ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

IJARCCE

International Journal of Advanced Research in Computer and Communication Engineering

SITES

Smart And Innovative Technologies In Engineering And Sciences

Gyan Ganga College of Technology

Vol. 5, Special Issue 3, November 2016

Machine Learning: A Critical Review of Classification Techniques

Dr. D. Durga Bhavani[1], A. Vasavi[1], P.T. Keshava[1]

Dept of CSE, NMREC, Hyderabad, India[1]

Abstract: Classification is a data mining (machine learning) technique used to predict group membership for data instances. The goal of this survey is to provide a comprehensive review of different classification techniques in data mining. Evaluation of information, gathered almost everywhere in our day to day life can help devising some efficient and personalized strategies. The amount of information stored in modern databases makes manual analysis intractable. Classification is a well known datamining technique that tells the class of an unknown object. For this purpose, classification predicts categorical (discrete, unordered) labels. Many classification algorithms have been proposed by researchers in statistics, machine learning and pattern recognition. In this paper, we have described various popular classification techniques in data mining.

Keywords: Data Mining, Machine Learning, Prediction, Affinity Grouping, Clustering, Estimation and Visualization.

1. INTRODUCTION

Data mining techniques have been a result of long process of research and product development. This evolution started when business application date used for transactional purposes started getting stored in computers, continued with improvements in date access, which allow users to process, and navigate through the data in real time. Data mining is the process of exploration and analysis, by automatic and semi automatic means of large quantities of data in order to discover meaningful patterns and rules. The six main data mining activities are classification, estimation, prediction, affinity grouping, clustering, estimation and visualization. Learning from data can be classified into supervised and unsupervised learning. Supervised learning (class prediction)[12] is to use the available data to build a model that describes one particular variable of interest in terms of rest of the data. Classification, Prediction and Estimation are the examples. Unsupervised learning (class discovery) [12] is where no variable is declared as target, the goal is to establish some relationship among all variables. Association, visualization, clustering are the examples.

1.1. What is classification:

Classification is the process of finding a model (or function) that describes and distinguishes data classes or concepts, for the purpose of being able to use the model to predict the class of objects whose class label is unknown. The derived model is based on the analysis of a set of training data. The derived model may be represented in various forms, such as classification (IF-THEN) rules, decision trees, mathematical formulae, or neural networks.

Data classification is a two-step process, in the first step; a classifier is built describing a predetermined set of data classes or concepts. This is the learning step (or training phase), where a classification algorithm builds the classifier by analyzing or "learning from" a training set made up of database tuples and their associated class labels.

As the class label of each training tuple is provided, this step is also known as supervised learning (i.e., the learning of the classifier is "supervised" in that it is told to which class each training tuple belongs). It contrasts with unsupervised learning (or clustering), in which the class label of each training tuple is not known, and the number or set of classes to be learned may not be known in advance.

This first step of the classification process can also be viewed as the learning of a mapping or function, y = f (**X**), that can predict the associated class label y of a given tuple **X**. In this view, we wish to learn a mapping or function that separates the data classes. Typically, this mapping is represented in the form of classification rules, decision trees, or mathematical formulae.

In the second step, the model is used for classification. First, the predictive accuracy of the classifier is estimated. If we were to use the training set to measure the accuracy of the classifier, this estimate would likely be optimistic, because the classifier tends to over fit the data (i.e., during learning it may incorporate some particular anomalies of the training data that are not present in the general data set overall). Therefore, a test set is used, made up of test tuples and their associated class labels. These tuples are randomly selected from the general data set. They are

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

IJARCCE

International Journal of Advanced Research in Computer and Communication Engineering

SITES

Smart And Innovative Technologies In Engineering And Sciences

Gyan Ganga College of Technology

Vol. 5, Special Issue 3, November 2016

independent of the training tuples, meaning that they are not used to construct the classifier.

The accuracy of a classifier on a given test set is the percentage of test set tuples that are correctly classified by the classifier. The associated class label of each test tuple is compared with the learned classifier's class prediction for that tuple. Several methods for estimating classifier accuracy. If the accuracy of the classifier is considered acceptable, the classifier can be used to classify future data tuples for which the class label is not known.

