

Classifier Conditional Posterior Probabilities

Robert P.W. Duin, David M.J. Tax

Pattern Recognition Group, Department of Applied Sciences
Delft University of Technology, The Netherlands

Abstract

Classifiers based on probability density estimates can be used to find posterior probabilities for the objects to be classified. These probabilities can be used for rejection or for combining classifiers. Posterior probabilities for other classifiers, however, have to be conditional for the classifier, i.e. they yield class probabilities for a given value of the classifier outcome instead for a given input feature vector. In this paper they are studied for a set of individual classifiers as well as for combination rules.

Keywords: classification reliability, reject, combining classifiers

1 Introduction

If the error rate of a classifier is known to be ε then the overall probability that an arbitrary new object is erroneously classified is ε . For individual objects, however, often more specific numbers for the expected classification accuracy may be computed, depending on the type of classifier used. Discriminant rules based on density estimation are usually able to yield estimates for the posterior probabilities $P(\omega_i | x)$, i.e. the probability that the correct label ω of x is ω_i . Such numbers might be interpreted as the probability that the generated label is correct, given the object x . This quantity is called the confidence of the classification result. Confidences can be used for rejecting objects or for combining the classifier with other, possibly uncertain, information. An important application is the combination of classifiers. This can be given a theoretical foundation in case the classifiers yield reliable posterior probabilities [4], [5].

There are just a few classifiers that are really based on probability density estimates, parametric ones, e.g. assuming normal densities, or non-parametric ones like the k -nearest neighbour rule (k -NN). Many others just produce linear discriminant functions, or are, like the 1-NN rule, based on very local information that can't be used for density estimation. Neural network classifiers usually have $[0,1]$ outputs that, if well trained, after normalization may be used as posterior probability estimates. These are, however, not based on densities and should thereby be interpreted differently from the above posterior probabilities.

In this paper it will be shown to be possible to define so-called classification confidences for a large set of classifiers, i.e. estimates for classifier conditional class probabilities $P(\omega_i | S(x))$ in which ω_i is the by $S(x)$ selected class for object x . These are class probabilities generated from continuous classifier outputs, e.g. some distance from x to the separation boundary. It should be emphasized that classifier conditional class probabilities can be entirely different for different classifiers as they generally reflect the underlying assumptions or models on which the classifiers are based.

In the next sections we will propose classifier conditional posterior probabilities for a range of classifiers. These will be experimentally evaluated on some simple

examples. Here also the consistency of classifier combination rules will be verified. The methods discussed in this paper are implemented in a Matlab toolbox [7] that can be obtained freely for academic purposes.

2 Conditional posterior probabilities for various classifiers

For classifiers based on probability density estimates, posterior probabilities can be directly related to these densities:

$$P(\omega_i|x) = \frac{P(x|\omega_i)P(\omega_i)}{\sum_i P(x|\omega_i)P(\omega_i)} = \frac{F_i(x)P(\omega_i)}{\sum_i F_i(x)P(\omega_i)} \quad (1)$$

$F_i(x)$ is the density estimate for class ω_i . In some rules, like the k-NN rule and decision trees, densities are related to cell frequencies. As these can be very small numbers we prefer the Bayes probability estimates over just frequencies, avoiding problems with classes not represented in some particular cells:

$$P(\omega_i|x) = \frac{n_i(x) + 1}{N(x) + c} \quad (2)$$

in which $n_i(x)$ is the number of training objects of class ω_i in the cell related to x . $N(x)$ is the total number of training objects in that cell (in the k-NN rule $N(x)=k$ everywhere). c is the number of classes. Equation (2) is not very informative for small k in the k-NN rule. For that reason we experimented with a distance related estimate described below.

Many classifiers just produce a distance to a separation boundary $S(x)=0$, e.g. Fisher's Linear Discriminant. In order to map such a distance on posterior probabilities they should be scaled properly. An attractive method to do this is using a 1-dimensional logistic classifier [6] which maximizes the likelihood of the classified training objects. This can be achieved by optimizing α in the function:

$$P(\omega_i|x) = \text{sigmoid}(\alpha S_i(x)) = \frac{1}{1 + \exp(-\alpha S_i(x))} \quad (3)$$

which maps the classifier on the $[0,1]$ interval. An important point is the consistency of the classifier with the posterior probabilities and the confidences: classifications derived from the latter should not differ from those derived from the first. For that reason the constant term is suppressed in order to keep the position of $S(x) = 0$ in the feature space identical to that of $P(\omega_i|x) = 0.5$. This method can be used for all linear and polynomial classifiers.

