Ensemble Learning Based on Multi-Task Class Labels

Qing Wang[1,2] and Liang Zhang[1]

[1] School of Computer Science, Fudan University, Shanghai, China
{wangqing,lzhang}@fudan.edu.cn
[2] School of Management Science and Engineering, Anhui University of Technology, Ma'anshan, China

Abstract. It is well known that diversity among component classifiers is crucial for constructing a strong ensemble. Most existing ensemble methods achieve this goal through resampling the training instances or input features. Inspired by MTForest and AODE that enumerate each input attribute together with the class attribute to create different component classifiers in the ensemble. In this paper, we propose a novel general ensemble method based on manipulating the class labels. It generates different biased new class labels through the Cartesian product of the class attribute and each input attribute, and then builds a component classifier for each of them. Extensive experiments, using decision tree and naive Bayes as base classifier respectively, show that the accuracy of our method is comparable to state-of-the-art ensemble methods. Finally, the bias-variance decomposition results reveal that the success of our method mainly lies in that it can significantly reduce the bias of the base learner.

1 Introduction

Ensemble learning algorithms train multiple base learners and then combine their predictions to make final decision. Since the generalization ability of an ensemble could be significantly better than that of a single learner, studying the methods for constructing good ensembles has attracted a lot of attentions in machine learning literature during the past decade [17]. Generally, the design of a classifier ensemble contains two subsequent steps, i.e. constructing multiple component classifiers and then combining their predictions.

It is well-recognized that in order to get a strong ensemble, the component classifiers should be with high accuracy as well as high diversity [9,22]. Most existing ensemble methods achieve this goal through resampling the training instances or input features. Among them, Bagging [3], Boosting [10], and Random Subspace [14] are three general techniques widely used in many applications. Both Bagging and Boosting train base classifiers by resampling training instances, while Random Subspace trains base classifiers by using different random subsets of input features.

Another general method for constructing classifier ensemble is to manipulate the class labels that given to the base classifier, i.e. by transforming the learning problem into a collection of related learning problems that use the same input instances but different assignment of the class labels. Most existing methods of this type achieve this by converting the given learning problem into a collection of different but related binary

M.J. Zaki et al. (Eds.): PAKDD 2010, Part II, LNAI 6119, pp. 464–475, 2010.

learning problems and then learning a component classifier for each of them. Representative examples include Error-Correcting Output Codes (ECOC)[8] and pairwise ensemble(also known as round robin ensemble) [12], etc. The main deficiency of existing ensemble methods based on manipulating the class labels lies in that, first, they can only be applied to multi-class learning problem; second, although they often can improve the classification accuracy of the base classifiers, their performance gain often is not as large as AdaBoost [12].

Inspired by MTForest[19] and AODE[20] that enumerate each input attribute together with the class attribute to create different component classifiers in the ensemble. In this paper, we propose a novel way to manipulate the class labels for constructing strong ensemble of classifiers by generating different biased new class labels through the Cartesian product of class attribute and each input attribute. This method can be applied to both binary and multi-class learning problems. Extensive experiments show that its performance is superior to AdaBoost. Bias-variance decomposition shows that, the success of it mainly owe to its ability to significantly reduce the bias of the base learner.

The rest of this paper is organized as follows. In section 2, we introduce the background and give a brief review on related work. In section 3, we propose the MACLEN (Multi-tAsk Class Labels based ENsemble) method. Then we report our experimental results in section 4. Finally, we conclude the paper in section 5.

2 Background and Related Work

2.1 Multi-Task Learning

Multi-Task Learning (MTL) trains multiple tasks simultaneously while using a shared representation and has been the focus of much interest in machine learning community over the past decade. It has been empirically [5] as well as theoretically [1] shown can often significantly improve performance relative to learning each task independently. When the training signals are for multiple tasks other than the main task, from the point of view of the main task, the other tasks are serving as a bias [5]. This multi-task bias causes the learner to prefer hypotheses that can explain more than one task, i.e. it must be biased to prefer hypotheses that have utility across these multiple tasks.

In most machine learning applications, however, we are only given the training data which is composed of input attributes and class attribute (main task) and we do not have any other related tasks information. So how to derive related tasks from the given data is crucial for utilizing MTL paradigm to improve the generalization performance. It has been shown in [6] that some of the attributes that attribute selection process discards can beneficially be used as extra related tasks for inductive bias transfer.