1.2 The classification methods are of two types:

1.2.1 Eager learners, when given a set of training tuples, will construct a generalization (i.e., classification) model before receiving new e.g., test) tuples to classify. We can think of the learned model as being ready and eager to classify previously unseen tuples. Decision tree induction, Bayesian classification, rule-based classification, classification by backpropagation, support vector machines, and classification based on association rule mining are all examples of eager learners.

1.2.2 Lazy learners, when given a training tuple, simply store it (or does only a little minor processing) and waits until it is given a test tuple. K-nearest neighbor classifiers and case-based reasoning classifiers are the examples of lazy learners.

2. EAGER LEARNERS' CLASSIFICATION METHODS

2.1 Classification by Decision Tree Induction

Decision tree induction [15] is the learning of decision trees from class-labelled training tuples. A decision tree is a flowchart-like tree structure, where each internal node (non leaf node) denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (or terminal node) holds a class label. The topmost node in a tree is the root node. ID3, C4.5, and CART adopt a greedy (i.e., non backtracking) approach in which decision trees are constructed in a top-down recursive divide-and-conquer manner. Most algorithms for decision tree induction [15] also follow such a top-down approach, which starts with a training set of tuples and their associated class labels. The training set is recursively partitioned into smaller subsets as the tree is being built.

Incremental versions of decision tree induction have also been proposed. When given new training data, these restructure the decision tree acquired from learning on previous training data, rather than relearning a new tree from scratch.

Differences in decision tree algorithms include how the attributes are selected in creating the tree and the mechanisms used for pruning. The basic algorithm described above requires one pass over the training tuples for each level of the tree. This can lead to long training times and lack of available memory when dealing with large databases.

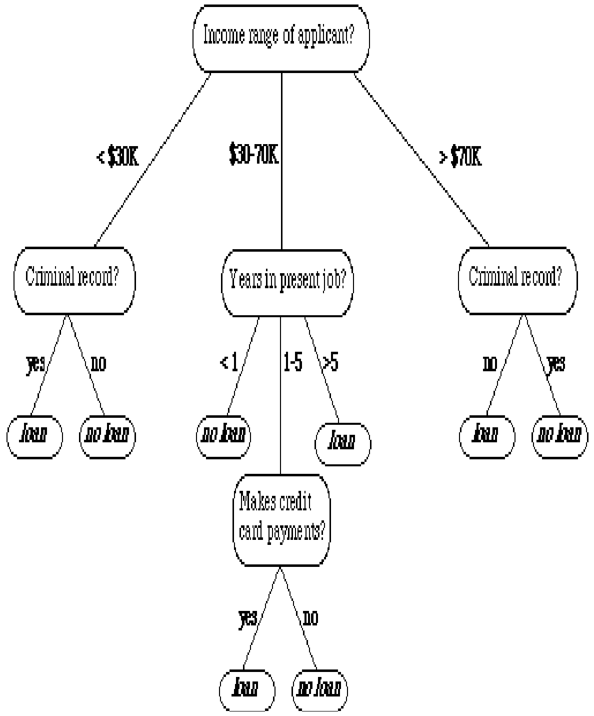


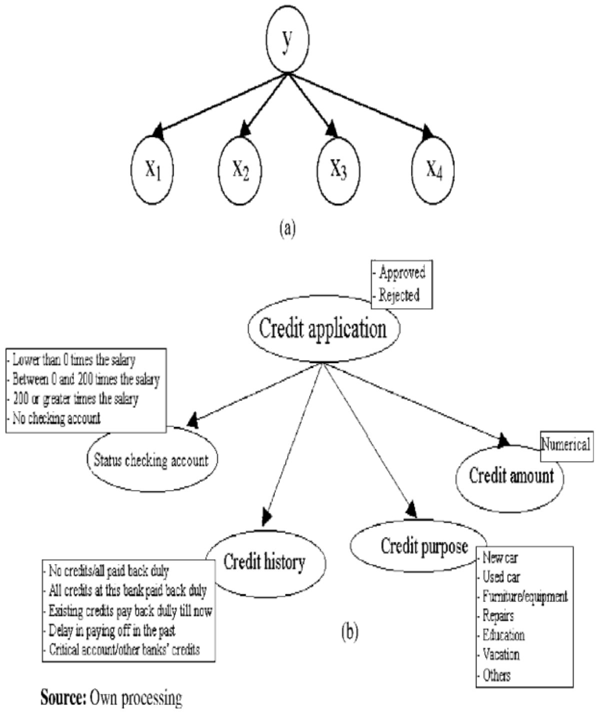
Figure: 1 Decision Tree Induction.