For the case of the k-NN rule we considered two possible estimators for the posterior probability. The most obvious one is based on (2):

$$P_i^1 = \frac{k_i + 1}{k + c} \quad (4)$$

in which k_i is the number of neighbours of class ω_i . The problem with this estimator is that it has a limited range of outcomes for small values of k . Therefore we investigated the possibility to include the object distance into the estimator. Let us assume that for a particular object x class ω_i has the majority with k_i neighbours. Let the distance to the

most remote object among these k_i neighbours be $d_i(x)$. The distances to the k_i -th object of the other classes are denoted by $d_j(x)$ ($j = 1, c, j \neq i$). As the class densities in x are inversely proportional to these distances we propose the following posterior probability estimator for class ω_i

$$P_i^2 = 1 - \frac{(c-1)d_i(x)}{\sum_{j=1, c} d_j(x)} \quad (5)$$

The two estimators (4) and (5) will be compared experimentally in the next section. We also experimented with a dimension dependency in (5) as densities are related to the multidimensional integrals over spheres determined by the nearest neighbour distances. Especially for high-dimensional situations this yielded very bad results as often the data itself has a lower and intrinsic dimensionality than the feature space that is often hard to determine.

Feed forward neural networks can easily be given continuous outcomes on the $[0,1]$ interval. For well trained and sufficiently large networks they approximate the posterior probabilities. In general they can at best be treated as classifier conditional posterior probabilities as an arbitrary, possibly nonoptimally trained neural network will not adapt itself to the underlying data distribution. Network outcomes can, however, often successfully be used for rejection and classifier combining purposes.

3 The confidence error

An important point of consideration is how the quality of the classification confidence estimators should be defined. Obviously unbiased estimators are desired: on the average a fraction p of all objects with confidence p should be classified correctly. Moreover, these estimators have to say something interesting. If all objects are given the same confidence $1-\epsilon$ (ϵ is the error rate) then the unbiasedness demand is satisfied but the result is not informative. Reliably classified objects should have larger confidences than objects close to the decision boundary. A good criterion to measure this over set of objects may be the likelihood: $-\sum \log(p_i)$. This measure, however, yields problems for very small probability estimates. Therefore we decided to define a confidence error q as the mean square error between the posterior probability estimate and the classification correctness indicator $C(x)$: $C(x) = 1$ if x is correctly classified and $C(x) = 0$ if x is erroneously classified (see also [8] and [9]):

$$q = E\{(C(x) - p(x))^2\} \quad (6)$$

$p(x)$ is the estimated confidence for object x . The expectation $E(\bullet)$ is taken over the weighted sum of the class conditional distributions of x . $C(x)$ depends on the classifier $S(x)$. As a result the confidence error q depends on the classification problem and the classifier under consideration. This classifier yields for each x two outputs: a class assignment ω_i and a confidence estimate p . Using an independent testset estimates for ϵ and q can be obtained. They characterize the behaviour of the classifier with respect to the data: ϵ describes the overall reliability of the hard classification ω_i , q describes the overall reliability of the confidence output $p(x)$.

A typical value for q is obtained by considering a classifier for which $p(x) = 1 \forall x$, i.e. it states to be sure on all its decisions. Easily can be derived that for this case $q = \epsilon \cdot 1^2 + (1-\epsilon) \cdot 0^2 = \epsilon$. This is not an absolute upperbound for q (which is 0.25 reached for $p(x) = 0.5 \forall x$) but a value that can easily be realized without any effort. A lower bound is found for the case in which a fraction of 2ϵ of the objects is entirely on the edge and has correctly $p(x) = 0.5$ and all other objects have $p(x) = 1$. So all unreliable classified objects are clustered to a set of size 2ϵ . Random assignment (these objects are on the decision boundary) thereby yields an error of ϵ . Now $q = 2 \epsilon \cdot 0.5^2 = \epsilon/2$. So in practice q is bounded by $\epsilon/2 < q < \epsilon$. For that reason we define

$$\rho = \frac{q}{\epsilon} - \frac{1}{2} \quad (7)$$

bounded by

$$0 < \rho < 1/2 \quad (8)$$

to be the normalized confidence error ρ . It will be studied in the below experiments. Note that $\rho = 0$ is only be reached if all unreliable objects are on the decision boundary and if this is detected by the classifier. $\rho = 1/2$ is reached if simply all confidences are set to 1. Values with $\rho > 1/2$ are possible and indicate that the individual confidences should not be used. Applied to rejection these bounds can be interpreted as: $\rho = 0$: rejection is very effective as the error decreases by $\delta/2$ if δ objects are rejected, $\rho = 1/2$: rejection is ineffective as the error decreases by $\delta\epsilon$ if δ objects are rejected.

