ISSN:0975-9646

Survey on Multiclass Classification Methods

Neha Mehra and Surendra Gupta

Computer Engineering Department
Shri Govindram Seksaria Institute of Technology and Science
Indore, India

Abstract-Supervised learning is based on the target value or the desired outputs. Various successful techniques have been proposed to solve the problem in the binary classification case. The multiclass classification case is more delicate one. In this short survey we investigate the various techniques for solving the multiclass classification problem. Various authors and research modified the multiclass classification approach such as one against one, one against all and Directed Acyclic Graph (DAG) which creates many binary classifiers and combines their results to determine the class label of a test pixel. They also describe the various extensible methods that are extended from binary class to solve the multiclass problem and also explain the method in which the classes are arranged into a tree.

***Keywords**: Multiclass classification, SVM, Neural Network, Hierarchical classification, KNN*

INTRODUCTION

In machine learning, the problem of classification is encountered in various areas, such as medicine to identify a disease of a patient, or industry to decide whether a defect has appeared or not, or to decide the temperature is low, middle or high. In these areas, multiclass classification is a major problem. Each instance in the learning set belongs to a number of set of previously defined labels in multiclass classification. The aim of supervised classification methods is to construct a learning model from a labeled training data set to be able to classify new objects with unknown labels.

Assume that a training data set is given of the form (x_i, y_i), where x_i € R^n is a vector of attributes of the i^{th} object and y_i is the i^{th} class label. We aim at finding a learning model H such that H(x_i) = y_i for new unlabeled objects. The problem is simply formulated to classify the samples into two classes +1 or -1. Several algorithms have been proposed to solve the problem in two class case and some algorithms are extended to solve the problem of multiclass case.

There are three groups of methods to solve the multiclass classification problems. The first group includes methods which can be extended from binary case. The second group includes methods for converting the multiclass classification problem into several binary classification problems. Third group is described by hierarchical classification methods.

EXTENSIBLE METHODS

The multiclass classification problem can be solved by extending the binary classification problem. These include neural networks, decision trees, k-Nearest Neighbor, Naive Bayes, and Support Vector Machines.

A. Neural Networks

Neural network learning is a type of supervised learning, meaning that we provide the network with example inputs and the correct answer for that input. Neural networks are commonly used for classification problems and regression problems. A multilayer feedforward neural network consists of a layer of input units, one or more layers of hidden units, and one output layer of units [1]. The network is not allowed to have cycles from later layers back to earlier layers, hence the name “feed-forward".

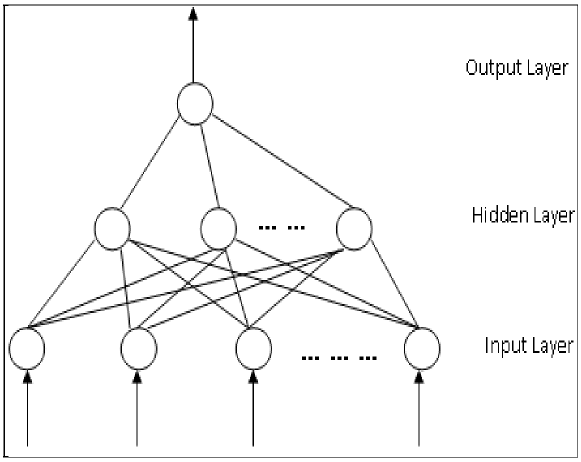

Fig 1: Typical Feed Forward Network composed of 3 layers

Determining the number of hidden units is a bit of an art form, and requires experimentation to determine the best number of hidden units. Too few hidden units will prevent the network from being able to learn the required function, because it will have too few degrees of freedom. Too many hidden units may cause the network to tend to overfit the training data, thus reducing generalization accuracy. In many applications, some minimum number of hidden units is needed to learn the target function accurately, but extra hidden units above this number do not significantly affect the generalization accuracy, as long as cross validation techniques can be used. Too many hidden units can also significantly increase the training time.