2.2 Bayesian Classification

Bayesian classifiers are statistical classifiers. They can predict class membership probabilities, such as the probability that a given tuple belongs to a particular class.

Studies comparing classification algorithms have found a simple Bayesian classifier known as the naive Bayesian classifier [7] to be comparable in performance with decision tree and selected neural network classifiers. Bayesian classifiers have also exhibited high accuracy and speed when applied to large databases. Bayesian classifiers are statistical classifiers. They can predict class membership probabilities, such as the probability that a given tuple belongs to a particular class.

Naïve Bayesian classifiers assume that the effect of an attribute value on a given class is independent of the values of the other attributes. This assumption is called class conditional independence. It is made to simplify the computations involved and, in this sense, is considered "naïve." Bayesian belief networks are graphical models, which unlike naïve Bayesian classifiers allow the representation of dependencies among subsets of attributes. Bayesian belief networks can also be used for classification.


y
X1
X2
X3
X4
(a)
- Approved
- Rejected
Credit application
- Lower than 0 times the salary
- Between 0 and 200 times the salary
- 200 or greater times the salary
- No checking account
Status checking account
Numerical
Credit amount
Credit history
Credit purpose
- No credits/all paid back duly
- All credits at this bank paid back duly
- Existing credits pay back dully till now
- Delay in paying off in the past
- Critical account/other banks' credits
(b)
- New car
- Used car
- Furniture/equipment
- Repairs
- Education
- Vacation
- Others

Source: Own processing

Figure: 2 Naive Bayesian Classification.

2.2.1 Bayesian Belief Networks

The naïve Bayesian classifier makes the assumption of class conditional independence, that is, given the class label of a tuple, the values of the attributes are assumed to be conditionally independent of one another. This simplifies computation. When the assumption holds true, then the naïve Bayesian classifier is the most accurate in comparison with all other classifiers. In practice, however, dependencies can exist between variables.

Bayesian belief networks [1] specify joint conditional probability distributions. They allow class conditional independencies to be defined between subsets of variables. They provide a graphical model of causal relationships, on which learning can be performed. Trained Bayesian belief networks[10] can be used for classification. Bayesian belief networks are also known as belief networks, Bayesian networks, and probabilistic networks. For brevity, we will refer to them as belief networks.

A belief network is defined by two components—a directed acyclic graph and a set of conditional probability table. Each node in the directed acyclic graph represents a random variable. The variables may be discrete or continuous-valued. They may correspond to actual attributes given in the data or to "hidden variables" believed to form a relationship.

2.2.2 Training Bayesian Belief Networks

In the learning or training of a belief network, a number of scenarios are possible. The network topology (or "layout" of nodes and arcs) may be given in advance or inferred from the data. The network variables may be observable or hidden in all or some of the training tuples. The case of hidden data is also referred to as missing values or incomplete data.

Several algorithms exist for learning the network topology from the training data given observable variables. The problem is one of discrete optimization. Experts must specify conditional probabilities for the nodes that participate in direct dependencies. These probabilities can then be used to compute the remaining probability values. If the network topology is known and the variables are observable, then training the network is straightforward. It consists of computing the CPT entries, as is similarly done when computing the probabilities involved in naive Bayesian classification.

When the network topology is given and some of the variables are hidden, there are various methods to choose from for training the belief network. We will describe a promising method of gradient descent. For those without an advanced math background, the description may look rather intimidating with its calculus-packed formulae.

