Using two-class classifiers for multiclass classification

David M.J. Tax and Robert P.W. Duin
Pattern Recognition Group
Faculty of Applied Science, Delft University of Technology
Lorentzweg 1, 2628 CJ Delft, The Netherlands
e-mail: {davidt,bob}@ph.tn.tudelft.nl

Abstract

The generalization from two-class classification to multiclass classification is not straightforward for discriminants which are not based on density estimation. Simple combining methods use voting, but this has the drawback of inconsequent labelings and ties. More advanced methods map the discriminant outputs to approximate posterior probability estimates and combine these, while other methods use error-correcting output codes. In this paper we want to show the possibilities of simple generalizations of the two-class classification, using voting and combinations of approximate posterior probabilities.

1 Introduction

Classifiers are often developed to distinguish between just two classes of objects. A discriminant function f_{AB} is optimized such that for values larger than a certain threshold value, the object is classified as class ω_A, and otherwise to class ω_B. This procedure is a direct generalization of the Bayes classifier, where for each of the classes the probability density function is estimated, and the object is assigned to the class with the highest posterior probability [3]. The discriminant function can directly be derived from the posterior probabilities by taking the difference between the posterior probabilities of the two classes.

A nice feature of using the posterior probabilities (and thus also of this Bayes classifier) is, that it can directly be extended to problems with more than two classes. It just requires the estimation of the extra class probability density (and the class prior probability). New objects are again assigned to the class with the largest posterior probability.

When the discriminant functions are not based on estimated densities but basically fit a discriminant between two classes (like the Fisher linear discriminant [7], the perceptron or the support vector classifier [11]), the multiclass generalization is not always straightforward [1]. It is in general not possible to compare the outputs of the discriminant functions directly to find the 'most confident' one. Several strategies are proposed to solve this problem.

The first possibility is to apply a voting mechanism, where each classifier votes for (or against) a certain class. The classifiers only have to output a binary answer. The object is assigned to the class with the highest number of votes. A second approach is to assume that the classifiers output an estimated class probability (or something related [10]), and assign the test object to the classifiers with the maximal ('most confident') classification output. This clearly requires classifiers which output a continuous confidence answer. Finally, it is possible to introduce error-correcting output codes, which changes the definition of the class a single classifier has to learn [4].

In this paper we show the voting approaches in comparison with the classifier output approaches for the extension to two-class classifiers to multiclass classifiers. For this, a very simple procedure to estimate posterior probabilities is proposed, to avoid ties and inconsequent labelings. We will use the same datasets as presented by [9].

2 Theory

Assume we have a labeled dataset $\mathcal{X}^{tr}$ containing N objects from K classes. The ultimate goal is to classify new objects from the same distribution in one of the K classes, ω_k, possibly with the option to reject the object in case of ties or contradictions. In the coming section, the classes will be labeled $\omega_A, \omega_B, \omega_C, ...$

Further we assume that we can train a discriminant function $f_{AB}(\mathbf{x};\mathbf{w})$ on a two-class classification problem involving classes ω_A and ω_B. The discriminant is optimized such that:

$$f_{AB}(\mathbf{x};\mathbf{w}) = \begin{cases} \geq 0 & \text{if} \quad \mathbf{x} \in \omega_A \\ < 0 & \text{if} \quad \mathbf{x} \in \omega_B \end{cases} \tag{1}$$

Note that objects $\mathbf{x}$ which do not originate from class ω_A or ω_B, are ignored. This definition of the discriminant does

1051-4651/02 $17.00 (c) 2002 IEEE

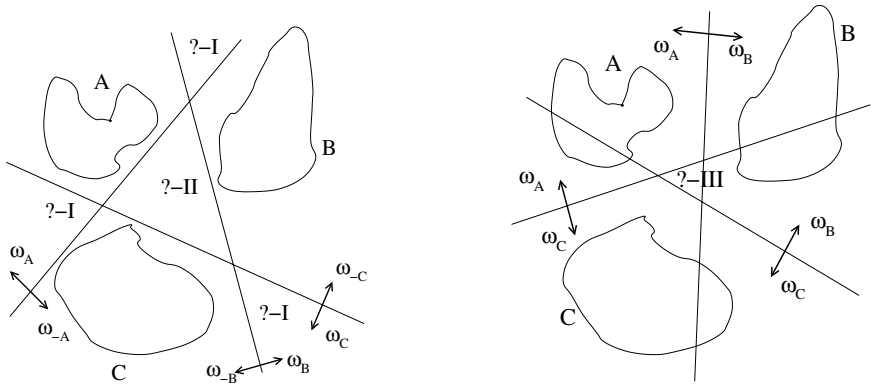

Figure 1. Two approaches in multiple classes: left shows one against the rest, right shows each pair of classes. The areas indicated with question marks show areas where there is inconsistent output.