4 Experiments

In this section confidence will be studied experimentally for a set of two-class problems and a set of classifiers. Here also various classifier combination rules will be considered as far as they are based on the posterior probabilities: minimum, median, maximum, mean and product.

The following datasets are considered:

- Two 2d Gaussian classes with different covariance matrices, 100 object per class
- Two 10d Gaussian classes with equal covariance matrices, 100 objects per class
- A real 8d data set with two classes having 130 and 67 objects (the 'Blood' dataset [10]), see fig. 1 for the first two principal components.

First we compared the two confidence estimators (4) and (5) for various k in the k -NN method using the datasets a and b. It appears, see fig. 2 and 3, that according to the normalized confidence error ρ the distance based estimator P^2 is only better for $k = 1$. So we decided to use P^2 just for the 1-NN method and for $k > 1$ we used P^1 , based on cell frequency counts.

For each of the three datasets we ran 25 experiments with randomly selected training sets of various sizes (3 up to 50 objects per class). A set of ten classifiers was trained (see below) and tested with the remaining objects. We also combined in each experiment all 10 classifiers into a single classifier based on the classifier conditional posterior probabilities. As combiner the maximum, median, minimum, mean and

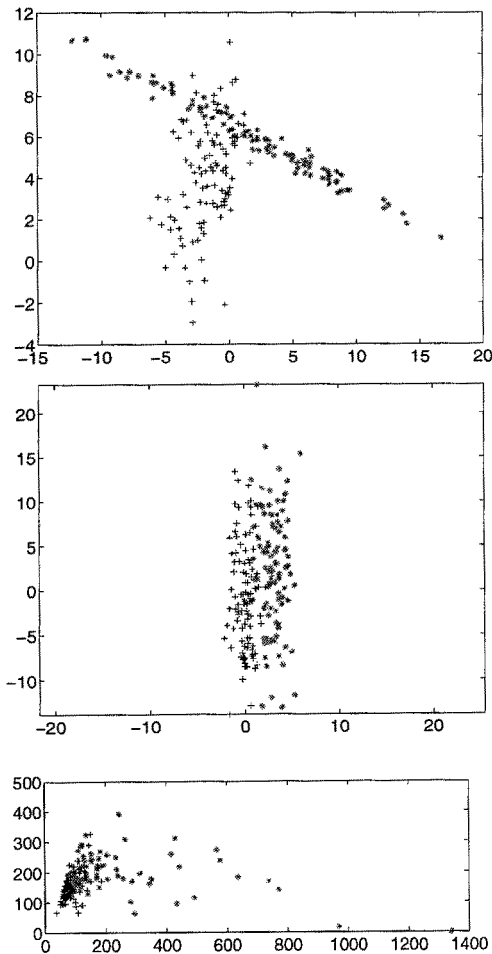


Fig. 1. Three datasets used:
 a) 2d Gaussian
 b) 10d Gaussian (features 3-10 are redundant)
 c) First two principal components of the
 'Blood' data.

Low values for the classification error ϵ and low values for the confidence error ρ are good, so classifiers in the bottom-left region of the plots match the data. The following observations can be drawn from the experiments:

Classification errors tend to decrease for increasing sizes of the training set. This does not hold for the normalized confidence errors as defined by (7). Partially this is caused by the normalization by the decreasing error. It is also the consequence of the fact that the accuracy of some density estimates fits itself to the data size like in the 1-NN rule and in decision trees in which the final nodes tend to have a constant number of training objects.

product rules were used. Averages for the classification error and for the normalized confidence error were computed for all individual and combined classifiers. In fig. 4 some ϵ - ρ plots are given. In table 1 the results for 20 objects per class are shown numerically.

The following classifiers are used: Nearest Mean (NM), Fisher linear discriminant (Fish), Bayes assuming Gaussian densities with different uncorrelated (diagonal) covariance matrices (BayU), Bayes assuming Gaussian densities with arbitrary covariance matrices (BayQ), 1-nearest neighbour rule (1-NN), the k nearest neighbour rule with built-in optimization of k over the leave-one-out-error in the training set (k -NN), a binary decision tree with maximum entropy node splitting and early pruning (Tree), see [2] and [3], Parzen density estimation with built-in optimization of the smoothing parameter over the leave-one-out likelihood estimation on the training set (Parz), see [1], a neural net with 3 hidden units trained by the back-propagation rule with a variable step-size (BPNC) and a neural net with 3 hidden units trained by the Levenberg-Marquardt rule (LMNC). We used the standard Matlab routines and trained them until no error decrease was found over the last $n/2$ steps.