2.2 MTForest

Because in multi-task learning extra task is served as additional inductive bias, an ensemble can be constructed by using different extra task to bias each component learner in the ensemble so as to generate different component learners [19]. The multi-task

learning theory reveals that the component learner will often be with higher accuracy if the extra task is related to the main task and the component learner will be with diversity if the each extra task represents different bias.

Inspired by this, we propose the MTForest [19] algorithm which enumerates each input attribute as extra task to introduce different additional inductive bias to generate component decision trees in the ensemble. In MTForest, the construction of each component decision trees is related to both the class attribute and the given input attribute. The learning process is similar to standard C4.5 decision tree learning algorithm except that the Information Gain criteria of each split S_i is calculated by combine the class attribute and the given input attribute, showing below:

$$\text{MTIG}(S_i) = \text{ClassAttributeIG}(S_i) + weight * \text{InputAttributeIG}(S_i)$$

The prediction of the ensemble is produced by aggregating the predictions of all these component decision trees. Experimental results show that MTForest can achieve significant improvement over Bagging and is robust to noise.

2.3 Averaged One-Dependence Estimators

Naive Bayes is an efficient and effective learning algorithm. Denote an instance x as a vector $\langle a_1, a_2, ..., a_n \rangle$, where a_i is the value of i-th attribute, and c is the class value . Naive Bayes simply assumes that the attributes are independent given the class, i.e.

$$P(x|c) = \prod_{i=1}^{n} P(a_i|c)$$

The conditional independence assumption unable to capture important attribute dependence information which sometimes hamper the performance of it seriously. So it needs to relax the assumption effectively to improve its classification performance.

In last fewer years, numerous semi-naive Bayesian classifiers that try to exploit attribute dependencies in moderate orders have been proposed and demonstrate improved performance on naive Bayes. Among those classifiers, approaches that utilize ODE(One-Dependence Estimator) have demonstrated remarkable performance [11,15,18,20]. Representative examples include TAN and SPTAN[11], HNB[15], SNODE[18] AODE and its variants, etc. ODE paradigm learning restricts that each attribute can only depend on one parent in addition to the class, and thus it follows that

$$P(x|c) = \prod_{i=1}^{n} P(a_i|pa(i), c)$$

where $pa(i)$ denotes the parent attribute of the i-th attribute, and is determined in the structure learning process.

Averaged One-Dependence Estimators(AODE) [20] is the ensemble technique of utilizing ODE. For simplicity and to avoid structure learning, AODE built an ODE for each attribute, in which the attribute is simply set to be the parent of all other attributes. The final prediction is produced by aggregating the predictions of all these ODEs. A lot of empirical study show that AODE can achieve significant improvement over naive Bayes and AdaBoost [15,20].

3 The MACLEN Method

The success of MTForest and AODE show that enumerating each input attribute together with the class attribute to create different component classifier is a powerful way to construct strong ensemble. However, MTForest and AODE are specifically for ensembling decision tree and naive bayes, respectively. Both of them need to revise the base classifiers to incorporate the information from the given input attribute. While most general ensemble method, such as Bagging, AdaBoost, Random Subspace, is transparent to the base classifier and can easily be applied to any base classifier. Therefore, it will be interesting to ask whether there exists another way to incorporate the information of the input attribute into the base classifier while is transparent to the base classifier.

In this and subsequent sections, we will argue and show that this can be achieved based on generating new class labels through the Cartesian product of the input attribute and class attribute. Since every new class label contains the information of both the class and input attribute, we call it multi-task class label and the ensemble method as MACLEN (Multi-tAsk Class Labels based ENsemble).

3.1 MACLEN: Algorithm Definition

In this subsection, we present the MACLEN method in detail. Assume that the instance in the training data set is represented by $n+1$-dimension vector $\langle A_1, A_2, ..., A_n, C\rangle$ where A_i is the i-th input attribute and C is the class attribute. An instance x is represented by a vector $\langle a_1, a_2, ..., a_n, c\rangle$, where a_i is the value of attribute A_i and c is the value of class attribute C respectively. For the sake of simplicity, in this paper, we suppose that all the input attributes are discrete.

In MACLEN (see Algorithm 1), an ensemble is constructed by using each input attribute together with the class attribute to generate different but related learning problems independently. When given an input attribute A_i, a new attribute C_i^* is constructed whose values are the Cartesian product of class attribute and this attribute. We then create a new data representation from the original input data by removing this attribute and class attribute and setting the new attribute C_i^* as the class attribute, i.e. the new data representation is $\langle A_1, ..., A_{i-1}, A_{i+1}, ..., A_n, C_i^*\rangle$. Next, each instance $x : \langle a_1, a_2, ..., a_n, c\rangle$ in the original input is transformed into the new data representation by deleting the value a_i and setting the class value as ca_i, i.e. the new transformed instance is $\langle a_1, ..., a_{i-1}, a_{i+1}..., a_n, ca_i\rangle$. Finally, this new relabeled data set is given to the base learning algorithm, which constructs a classifier h_i. By enumerating each input attribute, we obtain an ensemble of n component classifiers $\{h_1, h_2, ... , h_n\}$.

