Comparing Classifiers and Metaclassifiers

Elio Lozano[1] and Edgar Acuña[2]

[1] University of Puerto Rico at Bayamón, Computer Science Department,
Industrial Minillas 170 Carr 174, Bayamón, Puerto Rico
elio.lozano@upr.edu

[2] University of Puerto Rico at Mayagüez, Mathematics Department,
Calle Post, Mayagüez, Puerto Rico
edgar.acuna@upr.edu

Abstract. A metaclassifier is a technique that integrates multiple base classifiers. In this paper a hybrid meta-classifier algorithm based on generative and non-generative methods is proposed. Five well-know strong classifiers are used for the non-generative method and bagging was used for generative method. The performances of the five base classifiers, their ensembles based on bagging, and the proposed hybrid metaclassifier are compared using classification error rates. Eight different datasets coming from the UCI Machine Learning database repository are used in the experiments.

Keywords: Classifiers, Ensembles.

1 Introduction

Data mining and knowledge discovery play an important role in engineering, scientific, and medical databases. Different classifiers have been designed to solve different problems in this area. However, the performance of some of them is poor. For this reason, ensembles of base classifier algorithms, called metaclassifiers, are considered.

Ensembles of multiple classifiers [12], are found in several fields, such as the combination of estimators in econometrics, evidence in rule-based systems, and multi-sensor data fusion. Meta-classifiers are studied because they improve the efficiency of single classifiers, and also because they are robust.

In this paper a combination of two ensemble methods based on generative and non-generative methods is introduced. Each of these algorithms is based on five different base classifiers learned from centralized datasets. These classifiers are Radial Basis Function networks (RBF), C4.5 decision trees (C45), Kernel Density (KD), Naive Bayes (NB), and K-Nearest Neighbors (KNN). Single bagging for each of them and their combined bagging is carried out. The performance of the proposed algorithm is compared with the five base classifiers and other meta-classifiers. This performance is based on the classification error rate.

This paper is organized as follows: related works to this research are described in section two. Section three describes the ensemble methods and the proposed algorithm. Experimental results are presented in section four. Finally, conclusions are discussed in section five.

2 Related Work

Roli et al. [11] and Tumer and Ghosh [12] presented and analyzed combiners based on order statistics. They concluded that the combiner's robustness helps to improve the performance of certain individual classifiers. Their experimental results showed that, if there is significant variability among the classifiers, the order statistics-based combiners substantially outperform simple combiners.

Duin and Tax [5] concluded that combining classifiers trained in different feature sets is quite useful, especially when the probabilities are well estimated by the classifier in these feature sets. On the other hand, combining different classifiers trained in the same data set may also improve the classifier performance, but it is generally less useful. They concluded that there is no combiner winning rule: mean, median, majority in case of correlated errors, and the product of independent errors perform roughly as expected, but others may be good as well.

Dietterich [4] concluded that in low-noise cases, Adaboost [6] gives a good performance because it is able to optimize the ensemble without over-fitting. However, in high-noise cases, Adaboost puts a large amount of weight in the mislabeled examples, badly over-fitting the classifier. Bagging and randomization perform well in both the noise and noise-free cases because they are focusing in the statistical problem and noise increases this statistical problem. In very large data sets, randomization can be expected to do better than bagging, because bootstrap replications of a large training set are similar to the training set, hence the learned decision tree may not be very diverse. Randomization creates diversity under all conditions, but at the risk of generating low-quality decision trees. The author [4] was interested to see if the local algorithms such as radial basis functions and nearest neighbor methods can be profitably combined via Adaboost to yield interesting new learning algorithms.

Oza et al. [9] surveyed applications of ensemble methods to problems that present difficulties in classification such as: remote sensing, person recognition, one-versus-all recognition, and medicine.

Amin et al. [1] presented ensemble approached in single-layered complex value neural networks to solve classification problems. They applied two ensemble methods based on negative correlation learning and bagging.

