

# *Sequential combining in discriminant analysis*

Tomasz Górecki

Faculty of Mathematics and Computer Science  
Adam Mickiewicz University

European Conference on Data Analysis, 2013

In practice, it often happens that we have a number of base methods of classification. We are not able to clearly determine which method is optimal in the sense of the smallest error rate. Then we have a combined methods that allows us to consolidate information from multiple sources in a better classifier. Ensemble methods use multiple models to obtain better predictive performance than could be obtained from any of the constituent models. An ensemble is a technique for combining many learners in an attempt to produce a strong learner. By combining the outputs of a team of classifiers, we aim at a more accurate decision than that of the single best member of the team. Empirically, ensembles tend to yield better results when there is a significant diversity among the models. It has become clear that for more complicated data sets the traditional set of classifiers can be improved by various types of combining rules. Often none of the base classifiers is powerful enough to distinguish the classes optimally. Different classifiers may be desired for different features, or may reveal different possibilities for separating the data.

The question arises how the base classifiers should be combined. Various possibilities exist, based on fixed rules like mean or product combining and majority voting. In addition one may also train a classifier e.g. BKS method, Wernecke's method or fuzzy integral. The most intuitive one is a simple majority vote, whereby every base classifier computes a class label and the label that receives the most votes is the output of the ensemble. We can also combine the posterior probability of observing a successful treatment. This continuous measure can be combined using further simple functions: mean returns the mean probability of success by the classifiers; min yields the minimal probability of success (a pessimistic measure); max results in the maximal predicted probability of success (an optimistic measure); median returns the median probability.

I propose some fusion of parallel and stacked combining. I am using many different base classifiers (stacked combining) but in subsequent steps I partly change original data set (parallel combining). Six non-trainable combiners was used: mean, minimum, maximum, median, trimmed mean and product and five base methods: linear and quadratic discriminant classifier, naive Bayes classifier, 3 nearest neighbours method and binary decision tree. The essence of my method relies on the sequential joining of additional features, on which the classifiers are trained. These additional features are posterior probabilities obtained from a mean method of combining posterior probabilities from other combining method. All the time the same base methods of classification are used. In each step, however, we add the posterior probabilities obtained on a slightly different data set, because the posterior probabilities varies at each step. We therefore have many classifiers trained on slightly different data sets.

Suppose that a training sample has been collected by sampling from a population  $P$  consisting of  $K$  groups. The  $i$ th observation is a pair denoted by  $(x_i, y_i)$ , where  $x_i$  is a  $d$ -dimensional feature vector and  $y_i$  is the label for recording class membership. An automated classifier can be viewed as a method of estimating the posterior probability of membership. For a given  $x$ , a reasonable classification strategy is to assign  $x$  to that class with the highest posterior probability. This strategy is called the Bayes' rule classifier. We denote the posterior probability of membership by

$$p_k(x) = P(y = k|x).$$

Let  $\pi_k$  be the prior probability. Suppose also that the conditional multivariate probability density for the  $k$ th class is  $f_k(x)$ . Now Bayes' theorem yields the posterior probability

$$p_k(x) = \frac{f_k(x)\pi_k}{\sum_{i=1}^K f_i(x)\pi_i}.$$

Now we assume that all multivariate probability densities are multivariate normal having arbitrary mean vectors and a common covariance matrix. That is, we take  $f_k$  to be an  $N_p(\boldsymbol{\mu}_k, \boldsymbol{\Sigma})$  density. We shall call this model the linear discriminant classifier (LDC). Assuming that class-covariance matrices are different, that is  $f_k$  is an  $N_p(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$  density we obtain quadratic discriminant classifier (QDC).

A naive Bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem with independence assumptions. Simply speaking, we assume that the presence or absence of a particular feature is unrelated to the presence or absence of any other feature, given the class variable. Under this assumption we have:

$$p_k(x) \propto \pi_k \prod_{i=1}^d f_i(x_i),$$

where  $f_i$  is density of variable  $x_i$ . When dealing with continuous data, a typical assumption is that the continuous values associated with each class are distributed according to a normal distribution (one-dimensional). Despite the fact that the independence assumptions are often inaccurate, the naive Bayes classifier can be robust enough to ignore serious deficiencies in its underlying naive probability model.