In multi-layer feed forward neural networks, the sigmoid activation function, denoted by g(x) is normally used.

$$g\ (x) = \frac{1}{1+\exp\ (-x)} \qquad (1)$$

Instead of just having one neuron in the output layer, with binary output, we could have N binary neurons. The output codeword corresponding to each class can be chosen as follows:

1. One-per-class coding: Each output neuron is designated the task of identifying a given class [14]. The output code for that class should be 1 at this neuron and 0 for the others. Therefore, we will need N = K neurons in the output layer, where K is the number of classes. The output code with four class problem is shown in Table 1.
2. Distributed output coding: Each class is assigned a unique binary codeword from 0 to 2N − 1, where N is the number of output neurons [14]. When testing an

unknown example, the output codeword is compared to the codewords for the K classes, and the nearest codeword, according to some distance measure, is considered the winning class. Usually the Hamming distance is used in that case, which is the number of different bits between the two codewords. The output code with four class problem using N=5 is shown in Table 2.

Table 1: One per-class Coding

Class 1	1000
Class 2	0100
Class 3	0010
Class 4	0001

Table 2: Distributed Output Coding

Class 1	00000
Class 2	00111
Class 3	11001
Class 4	11110

B. Decision Trees

Decision trees are a powerful classification technique. Two widely known algorithms for building decision trees are Classification and Regression Trees [2]. The decision tree consists of nodes that form a *rooted tree*, meaning it is a *directed tree* with a node called "root" that has no incoming edges. All other nodes have exactly one incoming edge. A node with outgoing edges is called an *internal* or test node. All other nodes are called leaves (also known as terminal or decision nodes). In a decision tree, each internal node splits the instance space into two or more sub-spaces according to a certain discrete function of the input attributes values.

The goal of decision tree is to create a model that predicts the value of a target variable based on several input variables. The tree tries to infer a split of the training data based on the values of the available features to produce a good generalization. The split at each node is based on the feature that gives the maximum information gain. Each leaf node corresponds to a class label. A new example is classified by following a path from the root node to a leaf node, where at each node a test is performed on some feature of that example. The leaf node reached is considered the class label for that example. The algorithm can naturally handle binary or multiclass classification problems. The leaf nodes can refer to either of the K classes concerned.

To clearly understand the decision tree consider an example. Imagine you only ever do things at the weekend: go shopping, watch a movie, play tennis or just stay in. What you do depends on three things: the weather (windy, rainy or sunny); how much money you have (rich or poor) and whether your parents are visiting. You say to yourself, if my parents are visiting, we'll go to the cinema. If they are not visiting and it's sunny, then I'll play tennis, but if it's windy, and I'm rich, then I'll go to shopping. If they are not visiting and it's windy and I'm poor then I will go to the cinema. If they are not visiting, then I'll stay in.

To remember all this, you draw a flowchart (decision tree) which will enable you to read off your decision.

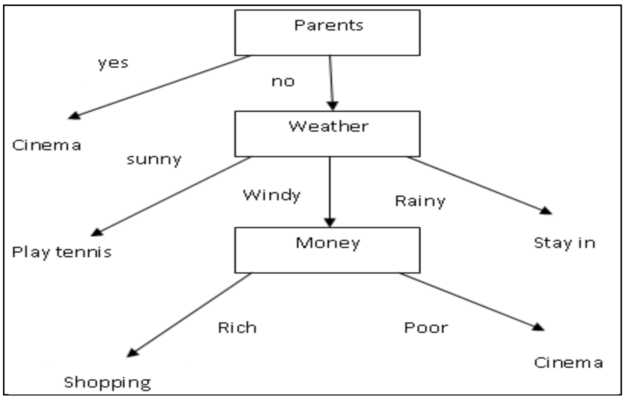

Fig 2: Typical Decision Tree Example

There are many specific decision tree algorithms some are ID3, FID3, C4.5, MARS, CART, and CHAID.

Some advantages of decision tree are, they are computationally simply to understand and interpret, can handle both numerical and categorical data and performs well on large data in a short time.

Some disadvantages are that it can create over-complex trees that do not generalize the data well.