not define the exact values for f_{AB} larger or smaller than 0. For the support vector classifier, the outputs are binarized $f_{AB} = \pm 1$. For other classifiers, like the Fisher discriminant, the output is the distance to the decision boundary ($-\infty < f_{AB} < \infty$).

2.1 Voting: one-against-rest or one-against-one

When it is assumed that the classifiers output a *binary* decision, there are two basic approaches. In the first approach a classifier between one class and the $K-1$ other classes is trained (in total K classifiers). This is called the 1-r (one-against-rest) approach. Each discriminant k outputs the decision ω_k or ω_{-k} ('not class k'). In the second approach a classifier is trained between each *pair* of classes (in total $K(K-1)/2$ classifiers). This classifier (labeled with double indices (k_1, k_2)) outputs the decision ω_{k1} or ω_{k2}. This is called the 1-1 (one-against-one) approach.

The objects are then classified by applying a combining rule on the set of decisions. One strategy is to use voting, where the object is labeled to the class with the highest number of votes (in the 1-r approach this means that a class can have 1 or 0 votes). In both cases we are faced with the possibility of ties or contradictory votings [5]. In figure 1 an example of a 2D dataset containing three classes is shown. For both scenarios linear classifiers are trained. When the voting combining is used, there are areas with inconsistent labelings, indicated by the question marks. In the first approach all classifiers might conclude it is not their class (so it is rejected by all classifiers, ?-II), or several classifiers label it as their class (?-I). In the second approach there will be cases where all classifiers disagree (?-III) or there are ties in the voting (not possible here for $K = 3$).

When rejected objects are not allowed in the application, these rejected objects should still be classified. The most simple method is to assign these objects to the class with largest *prior* probability P_{k^*}. The error rate on this (rejected) data will therefore be $\epsilon_r = 1 - P_{k^*}$ ($= (K-1)/K$ for equal prior probabilities). When a fraction f of the objects is rejected, and the rest is classified with an error ϵ, the total error ϵ_{tot} becomes $\epsilon_{tot} = (1-f)\epsilon + f(1-P_{k^*})$.

2.2 Confidence value estimation

To avoid these ties and inconsequent labelings, a real-valued confidence value for each classification has to be used. Of course, when we assumed we have (two-class) classifiers which just output a binary classification, there is no confidence level for the classification defined. In these cases we will use the distance from the object $\mathbf{x}$ to the decision boundary $f_{AB}(\mathbf{x})$. In order to map such a distance on posterior probabilities, we use a 1D logistic classifier [2] which maximizes the likelihood of the classified training objects. This is achieved by optimizing α in the function [6]:

$$\tilde{p}_{AB}(\omega_A|\mathbf{x};\mathbf{w}) = \frac{1}{1+\exp(-\alpha f_{AB}(\mathbf{x};\mathbf{w}))} \qquad (2)$$

This optimization will produce for each classification a confidence level, but it will not change the output label for the two-class problem.

2.3 Combining the confidence value estimations

The probability estimates of the classifiers will be combined using the maximum rule (the objects are assigned to the class with the maximal output). Combining *all* $K(K-1)/2$ classifiers in the 1-1 situation (using voting or using the estimated posterior probability) will often result in poor classification performances. Assume that the object

$\mathbf{x}$ originates from class ω_A. There will be $(K-1)(K-2)/2$ classifiers which have never seen objects from this class ω_A. Combining these ignorant classifiers will therefore result in almost random classification. This becomes even more prominent for larger numbers of classes. For a classification problem involving more than $K = 4$ classes, the majority of the classifiers are ignorant of class ω_A (and for a 10-class problem even 80%).