Hsieh et al. [8] proposed an ensemble classifier constructed by incorporating data mining techniques, such as associate binning to discretize continuous values, neural networks, support vector machine, and Bayesian networks to augment the ensemble classifier. They applied their ensemble in a credit scoring system to replace one existing hybrid system.

Goumas et al. [7] presented fusion methods of multiple classifiers through multi-modular architectures to improve the classification results and to contribute the robustness of the inspection system which is based on a technology of surface mounted devices.

3 Ensemble Methods

Experiments with classifier combining rules were described by Duin and Tax [5], and Dietterich [4]. The latter used various fixed and trained combining rules. Six methods for fusing multiple classifiers are presented by Roli et al. [11]. They measured classifiers's diversity and performance of such methods.

Meta-learning refers to learning from a prediction of base classifiers in a common validation dataset. The sequence of this process is as follows:

1. Classifiers are trained from the initial training sets.
2. A prediction is generated by the learned classifiers in a separate validation set.
3. A meta-level training set is composed from the validation set and the prediction generated by the classifiers in the validation set.
4. The final classifier (meta-classifier) is trained from the meta-level training set.

Valentini and Masulli [13] quoted two general ensemble Methods: generative and non-generative. The latter doesn't generate new base classifiers, instead it combines base classifiers in a suitable way to find the ensemble. An explanation of both methods is given below.

– Non-generative Methods
  These methods combine a set of base learning algorithms using a combiner module, which depends in its adaptivity of input and output of its base classifiers. If labels or if continuous outputs are hardened, the majority voting among the represented base classifiers are used. Weights can be assigned to each classifier output to optimize the combined classifier of the training set. Ensembles can be based in Bayes rule approach. For this purpose, the Behavior Knowledge Space method considers each possible combination of class labels. This method computes the frequency of each class corresponding to each combination of the classifiers, but this technique requires a huge size of training data. The base learners can be aggregated using operators such as minimum, maximum, average, product, and ordered weight averaging. Another method of combination is using second level learning machine. This learning algorithm takes the base leaner outputs as features in the intermediate space.
– Generative Methods
  Resampling methods may be used to generate different training sets. In order to produce multiple classifiers, base learning algorithms can be applied in these sets. Bagging draws samples with replacement. On the other hand, boosting uses different distribution or weighting over the training examples in each iteration. Another method to get training samples is leaving one disjoint subset out. This method is called cross-validation, which is a technique to sample without replacement. Randomized ensemble methods generate classifiers using random initial values to construct the classifier. For instance, in a radial basis function, the initial weights can be initialized randomly to obtain different classifiers. In this paper a combination of generative and non-generative method is proposed. A comparison between the

single base classifiers, their respective ensembles based on bagging, and the proposed ensemble is carried out.

3.1 Combination of Generative and Non-generative Ensemble

In this paper a combination of two ensemble methods is proposed: a generative method based on majority voting and a non-generative method based on bagging.

Five well-known base classifiers are used to build a meta-classifier algorithm based on majority voting. These classifiers are the C4.5 decision trees (C45) [10], Naive Bayes (NB) from Machine Learning, Kernel Density (KD) from statistics, Radial Basis Functions (RBF) from neural networks, and K-Nearest Neighbors (KNN).

The bagging algorithm was proposed by Breiman [3]. This algorithm consists in taking B bootstrap samples with replacement $\pounds_1, \pounds_2, ..., \pounds_B$, generated from a training sample $\pounds$. A classifier C_i is built for each bootstrap sample $\pounds_i$. Finally, a classifier C_A is generated, containing the most frequent class estimated by the C_i classifier.

In majority voting, an instance is classified in the most frequent class that appears in the classifier output.