To classify an unlabeled instance, $x : \langle a_1, a_2, ..., a_n\rangle$, we employ the following method. Each component classifier h_i in the ensemble provides probabilities for the class membership of x while the possible class labels is C_i^*. Since the probability output of h_i can be seen as the joint probability distribution of class attribute C and attribute A_i, we can derive the probability of x belonging to original class label c using the *marginal probability*, i.e.:

$$\hat{P}_{h_i}(c|x) = \sum_{a \in A_i} \hat{P}^*_{h_i}(ca|x) \tag{1}$$

Algorithm 1. MACLEN

Input: *BaseClassifier* - the base learning algorithm.
T - training data set in which instance is represented by $n+1$-dimension vector $\langle A_1, A_2, ..., A_n, C\rangle$.
Output: A ensemble of base classifiers *E*
E={};
for each attribute A_i in the input
1. Construct a new attribute C_i^* whose values are the Cartesian product of class attribute C and attributeA_i.
2. Generate a new data representation from the original input by removing attribute A_i and class attribute C and setting the new attribute C_i^* as class attribute.
3. Generate a new data set T^* by transforming each original input instance x in T into the new data representation.
4. h_i=*BaseClassifier*(T^*).
5. *E*=*E* $\bigcup$ h_i.

return *E*

where $\hat{P}_{h_i}(c|x)$ is the estimated probability of instance *x* belonging to original class label *c* according to classifier h_i, and $\hat{P}^*_{h_i}(ca|x)$ is the estimated probability of *x* belonging to new class label ca according to classifier h_i. Furthermore, if the value of attribute A_i is given (denoted as a_i), we can also derive the probability of *x* belonging to original class label *c* using the *conditional probability*, i.e.:

$$\hat{P}_{h_i}(c|x) = \frac{\hat{P}^*_{h_i}(ca_i|x)}{\sum_{y\in C}\hat{P}^*_{h_i}(ya_i|x)} \tag{2}$$

Compared to marginal probability (Eq.1), the conditional probability (Eq.2) can utilize extra information about the value of attribute A_i. So, in our implementation, we first choose Equation (2) to calculate the class membership probabilities of each component classifier, only when the Equation (2) is undefined [1], we then turn to use Equation (1) to calculate the class membership probabilities.

Then we compute the class membership probabilities for the entire ensemble as:

$$\hat{P}(c|x) = \frac{\sum_{i=1}^{n}(\hat{P}_{h_i}(c|x))}{n} \tag{3}$$

where $\hat{P}(c|x)$ is the probability of *x* belonging to class *c*. At last, we select the most probable class as the label for *x*, i.e. $E(x) = \arg\max_{c\in C}\hat{P}(c|x)$.

It is noteworthy that for MTCLAN, it enumerates each input attribute to generate each component classifiers in the ensemble, hence its ensemble size is equal to the number of input attributes which is same as MTForest and AODE but is different from most other ensemble methods such as Bagging and Boosting that need to specify the ensemble size. In addition, the building process of each component classifiers do not depend on each other, so MACLEN can easily be parallelized.

[1] The denominator of Equation (2) is 0 (For example, when decision tree is chosen as base classifier and some leafs do not contain revelent labels) or the value of attribute A_i is missing.

To illustrate this algorithm, consider the learning problem described in UCI data set *adult* [2]. In this problem, we are given 14 attributes representing 14 different aspect information of a person, such as the *age*, *martial-status*, *workclass*, *education*, *occupation*, *race*, *sex*, and so on, and the class attribute indicating whether the annual income of this person is larger than 50*k* or not. Our goal is to build a predictive model from the labeled training data which can be used to predict the income type of an unlabeled person given the values of these 14 attributes. In MACLEN, we use each of these 14 input attributes to generate different but related learning problems. For example, if attribute *sex* whose values are $\{male, female\}$ is chosen, we construct a new attribute as the class attribute whose values are $\{male :> 50k, male :\leq 50k, female :> 50k, female :\leq 50k\}$. This new learning problem is different from the original problem because it not only has to predict the income type of the person, but also the sex type of the person. Then we can construct a base classifiers h_{sex}. Assume that when to classify an unlabeled example and the given *sex* value is *female*, we can derive the probabilities for the original class membership $> 50k$ and $\leq 50k$ only from the predicted probabilities of new class membership $female :> 50k$ and $female :\leq 50k$ in h_{sex}. Through using different attribute we construct different learning problems which have different class labels, while these learning algorithms are also related to each other since they all have to predict the income type of the person.