However, packaged software exists to solve these equations, and the general idea is easy to follow. A gradient descent strategy is used to search for the values that best model the data, based on the assumption that each possible setting of is equally likely. Such a strategy is iterative. It searches for a solution along the negative of the gradient (i.e. steepest descent) of a criterion function. We want to find the set of weights, **W**, that maximize this function. To start with, the weights are initialized to random probability values. The gradient descent method performs greedy hill-climbing in that, at each iteration or step along the way, the algorithm moves toward what appears to be the best solution at the moment, without backtracking. The weights are updated at each iteration. Eventually, they converge to a local optimum solution.

2.3 Classification by Backpropagation

Backpropagation [11] is a neural network learning algorithm. A neural network is a set of connected input/output units in which each connection has a weight associated with it. During the learning phase, the network learns by adjusting the weights so as to be able to predict the correct class label of the input tuples. Neural network learning is also referred to as connectionist learning due to the connections between units.

Neural networks involve long training times and are therefore more suitable for applications where this is feasible. Neural networks have been criticized for their poor interpretability. For example, it is difficult for humans to interpret the symbolic meaning behind the learned weights and of "hidden units" in the network. These features initially made neural networks less desirable for data mining.

Advantages of neural networks, however, include their high tolerance of noisy data as well as their ability to classify patterns on which they have not been trained. They can be used when you may have little knowledge of the relationships between attributes and classes. They are well-suited for continuous-valued inputs and outputs, unlike most decision tree algorithms. They have been successful on a wide array of real-world data, including handwritten character recognition, pathology and laboratory medicine, and training a computer to pronounce English text. Neural network algorithms are inherently parallel; parallelization techniques can be used to speed up the computation process. There are many different kinds of neural networks and neural network algorithms. The most popular neural network algorithm is backpropagation, which gained repute in the 1980s.

2.3.1 A Multilayer Feed-Forward Neural Network

The backpropagation algorithm performs learning on a multilayer feed-forward neural network. It iteratively learns a set of weights for prediction of the class label of tuples. A multilayer feed-forward neural network consists of an input layer, one or more hidden layers, and an output layer.

2.3.2 Defining a Network Topology

Before training can begin, the user must decide on the network topology by specifying the number of units in the input layer, the number of hidden layers (if more than one), the number of units in each hidden layer, and the number of units in the output layer. Normalizing the input values for each attribute measured in the training tuples will help speed up the learning phase. Typically, input values are normalized so as to fall between 0:0 and 1:0. Discrete-valued attributes may be encoded such that there is one input unit per domain value.

There are no clear rules as to the "best" number of hidden layer units. Network design is a trial-and- error process and may affect the accuracy of the resulting trained network. The initial values of the weights may also affect the resulting accuracy. Once a network has been trained and its accuracy is not considered acceptable, it is common to repeat the training process with a different network topology or a different set of initial weights.

Cross-validation techniques for accuracy estimation can be used to help decide when an acceptable network has been found. A number of automated techniques have been proposed that search for a "good" network structure. These typically use a hill-climbing approach that starts with an initial structure that is selectively modified.

2.3.3 Backpropagation

Backpropagation [11] learns by iteratively processing a data set of training tuples, comparing the network's prediction for each tuple with the actual known target value. The target value may be the known class label of the training tuple (for classification problems) or a continuous value (for prediction). For each training tuple, the weights are modified so as to minimize the mean squared error between the network's prediction and the actual target value. These modifications are made in the "backwards" direction, that is, from the output layer, through each hidden layer down to the first hidden layer. Although it is not guaranteed, in general the weights will eventually converge, and the learning process stops.

How efficient is backpropagation?
The computational efficiency depends on the time spent training the network. Given the tuples and w weights, each epoch requires time. However, in the worst-case scenario, the number of epochs can be exponential in n, the number of inputs. In practice, the time required for the networks to converge is highly variable. A number of techniques[11] exist that help speed up the training time. For example, a technique known as simulated annealing can be used, which also ensures convergence to a global optimum.