Combining bagging and majority voting consists in generating different base classifiers for each bootstrap sample. Then a majority voting method is applied to these classifiers. The final output is taken as the classifier output for each bootstrap sample in the Bagging algorithm (Fig. 1).

```
Input $training\ sample\ \pounds,\ classifier\ C,\ bootstrap\ samples\ B$
$for\ i = 1\ to\ B\ \{$
  $\pounds' = bootstrap\ sample\ from\ \pounds$
  $EC_i = majority\ vote\ \{BC_j(\pounds')\}$
   $where\ BC_j\ is\ a\ base\ Classifier,$
    $j = 1, ..., \#\ of\ base\ classifiers$
$\}$
$EC_A(x) = \underset{y \in Y}{\operatorname{argmax}} \sum_{i:EC_i(x)=y} \mathbf{1},\ (the\ most\ frequent\ class)$
          $Y = \{1, 2, ..., g\}$
Output $Classifier\ EC_A.$
```

Fig. 1. Proposed Ensemble Algorithm

4 Experimental Results

The implementation of the base algorithms and the ensembles were made in the C++ language. For the decision tree classifier, a wrapper of the C4.5 software was used. This software was developed by Quinlan [10] which is available in the author's web page. The RBF classifier uses the CPPLapack library to find the pseudo inverse of a matrix using the general svd procedures from the LAPACK library. This library is an open source C++ wrapper for BLAS and LAPACK library and it is available on `http://cpplapack.sourceforge.net/`

Table 1. Parameters of Base and Ensemble Algorithms

Dataset	iris	diabetes	ionosphere	breawst	bupa	vehicle	segment	landsat
NH	5	5	6	3	9	20	25	36
SCL	1	1	1	1	1	1000	100	10000
NN	4	7	3	7	7	3	3	11

Table 2. Classification Error Rates of Base and Ensemble Algorithms

Dataset	C4.5	RBF	KD	NB	KNN	$B^{C4.5}$	B^{RBF}	B^{KD}	B^{NB}	B^{KNN}	B^1
Breawst	0.013	0.032	0.049	0.037	0.028	0.041	0.032	0.049	0.037	0.029	0.038
Bupa	0.171	0.359	0.359	0.420	0.359	0.335	0.348	0.379	0.414	0.358	0.368
Diabetes	0.259	0.277	0.264	0.246	0.255	0.242	0.257	0.265	0.247	0.261	0.238
Ionosphere	0.125	0.099	0.108	0.262	0.162	0.080	0.100	0.114	0.262	0.162	0.066
Iris	0.080	0.047	0.046	0.053	0.400	0.060	0.038	0.038	0.048	0.053	0.038
Landsat	0.262	0.189	0.134	0.219	0.207	0.211	0.157	0.129	0.219	0.199	0.161
Segment	0.062	0.142	0.134	0.230	0.111	0.056	0.122	0.136	0.230	0.110	0.067
Vehicle	0.271	0.335	0.373	0.508	0.353	0.262	0.245	0.347	0.507	0.357	0.241

Table 3. Ranking of Classifiers and Ensembles

Dataset	1st	2nd	3th	4th	5th	6th	7th	8th	9th	10th	11st
Breawst	C4.5	KNN	B^{KNN}	RBF	B^{RBF}	NB	B^{NB}	B^1	$B^{C4.5}$	KD	B^{KD}
Bupa	C4.5	$B^{C4.5}$	B^{RBF}	B^{KNN}	RBF	KD	KNN	B^1	B^{KD}	B^{NB}	NB
Diabetes	B^1	$B^{C4.5}$	NB	B^{NB}	B^{RBF}	C4.5	KNN	B^{KNN}	KD	B^{KD}	RBF
Ionosphere	B^1	$B^{C4.5}$	RBF	B^{RBF}	KD	B^{KD}	C4.5	KNN	B^{KNN}	NB	B^{NB}
Iris	B^1	B^{RBF}	B^{KD}	KNN	KD	RBF	B^{NB}	NB	B^{KNN}	$B^{C4.5}$	C4.5
Landsat	B^{KD}	KD	B^{RBF}	B^1	RBF	B^{KNN}	KNN	$B^{C4.5}$	NB	B^{NB}	C4.5
Segment	$B^{C4.5}$	C4.5	B^1	B^{KNN}	KNN	B^{RBF}	KD	B^{KD}	RBF	NB	B^{NB}
Vehicle	B^1	B^{RBF}	$B^{C4.5}$	C4.5	RBF	B^{KD}	KNN	B^{KNN}	KD	B^{NB}	NB

Eight data sets from the UCI Machine Learning Database Repository [2] were used for the experiments. These datasets were iris, diabetes, ionosphere, breawst, bupa, vehicle, segment, and landsat. The error rates of classifiers were estimated using 10-fold cross validation technique. Table 2 shows these error rates, which are the average of 10 runs.

