - \sum_{i=1}^{L}\mu_i\xi_i$$

Differentiating with respect to w, b and ξ_i and setting the derivatives to zero:

$$\frac{\partial L_P}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^{L}\alpha_i y_i x_i$$

$$\frac{\partial L_P}{\partial b} = 0 \Rightarrow \sum_{i=1}^{L}\alpha_i y_i = 0$$

$$\frac{\partial L_P}{\partial \xi_i} = 0 \Rightarrow C = \alpha_i + \mu_i$$

So we need to find:

$$\max_{\alpha}\left[\sum_{i=1}^{L}\alpha_i - \frac{1}{2}\alpha^T H\alpha\right]$$

$$s.t.\quad 0 \leq \alpha_i \leq C \text{ and } \sum_{i=1}^{L}\alpha_i y_i = 0$$

When applying SVM to nonlinearly dataset, we need define a feature mapping function $x \mapsto \phi(x)$. The feature mapping function is called kernel function. In the feature space, optimal hyper plane (fig.1) can get.

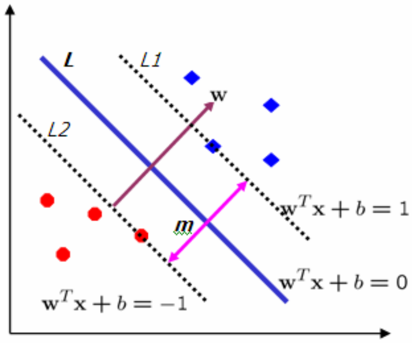

L
L1
w
L2
m
$\mathbf{w}^T\mathbf{x}+b=1$
$\mathbf{w}^T\mathbf{x}+b=0$
$\mathbf{w}^T\mathbf{x}+b=-1$

Fig.1. Optimal Hyper plane

There are three common kernel functions:

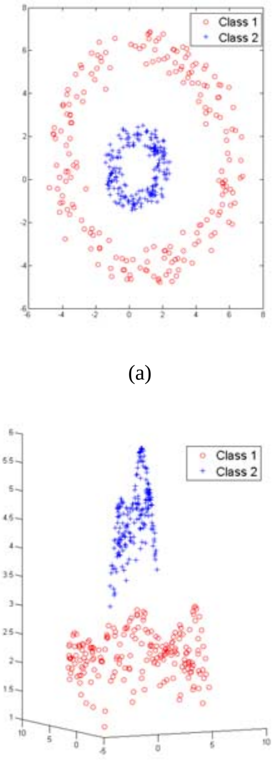
Polynomial Kernel

$$k(x_i, x_j) = (x_i \cdot x_j + a)^b$$

Radial Basis Kernel

$$k(x_i, x_j) = e^{-\left(\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)}$$

Then a dataset that is nonlinearly separable in the two dimensional data space X (as in the fig.2(a)) is separable in the nonlinear feature space (as in the fig. 2(b)) defined implicitly by this nonlinear kernel function – known as a Radial Basis Kernel.



(a)

(b)

Fig.2. Radial Basis Kernel[11]

Sigmoidal Kernel

$$k(x_i, x_j) = \tanh(ax_i \cdot x_j - b)$$

Where a and b are parameters defining the kernel's behavior.

In order to use SVM to solve a classification or regression problem on dataset that is nonlinearly separable, we need to first choose a kernel and relevant parameters which you expect might map the nonlinearly separable data into a feature space where it is linearly separable. This is more of an art than an exact science and can be achieved empirically – e.g. by trial and error. Sensible kernels to start with are Polynomial , Radial Basis and Sigmoid kernels.

For classification, we need to:

Create H, where $H_{ij} = y_i y_j \phi(x_i)\phi(x_j)$

Choose how significantly misclassifications should be treated, by selecting a suitable value for the parameter C.

Find α so that,

$$\max_{\alpha}\left[\sum_{i=1}^{L}\alpha_i - \frac{1}{2}\alpha^T H\alpha\right]$$

$$s.t.\ \ 0 \le \alpha_i \le C \ \text{ and } \sum_{i=1}^{L}\alpha_i y_i = 0$$

Calculate $w = \sum_{i=1}^{L}\alpha_i y_i \phi(x_i)$

Determine the set of support vectors V, by finding the indices such that $0 \le \alpha_i \le C$.

Calculate $b = \frac{1}{N_v}\sum_{v\in V}\left(y_v - \sum_{i\in V}\alpha_i y_i \phi(x_i)\phi(x_v)\right)$.