3.2 Discussion

In MACLEN, we only deal with discrete input attributes. For numeric input attribute, some discretize method must be used ahead. Since discretize methods based on different split criteria often result in different number of discrete values and different intervals for the same numerical attribute, this provides a way to build several different component classifiers from a single numerical attribute which can be used to enlarge the ensemble size of the MACLEN.

Note that the existing of discrete attributes which have large number of values may hamper the performance of MACLEN. These attributes make the corresponding new class label set very large. Since the given training instance number is fixed, the instance-to-class ratio will decrease dramatically, which likely to result in inaccurate component classifier. So it may be better to cluster the values of these attributes into small number of groups respectively, and assign all the values in the same group the same discrete value. By using different cluster method, we can also build several different component classifiers from a single discrete attribute.

4 Experiments and Results

4.1 Experimental Setup

We conduct experiments under the framework of *Weka*[21]. For the purpose of our study, we use the 30 well-recognized data sets from the UCI repositories [2] which represent a wide range of domains and data characteristics. A brief description of these data sets is shown in Table 1. We adopted the following three steps to preprocess each

Table 1. Description of the data sets used in the experiments

Datasets	Size	Attribute	Classes	Datasets	Size	Attribute	Classes
adult	48842	15	2	ionosphere	351	35	2
anneal	898	39	6	iris	150	5	3
autos	205	26	7	kr-vs-kp	3196	37	2
balance-scale	625	5	3	letter	20000	17	26
breast-cancer	286	10	2	mushroom	8124	23	2
car	1728	7	4	nursery	12960	9	5
colic	368	23	2	primary-tumor	339	18	21
credit-a	690	16	2	segment	2310	20	7
credit-g	1000	21	2	sick	3772	30	2
diabetes	768	9	2	soybean	683	36	19
glass	214	10	7	vehicle	846	19	4
heart-c	303	14	5	vowel	990	14	11
heart-h	294	14	5	waveform	5000	41	3
hepatitis	155	20	2	yeast	1484	10	10
hypothyroid	3772	30	4	zoo	101	18	7

data set. First, missing values in each data set are filled in using the unsupervised filter *ReplaceMissingValues* in *Weka*; Second, numeric attributes are discretized using the supervised filter *Discretize* in *Weka* which use the MDL discretization method; Third, we use the unsupervised filter *Remove* in *Weka* to delete attributes that do not provide any information to the class, two occurred within the 30 data sets, namely *Sequence Name* in data set *yeast* and *Animal* in data set *zoo*.

In our experiments, we compare MACLEN to Bagging, Boosting and Random Forest when using C4.5 as base classifier. It is well recognized that Bagging is often unable to enhance the performance of stable classifiers. So we compare MACLEN to Boosting and AODE when using naive Bayes as base classifier. And we study MTCLAN for naive Bayes using the Laplace estimation and M-estimation to smooth probability esitmation which is denoted as MACLAN(L) and MACLAN(M) , respectively. The C4.5, Bagging, Boosting (we use the multi-class version AdaBoost.M1 [10]), Random Forest, Naive Bayes and AODE are all already implemented in *Weka*, so we only implement our algorithm under the framework of *Weka*. We set the ensemble size as 50 for all compared methods and keep other parameters at their default values in Weka. It is noteworthy that for our method, the ensemble size is just the number of input attributes which is often far smaller than 50 on these data sets. The classification accuracy and standard deviation of each algorithm on a data set was obtained via 10 runs of ten-fold cross validation. Runs with the various algorithms were carried out on the same training sets and evaluated on the same test sets.

To compare two learning algorithms across all these domains, we first adopt the widely used pairwise two-tailed t-test with 95% confidence level to compare our algorithm with other algorithms. Recently, it has been proposed that the best method to compare multiple algorithms over multiple data sets is to compare their overall ranks [7]. So, we also present the overall ranks of our algorithm and other ensemble methods compared. Note that the smaller the rank, the better the method.