The three previous classifiers assumed that the features are normally distributed. Most often we do not have sufficient knowledge of the underlying distributions. One of the important nonparametric classifiers is a  $j$ -nearest neighbor classifier ( $j$ NN classifier). The estimator of  $p_k(x)$  produced by the  $j$ NN classifier is the sample proportion of the  $j$ -nearest neighbors belonging to  $G_k$  (objects are assigned to the class having the majority in the  $j$  nearest neighbours in the training set):

$$\hat{p}_k(x) = \frac{1}{j} \sum_{i=1}^N I(\rho(x, x_i) \leq d_j(x)) I(y_i = k), \quad k = 1, \dots, K,$$

where  $I(A)$  is the indicator function of the event  $A$ ,  $d_j(x_0)$  is the  $j$ -th distance from the point  $x$  to the points  $x_1, \dots, x_N$  and  $\rho$  is a given measure. Usually  $\rho$  is an Euclidean metric. For  $j > 2$  posterior probabilities are estimated using the class frequencies in the set of  $j$  neighbours.



Another example of nonparametric classifier is a decision tree. They are widely used for building classifier ensembles. They are intuitive because the decision process can be traced as a sequence of simple decisions. To construct a decision tree we start with the root and continue splitting the tree. This process is repeated on each derived subset in a recursive manner called recursive partitioning. That part is subsequently split into smaller part until a termination criterion is met. To automate the tree construction it is reasonable to choose binary tree, that is, each nonterminal node has exactly two children nodes. Our algorithm computes a binary decision tree. Thresholds are set such that the Gini impurity is minimized in each step. Posterior probabilities are estimated by the class frequencies of the training set in each end node.

In the classifier combination problem with confidence score outputs, inputs to the combiner are the posterior scores belonging to different classes obtained from the base classifiers. Let  $p_k^i$  be the posterior score of class  $i = 1, 2, \dots, K$  obtained from classifier  $k = 1, 2, \dots, L$  for any data instance. Let  $p_k = (p_k^1, p_k^2, \dots, p_k^K)'$ , then the input to the combiner is  $f = (p_1, p_2, \dots, p_L)'$ . Outputs of the combiner are  $K$  different scores representing the degree of support for each class. Let  $r^i$  be the combined score of class  $i$  and let  $r = (r^1, r^2, \dots, r^K)'$ . In general the combiner is defined as a function  $g$  such that  $r = g(f)$ . Label of a data instance  $x$  is assigned as follows:

$$d(x) = \arg \max_{i=1, \dots, K} r^i.$$

For equal priors we have following combining classifiers:

$$r_{\text{mean}}^i(x) = \frac{1}{L} \sum_{k=1}^L p_k^i(x),$$

$$r_{\text{prod}}^i(x) = \prod_{k=1}^L p_k^i(x),$$

$$r_{\text{min}}^i(x) = \min_k p_k^i(x),$$

$$r_{\text{max}}^i(x) = \max_k p_k^i(x),$$

$$r_{\text{med}}^i(x) = \text{median}_k p_k^i(x),$$

The  $n\%$  trimmed (or truncated) mean is obtained by discarding the  $n\%$  lowest and highest probabilities. This combination rule tries to eliminate harms of the outliers in the ensemble for the mean rule. Some classifiers may give unusually high or low scores to a particular class and these scores are not included in the averaging.

The arithmetic mean is most resilient to estimation errors and usually outperforms other classifier combinations schemas. For combination rules based on the sum, such as the arithmetic mean, and for the case of classifiers working on different feature spaces, the arithmetic mean is less sensitive to errors than the geometric mean. The combining rules based on the product give better results when all classifiers produce small errors. If at least one of the classifiers makes large errors then the arithmetic mean rule gives better results. The product rule is sensitive to errors in the posterior probability estimates, and deteriorates more rapidly than the mean rule as the estimation errors increase. The motivation behind minimum rule is to assign the test instance to the class for which there is no base classifier that disagrees on the decision. Performance of the minimum rule tends to increase as the base classifiers are all strong. For the maximum rule, if one base methods insists on a particular class for a given test instance, final decision assigns the test instance to that class; even if all other base methods disagree. Median rule also provide robustness to outliers, like the trimmed mean rule.

## Sequential combining

The method presented here relies on the sequential joining of additional features on which the classifier is trained. Features are posterior probabilities derived from the mean method of combining posterior probabilities. In each step, however, we add the posterior probabilities obtained on a slightly different data set, because the posterior probabilities varies at each step. Some features are fixed (base features of observation), and the new added features change at every step. Each base method gives us the posterior probabilities, these probabilities are combined by six non-trainable combining method giving us new posterior probabilities. Finally this probabilities are combine once again by mean method of combining classifiers. At each step, we replace the previously attached posterior probabilities (resubstitution method of estimating error rate). Schemes will be denoted CLDC, CQDC, CNB, C3NN and CTREE (prefix C). The second problem concerns when to stop the procedure. A very simple criterion was chosen, which makes the algorithm stop if at the next step it does not decrease the classification error (the maximum number of steps was fixed at 30).