The RBF and KNN algorithms have their own parameters for each dataset. These parameters are chosen according to their lowest classification error rate by cross validation. The RBF classifier has hidden nodes, tested from 2 to 30, and a scale parameter used to normalize the values of distance matrices, tested from 1 to 10000. The KNN classifier used the number of neighbors for each dataset. Table 1 shows these parameters for each dataset used in this research.

Table 2 shows the classification error rates for the proposed combined voting algorithm (B^1), five base classifiers (C4.5, RBF, KD, NB, and KNN) and their ensembles based on bagging (B$^{C4.5}$, B^{RBF}, B^{KD}, B^{NB}, B^{KNN}). The results show that for each classifier the bagging algorithm tends to reduce the classification error rate. In almost all datasets, the proposed algorithm gives better results and is more robust when compared to single ones and their ensembles. Table 3 shows a summary of ranking of each base classifier, their ensembles, and the proposed combined voting scheme.

5 Conclusions

In this paper the performances based on classification error rate of five well-known base classifier algorithms (C4.5 decision tree, Naive Bayes, Kernel Density, Radial Basis Functions and K-Nearest Neighbors) were compared, as well as their ensemble bagging algorithms. A new ensemble algorithm (a combination of generated and non generated ensembles) was introduced. The misclassification error rate, based on ten-fold cross validation, was used to compare the performances of the base classifiers and the ensembles. The proposed algorithm was ranked first for diabetes, ionosphere, iris, and vehicle; third for segment; fourth for landsat; and eighth for bupa and breawst datasets. It can be concluded that the proposed algorithm yielded better results for almost all datasets, and it was robust compared to the single ones and other ensemble algorithms.

Acknowledgments. The authors would like to acknowledge

1. The University of Puerto Rico at Bayamón supported in part this work
2. The High Performance Computing Facilities of the University of Puerto Rico at Rio Piedras for providing us the computing equipment to realize the experiments.
3. Prof. Rose Hernández (English Department of UPRB) and Prof. Antonio Huertas (Computer Science Department of UPRB) for reviewing this article.

References

1. Amin, F., Islam, M., Murase, K.: Ensemble of single-layered complex-valued neural networks for classification tasks. Neurocomputing 72, 2227–2234 (2009)
2. Blake, C.L., Mertz, C.J., Newman, D.J., Hettich, C.L.: UCI Repository of machine learning databases. University of California, Department of Information and Computer Science, Irvine, CA (1998), `http://www.ics.uci.edu/~mlearn/MLRepository.html`
3. Breiman, L.: Bagging predictors. Machine Learning, Technical Report. Department of Statistics, University of California (1994)
4. Dietterich, T.: Ensemble Methods in Machine Learning. In: Kittler, J., Roli, F. (eds.) MCS 2000. LNCS, vol. 1857, pp. 1–15. Springer, Heidelberg (2000)
5. Duin, W., Tax, D.: Experiments with Classifier Combining Rules. In: Pattern Recognition Group, pp. 16–29. University of Technology and Springer, The Netherlands, Heidelberg (2000)