4.2 Experimental Results

Table 2 shows the detailed experimental results of the mean classification accuracy and standard deviation of C4.5, Random Forest and three ensemble methods using C4.5 as base classifier on each data set. And the mean values, overall ranks and the pairwise *t*-test results are summarized at the bottom of the table. From Table 2 we can see that MACLEN can achieve substantial improvement over C4.5 on most data set (11 wins and 3 losses) which suggests that MACLEN is potentially a good ensemble technique for decision trees. MACLEN can also gain significantly improvement over Bagging (7 wins and 2 losses) and is comparable to two state-of-the-art ensemble technique for decision trees, AdaBoost (6 wins and 4 losses) and RandomForest (4 wins and 4 losses). The overall rank of MACLEN on these 30 data sets is 2.22 which the smallest among all these ensemble methods.

Table 3 shows the detailed experimental results of the mean classification accuracy and standard deviation of naive Bayes, AODE, MACLAN(L) and MACLAN(M) using naive Bayes as base classifier on each data set. The mean values, overall ranks and the pairwise *t*-test results are also summarized at the bottom of the table. From Table 3, we can see that MACLEN(M) is significantly better than MACLEN(L). This is very likely due to that, compared to original problem, the number of instance belong to each new class label is smaller and the number of class labels is larger. Thus , compared to Laplace estimation, the M-estimation which places more emphasis on data than prior could make probability estimation effectively in this situation. MACLEN can achieve substantial improvement (12 wins and 1 loss, 16 wins and 1 loss, respectively) over naive Bayes and gain improvement over AdaBoost (8 wins and 5 losses, 9 wins and 5 loss, respectvely). Furthermore, MACLEN(M) is comparable to AODE which is the state-of-the-art ensemble method for naive Bayes. Note that MACLEN is a general ensemble method while AODE not.

We also test our method on these 30 UCI data sets under artificial noise in the class labels to study its robustness. Following the method in [4], the noisy version of each training data set is generated by choosing 5% instances and changing their class labels to other incorrect labels randomly. Due to space limited, we do not list the detailed results of the accuracy and standard deviation on each data set here. The experimental results show that MACLEN can significantly outperform AdaBoost (12 wins 1 loss for C4.5 as base classifier, 13 wins 2 losses for naive Bayes as base classifier)in this situation, and is comparable to RandomForest and AODE respectively.

4.3 Bias-Variance Decomposition

To understand the working mechanism of MACLEN, we use the bias-variance decomposition to analysis it. The bias-variance decomposition is a powerful tool for investigating the working mechanism of learning algorithms. Given a learning target and the size of training set, it breaks the expected error of a learning approach into the sum of three non-negative quantities, i.e. the intrinsic noise, the bias, and the variance. At present there exist several kinds of bias-variance decomposition schemes [13]. In this paper, we adopt the one proposed by [16] which has already been implemented in *weka*.

The training data are divided into training and test sets each containing half the data. 50 local training sets are sampled from the training set, each local set containing 50% of

Table 2. Accuracy and standard deviation of C4.5, Bagging, AdaBoost, Random Forest and MACLEN on the 30 UCI data sets. Bottom rows of the table present the mean values, overall ranks and Win-Loss-Tie (*w/l/t*) comparisons between MACLEN against other algorithms using pairwise *t*-tests at 95% significance level, respectively.