Each new point x' is classified by evaluating $y' = \operatorname{sgn}(w\phi(x') + b)$

2.1.2 *Genetic algorithm* [16][17][18]

Genetic algorithms are implemented in a computer simulation in which a population of abstract representations (called chromosomes or the genotype of the genome) of candidate solutions (called individuals, creatures, or phenotypes) to an optimization problem evolves toward better solutions. Traditionally, solutions are represented in binary as strings of 0s and 1s, but other encodings are also possible. The evolution usually starts from a population of randomly generated individuals and happens in generations. In each generation, the fitness of every individual in the population is evaluated, multiple individuals are stochastically selected from the current population (based on their fitness), and modified (recombined and possibly randomly mutated) to form a new population. The new population is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population. If the algorithm has terminated due to a maximum number of generations, a satisfactory solution may or may not have been reached.

Genetic algorithms find application in bioinformatics, phylogenetics, computational science, engineering, economics, chemistry, manufacturing, mathematics, physics and other fields.

A typical genetic algorithm requires:

A genetic representation of the solution domain;

A fitness function is used to evaluate the solution domain.

A standard representation of the solution is as an array of bits. Arrays of other types and structures can be used in essentially the same way. The main property that makes these genetic representations convenient is that their parts are easily aligned due to their fixed size, which facilitates simple crossover operations. Variable length representations may also be used, but crossover implementation is more complex in this case. Tree-like representations are explored in genetic programming and graph-form representations are explored in evolutionary programming.

The fitness function is defined over the genetic representation and measures the quality of the represented solution. The fitness function is always problem dependent. For instance, in the knapsack problem one wants to maximize the total value of objects that can be put in a knapsack of some fixed capacity. A representation of a solution might be an array of bits, where each bit represents a different object, and the value of the bit (0 or 1) represents whether or not the object is in the knapsack. Not every such representation is valid, as the size of objects may exceed the capacity of the knapsack. The fitness of the solution is the sum of values of all objects in the knapsack if the representation is valid or 0 otherwise. In some problems, it is hard or even impossible to define the fitness expression; in these cases, interactive genetic algorithms are used.

Once we have the genetic representation and the fitness function defined, GA proceeds to initialize a population of solutions randomly, then improve it through repetitive application of mutation, crossover, inversion and selection operators.

2.1.3 *GA-SVM*

There are two methods that are used to combine Genetic algorithm and SVM.

1) Dealing with dataset[19][20]

The initial training dataset are optimized with GA in order to find a sample subset including the important samples that can preserve or improve the discrimination ability of SVM. Training on the subset is equal to that on the initial sample sets. The training time is greatly shortened.

There are two available sets of training data: class1 $\{z_1, z_2, \cdots z_n\}$ and class2 $\{z'_1, z'_2, \cdots z'_n\}$. z_i or z'_j is an example, or one feature vector for SVM. An intuition idea is to find out the important examples that affect the classification results greatly. If these feature vectors are removed, the separating boundary changes the most. The key important question is how to find out these important training data from all the examples with GA.

2) Defining parameters[21] [22]

The value of parameters in Support Vector Machines is important to algorithm's performance. Ángel Kuri-Morales and Iván Mejía-Guevara presented a methodology to train SVM where the regularization parameter (C) was determined automatically via an efficient Genetic Algorithm in order to solve multiple category classification problems.

In previous works, the support vectors have been determined from the application of Lagrange Multipliers, but are not applicable to search for "C". In fact, GA are used to solve the constrained QP. One advantage of using GA for this kind of problems is that restrictions are not imposed in the form of the objective function: a neither the objective function nor the constraints of the problem must be derivable in order to solve problems. In some cases, each individual represents a LM ($\alpha_i, i = 1,2,3,\cdots N$), where N is the number of points in the training set for the dual SVM problem.

This algorithm combing GA and SVM has been applied many fields, such as fault detection[23], protein sequences classification[24][25], network intrusion detection[26], daily flow forecasting[27], Short-term Load Forecasting[28], Evaluation of competitiveness of power plants[29], stock index forecasting[30].

