วารสารวิชาการทางเทคโนโลยีคอมพิวเตอร์และระบบสารสนเทศ ปีที่ 1 ฉบับที่ 1 เมษายน - กันยายน 2554

Ensemble Data Classification based on Decision Tree, Artificial Neuron Network and Support Vector Machine Weighting Classifier by Adaboost

Dech Thammasiri [1] Phayung Meesad [2]
[1] Faculty of Information Technology King Mongkut's University of Technology North Bangkok
[2] Faculty of Technical Education King Mongkut's University of Technology North Bangkok
Bangsue, Bangkok, Thailand 10800
dechit@msn.com pym@kmutnb.ac.th

Abstract— **In this research we propose an ensemble classification technique based on decision tree, artificial neural network, and support vector machine models weighting classifier by adaboost in order to increase classification accuracy. Testing Diabites Data from UCI, classification accuracy tests on Diabites Data found that the proposed ensemble classification models weighting classifier by Adaboost yields better performance than that of a single model with the same type of classifier. The result as follows, Diabites Data achieved the best performance with 75.21%.**

Keywords— **Decision Tree, Artificial Neuron Network, Support Vector Machine, Ensemble, Adaboost**

I. INTRODUCTION

A. Background

Many researchers are now focused on problems that require an accurate method of classification. Presented by various methods such as Application of Artificial Intelligence Techniques applied Decision Tree Models (DT) [1], Artificial Neural Networks (ANN)[2] and Support Vector Machine (SVM)[3] etc.

Although the above techniques will result in accurate classification, a single model that results in the collection of data used in learning including a fixed set of parameters sometimes has the problem of bias. Techniques to reduce the tendency use the ensemble method. However, the ensemble method has good performance. Wang *et al.* [4] advised that "In general, the efficiency of ensemble depends on the variety and accuracy of the agency to classify data".

Yao *et al.* [5] presented combining ensemble with decision tree to boost the results. This method provides better performance than a single model. Jian *et al.* [6] used ensemble techniques with neural networks. This showed that the method can be improved in terms of accurate results. Obviously, ensemble technique can improve the efficiency of the model.

In terms of variety of classification, researchers have proposed such as Ying *et al.* [8] using support vector machines, neural networks and decision trees in order to classify spam e-mail. Yong *et al.* [9] used KNN, Bayes, and Genetic methods to optimize the classifier. Zhou *et al.* [10] and Chen *et al.* [7] proposed classification by using support vector machines that configured variety kernel function as well as parameters. Therefore, a variety of classifiers are able to solve a variety of problems and also increase the overall performance of the model.

For appropriate weight of each classifier in the ensemble classification, Lei *et al.* [11] used AdaBoost techniques to classify tourism data in China. The results showed that AdaBoost has more accuracy than using just a single model and Bagging technique. Also, Khaleghian *et al.* [12] used Adaboost in order to increase the efficiency of Tied Factor Analysis (TFA) for face detection.

In this paper, we propose a technique to solve these problems and find a suitable method to classify data. By creating classification from a variety of techniques and choosing the appropriate parameters to optimize the classification with the weight to the appropriate classification technique, AdaBoost will create a new model for efficient data classification and more accuracy.

II. THEORY AND METHODOLOGY

A. Decision Tree

Decision tree learning [13] is a means for approximating discrete-valued target functions, in which the learned function is represented by a decision tree. Learned trees can also be re-represented as sets of if–then rules to improve human readability. Suppose, in a set of records, each record has the same structure, consisting of a number of attribute/value pairs. One of these at tributes represents the category of the record. The problem is to determine a decision tree that, on the basis of answers to question about the noncategory attributes, predicts correctly the value of the category attribute. In the decision tree, each node corresponds to a noncategorical attribute and each arc to a possible value of that attribute. A leaf of the tree specifies the expected value of the categorical attribute for the records described by the path from the root to that leaf. There are many decision tree algorithms. The results of [14] shows the C4.5 treeinduction algorithm provides good classification accuracy. C4.5 uses the gain ratio criterion, which is based on information theory and produces suboptimal trees heuristically [15]. Firstly, a C4.5 decision tree is built using the training set. Secondly, pruning of the decision tree is done by replacing a whole subtree by a leaf node. If a decision rule establishes that the expected error rate in the subtree is greater than in the single leaf, the replacement takes place. Thirdly, decision trees

can sometimes be quite difficult to understand. Thus, the rule sets that consist of simple if–then rules are derived from a decision tree: write a rule for each path in the decision tree from the root to a leaf. In that rule, the leaf-hand side is easily built from the label of the nodes and the labels of the arcs. Rules are ordered by class and sub ordered by confidence and a default rule is created for dealing with instances that are not covered by any of the generated rules. The default rule has no antecedent and its consequence is the class that contains the most training instances not covered by any rule. Each of the rule sets produced are then evaluated on the original training data and on the test data.