C. K-Nearest Neighbor (kNN)

K-nearest neighbor (*k*NN) classification, finds a group of *k* objects in the training set that are closest to the test object, and bases the assignment of a label on the predominance of a particular class in this neighborhood [3]. There are three key elements of this approach: a set of labeled objects, e.g., a set of stored records, a distance or similarity metric to compute distance between objects, and the value of *k*, the number of nearest neighbors. To classify an unlabeled object, the distance of this object to the labeled objects is computed, its *k*-nearest neighbors are identified, and the class labels of these nearest neighbors are then used to determine the class label of the object. kNN is a type of instance-based learning or lazy learning, where the function is only approximated locally and all computation is deferred until classification.

kNN is called lazy learning which means that it does not use the training data points to do any generalization. In other words, there is no explicit training phase or it is very minimal. This means the training phase is pretty fast. Lack of generalization means that kNN keeps all the training data. More exactly, all the training data is needed during the testing phase.

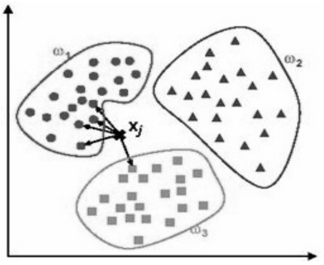

Fig 3: Typical Example of kNN Classifer

Some assumptions of kNN classifer are that, kNN assumes all data is in a feature space.

Some advantages of kNN classifier is that, it is simple in implementation, it is nearly optimal in the large sample limit. Some disadvantages are that, it requires large storage and highly susceptible to the curse of dimensionality

D. Naïve Bayes Classifer

The Naive Bayes algorithm is a classification algorithm based on the Bayes rule. A naive bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumptions [4]. It assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature, given the class variable. It is particularly suited when the dimensionality of the inputs is high. Parameter estimation for naive Bayes models uses the method of maximum likelihood. An advantage of the naive Bayes classifier is that it only requires a small amount of training data to estimate the parameters (means and variances of the variables) necessary for classification.

Given a problem with K classes {C_1, . . . ,C_K} with so-called prior probabilities P(C_1), . . . , P(C_K), we can assign the class label c to an unknown example with features x = (x_1, . . . , x_N) such that c = argmax$_c$P(C = c||x_1, . . . , x_N), that is choose the class with the maximum a posterior probability given the observed data.

$$P(C = c||x_1 \dots x_N) = \frac{P(C=c)P(x_1 \dots\dots x_N||C=c)}{P(x_1 \dots\dots x_N)} \qquad (2)$$

E. Support Vector machine

SVM are a group of supervised learning methods that can be used for classification. It is used for the purpose of classification and regression, can analyze data and recognize patterns. It does not have prior knowledge of the problem but learns about it during training. The major advantage of SVM is its generalization capability [5][6]. This feature makes it better than most of the other models present in this field .It takes set of inputs data and predicts, for each given input, which two possible classes forms the input.

There are two types of problems linearly separable and non-linearly separable problem. Linearly separable problem can be easily separated by a straight hyperplane but non-linearly separable problem cannot. To solve non-linearly separable problem data are transformed from input space to higher dimensional feature space because in the higher dimensional feature space it is easier to separate the input data

A kernel function $\emptyset: X \rightarrow F$ is a mapping from the input space to the feature space F, where patterns are more easily separated, and $w^T\phi(x_i) + b = 0$ is the hyperplane to be derived with w (perpendicular to the separating hyperplane), and b being weight vector and offset, respectively. The maximum margin of the separating hyperplane is 2/||w||.