6. Freund, Y., Schapire, R.: Experiments with a new booosting algorithm. In: Proceedings of the Thirteenth International Conference on Machine Learning, pp. 148–156. Morgan Kauffman, San Francisco (1996)
7. Goumas, S.K., Dimou, I.N., Zervakis, M.E.: Combination of multiple classifiers for post-placement quality inspection of components: A comparative study. Information Fusion 11(2), 149–162 (2010)
8. Hsieh, N.C., Hung, L.P.: A data driven ensemble classifier for credit scoring analysis. Expert Systems with Applications 37(1), 534–545 (2010)
9. Oza, N.C., Tumer, K.: Classifier ensembles: Select real-world applications. Information Fusion 9(1), 4–20 (2008)
10. Quinlan, R.: C4.5 Programs for Machine Learning. Published by Morgan Kaufmann Series in Machine Learning (1993)
11. Roli, F., Giacinto, G., Vernazza, G.: Methods for Designing Multiple Classifier Systems. In: Kittler, J., Roli, F. (eds.) MCS 2001. LNCS, vol. 2096, pp. 78–87. Springer, Heidelberg (2001)
12. Tumer, K., Ghosh, J.: Robust Order Statistics-based ensembles for distributed data mining. In: Advances in Distributd and Parallel Knowledge Discovery. AAAI Press/The MIT Press (2000)
13. Valentini, G., Masulli, F.: Ensembles of Learning Machines. In: Marinaro, M., Tagliaferri, R. (eds.) WIRN 2002. LNCS, vol. 2486, pp. 3–19. Springer, Heidelberg (2002)

Appendix

In this paper, five base classifiers are used in order to build a meta-classifier algorithm. They are the C4.5 decision trees [10], Naive Bayes from Machine Learning, Kernel density from statistics, radial basis functions from neural networks, and K-nearest neighbors.

These algorithms are the following:

A The C4.5 algorithm

It is a decision tree algorithm introduced by Quinlan [10]. Given a training sample with known labels; this algorithm constructs a decision tree. On each node a test for each attribute is made. Finally, given an input vector $\boldsymbol{x}$ (unlabeled) the decision tree determines to which class is assigned.

The following steps explain the C4.5 algorithm. Let T be the training sample and let $C_1, C_2, ..., C_k$ be the set of possible classes of the instances in T. The decision tree construction is as follows:

1. If T contains instances belonging to a single class, then the decision tree for T is a leaf identifying a class C_j.
2. If T doesn't contain any samples, then T is a leaf.
3. If T contains instances with mixture classes. T is partitioned into T_i (mixture classes). The decision tree of T consists in a decision node.
4. The same process of tree construction is applied recursively to each no leaf node.

In step 3 the criterion to select an attribute is given by the info-gain measure.

$$Info(T) = -\sum_{i=1}^{k}((freq(C_i, T)/|T|)log_2(freq(C_i, T)/|T|)$$

$$Info_X(T) = -\sum_{i=1}^{n}((|T_i|/|T|)Info(T_i))$$

$$Gain(X) = Info(T) - Info_X(T)$$

Where X is a vector of instance. For discrete attributes, frequencies are used to find the maximum info-gain. For continuous attributes binary tests $Y \leq Z$ and $Y \geq Z$ are defined. In the last case the training instances are first sorted; there are only m-1 possible splits, each of them should be examined. Finally the value with the highest information gain is selected.

B Radial Basis Function Networks

The radial basis functions (RBF neural networks) are defined as the combination of radially symmetric linear basic functions. These functions transform an input $\boldsymbol{x} \in R^p$ in a C dimensional space, it is

$$g_j(\boldsymbol{x}) = \sum_{i=1}^{m} w_{ij}\phi_i(\|\boldsymbol{x} - \mu_i\|) + w_{j0}$$

the parameters w_{ij} ($j = 1, ..., C$, C = number of classes) are called the weights and μ_i the centers ($i = 1, ..., m$).

For instance two kind of basic functions are: Thin plate $\phi(z) = z^2 log(z)$ and gaussian $\phi(z) = exp(-z^2)$.

The centers u_i can be obtained by the following procedures: 1) Random selection. 2) Using clustering algorithms like k-means. 3) Gaussian mixtures. 4) K-nearest neighbors.

The weights w_{ij} can be found using minimum least squares procedure.