Experiments were performed on 22 data sets. The number of classes varies from 2 to 11, the number of features varies from 3 to 33, and the number of instances varies from 150 to 2201. All the data sets originate from the UCI Machine Learning Repository and KEEL data set repository, which are the collections of databases used by the machine learning community for the empirical analysis of machine learning algorithms.

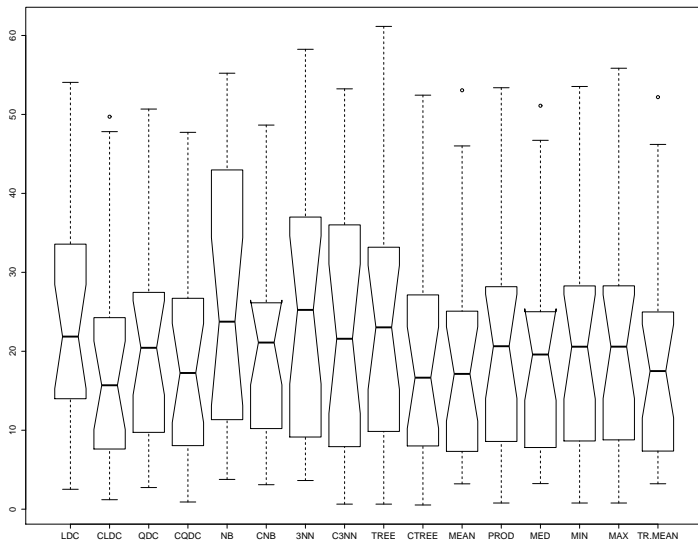
# Data sets and experimental setup

Data set	Number of features	Number of classes	Number of instances	Origin
Australian	14	2	690	KEEL
Balance	4	3	625	UCI
Bands	19	2	365	KEEL
BreastWisconsin	9	2	683	UCI
Car	6	4	1728	UCI
Contraceptive	9	3	1473	KEEL
Diabetes	8	2	768	UCI
Faults	27	7	1941	UCI
German	20	2	1000	UCI
Haberman	3	2	306	KEEL
Ionosphere	33	2	351	UCI
Iris	4	3	<b>150</b>	UCI
Liver	6	2	345	UCI
Mammographic	5	2	830	KEEL
Monk-2	6	2	432	KEEL
Parkinsons	22	2	195	UCI
Thyroid	5	3	215	UCI
Titanic	3	2	<b>2201</b>	KEEL
Vehicle	18	4	846	KEEL
Vowel	10	<b>11</b>	990	UCI
Wdbc	30	2	569	KEEL

A bootstrap method for estimating the classification accuracy was employed. An essential feature of the bootstrap method is that observations of the training sample play the role of a general population and are used to determine the optimistic bias of the resubstitution estimate. A bootstrap training sample is a random sample with replacement from the training sample. A classification rule is designed using a bootstrap training sample and is tested twice. 50 repetitions for each data set were used. The 50 values of classification accuracy are averaged to get the final estimate.



These results (absolute bootstrap error rates) are represented as a box plot. A notches was drawn in each side of the boxes. If the notches of two plots do not overlap this is strong evidence that the two medians differ.



In the following Table are shown relative error rates (in %). For two methods A and B with errors  $\varepsilon_A$  and  $\varepsilon_B$  the relative error rate is

$$\frac{\varepsilon_B - \varepsilon_A}{\varepsilon_A}.$$

The mean ratio of relative error rates is the average (over all data sets) of the relative errors between the pair of compared method. A value of the mean ratio of relative error rates less than 0 represents an improvement due to the base method. We can see that we obtain, in the case of the LDC, the highest average reduction in the relative error rates equals to 22.80. In the case of the QDC we get a much lower average reduction in relative classification error rates equals to 14.07. Similarly for the NB we have 20.57 reduction, for the 3NN method 17.42 and for decision trees 21.68. We see that in each case it is very high reduction of relative error rate.