Data sets	C4.5	Bagging	AdaBoost	Random Forest	MACLEN
adult	86.74±0.39	86.94±0.36	85.49±0.40	85.59±0.40	85.79±0.37
anneal	98.78±0.91	98.79±0.87	99.61±0.62	99.42±0.84	99.19±0.82
autos	76.39±9.55	83.89±8.51	86.38±6.94	87.06±6.86	87.02±7.64
balance-scale	69.32±3.89	69.26±3.81	69.31±3.90	69.01±3.84	69.39±3.72
breast-cancer	75.26±5.04	73.76±5.85	66.04±8.21	70.37±7.34	67.82±7.22
car	92.22±2.01	93.59±1.79	96.72±1.50	94.42±1.54	94.43±1.71
colic	84.72±5.94	85.10±5.68	79.48±6.36	73.29±5.52	84.48±5.26
credit-a	86.58±3.53	86.17±3.56	82.72±4.36	84.36±4.01	84.71±3.98
credit-g	72.17±3.49	73.66±3.69	71.69±3.98	73.76±3.63	72.59±3.44
diabetes	77.34±4.91	77.33±4.72	77.16±4.38	76.59±4.78	77.19±4.60
glass	75.23±9.46	77.10±9.13	75.28±9.41	76.94±8.07	76.99±9.21
heart-c	77.32±6.20	80.41±6.38	79.57±6.57	80.78±5.83	80.23±5.61
heart-h	80.96±6.91	80.04±7.25	82.34±6.32	81.84±6.35	81.31±6.97
hepatitis	81.32±9.48	83.26±10.18	83.61±8.93	85.52±8.03	83.94±9.74
hypothyroid	99.28±0.42	99.31±0.41	99.50±0.37	99.27±0.40	99.44±0.38
ionosphere	89.49±5.12	91.03±5.33	93.11±4.08	92.99±3.88	91.77±4.49
iris	93.87±4.89	94.47±5.02	94.61±5.42	94.19±5.81	94.87±5.35
kr-vs-kp	99.44±0.37	99.46±0.37	99.60±0.31	99.23±0.46	99.54±0.36
letter	78.75±0.78	81.87±0.95	89.94±0.75	92.01±0.61	90.72±0.68
mushroom	100.0±0.00	100.0±0.00	100.0±0.00	100.0±0.00	100.0±0.00
nursery	97.18±0.46	97.42±0.43	99.75±0.17	99.06±0.33	98.85±0.35
primary-tumor	41.01±6.59	45.19±6.16	42.63±6.61	41.27±6.09	43.16±6.52
segment	95.23±1.37	95.67±1.22	96.51±1.22	96.91±1.05	97.02±1.01
sick	97.82±0.75	97.84±0.76	97.63±0.78	97.70±0.73	97.81±0.76
soybean	92.63±2.71	94.05±2.61	94.04±2.61	93.73±2.69	93.44±2.73
vehicle	70.77±3.86	70.84±4.01	72.53±3.95	73.25±3.72	72.29±4.10
vowel	79.54±4.01	82.41±3.72	86.48±3.25	90.04±3.02	86.59±3.19
waveform	76.36±1.77	78.78±1.77	82.19±1.59	84.47±1.54	83.71±1.62
yeast	59.46±3.40	59.76±3.46	59.46±3.41	59.50±3.42	60.14±3.69
zoo	92.61±7.33	93.10±7.48	96.07±5.89	92.85±7.07	93.41±7.28
mean value	83.26±3.85	84.35±3.84	84.66±3.74	84.85±3.60	84.93±3.76
overall rank	–	2.68	2.57	2.53	2.22
w/l/t	11-3-16	7-2-21	6-4-20	4-4-22	–

the training set, which is 25% of the full data set. A classifier is constructed from each local training set and bias, variance, and error are estimated on the test set. Since the bias-variance evaluation procedure utilizes training sets containing only 25% of each data set, we only do this decomposition on the 12 UCI data set which size is larger than 1000. Table 4 shows the detailed decomposition results of C4.5 , Naive Bayes and MACLEN using each of them as base learners respectively. Note that the actual bias-variance decomposition scheme of [16] generates a bias term that includes the intrinsic noise, so the errors shown in Table 4 are only composed of bias and variance.

Table 3. Accuracy and standard deviation of Naive Bayes, AdaBoost, AODE and MACLEN using Laplace estimation and M-estimation naive Bayes respectively, on the 30 UCI data sets. Bottom rows of the table present the mean values, overall ranks and Win-Loss-Tie (*w/l/t*) comparisons between MACLEN(L) and MACLAN(M) against other algorithms using pairwise *t*-tests at 95% significance level, respectively.