2.1.4 *gr-GA-SVM*

In parameters of genetic algorithm, the rate of crossover and the rate of mutation affect the speed of convergence. The rates of crossover affect new population. Higher the rate of crossover is, faster new individuation product. High rate of crossover will destroy the schemas of good individuations, while low rate of crossover will postpone production of new individuations. The rate of mutation is a key factor for algorithms to step out from local optimal solution. If the rate of mutation is larger, GA will be a random search algorithm. If the rate of mutation is lesser, it is difficult to produce new schemas for individuations. In this work, an adaptive technique is applied in a new models, gr-GA-SVM. In new models, the rate of crossover and the rate of mutation are changeable as following:

$$P_c = \begin{cases} \dfrac{c_1\left(f_{\max} - f'\right)}{f_{\max} - f_{avg}} & f' \geq f_{avg} \\ c_2 & f' < f_{avg} \end{cases}$$

$$P_m = \begin{cases} \dfrac{c_3\left(f_{\max} - f'\right)}{f_{\max} - f_{avg}} & f' \geq f_{avg} \\ c_4 & f' < f_{avg} \end{cases}$$

Where,

c_1, c_2, c_3 *and* c_4 are constants,

$f_{\max}$ is the maximizing value of the fitness function during the processing of the iterations,

f_{avg} is the average of the fitness function during the processing of the iterations,

f' is the value of the fitness function in one iteration.

gr-GA-SVM algorithm can be showed as following in detail,

Step 1:

Parameter settings.

Setting the population size- *PopSize*, crossover probability $P_{c,}$ mutation probability $P_{m,}$.

c_1, c_2, c_3 *and* c_4 are also set.

Step 2:

Classify dataset by SVM, calculate the Accuracy Rate that will be the fitness function in Genetic algorithm.

Step 3:

Set the fitness function.

Fitness function is the optimization goal, where is denoted by Accuracy Rate.

Step 4 :

Population initialization.

Each individuals within the population *PopSize* is initialized.

Step 6:

Select operation.

Calculate all the individuals' fitness $F_{i\,,}$ in population PopSize. $F_{\,i}$ is predictive rate for the unknown data.

Step 7:

Crossover operation.

The group of M chromosomes make pairs randomly and form PopSize/2 pairs of chromosomes for crossover operation. This paper uses double-point crossover.

Step 8:

Mutation operation.

Each gene of chromosomes can mutate randomly according to the probability P_m.

Step 9:

Recalculate fitness values of *PopSize* individuals and go back to step 2.

The flowchart of gr-GA-SVM algorithm is showed in fig.3 in detail.

2.2. *Materials*

In this study, numerical experiments use two datasets, German credit dataset and Australian credit dataset from UCI Machine Learning Repository[31].

German credit card dataset has 1000 instances. There are twenty-four numerical attributes in this dataset. Number of customers as “good” is 700, and number of customers as “bad” is 300. Australian credit card dataset has 690 Instances. Number of attributes of each instance is 14. This dataset is a good mix of attributes -- continuous, nominal with small numbers of values, and nominal with larger numbers of values. There are six numerical and eight categorical attributes in this dataset. There are 307 customers as “good” and 383 customers as “bad”. In two datasets, “0” denotes a customer as “bad”, and “1” denotes a customer as “good”. Table1 shows the detail of two datasets. German credit dataset and Australian credit dataset choose 800 and 500 instances and as train datasets respectively. Rests of two datasets are as test datasets. Continuous attributes are normalized firstly, and then used to train and test.

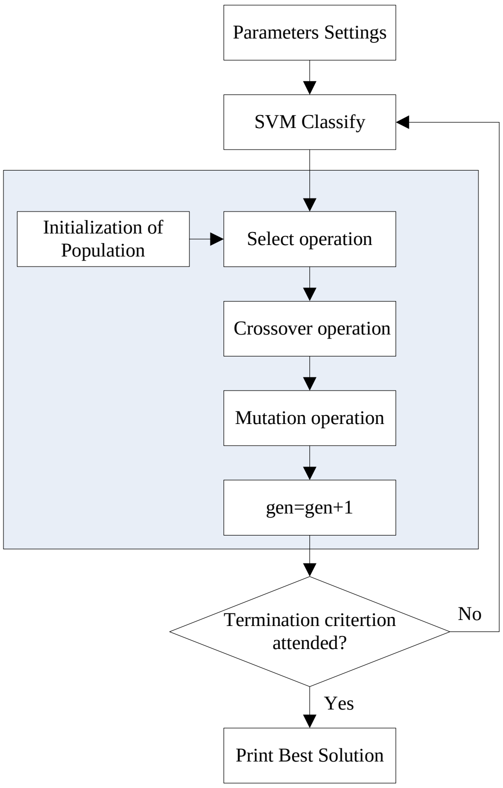


Fig.3. The flowchart of gr-GA-SVM

Table1. Credit Dataset

Dataset	Number of Instances	Number of Attributes	Good Credit	Bad Credit
German credit dataset	1000	24	700	300
Australian credit dataset	690	14	307	383

3. Results and discussion

For the compare of performance between traditional SVM, GA-SVM and gr-GA-SVM, these models are run several times. The program of the new algorithm is written by Matlab 7.0 (R14) and run on a computer with 2.0 GHz CPU, 1GB DDR RAM. Table 2 lists the appropriate values of these parameters in three algorithms. In order to use SVM to solve credit scoring problems on two datasets that is nonlinearly separable, we first choose a radial basis kernel because we find that SVM based on radial basis kernel is faster than SVM based on polynomial kernel.