The two neural classifiers perform bad, especially the training by the Levenberg-Marquardt rule. Inspection of what happened revealed that in a number of experiments training didn't start at all and was terminated due to a constant error (we used the standard Matlab Neural Network routines).

Confidences computed for the max-combination rule are sometimes very bad (often $\rho > 1$) and are omitted from some figures. This behaviour can be understood by observing that by maximizing over a large set of different classifiers for both classes large posterior probabilities might be obtained. After normalization (the sum of probabilities should be one) the confidences generally tend in the direction of $p = 0.5$, which causes a large error contribution.

It is interesting to see what the classifier combining rules do. In our experiments we combined a set of different and not so different classifiers. So averaging and multiplying posterior probabilities are theoretically not applicable as they are based on independence assumptions, see [4]. This also holds for the medium, being a robust estimator of the mean. This combination rule, however, does relatively well. The minimum and the maximum rule are in fact the best rules for a situation of dependent classifiers. How to combine the input confidences to a reliable output confidence should be studied further.

Table 1. Classification errors, confidences and confidence errors for classifiers trained with 20 objects/class.

Method	2D Gauss			10D Gauss			Blood		
	ϵ	q	ρ	ϵ	q	ρ	ϵ	q	ρ
NM	0.220	0.177	0.30	0.315	0.207	0.16	0.102	0.083	0.31
Fish	0.176	0.133	0.25	0.068	0.053	0.28	0.130	0.090	0.19
BayU	0.142	0.091	0.14	0.071	0.056	0.29	0.066	0.065	0.49
BayQ	0.042	0.035	0.33	0.100	0.079	0.29	0.071	0.066	0.43
1-NN	0.076	0.055	0.22	0.177	0.169	0.45	0.175	0.141	0.31
k-NN	0.086	0.071	0.33	0.173	0.139	0.30	0.119	0.116	0.48
Tree	0.125	0.078	0.12	0.106	0.083	0.29	0.094	0.119	0.76
Parz	0.092	0.061	0.16	0.141	0.110	0.28	0.095	0.092	0.47
BPXNC	0.097	0.101	0.54	0.081	0.062	0.26	0.125	0.098	0.29
LMNC	0.419	0.346	0.33	0.441	0.329	0.25	0.376	0.313	0.33
Max	0.075	0.156	1.58	0.090	0.166	1.34	0.191	0.157	0.32
Med	0.082	0.060	0.23	0.082	0.063	0.27	0.076	0.075	0.48
Min	0.075	0.058	0.27	0.090	0.071	0.29	0.191	0.150	0.29
Mean	0.076	0.071	0.43	0.080	0.079	0.49	0.074	0.085	0.65
Prod	0.071	0.050	0.21	0.081	0.059	0.22	0.110	0.106	0.46

5 Conclusion

The main purpose of our study was to show that classifiers can be more informative than generating class labels. Classifier conditional posterior probabilities can be consistently computed for almost any classifier and can be used for obtaining classification confidences, rejects and as inputs for classifier combination rules. The estimation of posterior probabilities for the outputs of combination rules, in particular the maximum rule, should be further investigated.

6 Acknowledgment

This work is supported by the Foundation for Applied Sciences (STW) and the Dutch Organization for Scientific Research (NWO).