The choices of kernel function are

1. Linear kernel

$$k(x,y) = x\,.y + c \qquad (3)$$

2. Polynomial Kernel

$$k(x,y) = (ax^T + c)^d \qquad (4)$$

3. Gaussian Kernel Function

$$k(x,y) = exp - \frac{||x-y||^2}{2\sigma^2} \qquad (5)$$

4. Sigmoid kernel:

$$k(x_i, x_j) = tanh(\gamma\, x_i^T\, x_j + r) \qquad (6)$$

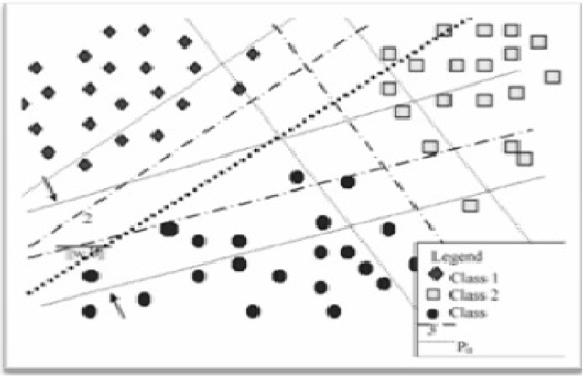

Fig 4: SVM with maximum margin 2/||w||

There are some patterns which are misclassified as and they should be penalized. Therefore, slack variables are introduced to account for misclassifications. The objective function and constraints of the classification problem can be formulated as:

$$\min_{w,b} \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i \qquad (7)$$
$$s.t. y_i\,(w^T \phi(x_i) + b) \geq 1 - \xi_i\,, \xi_i \geq 0$$
$$i = 1,2 \dots, l,$$

Where l is the number of training patterns, C is a parameter, which gives a tradeoff between maximum margin and classification error, and y_i, being +1 or -1, is the target label of pattern x_i.

II Decomposing Into Binary Classification

The most popular approach used in multiclass classification is to decompose the problem into multiple two-class classification problems and then solve those using efficient binary classifiers [7][8][13]. The most successful and widely used binary classifiers are the support vector machine. There are a number of different approaches to decompose a k-class classification problem into two-class problems.

A. One against All Approach (OVA)

Suppose the dataset is to be classified into *K* classes. Therefore, *K* binary SVM classifiers may be created where each classifier is trained to distinguish one class from the remaining *K*-1 classes. For this approach, we require N = K binary classifiers, where the kth classifier is trained with positive examples belonging to class k and negative examples belonging to the other K − 1 classes.

During the testing, samples are classified by finding margin from the linear separating hyperplane. The final output is the class that corresponds to the SVM with the largest margin. However, if the outputs corresponding to two or more classes are very close to each other, those points are labeled as *unclassified*.

This multiclass method has an advantage that the number of binary classifiers to construct equals the number of classes. However, there are some drawbacks. First, during the training phase, the memory requirement is very high and amounts to at the square of the total number of training samples. This may cause problems for large training data sets and may lead to computer memory problems. Second, suppose there are *K* classes and each has an equal number of training samples. During the training phase, the ratio of training samples of one class to rest of the classes will be 1:

(K −1). This ratio, therefore, shows that training sample sizes will be unbalanced.

Because of these limitations, the *one against one* approach of multiclass classification has been proposed.

B. One against One Approach (OAO)

In this method, SVM classifiers for all possible pairs of classes are created. Therefore, for *K* classes, there will be binary classifiers. The output from each classifier in the form of a class label is obtained. The class label that occur the most is assigned to that point in the samples. The number of classifiers created by this method is generally much larger than the previous method. However, the number of training data vectors required for each classifier is much smaller. This method constructs k (k-1)/2 classifiers where each one is trained on data from two classes. For training data from the i^{th} and the j^{th} classes, we solve the following binary classification problem. When testing a new example, a voting is performed among the classifiers and the class with the maximum number of votes wins.

The main disadvantage of this method is the increase in the number of classifiers as the number of class increases but its gives better results than the one against all approach.

C. Directed Acyclic Graph SVM (DAGSVM)

This method is based on the *Decision Directed Acyclic Graph* (DDAG) structure that forms a tree-like structure. Its training phase is same as one against one approach. For k class classification problem, the number of binary classifiers is equal to k (k-1)/2 and each classifier is trained to classify two classes of interest. Each classifier is treated as a node in the graph structure. Nodes in DDAG are organized in a triangle with the single root node at the top and increasing thereafter in an increment of one in each layer until the last layer that will have *k* nodes.