Table 3 shows the accuracy comparison of SVM, GA-SVM and gr-GA-SVM. For the German dataset, the accuracy of gr-GA-SVM is 75.50%, GA-SVM is 72.50% and SVM is 70.50%. For the Australian dataset, the accuracy of gr-GA-SVM is 86.84%, GA-SVM is 85.26% and SVM is 81.58%. The results of empirical analysis showed that the predictive ability of all the models is acceptable. However, the gr-GA-SVM results outperformed than the other methods. Therefore, gr-GA-SVM is a more effective model to predict credit scoring in two datasets.

Table2. Parameters setting of Algorithms

Parameter	PopSize	Iteration	p_c	p_m	c_1	c_2	c_3	c_4
Value	20	50	0.5	0.005	0.8	0.5	0.008	0.005

Table3. Comparison of models

Dataset	Algorithm	Forecasting Accuracy total	Forecasting Accuracy in Good Credit	Forecasting Accuracy in Bad Credit
German credit dataset	SVM	0.7050	0.9058	0.2581
	GA-SVM	0.7250	0.9855	0.1452
	gr-GA-SVM	0.7550	0.9275	0.3710
Australian credit dataset	SVM	0.8158	0.8861	0.7658
	GA-SVM	0.8526	0.8481	0.8559
	gr-GA-SVM	0.8684	0.9114	0.8378

4. Conclusion

In the last few decades, several credit scoring models have been developed for the credit granting decision. The objective of quantitative credit scoring models is to assign credit applicants to one of two groups: a "good credit" group that is likely to repay the financial obligation, or a "bad credit" group that should be denied credit because of a high likelihood of defaulting on the financial obligation. In this paper, we design a new credit scoring model, gr-GA-SVM. The experimental research results show the novel algorithm is better than SVM and GA-SVM.

Acknowledgements

The authors would like to thank anonymous reviewers for their constructive and enlightening comments, which improved the manuscript. This work has been supported by grants from Program for Excellent and Creative Young Talents in Universities of Guangdong Province (LYM08074 and LYM10097). The authors is partly supported by the grants from the Fundamental Research Funds for the Central Universities, SCUT (20092M0103), Guangdong Natural Science Foundation (x2jsB6100260), and Comprehensive Strategic Cooperation of Guangdong Province and Chinese Academy(2009B091300069).

References

1. McAllister, Patrick H., and John J. Mingo. Commercial Loan Risk Management, Credit-Scoring, and Pricing: The Need for a New Shared Database. *Journal of Commercial Lending*, 1994,76(9), pp. 6-22.
2 . MESTER L. What’s the point of credit scoring?. *Business review*, 1997(3), pp.3–16.
3. Atiya, A. F.. Bankruptcy prediction for credit risk using neural networks: a survey and new results. *IEEE Transactions on Neural Networks*, 2001,12(4), pp.929－935.
4. Huang, Z., Chen, H., Hsu, C.-J., Chen, W.-H., & Wu, S.. Credit rating analysis with support vector machines and neural networks: A market comparative study. *Decision Support Systems*, 2004(37), pp.543－558.
5. Lee, T.-S., Chiu, C.-C., Chou, Y.-C., & Lu, C.-J.. Mining the customer credit using classification and regression

tree and multivariate adaptive regression splines. *Computational Statistics and Data Analysis*, 2006(50), pp.1113﹣1130.

6 .R. Stecking and K. B. Schebesch. Combining Support Vector Machines for Credit Scoring. *Proceedings of Operations Research 2006*, pp.135-140.

7 . Martens D, Baesens B, Van Gestel T, et al. Comprehensible credit scoring models using rule extraction from support vector machines. *European Journal of Operational Research*, 2007, 183(3), pp. 1466-1476.

8. Luo S, Cheng B, Hsieh C. Prediction model building with clustering-launched classification and support vector machines in credit scoring. *Expert Systems with Applications*, 2009, 36(4), pp. 7562-7566.