Note that when the classifiers are combined using the estimated probabilities, it is still possible to reject the least confident outputs. In that case a threshold on the posterior probability can be set, but the user has to define the fraction of the objects which can be rejected. This will not be investigated in this paper.

3 Experiments

In this section we will use the datasets presented in [8]. In table 1 the characteristics of these datasets are given, showing the variety of training set sizes, number of classes and dimensionality. Ten-fold cross-validation is used. Three classifiers are considered: the normal density based linear classifier (LDA), the Fisher linear discriminant and the linear support vector classifier (SVC). For the first classifier the output is defined as the probability density estimates for each of the classes. In that case the fitting from section 2.2 is not required. For the second two classifiers the confidence estimates and combining rules are used.

Table 1. Dataset characteristics

dataset	#train	#test	#class	dim.
vowel	528	462	11	10
waveform	300	500	3	21
vehicle	423	423	4	18
crabs	80	120	4	5
digits	1000	1000	10	256

In the experiments we will investigate the classification errors and rejection rates obtained by 4 different methods for generalizing two-class classifiers (see section 2.1 for explanation of '1-r' and '1-1').

vote 1-r: K classifiers are trained and the binarized output to vote for the output class is used. In case of ties or inconsequent votes, the object is rejected or randomly assigned to one of K classes.

prob 1-r: K classifiers are trained and the output of the classifiers is mapped to a posterior probability estimate. The object is assigned to the class with the largest output.

vote 1-1: here $K(K-1)/2$ classifiers are trained. Again the binarized output is used to vote for the output class. In case of ties or inconsequent votes, the object is rejected or randomly assigned.

prob 1-1: $K(K-1)/2$ classifiers are trained, but the output of the classifiers is mapped to a posterior probability estimate. The object is assigned to the class with the largest output.

In table 2 the classification errors and rejection rates (if applicable) are shown for all datasets and all multiclass generalizations. For the rules involving voting, a third result is given: the total classification error ϵ_{tot} when the rejected objects are randomly assigned as in section 2.1.

We can conclude, that vote 1-r rejects large fractions of the data, while vote 1-1 has a low rejection rate (often even 0%). This is clearly visible for the LDA in datasets which contain many classes (vowel and digits). On the other hand, when the objects are rejected, the performance on the remaining data is often much better (compare the error of vote 1-r and ϵ_{tot} of vote 1-r or error of prob 1-r). When the rejected objects are randomly assigned, very poor results appear (the ϵ_{tot} column is much higher than the error column).

When classifiers output are mapped to probabilities and combined, much better results appear for the 1-r case (see also section 2.3). The classification performance is not better than the vote 1-r with reject, but consistently better than vote 1-r with random assignment of rejected objects.

The vote 1-1 including rejection is worse than vote 1-r, but better when the rejected objects are randomly assigned. This is mainly due to the low rejection rate. When vote 1-1 shows high rejection rate (for instance in the digit dataset), performances are comparable. Furthermore, prob 1-1 performs well if the outputs are well estimated (as for the LDA) but otherwise it performs very poorly (both in Fisher and SVC).

4 Conclusions

Multiclass classification problems can be solved by combining the hard output labels (voting) or probability estimates of standard two-class classifiers. Voting can result in many rejects, especially when K classifiers are trained to distinguish between one class and the rest (1-r case), but the performance on the accepted objects is very good (the 'difficult' objects are rejected). For the one-against-one (1-1) case, much less objects are rejected. When these rejectes are not acceptable, and the classifiers do not provide a probability estimate, we propose to fit a simple logistic function on the outputs and combine them with the maximum combining rule in the 1-r case. This does not improve over the voting method with rejects, but no objects are rejected

Table 2. Classification errors (in %) for all datasets and all multiclass generalization rules. For the voting rules also the rejection rate and classification error after random assignment of the rejected objects is shown. Values between brackets are the standard deviation over 10 runs.