The classification rule using radial basis function is: Assign $\boldsymbol{x}$ to the class C_i if:

$$g_i(x) = \max_j g_j(x) \quad i = 1, ..., C$$

$\boldsymbol{x}$ is assigned to the class for which the discriminant function has the largest value.

C The Kernel Density Classifier

Given a univariate data set x_1, ... ,x_n; its empirical distribution function can be written as:

$$\hat{F}(x) = \frac{\#\text{ observations} \leq x}{n} \qquad (1)$$

Since the density function of a random variable X is the derivative of its distribution function.

In the multivariate case ($\boldsymbol{x} \in R^p$), the estimation of the density function is given by:

$$\hat{f}(\boldsymbol{x}) = \frac{1}{nh_1h_2 \cdots h_p} \sum_{i=1}^{n} K_p \left(\frac{x_1 - x_{i1}}{h_1}, ..., \frac{x_p - x_{ip}}{h_p} \right) \tag{2}$$

Where the bandwidth parameters are estimated by:

$$h_{opt} = s \left\{ \frac{4}{n(p+2)} \right\}^{\frac{1}{p+4}} \tag{3}$$

The kernel product estimator is defined by:

$$K_p(\boldsymbol{x}) = \prod_{v=1}^{p} K(x_v) \tag{4}$$

Where $K(\cdot)$ is a univariate density function. The estimation of density using a variable bandwidth is:

$$\hat{f}(\boldsymbol{x}) = \frac{1}{nh_1h_2...h_p} \sum_{i=1}^{n} \left(\frac{1}{\lambda_i} \right)^p \prod_{v=1}^{p} K \left(\frac{x_v - x_{iv}}{h_v \lambda_i} \right) \tag{5}$$

Where λ_i is calculated as in the univariate case.

An object x is assigned to the class i where the $\pi_i \hat{f}(x/C_i)$ is maximum (the π_is are the priors, and $\hat{f}(x/C_i)$ is the class conditional function estimated by the kernel product).

D The K-Nearest Neighbors Classifier

The probability that a point x falls in a volume V centered at a point x is given by

$$\theta = \int_{V(x)} p(x) dx$$

The integration is taken over the volume V. For small samples $\theta \sim p(x)V$, the probability θ may be approximated by the proportion of samples falling within V. If k is the number of instances, out of total n, falling within V, then $\theta \sim k/n$. Now the density can be approximated by:

$$p(x) = \frac{k}{nV}$$

If x_k is the kth nearest neighbor point of x, then V may be taken to be a sphere, centered at x, of radius $\|x - x_k\|$ (the volume of a sphere in n dimension is $2r^n \pi^{n/2}/n\Gamma(n/2)$, where $\Gamma(x)$ denotes the gamma function).

The classification rule of the k-nearest neighbor algorithm is as follows: Given an instance of testing data sample, k nearest neighbors of a training data is computed first. Then, the testing instance is assigned to the most similar class of its k nearest neighbors.

E The Naive Bayes Classifier

Naive Bayes classifier relies in the classical Bayes theorem. The class posterior probability given a feature vector $\boldsymbol{x}$, is $f_i(\boldsymbol{x}) = P(C = i|X = \boldsymbol{x})$. But, $P(C = i|X = \boldsymbol{x}) = \frac{P(X = \boldsymbol{x}|C = i)P(C = i)}{P(X = \boldsymbol{x})}$ by Bayes theorem. Therefore, $f_i(\boldsymbol{x}) \propto P(X = \boldsymbol{x}|C = i)P(C = i)$. The Bayesian classifier is defined as:

$$h(\boldsymbol{x}) = arg \max_i P(X = \boldsymbol{x}|C = i)P(C = i) \quad i = 1, ..., g(\# \; of \; classes)$$

When the feature space is high dimensional, the Naive Bayes classifier assumes that features are independent. Therefore, the discriminant function is given by:

$$f_i^{NB}(x) = \prod_{j=1}^{n} P(X_j = x_j|C = i)P(C = i) \quad n : number \; of \; features$$