Learning from Imbalanced Data Sets with Boosting and Data Generation: The DataBoost-IM Approach

Hongyu Guo
School of Information Technology and Engineering,
University of Ottawa
800 King Edward Road, Ottawa, Ontario, Canada, K1N 6N5
hguo028@site.uottawa.ca

Herna L Viktor
School of Information Technology and Engineering,
University of Ottawa
800 King Edward Road, Ottawa, Ontario, Canada, K1N 6N5
hlviktor@site.uottawa.ca

ABSTRACT

Learning from imbalanced data sets, where the number of examples of one (majority) class is much higher than the others, presents an important challenge to the machine learning community. Traditional machine learning algorithms may be biased towards the majority class, thus producing poor predictive accuracy over the minority class. In this paper, we describe a new approach that combines boosting, an ensemble-based learning algorithm, with data generation to improve the predictive power of classifiers against imbalanced data sets consisting of two classes. In the DataBoost-IM method, hard examples from both the majority and minority classes are identified during execution of the boosting algorithm. Subsequently, the hard examples are used to separately generate synthetic examples for the majority and minority classes. The synthetic data are then added to the original training set, and the class distribution and the total weights of the different classes in the new training set are rebalanced. The DataBoost-IM method was evaluated, in terms of the *F-measures*, *G-mean* and *overall accuracy*, against seventeen highly and moderately imbalanced data sets using decision trees as base classifiers. Our results are promising and show that the DataBoost-IM method compares well in comparison with a base classifier, a standard benchmarking boosting algorithm and three advanced boosting-based algorithms for imbalanced data set. Results indicate that our approach does not sacrifice one class in favor of the other, but produces high predictions against both minority and majority classes.

Keywords

Data mining, Imbalanced data sets, Ensembles of classifiers, Boosting

1. INTRODUCTION

The class imbalance problem corresponds to domains for which one class is represented by a large number of examples while the other is represented by only a few [1]. Many real world applications involve learning from imbalanced sets, such as fraud detection, telecommunications management, oil spill detection and text classification [2]. When learning from imbalanced data sets, machine learning algorithms tend to produce high predictive accuracy over the majority class, but poor predictive accuracy over the minority class [3].

There have been several proposals for coping with imbalanced data sets [1]. Kubat et al. under-sampled examples of the majority class [5]; Ling and Li over-sampled examples of the minority class [3]; Chawla et al. over-sampled the minority class and under-sampled the majority class [2]; Cardie et al. weighted examples in an effort to bias the learning toward the minority class [3]; Joshi et al. evaluated boosting algorithms to classify rare classes [6]; and Chawla et al. combined boosting and synthetic data to improve the prediction of the minority class [7].

Over the past few years, ensembles have emerged as a promising technique with the ability to improve the performance of weak classification algorithms [8, 9]. Ensembles of classifiers consist of a set of individually trained classifiers whose predictions are combined to classify new instances [8, 9]. In particular, boosting is an ensemble method where the performance of weak classifiers is improved by focusing on hard examples which are difficult to classify. Boosting produces a series of classifiers and the outputs of these classifiers are combined using weighted voting in the final prediction of the model [10]. In each step of the series, the training examples are re-weighted and selected based on the performance of earlier classifiers in the training series. This produces a set of "easy" examples with low weights and a set of hard ones with high weights. During each of the iterations, boosting attempts to produce new classifiers that are better able to predict examples for which the previous classifier's performance is poor. This is achieved by concentrating on classifying the hard examples correctly. Recent studies have indicated that boosting algorithm is applicable to a broad spectrum of problems with great success [10, 11].