Data sets	Naive Bayes	AdaBoost	AODE	MACLEN(L)	MACLEN(M)
adult	84.07±0.42	85.19±0.41	85.20±0.41	85.04±0.39	85.10±0.37
anneal	96.13±2.16	99.32±0.86	98.05±1.37	96.85±1.93	97.11±1.79
autos	72.30±10.31	81.28±8.45	81.14±8.50	78.45±9.30	81.95±8.35
balance-scale	71.08±4.29	71.08±4.29	69.34±3.82	69.26±3.83	69.21±3.83
breast-cancer	72.94±7.71	68.87±8.64	72.73±7.01	72.35±7.10	70.95±7.69
car	85.46±2.56	90.25±2.48	91.41±2.06	91.51±2.06	91.76±2.04
colic	81.39±5.74	81.28±6.84	82.37±5.72	82.53±5.56	82.39±5.62
credit-a	86.25±4.01	85.83±3.92	86.55±3.82	86.32±3.82	86.23±3.89
credit-g	75.42±3.84	75.22±3.91	76.44±3.89	75.80±3.92	75.84±3.97
diabetes	77.85±4.67	77.84±4.74	78.07±4.56	78.06±4.51	78.02±4.50
glass	74.39±7.95	74.34±8.10	76.49±7.71	75.10±8.19	76.99±8.36
heart-c	83.60±6.42	83.26±6.48	83.26±6.19	83.33±6.25	83.33±6.29
heart-h	84.46±5.92	83.51±6.32	84.43±5.92	84.53±5.89	84.49±5.87
hepatitis	84.22±9.41	85.76±7.93	85.42±8.96	84.61±9.37	85.38±9.24
hypothyroid	98.48±0.59	99.32±0.41	98.74±0.54	98.55±0.56	98.78±0.54
ionosphere	90.77±4.76	93.62±4.08	92.97±4.32	92.99±4.07	93.32±3.83
iris	94.47±5.61	94.53±5.87	93.20±5.76	93.27±5.65	93.33±5.61
kr-vs-kp	87.79±1.91	95.14±1.23	91.03±1.66	89.24±1.86	89.29±1.87
letter	74.00±0.88	74.00±0.88	88.76±0.70	86.65±0.72	88.53±0.67
mushroom	95.52±0.78	100.0±0.00	99.95±0.07	99.35±0.29	99.81±0.16
nursery	90.30±0.72	91.91±0.76	92.73±0.62	92.60±0.62	92.60±0.63
primary-tumor	47.19±6.02	47.19±6.02	47.87±6.37	48.14±6.45	48.64±5.77
segment	91.71±1.68	93.21±1.46	95.77±1.24	95.06±1.40	96.48±1.14
sick	97.10±0.84	97.01±0.81	97.39±0.79	97.29±0.80	97.26±0.81
soybean	92.20±3.23	91.99±3.49	93.31±2.85	92.78±3.01	93.61±2.78
vehicle	62.51±3.81	62.52±3.81	72.31±3.62	71.44±3.46	72.24±3.38
vowel	65.23±4.53	72.08±4.73	80.88±3.81	79.96±4.09	83.72±3.73
waveform	80.72±1.50	80.72±1.50	86.03±1.56	82.25±1.31	82.34±1.30
yeast	59.16±3.80	59.16±3.80	59.72±3.86	59.54±3.85	59.81±3.79
zoo	93.21±7.35	93.61±7.02	94.66±6.38	94.66±6.38	97.21±5.13
mean value	81.66±4.11	82.97±3.97	84.54±3.80	83.92±3.89	84.52±3.76
overall rank	–	3.05	2.07	2.78	2.10
w/l/t	12-1-17	8-5-17	0-8-22	–	–
w/l/t	16-1-13	9-5-16	3-4-23	7-0-23	–

From Table 4, we can see that in most cases, the error reduction of MACLEN compared to its base learners is mainly due to its improvement of the bias. The mean values shown at the bottom of the table also clearly demonstrate this. In summary, the success of MACLEN mainly lies in that it can significantly reduce the bias of the base learner. This suggests that MACLEN is similar to Boosting style algorithm which has been shown that its improvement is mainly due to its ability to reduce the bias of the base learner.

Table 4. Bias-variance decomposition results of C4.5, Naive Bayes and MACLEN using each of them as base classifier respectively for the 12 UCI data sets which size are larger than 1000

Data sets	MACLEN(C4.5)			C4.5			MACLEN(NB)			Naive Bayes		
	Error	Bias	Var	Error	Bias	Var	Error	Bias	Var	Error	Bias	Var
adult	0.152	0.110	0.041	0.141	0.121	0.020	0.153	0.145	0.008	0.162	0.156	0.006
car	0.128	0.065	0.062	0.157	0.092	0.064	0.128	0.069	0.058	0.166	0.098	0.067
credit-g	0.284	0.182	0.101	0.290	0.204	0.085	0.245	0.192	0.052	0.246	0.195	0.050
hypothyroid	0.013	0.008	0.005	0.015	0.012	0.004	0.024	0.020	0.004	0.025	0.021	0.004
kr-vs-kp	0.015	0.008	0.007	0.020	0.011	0.009	0.121	0.094	0.026	0.136	0.109	0.027
letter	0.181	0.087	0.092	0.312	0.164	0.147	0.186	0.139	0.047	0.280	0.228	0.051
mushroom	0.000	0.000	0.000	0.000	0.000	0.000	0.012	0.010	0.002	0.062	0.058	0.003
nursery	0.049	0.021	0.028	0.066	0.038	0.028	0.075	0.063	0.011	0.094	0.083	0.011
segment	0.056	0.035	0.021	0.105	0.064	0.040	0.065	0.045	0.020	0.097	0.073	0.023
sick	0.025	0.019	0.006	0.025	0.019	0.006	0.024	0.021	0.002	0.025	0.022	0.002
waveform	0.237	0.122	0.115	0.281	0.159	0.120	0.179	0.166	0.013	0.191	0.179	0.012
yeast	0.431	0.317	0.113	0.453	0.328	0.124	0.419	0.338	0.080	0.419	0.339	0.079
mean	0.131	0.081	0.049	0.155	0.101	0.054	0.136	0.108	0.027	0.159	0.130	0.028