9. Ong, C.-S., Huang, J.-J., & Tzeng, G.-H.. Building credit scoring models using genetic programming. *Expert Systems with Applications*, 2005(29), pp.41﹣47.

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

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

12. Tsai C, Wu J. Using neural network ensembles for bankruptcy prediction and credit scoring. *Expert Systems with Applications*, 2008, 34(4),pp. 2639-49.

13. Tristan Fletcher. Support Vector Machines Explained. http://www.tristanfletcher.co.uk/SVM%20Explained.pdf. [2010-2-6]

14. Anjum Reyaz-Ahmed, Yan-Qing Zhang and Robert W. Harrison. Granular Decision Tree and Evolutionary Neural SVM for Protein Secondary Structure Prediction. *International Journal of Computational Intelligence Systems*, 2009, 2(4), pp. 343 - 352.

15 . Wensen An, Cecilio Angulo and Yanguang Sun.Support Vector Regression with Interval-Input Interval-Output. *International Journal of Computational Intelligence Systems*, 2008, 1(4), pp. 299 - 303.

16. Genetic algorithm.
http://en.wikipedia.org/wiki/Genetic_algorithm.[2010-9-6]

17 . Alireza Fasih, Jean Chamberlain Chedjou and Kyandoghere Kyamakya. Cellular Neural Networks-Based Genetic Algorithm for Optimizing the Behavior of an Unstructured Robot. *International Journal of Computational Intelligence Systems*, 2009, 2(2), pp. 124–131.

18. R.Sampath Kumar and N. Alagumurthi. Integrated total cost and Tolerance Optimization with Genetic Algorithm. *International Journal of Computational Intelligence Systems*, 2010, 3(3), pp. 325 – 333.

19. Zhang, X.R., and Liu, F. A patten classification method based on GA and SVM, *Proceedings of 6th International Conference on Signal Processing Proceedings(2002)* , pp. 110-113.

20. Liu, J.J., Cutler, G., Li, W.X., Pan, Z., Peng, S.H., Hoey, T., Chen, L.B., and Ling, X.F.B. Multiclass cancer classification and biomarker discovery using GA-based algorithms, *Bioinformatics*, 2005, 21(11), pp. 2691-2697.

21. Liu, S., Jia, C.Y. and Ma, H. A new weighted support vector machine with GA-based parameter selection, *Proceedings of 4th International Conference on Machine Learning and Cybernetics(2005)*, pp. 4351-4355

22 . Kuri-Morales A, MEJIA-Guevara I. Evolutionary training of SVM for multiple category classification problems with self-adaptive parameters. *Advances in Artificial Intelligence, LNCS 4140*, pp. 329-338.

23. Nguyen N.T. and Lee H.H. An Application of Support Vector Machines for Induction Motor Fault Diagnosis with Using Genetic Algorithm, *Advanced Intelligent Computing Theories and Applications*, *LNCS 5227*, pp. 190-200.

24. Zhao X.M., Huang D.S., Cheung Y.M., Wang H.Q. and Huang X. A novel hybrid GA/SVM system for protein sequences classification, *LNCS 3177*, pp. 11-16.

25. Li S.T., Wu X.X. and Hu X.Y. Gene selection using genetic algorithm and support vectors machines, *Soft Computing*, 2008, 12 (7), pp. 693-698

26. Kim D.S. and Park G.S. Modeling network intrusion detection system using feature selection and parameters optimization, *IEICE Transactions on Information and Systems*, 2008 (4), pp. 1050-1057

27. Jianzhong W., Ling L. and Juan C. Combination of genetic algorithm and support vector machine for daily flow forecasting, *Proceedings of Fourth International Conference on Natural Computation (2008)*, pp. 31-35

28. Ma L.H., Zhou S.G. and Lin M. Support Vector Machine Optimized with Genetic Algorithm for Short-term Load Forecasting, *Proceedings of International Symposium on Knowledge Acquisition and Modeling (2008)*, pp. 654-657.

29. Wei, S., and Jie, Z. Evaluation of competitiveness of power plants based on optimized SVM using GA and AIS. *Proceedings of International Conference on Risk Management & Engineering Management(2008)*, pp. 648-652.

30. Huang S.C. and Wu T.K. Integrating GA-based time-scale feature extractions with SVMs for stock index forecasting, *Expert Systems with Applications*, 2008, 35 (4), pp. 2080-2088

31. UCI Machine Learning Repository.
http://archive.ics.uci.edu/ml/ .