LDA								
		vote 1-r		prob 1-r		vote 1-1		prob 1-1
	error ϵ	reject f	ϵ_{tot}	error	error ϵ	reject f	ϵ_{tot}	error
vowel	27.0 (4.0)	75.0 (2.0)	75.0 (1.1)	49.7 (1.2)	25.4 (1.5)	5.5 (1.0)	29.0 (1.8)	30.3 (2.7)
waveform	6.8 (0.8)	32.5 (1.4)	26.3 (1.0)	16.3 (1.1)	15.0 (1.2)	0.0 (0.0)	15.0 (1.2)	20.0 (1.1)
vehicle	15.6 (1.4)	26.0 (2.0)	31.1 (0.7)	23.9 (1.4)	18.7 (1.2)	2.0 (0.6)	19.8 (1.2)	18.9 (1.4)
crabs	2.2 (1.4)	7.6 (2.2)	7.7 (2.0)	5.4 (2.6)	2.9 (1.1)	0.0 (0.0)	2.9 (1.1)	2.7 (1.4)
digits	10.4 (1.1)	21.3 (1.4)	27.4 (1.1)	18.6 (0.7)	74.5 (0.8)	34.5 (7.3)	79.8 (1.3)	44.7 (4.4)
Fisher								
		vote 1-r		prob 1-r		vote 1-1		prob 1-1
vowel	5.3 (6.3)	96.1 (0.8)	87.6 (0.6)	51.1 (2.0)	25.4 (1.5)	5.5 (1.0)	29.0 (1.8)	82.8 (4.6)
waveform	7.3 (0.8)	31.5 (1.4)	26.0 (0.9)	16.4 (1.4)	15.0 (1.2)	0.0 (0.0)	15.0 (1.2)	22.1 (2.0)
vehicle	15.2 (1.7)	27.3 (1.9)	31.5 (0.6)	23.1 (1.5)	18.7 (1.2)	2.0 (0.6)	19.8 (1.2)	42.6 (5.4)
crabs	1.9 (1.4)	8.1 (2.0)	7.8 (2.1)	5.1 (2.0)	2.9 (1.1)	0.0 (0.0)	2.9 (1.1)	50.0 (13.0)
digits	9.0 (0.8)	23.8 (0.8)	28.3 (0.8)	17.9 (1.0)	16.6 (1.1)	17.2 (1.1)	29.2 (1.4)	88.9 (1.6)
SVC linear								
		vote 1-r		prob 1-r		vote 1-1		prob 1-1
vowel	16.9 (3.6)	73.0 (2.0)	71.0 (1.0)	69.6 (2.4)	19.8 (2.1)	7.8 (1.1)	25.4 (1.9)	81.3 (2.3)
waveform	7.5 (0.8)	28.5 (0.6)	24.4 (0.5)	16.0 (1.2)	15.2 (0.6)	0.0 (0.0)	15.3 (0.6)	49.2 (2.5)
vehicle	15.7 (1.9)	24.6 (2.6)	30.3 (1.5)	23.9 (2.5)	19.3 (1.4)	3.7 (1.8)	21.4 (1.4)	40.4 (10.6)
crabs	2.4 (1.3)	9.9 (2.5)	9.6 (2.3)	7.1 (1.5)	4.1 (1.0)	0.6 (0.6)	4.5 (0.9)	67.0 (15.6)
digits	5.4 (0.7)	18.7 (1.0)	21.2 (1.0)	12.2 (1.0)	5.9 (0.8)	1.9 (0.4)	7.5 (0.5)	67.1 (4.1)

and the performance is significantly better than the voting in which the rejects are randomly assigned. Surprisingly, the voting with the random assignment often gives the best performance in the 1-1 case.

References

[1] E. Allwein, R. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. In *Proc. 17th International Conf. on Machine Learning*, pages 9–16. Morgan Kaufmann, San Francisco, CA, 2000.

[2] J. Anderson. Logistic discrimination, 1982.

[3] C. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Walton Street, Oxford OX2 6DP, 1995.

[4] T. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

[5] R. Duda, P. Hart, and D. Stork. *Pattern classification*. Wiley, 2001.

[6] R. P. W. Duin and D. M. J. Tax. Classifier conditional posterior probabilities. *Lecture Notes in Computer Science*, 1451:611–619, 1998.

[7] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179–188, 1936.

[8] T. Hastie and W. Stuetzle. Principal curves. *Journal of the American Statistical Association*, 84(406):502–516, June 1989.

[9] T. Hastie and R. Tibshirani. Classification by pairwise coupling. Technical report, Standford university and University of Toronto, November 1996.

[10] U. Kressel. Pairwise classification and support vector machines, 1999.

[11] V. Vapnik. *Statistical Learning Theory*. Wiley, 1998.