B. Neural Network

A neural network is a computer-intensive, algorithmic procedure for transforming inputs into desired outputs using highly inter-connected networks of relatively simple processing elements (often termed neurons, units or nodes- we will use them interchangeably thereafter). Neural networks are modeled following the neural activity in the human brain. The three essential features of a neural network are the nodes, the network architecture describing the connections between the nodes, and the training algorithm used to find values of the network parameters (weights) for performing a particular network. The nodes are connected to one another in the sense that the output from one node can be served as the input to another node. Each node transforms an input to an output using some specified function that is typically monotone, but otherwise arbitrary. This function depends on constants (parameters) whose values must be determined with a training set of inputs and outputs. Network architecture is the organization of nodes and the types of connections permitted. The nodes are arranged in a series of layers with connections between nodes in different layers, but not between nodes in the same layer. The layer receiving the inputs is the input layer. The final layer provides the target output signal is the output layer. Any layers between the input and output layers are hidden layers. A simple representation of a neural network is shown in Figure 1.

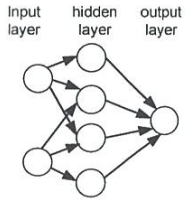

Fig. 1. A Three-Layer Backpropagation Neural Network

C. Support Vector Machine

Given a set of training samples $\mathbf{T}=\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, where $\mathbf{x}_i \in \mathbf{R}^n$ is the input pattern, $y_i \in \{-1,1\}$ is the class label for a 2-class problem, SVM for classification attempts to find a classifier $f(\mathbf{x})$, which minimizes the expected misclassification rate. A linear classifier $f(\mathbf{x})$ is a hyperplane, and can be represented as classifier $f(\mathbf{x}) = \mathrm{sgn}(\mathbf{W}^{\mathrm{T}}\mathbf{x} + b)$. Finding the optimal classifier $f(\mathbf{x})$ in SVM is equivalent to solving a convex quadratic optimization problem in (1)

$$\max_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\varepsilon_i \qquad (1)$$

subject to $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \varepsilon_i$, and $\varepsilon_i \geq 0$,
where C is called the regularization parameter, and is used to balance the classifier's complexity and classification accuracy on the training set T. This quadratic problem is generally solved through its dual formulation [16]. Simply replacing the involved vector inner-product with a non-linear kernel function converts linear SVM into a more Hexible non-linear SVM, which is the essence of the famous *kernel trick*. Please refer to [17] for more details on SVM for classification

D. Adaboost

Given a set of training samples, AdaBoost [18] maintains a weight distribution, W, over these samples. This distribution is initially set uniform, shown in Figure 2.

Algorithm : Adaboost
1. Input: a set of training samples with labels $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$ where $\mathbf{x}_i \in \mathbf{X}, y_i \in Y = \{-1, +1\}$, a ComponentLearn algorithm, the number of cycles T.
2. Initialize: the weights of training samples: $w_i^1 = \frac{1}{N}$; for all $i = 1, \ldots, N$
3. Do for $t = 1, \ldots, T$
   (1) Use the ComponentLearn algorithm to train a component classifier h_t, on the weighted training samples.
   (2) Calculate the training error of h_t : $\varepsilon_t = \sum_{i=1}^{N} W_i^t, y_i \neq h_t(x_i)$
   (3) Set weight for the component classifier h_t : $\alpha_t = \frac{1}{2} ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$
   (4) Update the weights of training samples: $w_i^{t+1} = \frac{w_i^t \, exp\{-\alpha_t y_i h_i(x_i)\}}{C_t}$, $i = 1, \ldots, N$
   where C_t is a normalization constant, and $\sum_{i=1}^{N} w_i^{t+1} = 1$
4. Output: $f(\mathbf{x}) = sign(\sum_{i=1}^{t} \alpha_t h_t(\mathbf{x}))$

Fig 2. The AdaBoost algorithm

Then, AdaBoost calls ComponentLearn algorithm repeatedly in a series of cycles shown in Figure 2. At cycle t, AdaBoost provides training samples with a distribution W, to ComponentLearn. In response, the ComponentLearn trains a classifier h_t. The distribution W, is updated after each cycle according to the prediction results on the training samples. ''Easy'' samples that are correctly classified h_t get lower weights, and ''hard'' samples that are misclassified get higher

weights. Thus, AdaBoost focuses on the samples with higher weights, which seem to be harder for ComponentLearn. This process continues for T cycles, and finally, AdaBoost linearly combines all the component classifiers into a single final hypothesis f. Greater weights are given to component classifiers with lower training errors. The important theoretical property of AdaBoost is that if the component classifiers consistently have accuracy only slightly better than half, then the training error of the final hypothesis drops to zero exponentially fast. This means that the component classifiers need to be only slightly better than random.