2.3.4 Backpropagation and Interpretability

Neural networks are like a black box. How can we 'understand' what the backpropagation network has learned? A major disadvantage of neural networks lies in their knowledge representation. Acquired knowledge in the form of a network of units connected by weighted links is difficult for humans to interpret. This factor has motivated research in extracting the knowledge embedded in trained neural networks and in representing that knowledge symbolically. Methods include extracting rules from networks and sensitivity analysis. Various algorithms for the extraction of rules have been proposed. The methods typically impose restrictions regarding procedures used in training the given neural network, the network topology, and the discretization of input values.

Fully connected networks are difficult to articulate. Hence, often the first step toward extracting rules from neural networks is network pruning. This consists of simplifying the network structure by removing weighted links that have the least effect on the trained network. For example, a weighted link may be deleted if such removal does not result in a decrease in the classification accuracy of the network.

2.4 Support Vector Machines

Support Vector Machines [2], a promising new method for the classification of both linear and nonlinear data. In a nutshell, a support vector machine (or SVM) is an algorithm that works as follows. It uses a nonlinear mapping to transform the original training data into a higher dimension. Within this new dimension, it searches for the linear optimal separating hyperplane (that is, a “decision boundary” separating the tuples of one class from another). With an appropriate nonlinear mapping to a sufficiently high dimension, data from two classes can always be separated by a hyperplane. The SVM finds this hyperplane using support vectors and margins (defined by the support vectors).

The SVMs seem to be attracting more attention lately. The reason being that the first paper on support vector machines was presented in 1992 by Vladimir Vapnik and colleagues Bernhard Boser and Isabelle Guyon, although the groundwork for SVMs has been around since the 1960’s. Although the training time of even the fastest SVMs can be extremely slow, they are highly accurate, owing to their ability to model complex nonlinear decision boundaries. They are much less prone to over fitting than other methods. The support vectors found also provide a compact description of the learned model. SVMs can be used for prediction as well as classification. They have been applied to a number of areas, including handwritten digit recognition, object recognition, and speaker identification, as well as benchmark time-series prediction tests.

A popular research area in the support vector machines is the reduced support vector machine [2]. In dealing with large data sets, the reduced support vector machine (RSVM) was proposed for the practical objective to overcome some computational difficulties as well as to reduce the model complexity. The RSVM uses a reduced mixture with kernels sampled from certain candidate set.

2.5 Associative Classification: Classification by Association Rule Analysis

Frequent patterns and their corresponding association or correlation rules characterize interesting relationships between attribute conditions and class labels, and thus have been recently used for effective classification. Association rules [4] show strong associations between attribute-value pairs (or items) that occur frequently in a given data set. Association rules are commonly used to analyze the purchasing patterns of customers in a store. Such analysis is useful in many decision-making processes, such as product placement, catalogue design, and cross-marketing. Association rules are mined in a two-step process consisting of frequent itemset mining, followed by rule generation.

The first step searches for patterns of attribute-value pairs that occur repeatedly in a data set, where each attribute-value pair is considered an item. The resulting attribute value pairs form frequent itemsets. The second step analyzes the frequent itemsets in order to generate association rules. All association rules must satisfy certain criteria regarding their “accuracy” and the proportion of the data set that they actually represent (referred to as support). One of the earliest and simplest algorithms for associative classification is CBA (Classification-Based Association)[8]. CBA uses an iterative approach to frequent itemset Mining.

CMAR (Classification based on Multiple Association Rules)[8] differs from CBA in its strategy for frequent itemset mining and its construction of the classifier. It also employs several rule pruning strategies with the help of a tree structure for efficient storage and retrieval of rules. CMAR adopts a variant of the FP-growth algorithm to find the complete set of rules satisfying the minimum confidence and minimum support thresholds. FP-growth uses a tree structure, called an FP-tree, to register all of the frequent itemset information contained in the given data Set. CBA and CMAR adopt methods of frequent itemset mining to generate candidate association rules, which include all conjunctions of attribute-value pairs (items) satisfying minimum support. These rules are then examined, and a subset is chosen to represent the classifier.