7 References

- [1]R.P.W. Duin, On the choice of the smoothing parameters for Parzen estimators of probability density functions, *IEEE Trans. Computers*, vol. C-25, no. 11, 1976, Nov., 1175-1179.
- [2]J.R. Quinlan, Induction of decision trees, *Machine Learning*, vol. 1, pp. 81 - 106, 1986.
- [3]J.R. Quinlan, Simplifying decision trees, *Int. J. Man - Machine Studies*, vol. 27, pp. 221-234, 1987.
- [4]J. Kittler, M. Hatef, R.P.W. Duin, and J. Matas, On Combining Classifiers, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, 1998.
- [5]D.M.J. Tax, M. van Breukelen, R.P.W. Duin, and J. Kittler, Combining multiple classifiers by averaging or by multiplying?, submitted, september 1997.
- [6]J. A. Anderson, Logistic discrimination, in: P. R. Krishnaiah and L. N. Kanal (eds.), *Handbook of Statistics 2: Classification, Pattern Recognition and Reduction of Dimensionality*, North Holland, Amsterdam, 1982, 169--191.
- [7]R.P.W. Duin, PRTTools, *A Matlab toolbox for pattern recognition, version 2.1*, 1997, see <ftp://ph.tn.tudelft.nl/pub/bob>
- [8]A. Hoekstra, S.A. Tholen, and R.P.W. Duin, Estimating the reliability of neural network classifiers, in: C. von der Malsburg, W. von Seelen, J.C. Vorbruggen, B. Sendhoff (eds.), *Artificial Neural Networks - ICANN'96, Proceedings of the 1996 International Conference on Artificial Neural Networks* (Bochum, Germany, July 16-19, 1996), 53 - 58.
- [9]A. Hoekstra, R.P.W. Duin, and M.A. Kraaijveld, Neural Networks Applied to Data Analysis, in: C.T. Leondes (eds.), *Neural Network Systems Techniques and Applications*, Academic Press, in press.
- [10]Cox, L.H., M.M. Johnson, K. Kafadar (1982), Exposition of statistical graphics technology, *ASA Proceedings Statistical Computation Section*, page 55-56.

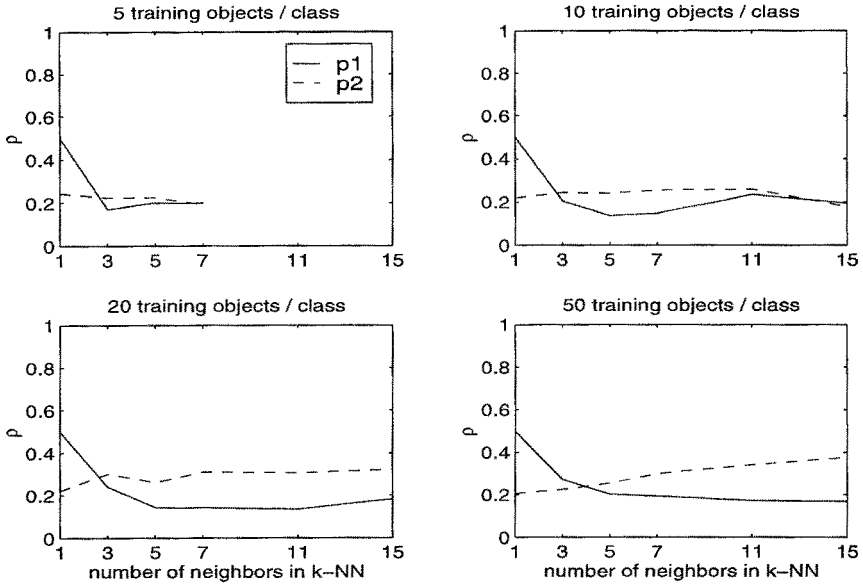


Fig. 2. Normalized confidence error versus numbers of neighbors in k-NN rule for the two estimators (4) and (5) for dataset a.

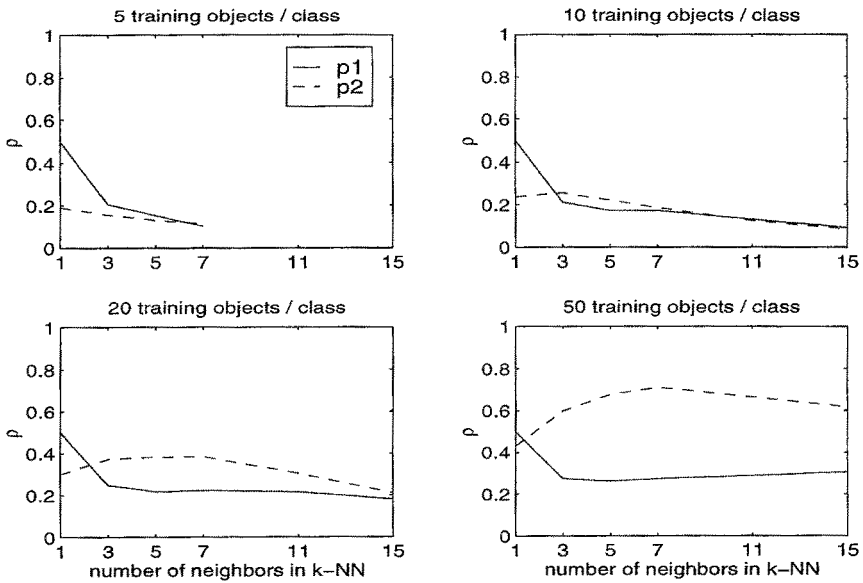


Fig. 3. Normalized confidence error versus numbers of neighbors in k-NN rule for the two estimators (4) and (5) for dataset b.