III. EXPERIMENTS

A. Dataset

The Pima Indian diabetes dataset [19], includes 768 complete instances described by 8 features (labeled as number of times pregnant, glucose tolerance test, diastolic blood pressure, triceps skin fold thickness, 2-hour serum insulin body mass index, diabetes pedigree function and age). The class distribution is class value 1 interpreted as "tested positive for diabetes" in 268 numbers of instances and class value 0 in 500 numbers of instances. There is no missing data present in the training dataset.

B. Accuracy calculation

The performance of the proposed approach is analyzed using the accuracy of the classifier. The accuracy of the classifier is calculated using the formula [20] (2).

$$Accuracy = (TP + TN)/(TP + TN + FP + FN) \quad (2)$$

where TP is the number of true positives (number of 'YES' patients predicted correctly); TN is the number of true negatives (number of 'NO' patients predicted correctly); FP is the number of false positives (number of 'YES' patients predicted as 'NO') and FN is the number of false negatives (number of 'NO' patients predicted as 'YES').

C. DNSA Modeling

In this research we use normalization by adjusting the data in the same format. Then show how to create a variety of classifiers by using random samples in order to make a different group data for training sets, which include N number of samples equal to the classifier. For this study, we used a total of 30 classifiers. The technique generated random data called Bootstrap [21]. Data used to train the classifier are decision tree technique, neural networks and support vector machine as in Figure 3.

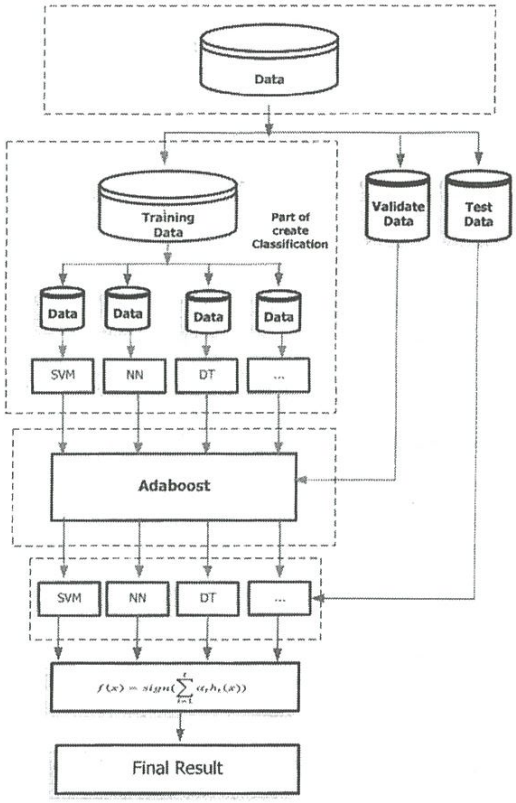

Fig. 3. Prototype of DNSA model

D. Models for comparison

In this research, the researcher effectively classified 5 models shown in Table I.

TABLE I. Comparison of estimates of missing data

Model	Detail	initial
1	Decision Tree	DT
2	Neural Network	NN
3	Support Vector Machine	SVM
4	NN+Adaboost	NNAB
5	DT+NN+SVM+Adaboost	DNSA

IV. RESULTS

Results obtained from the research of Diabetes data showed that the model DNSA has accurate results at 75.21%, followed by the model NNAB will accuracy at 75.02%, DT was at 74.66% accuracy similar to the SVM model. NN had an accuracy of 72.66%, respectively, as shown in Table 2 and Figure 4.

TABLE II. Comparison of classify data

Diabetes data	
Model	Accuracy(%)
DT	74.66
NN	72.66
SVM	72.66
NNAB	75.02
DNSA	75.21

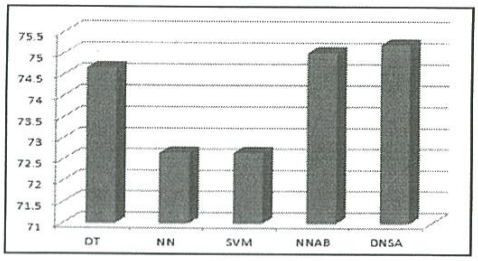

Fig. 4. Experimental results of models

From the results, it can be observed that the accuracy rate of AdaBoost ensemble method is quite better than the compared single classifier. Due to ensemble techniques, there is a reduction of the bias problem that arises from the collection of data used in learning including the fixed set of parameters. Moreover, the adaboost aid in weight to each of the classification enhances accuracy.

V. Discussion and Conclusions

This research aimed to create an effective model for data classification. The result shows that the ensemble classification technique based on decision tree, artificial neural network and support vector machine models weighting classifier by AdaBoost yields better performance than that of single model and same type classifier.