3. LAZY LEARNERS (OR LEARNING FROM YOUR NEIGHBOURS)

3.1 K-Nearest-Neighbor Classifiers

Nearest-neighbor classifiers are based on learning by analogy, that is, by comparing a given test tuple with training tuples that are similar to it. The training tuples are described by n attributes. Each tuple represents a point in an n- dimensional space. In this way, all of the training tuples are stored in an n-dimensional pattern space. When given an unknown tuple, a **k**-nearest-neighbor [5] classifier searches the pattern space for the k training tuples that are closest to the unknown tuple. These k training tuples are the k “nearest neighbors” of the unknown tuple.

3.2 Case-Based Reasoning

A case-based reasoned[14] will first check if an identical training case exists. If one is found, then the accompanying solution to that case is returned. If no identical case is found, then the case-based reasoner will search for training cases having components that are similar to those of the new case. Conceptually, these training cases may be considered as neighbors of the new case. If cases are represented as graphs, this involves searching for subgraphs that are similar to subgraphs within the new case. The case-based [14] reasoner tries to combine the solutions of the neighboring training cases in order to propose a solution for the new case. If incompatibilities arise with the individual solutions, then backtracking to search for other solutions may be necessary. The case-based reasoner may employ background knowledge and problem-solving strategies in order to propose a feasible combined solution.

4. OTHER CLASSIFICATION METHODS

4.1 Genetic Algorithms

Genetic algorithms [9] attempt to incorporate ideas of natural evolution. An initial population is created consisting of randomly generated rules. Each rule can be represented by a string of bits. As a simple example, suppose that samples in a given training set are described by two Boolean attributes, A1 and A2, and that there are two classes, C1 andC2. The rule “IF A1 ANDNOT A2 THENC2” can be encoded as the bit string “100,” where the two leftmost bits represent attributes A1 and A2, respectively, and the rightmost bit represents the class. Similarly, the rule “IF NOT A1 AND NOT A2 THENC1” can be encoded as “001.” If an attribute has k values, where k > 2, then k bits may be used to encode the attribute’s values. Classes can be encoded in a similar fashion. Based on the notion of survival of the fittest, a new population is formed to consist of the fittest rules in the current population, as well as offspring of these rules. Typically, the fitness of a rule is assessed by its classification accuracy on a set of training samples. Offspring are created by applying genetic operators such as crossover and mutation.

In crossover, substrings from pairs of rules are swapped to form new pairs of rules. In mutation, randomly selected bits in a rule’s string are inverted. The process of generating new populations based on prior populations of rules continues until a population, P, evolves where each rule in P satisfies a pre-specified fitness threshold. Genetic algorithms [9] are easily parallelizable and have been used for classification as well as other optimization problems.

4.2 Rough Set Approach

Rough set theory[13] is based on the establishment of equivalence classes within the given training data. All of the data tuples forming an equivalence class are indiscernible, that is, the samples are identical with respect to the attributes describing the data. Given real world data, it is common that some classes cannot be distinguished in terms of the available attributes. Rough sets can be used to approximately or “roughly” define such classes.

A rough approximation of C and an upper approximation of C. The lower approximation of C consists of all of the data tuples that, based on the knowledge of the attributes, are certain to belong to C without ambiguity. The upper approximation of C consists of all of the tuples that, based on the knowledge of the attributes, cannot be described as not belonging to C. Decision rules is generated for each class. Typically, a decision table is used to represent the rules.

4.3 Fuzzy Set Approaches

Rule-based systems [3] for classification have the disadvantage that they involve sharp cut-offs for continuous attributes. For example, consider the following rule for customer credit application approval. The rule essentially says that applications for customers who have had a job for two or more years and who have a high income are approved:

IF (years employed _ 2) AND (income _ 50K) THEN credit = approved;

A customer who has had a job for at least two years will receive credit if her income is, say, $50,000, but not if it is $49,000. Such harsh thresholding may seem unfair. Instead, we can discretize income into categories such as flow income, medium income, high incomes, and then apply fuzzy logic to allow “fuzzy” thresholds or boundaries to be defined for each category. Rather than having a precise cut-off between categories, fuzzy logic uses truth values between 0:0 and 1:0 to represent the degree of membership that a certain value has in a given category. Each category then represents a fuzzy set. Hence, with fuzzy logic [3], we can capture the notion that an income of $49,000 is, more or less, high, although not as high as an income of $50,000. Fuzzy logic systems typically provide graphical tools to assist users in converting attribute values to fuzzy truth values.