5 Conclusion

In this paper, we propose the MACLEN method, a new general ensemble method based on manipulating the class labels. It has several appealing properties: first, it can be applied to both binary and multi-class learning problems; second, its performance is superior to AdaBoost; third, it is simple, easy to parallelized and robust to noise. These demonstrate that manipulating class labels is also a general powerful way to generate strong ensemble besides the popular way of resampling the input instances or features.

Acknowledgements

We would like to thank the anonymous reviewers for their helpful comments. This work was supported in part by NKBRP(2005CB321905), NSFC (60873115) and AHEDUSF (2009SQRZ075).

References

1. Baxter, J.: A model for inductive bias learning. Journal of Artificial Intelligence Research 12(2), 149–198 (2000)
2. Blake, C.L., Merz, C.J.: UCI repository of machine learning databases. University of California, Dept. of C.S., `http://www.ics.uci.edu.learn/MLRepository.html`
3. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)
4. Breiman, L.: Random forests. Machine Learning 45(1), 5–32 (2001)
5. Caruana, R.: Multi-task learning. Machine Learning 28(1), 41–75 (1997)
6. Caruana, R., Sa, V.R.: Benefitting from the variables that variable selection discards. Journal of Machine Learning Research 3(2), 1245–1264 (2003)
7. Demsar, J.: Statistical Comparison of Classifiers over Multiple Data Sets. Journal of Machine Learning Research 7, 1–30 (2006)

8. Dietterich, T.G., Bakiri, G.: Solving multiclass learning problems via error-correcting output codes. Journal of Artificial Intelligence Research 2, 263–286 (1995)
9. Dietterich, T.G.: Ensemble Methods in Machine Learning. In: Kittler, J., Roli, F. (eds.) MCS 2000. LNCS, vol. 1857, pp. 1–15. Springer, Heidelberg (2000)
10. Freund, Y., Schapire, R.E.: Experiments with a new boosting algorithm. In: Proceedings of the 13th International Conference on Machine Learning, pp. 123–140 (1996)
11. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian network classifiers. Machine Learning 29, 131–163 (1997)
12. Furnkranz, J.: Pairwise classification as an ensemble technique. In: Proceedings of the 13th European Conference on Machine Learning, pp. 97–110 (2002)
13. Goebel, M., Riddle, P.J., Barley, M.: A unified decomposition of ensemble loss for predicting ensemble performance. In: Proceedings of the 19th International Conference on Machine Learning, pp. 211–218 (2002)
14. Ho, T.K.: The random subspace method for constructing decision forests. IEEE Trans. Pattern Analysis and Machine Intelligence 20(8), 832–844 (1998)
15. Jiang, L., Zhang, H., Cai, Z.: A novel bayes model: hidden naive Bayes. IEEE Trans. Knowledge and Data Engineering 21(10), 1361–1371 (2009)
16. Kohavi, R., Wolpert, D.: Bias plus variance decomposition for zero-one loss functions. In: Proceedings of the 13th International Conf. on Machine Learning, pp. 275–283 (1996)
17. Kuncheva, L.I.: Combining Pattern Classifiers: Methods and Algorithms. John Wiley and Sons, Chichester (2004)
18. Li, N., Yu, Y., Zhou, Z.H.: Semi-naive exploitation of one-dependence estimators. In: Proceedings of the 9th IEEE International Conf. on Data Mining, pp. 278–287 (2009)
19. Wang, Q., Zhang, L., Chi, M.M., Guo, J.K.: MTForest: Ensemble decision trees based on multi-task learning. In: Proceedings of the 18th European Conference on Artificial Intelligence, pp. 122–126 (2008)
20. Webb, G.I., Boughton, J., Wang, Z.: Not so naive bayes: Aggregating one-dependence estimators. Machine Learning 58(1), 5–24 (2005)
21. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann, San Francisco (2000)
22. Zhou, Z.H., Wu, J.X., Tang, W.: Ensembling neural networks: Many could be better than all. Artificial Intelligence 137(1), 239–263 (2002)