Usually, ensemble learning models outperform single learning models, whose performance is limited by the imperfection of feature extraction, learning algorithms, and the inadequacy of training data. Another reason supporting this proposition is that different single learning models have their inherent drawbacks. Aggregating them may thus lead to a better model with a high generalization capability. From the above descriptions, we can conclude that there are two essential requirements in the model. The first is that the ensemble members or learning agents must be diverse or complementary, i.e., agents must exhibit different properties. Another condition is that an optimal ensemble strategy is also required to fuse a set of diverse by AdaBoost.

In future research, we have ideas on how to select the appropriate classification by employing genetic algorithms to increase accuracy of classification.

References

[1] K. Zeitouni and N. Chelghoum, "Spatial Decision Tree-Application to Traffic Risk Analysis," in Proceedings of the ACS/IEEE International Conference on Computer Systems and Applications: IEEE Computer Society, 2001.

[2] D. Thammasiri N. Phothi V. Nuipain P. Sangsiri P. Aummarukkarat and P. Meesad, "Credit scoring with a data mining approach based on Back-Propagation Neural Network", 2rd National Conference on Information Technology, 2008, pp. 476-483.

[3] D. Thammasiri and P. Meesad, "Ensemble Data Classification based on Support Vector Machine Optimized parameters by Genetic Algorithm also Feature Selection by Correlation Coefficient", The 6th National Conference on Computing and Information Technology (NCCIT'10), 2010, pp.686–691.

[4] S. j. Wang, A. Mathew, Y. Chen, L. f. Xi, L. Ma, and J. Lee, "Empirical analysis of support vector machine ensemble classifiers," Expert Syst. Appl., vol. 36, pp. 6466-6476, 2009.

[5] Y. Yao, Z. Fu, X. Zhao and W. Cheng, "Combining Classifier Based on Decision Tree," icie, vol. 2, pp.37-40, 2009 WASE International Conference on Information Engineering, 2009.

[6] J. Wang, J. Yang, S. Li, Q. Dai, and J. Xie, "Number Image Recognition Based on Neural Network Ensemble," in *Proceedings of the Third International Conference on Natural Computation - Volume 01*: IEEE Computer Society, 2007.

[7] S. Chen, W. Wang, and H. V. Zuylen, "Construct support vector machine ensemble to detect traffic incident," *Expert Systems with Applications,* vol. 36, pp. 10976-10986, 2009.

[8] K. C. Ying, S.-W. Lin, Z. J. Lee, and Y. T. Lin, "An ensemble approach applied to classify spam e-mails," *Expert Syst. Appl.,* vol. 37, pp. 2197-2201, 2010.

[9] X. Pei-Yong, D. X. Qian, and J. B. Ning, "A GA-based feature selection and ensemble learning for high-dimensional datasets," in *Machine Learning and Cybernetics, 2009 International Conference on*, 2009, pp. 7-12.

[10] L. Zhou, K. K. Lai, and L. Yu, "Least squares support vector machines ensemble models for credit scoring," *Expert Systems with Applications,* vol. 37, pp. 127-133, 2010.

[11] W. Yu and L. Cheng De, "Learning by Bagging and Adaboost based on Support Vector Machine," in *Industrial Informatics, 2007 5th IEEE International Conference on*, 2007, pp. 663-668.

[12] S. Khaleghian, H. R. Rabiee, and M. H. Rohban, "Face recognition across large pose variations via Boosted Tied Factor Analysis," in *Applications of Computer Vision (WACV), 2011 IEEE Workshop on*, 2011, pp. 190-195.

[13] M. T. Mitchell, Machine Learning. New York: McGraw-Hill, 1997.

[14] T. S. Lim,W. Y. Loh, and Y. S. Shih, "A comparison of prediction accuracy, complexity, and training time of thirty_tree old and new classification algorithm," Mach. Learn., vol. 40, no. 3, pp. 203–228, Sep. 2000.

[15] J. R. Quinlan, C4.5: Programs forMachine Learning. San Francisco,CA: Morgan Kaufmann, 1993.

[16] C.J. Burges, A tutorial on support vector machines for pattern recognition, Data Mining and Knowledge Discovery 2 (2) (1998) 121–167.

[17] B. SchP olkopf, A. Smola, Learning with Kernels, MIT Press, Cambridge, MA, 2002.

[18] Schapire, R.E., Singer, Y., 1999. Improved boosting algorithms using confidence-rated predictions. Machine Learning 37 (3), 297–336.

[19] C.L. Blake, C.J. Merz, UCI repository of machine learning databases, Website: http://www.ics.uci.edu/~mlearn/MLRepository.html, 1998.

[20] Rayner Alfred, "Knowledge Discovery: Enhancing Data Mining and Decision Support Integration",2005.

[21] L. Breiman, "Bagging predictors," Machine Learning, vol. 24, pp. 123-140, 1996.