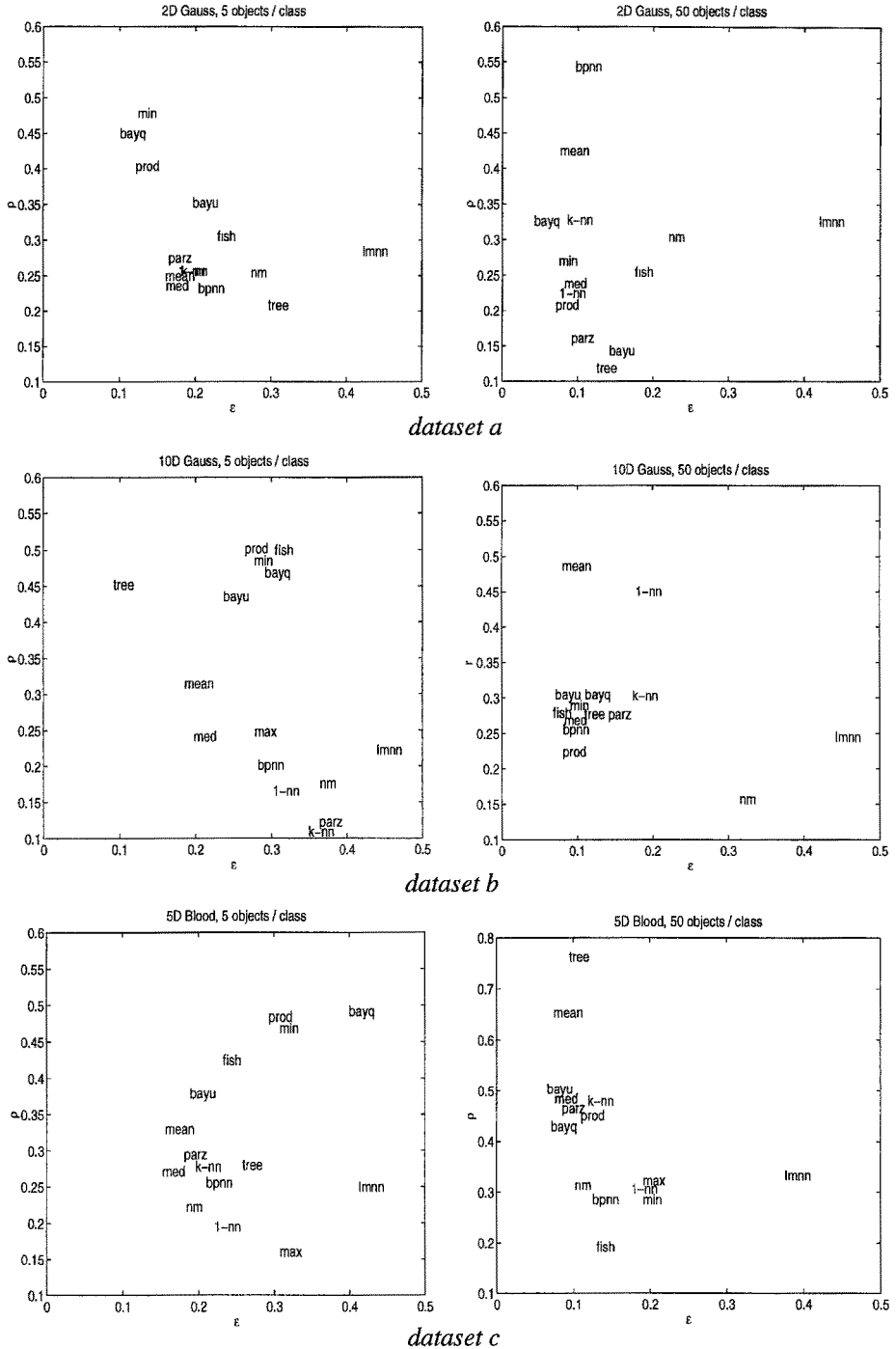


Fig. 4. Scatter plots for the classification error versus normalized confidence errors for the three datasets and for sample sizes 5 and 50 objects per class (note the different scaling of the right plot of dataset c). Figures are slightly adapted to improve readability.