Fuzzy set theory [6] is also known as possibility theory. It was proposed in 1965 as an alternative to traditional two-value logic and probability theory. Fuzzy logic systems have been used in numerous areas for classification, including market research, finance, health care, and environmental engineering.

5. CONCLUSION

In this paper, various data classification techniques such as Decision tree classifiers, Bayesian classifiers, Bayesian belief networks and backpropagation (a neural network technique) have been discussed. More recent approaches to classification such as state vector machines have also been surveyed. K-nearest classifier and case based reasoning, which come under lazy learners have also been discussed. Potentially upcoming research areas such as genetic algorithms, rough sets and fuzzy logic techniques have also been surveyed.

REFERENCES

[1] Vladimir Cherkassky, “Multiple Model Regression Estimation,” IEEE Trans. Neural Networks, vol.16, no. 4, 2005.

[2] Yuh-Jye Lee, Su-Yun Huang, “Reduced Support Vector Machines: A Statistical Theory,” IEEE Trans. Neural Networks, vol.18, no. 1, 2007.

[3] Yu Lin, Hongbo Wang, Geng-Sheng (G.S.) Kuo, Shi duan Cheng, “A Fuzzy-Based Approach to Remove Clock Skew and Reset From

One-Way Delay Measurement," IEEE Trans. Neural Networks, vol.16, no. 5, 2005.

[4] Guilherme A. Barreto, Luis G. M. Souza, Rewbenio A. Frota, Leonardo Aguayo, "Condition Monitoring of 3G Cellular Networks Through Competitive Neural Models," IEEE Trans. Neural Networks, vol.16, no. 5, 2005.

[5] KiYoung Lee, Dae-Won Kim, Kwang H. Lee, and Doheon Lee, "Density-Induced Support Vector Data Description," IEEE Trans. Neural Networks, vol.18, no. 1, 2007.

[6] Mang-Hui Wang, "Extension Neural Network-Type 2 and Its Applications," IEEE Trans. Neural Networks, vol.16, no. 6, 2005.

[7] K. Blekas, A. Likas, N. P. Galatsanos, and I. E. Lagaris, "A Spatially Constrained Mixture Model for Image Segmentation," IEEE Trans. Neural Networks, vol.18, no. 1, 2007.

[8] B. Liu,W. Hsu, and Y.Ma. Integrating classification and association rule mining. In Proc. 1998 Int. Conf. Knowledge Discovery and DataMining (KDD'98), pages 80–86,New York, NY, Aug. 1998.

[9] Slavi˘sa Sarafijanovic´, Jean-Yves Le Boudec, "An Artificial Immune System Approach With Secondary Response for Misbehavior Detection in Mobile ad hoc Networks," IEEE Trans. Neural Networks, vol.16, no. 5, 2005.

[10] Tom Auld, Andrew W. Moore, Stephen F. Gull, "Bayesian Neural Networks for Internet Traffic Classification," IEEE Trans. .Neural Networks, vol.18, no. 1, 2007.

[11] Nan Xie, Henry Leung, "Blind Equalization Using a Predictive Radial Basis Function Neural Network," IEEE Trans. Neural Networks, vol.16, no. 3, 2005.

[12] Rui Xu, Donald Wunsch II, "Survey of Clustering Algorithms," IEEE Trans. Neural Networks, vol.16, no. 3, 2005.

[13] Y. Jiang, K. J. Chen, and Z. H. Zhou, "Som-based image segmentation," in Proc. 9th Conf. Rough Sets, Fuzzy Sets, Data Mining and Gradnular Computing, Chongqi, China, 2003.

[14] C. Riesbeck and R. Schank. Inside Case-Based Reasoning. Lawrence Erlbaum, 1989.

[15] P. E. Utgoff, N. C. Berkman, and J. A. Clouse. Decision tree induction based on efficient tree restructuring. Machine Learning, 29:5–44, 1997.