The DDAG evaluates an input vector **x** starting at the root node and moves to the next layer based on the output values. For instance, it exits to the left edge if the output from the binary classifier is negative, and it exits to the right edge if the output from the binary classifier is positive. The binary classifier of the next node is then evaluated. The path followed is called the *evaluation path.* The DDAG method basically eliminates one class out from a list. Initially the list contains all classes. Each node evaluates the first class against the last class in the list. For example, the root node evaluates class 1 against class *k*. If the evaluation results in one class out of two classes, the other is eliminated from the list. The process then tests the first and the last class in the new list. It is terminated when only one class remains in the list. The class label associated with the input data will be the class label of the node in the final layer of the evaluation path or the class remained in the list.

An advantage of using a DAG is that some analysis of generalization can be established. There are still no similar theoretical results for one against the rest and one against one method yet. In addition, its testing time is less than the one against one method.

D. Error Correcting Output Coding

The concept of Error Correcting Output Coding (ECOC) based multi-class method is to apply binary (two-class) classifiers to solve the multi-class classification problems [9]. This approach works by converting *K* class classification problem into a large number *L* of 2-class classification problems. ECOC assigns a unique code word to a class instead of assigning each class a label. A (*L*, *M*, *d*) error correcting code is a *L* bit long, having *C* unique code words with a Hamming distance of d. The hamming distance between two code words is the number of bit positions in which both differs. In a classification problem *K* is the number of classes and *L* is a number decided by the method used to generate error-correcting codes.

Table 3 shows an example for K = 5 classes and N = 7 bit codewords.

Table 3: ECOC Example

	f1	f2	f3	f4	f5	f6	f7
Class 1	0	0	0	0	0	0	0
Class 2	0	1	1	0	0	1	1
Class 3	0	1	1	1	1	0	0
Class 4	1	0	1	1	0	1	0
Class 5	1	1	0	1	0	0	1

Each class is given a row of the matrix. Each column is used to train a distinct binary classifier. When testing an unseen example, the output codeword from the N classifiers is compared to the given K codewords, and the one with the minimum hamming distance is considered the class label for that example.

III HIERARCHICAL CLASSIFICATION

In this method the classes are arranged into a tree. The tree is created such that the classes at each parent node are divided into a number of clusters, one for each child node. The process continues until the leaf nodes contain only a single class. At each node of the tree, a simple classifier, usually a binary classifier makes the discrimination between the different child class clusters [10]. Following a path from the root node to a leaf node leads to a classification of a new pattern. Figure 4 shows the example of 5-class problem.

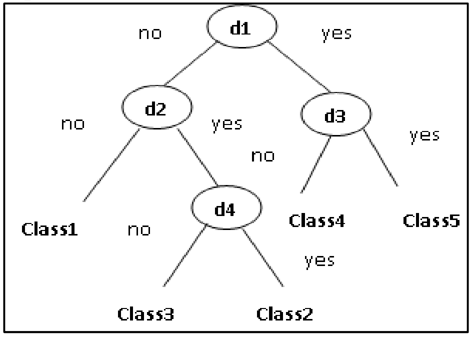

Fig 5: Example of 5-class problem

Kumar et al. [11] proposed a method called Binary Hierarchical Classifier (BHS). The method uses K−1 binary classifiers to classify a K-class problem. The binary classifiers are arranged in a binary tree with K leaf nodes, each corresponding to a given class.

Vural and Dy [12] work on a similar approach of building a binary tree of K −1 binary classifiers, which they call Divide-By-2 (DB2). The split of classes into two clusters at each step is performed by either using k-means algorithm for clustering the class means into two groups or by using the classes grand mean as a threshold and putting classes with means smaller to the grand mean in one cluster and those with larger mean into the other.

IV Experiments And Results

All the classification methods have been tested on various dataset such as iris, wine, glass and vowel. The result in terms of accuracy on various dataset is shown in table no.4.

Table 4: Experimental Results

	Iris	Wine	Glass	Vowel
MLFFN	96.825%	98.88%	70.09%	89.8%
Decision Tree	94%	52.80%	70.7%	81.1
kNN	99.33%	98.84%	70.90%	98.08%
Naïve Bayes Classifer	98.66%	99.42%	63.99%	77.39%
SVM	98%	100%	71.96%	95.64%
OVA	96.66%	98.87%	71.96%	98.485%
OAO	97.33%	99.43%	71.49%	99.053%
DAGSVM	96.66%	98.87%	73.83%	98.674%

Experimentations had performed on hierarchical classification in which vowel and segment dataset is used and result is shown in table no.5.