Data set	LDC vs CLDC	QDC vs CQDC	NB vs CNB	3NN vs C3NN	TREE vs CTREE
Australian	-2.83	-10.67	-16.96	-2.17	-6.05
Balance	-4.85	-4.28	-3.13	-32.93	-44.18
Bands	-7.39	-6.87	-14.52	-2.85	-14.29
Breast_W	-25.09	-13.69	-4.14	-15.78	-27.94
Car	-61.59	-28.12	-45.60	-46.19	-10.24
Contraceptive	-2.66	-2.67	-8.37	-3.69	-2.98
Diabetes	-1.32	-1.94	-4.21	-1.89	-6.09
Faults	-32.64	-19.26	-23.18	-0.70	-15.48
German	-1.69	-2.76	-5.23	-10.27	-9.45
Haberman	-3.83	-3.61	-4.96	-8.12	-10.76
Ionosphere	-54.18	-7.52	-32.35	-50.95	-37.09
Iris	-12.50	-14.12	-25.91	-30.26	-31.24
Liver	-6.13	-13.48	-22.02	-4.47	-9.93
Mammographic	-6.41	-5.49	-4.84	-5.21	-2.96
Monk-2	-94.40	-88.93	-61.68	-90.34	-17.82
Parkinsons	-23.22	-20.76	-19.11	-7.69	-23.63
Tae	-8.04	-6.36	-6.34	-8.60	-14.25
Thyroid	-46.12	-12.95	-17.79	-21.13	-52.55
Titanic	-2.21	-0.12	0.00	22.10	2.82
Vehicle	-12.82	-0.93	-52.67	-2.02	-31.00
Vowel	-72.96	-35.56	-54.75	-11.32	-64.46
Wdbc	-18.72	-9.50	-24.85	-4.49	-41.76
Mean ratio	-22.80	-14.07	-20.57	-17.42	-21.68





The average number of steps after which the method was completed was 2.05 for the CLDC method, 1.91 for CQDC, 1.89 for CNB, 1.97 for C3NN and 2.28 for CTREE. Hence the differences are rather slight. Note that for a fixed threshold of 30, the algorithm always converges to the solution. The largest number of steps which algorithms took to complete was 20 for CQDC, C3NN and CTREE. In addition, the whole procedure works quickly. The operating time of procedures increases proportionally to the square of the number of analyzed models. Because this number is not too large (up to 20, averaging ca. 2), the whole procedure is relatively fast.

	CLDC	CQDC	CNB	C3NN	CTREE
Min number of steps	1	1	1	1	1
Mean number of steps	2.05	1.91	1.89	1.97	2.28
Max number of steps	19	20	15	20	20

Finally, to confirm that the sequential combining classifiers are superior to base methods, we present a statistical comparison of their bootstrap error rates on all 22 data sets.

To statistically compare two classifiers over multiple data sets, Demšar (2006) recommends the Wilcoxon signed-ranks test. The Wilcoxon signed-ranks test is a non-parametric alternative to the paired  $t$ -test, which ranks the differences in performances of two classifiers for each data set, ignoring the signs, and compares the ranks for the positive and the negative differences. In our case we obtain a  $p$ -value equals to 0.0000004768 for LDC vs CLDC, QDC vs CQDC, 3NN vs C3NN and TREE vs CTREE and 0.00006412 for NB vs CNB. We see that sequential combining classifiers are significantly better than base methods at a significance level of  $\alpha = 0.05$  (and even for  $\alpha = 0.01$ ).

-  Alcalá-Fdez, J., Fernandez, A., Luengo, J., Derrac, J., García, S., Sánchez, L., Herrera, F. (2011). KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework. *Journal of Multiple-Valued Logic and Soft Computing* 17(2-3):255–287.
-  Demšar, J. (2006). Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research* 7:1–30.
-  Duin, R., Tax, D. (2000). Experiments with Classifier Combining Rules. *Lecture Notes in Computer Science* 1857:16–29.
-  Frank, A. Asuncion, A. (2010) UCI Machine Learning Repository. <http://archive.ics.uci.edu/ml> Irvine, CA: University of California, School of Information and Computer Science.

-  Górecki, T. (2013). Sequential correction of linear classifiers. *Journal of Applied Statistics* 40(4):763–776.
-  Kuncheva, L. (2004). *Combining Pattern Classifiers*. John Wiley & Sons.
-  Mojirsheibani, M. (2002). A comparison Study of Some Combined Classifiers. *Communications in Statistics - Simulation and Computation* 31(2):245–260.
-  Rokach, L. (2010). Ensemble-based classifiers. *Artificial Intelligence Review* 33:1–39.