Table 5: Experimental result on Hierarchical Classification

	Vowel	Segment
Hierarchical Classification	91.42%	92.57%

V Conclusion

This survey presented the different approaches employed to solve the problem of multiclass classification. It explains how two class classification methods can be extended to solve multiclass problem and explains how multiclass problem can be reduced to multiple binary class problem. It also explains how classes can be arranged in a tree, usually a binary tree, and how to utilize a number of binary classifiers at the nodes of the tree till a leaf node is reached. Depending on the need one of the method can be used for the classification purpose.

It shows the result that iris dataset gives best result when we use kNN classifer and SVM classifer. Wine dataset give best result when SVM, Naïve Bayes and One against all method is used. Glass dataset gives best result when it is classified with DAGSVM.

References

[1] Daniel Svozil, Vladimir KvasniEka, JiE Pospichal, Introduction to multi-layer feed-forward neural networks, Chemometrics and Intelligent Laboratory Systems 39 (1997) 43-62.

[2] Ravindra Changala, Annapurna Gummadi, G Yedukondalu, UNPG Raju, Classification by Decision Tree Induction Algorithm to Learn Decision Trees from the class-Labeled Training Tuples, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 4, April 2012.

[3] Yun-lei Cai, Duo Ji ,Dong-feng Cai,A KNN Research Paper Classification Method Based on Shared Nearest Neighbor, Natural Language Processing Research Laboratory, Shenyang Institute of Aeronautical Engineering, Shenyang, China, June 15–18, 2010**.**

[4] Irina Rish. An empirical study of the naive bayes classifier. In IJCAI Workshop on Empirical Methods in Artificial Intelligence, 2001.

[5] Corinna Cortes and Vladimir Vapnik. Support-vector networks. Machine Learning, pages 273–297, 1995.

[6] Tomer Hertz Tomboy, Aharon Bar Hillel and Aharonbh Daphna Weinshall, “Learning a Kernel Function for Classification with Small Training Samples,” School of Computer Science and Engineering, The Center for Neural Computation, The Hebrew University of Jerusalem, Jerusalem, Israel.

[7] Erin Allwein, Robert Shapire, and Yoram Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. Journal of Machine mLearning Research, pages 113–141, 2000.

[8] Chih-Wei Hsu and Chih-Jen Lin, “A Comparison of Methods for Multiclass Support Vector Machines,” IEEE Transactions on neural networks vol.13, no. 2, march 2002.

[9] T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error correcting output codes. Journal of Artificial Intelligence Research, 39:1–38, 1995.

[10] Volkan Vural and Jennifer G. Dy. A hierarchical method for multi-class support vector machines. In Proceedings of the twenty-first international conference on Machine learning, pages 105–112, 2004.

[11] S. Kumar, J. Ghosh, M.M. Crawford, Hierarchical fusion of multiple classifiers for hyperspectral data analysis, Pattern Analysis& Applications, 5:210-220, 2002.

[12] V. Vural, J.G. Dy, A hierarchical method for multi-class support vector machines. In Proceedings of the Twenty-First International Conference on Machine Learning, 105-112, 2004.

[13] Mahesh Pal, Multiclass Approaches for Support Vector Machine Based Land Cove Classification. Lecturer, Department of Civil engineering National Institute of Technology Kurukshetra, 136119, Haryana (India), 2008.

[14] Mohamed Aly, Survey on Multiclass Classification Methods, Technical Report, Caltech, USA, 2005.

Authors

Neha Mehra received her Bachelor of Engineering degree in Computer Science from RGPV University, India in 2010. She is currently pursuing Master of Engineering in Computer Engineering from SGSITS, Indore, India. Her research interests include machine learning.

Surendra Gupta received the Bachelor of Engineering degree in computer science and engineering from Barkatullah University, India in 1997 and Master of Engineering degree in computer engineering from DAVV University, India in 2000. He is currently working as Assistance Professors in computer engineering department at SGSITS Indore, India. His interests are in machine learning and optimization. He is a member of the computer society of India.