In this paper, we discuss a novel approach for learning from imbalanced data sets, DataBoost-IM, that combines data generation and boosting procedures to improve the predictive accuracies of both the majority and minority classes, without forgoing one of the two classes. That is, the aim of our approach is to ensure that the resultant predictive accuracies of both classes are high. Our approach differs from prior work in the following ways. Firstly, we separately identify hard examples from, and generate synthetic examples for, the minority as well as the majority classes. Secondly, we generate synthetic examples with *bias* information toward the hard examples on which the next component classifier in the boosting procedures needs to focus. That is, we provide additional knowledge for the majority as well as the minority classes and thus prevent boosting over-emphasizing the hard examples. Thirdly, the class frequencies in the new training set are rebalanced to alleviate the learning algorithm's bias toward the majority class. Rebalancing thus involves the utilization of a reduced number of examples from the majority and minority classes to ensure that both classes are represented during training. Fourthly, the total weights of the

different classes in the new training set are rebalanced to force the boosting algorithm to focus on not only the hard examples, but also the minority class examples. In this way, we focus on improving the predictions of both the minority and majority classes.

This paper is organized as follows. Section 2 describes the DataBoost-IM algorithm. This is followed, in Section 3, with a comparative evaluation of the DataBoost-IM algorithm against seventeen data sets. Finally, Section 4 concludes the paper.

ALGORITHM DataBoost-IM

Input: Sequence of m examples $\langle (x_1, y_1),...,(x_m, y_m) \rangle$ with labels $y_i \in Y = \{1,...,k\}$

Weak learning algorithm **WeakLearn**

Integer $\boldsymbol{T}$ specifying number of iterations

Initialize $D_1(i) = 1/m$ for all i.

Do for $t = 1, 2, \ldots, \boldsymbol{T}$

1. Identify hard examples from the original data set for different classes
2. Generate synthetic data to balance the training knowledge of different classes
3. Add synthetic data to the original training set to form a new training data set
4. Update and balance the total weights of the different classes in the new training data set
5. Call **WeakLearn**, providing it with the new training set with synthetic data and rebalanced weights
6. Get back a hypothesis $h_t : X \rightarrow Y$.
7. Calculate the error of $h_t : e_t = \sum_{i:h_t(x_i) \neq y_i} D_t(i)$ If $e_t > 1/2$, then set $\boldsymbol{T} = t - 1$ and abort loop.
8. Set $b_t = e_t / (1 - e_t)$.
9. Update distribution $D_t : D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} b_t & \leftarrow if h_t(x_i) = y_i \\ 1 & \leftarrow otherwise \end{cases}$, where Z_t is a normalization constant (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis: $h_{fin}(x) = \arg\max_{y \in Y} \sum_{t:h_t(x)=y} \log \frac{1}{b_t}$

Figure 1: Pseudo-code of the DataBoost-IM algorithm

2. DATABOOST-IM ALGORITHM

The DataBoost-IM approach extends our earlier DataBoost algorithm which was successfully used to produce highly accuracy classifiers in balanced domains containing hard to learn examples [13]. In this section, we describe a variation, the DataBoost-IM algorithm, which is applied to imbalanced data sets. This approach extends the original DataBoost algorithm as follows. Firstly, we *separately* identify hard examples from and generate synthetic examples for different classes. Secondly, the class distribution and the total weights of different classes are *rebalanced* to alleviate the learning algorithms' bias toward the majority class**,** by choosing a reduced number of representative (seed) examples from both classes.

Recall that boosting involves the creation of a series of classifiers which aim to correctly classify hard to learn examples, through focusing on these hard examples during training. Following this mechanism, the DataBoost-IM algorithm, as shown in Figure 1, consists of the following three stages. Firstly, each example of the original training set is assigned an equal weight. The original training set is used to train the first classifier of the DataBoost-IM ensembles. Secondly, the hard examples (so-called seed examples) are identified and for each of these seed examples, a set of synthetic examples is generated. During the third stage of the algorithm, the synthetic examples are added to the original training set and the class distribution and the total weights of different classes are rebalanced. The second and third stages of the DataBoost-IM algorithm are re-executed until reaching a user-specified number of iterations or the current component classifier's error rate is worse than a threshold value. Following the AdaBoostM1 ensemble method, this threshold is set to 0.5 [8,9].

The seed selection, data generation and re-balancing process of the DataBoost-IM algorithm are described next. Throughout this discussion, the Hepatitis data set is used as an illustrative example [12]. The Hepatitis data set contains 155 examples of Hepatitis patients, described by 19 continuous and discrete attributes. Of these cases, 123 corresponds to the patients who survived treatment (class 'Live') and 32 examples of mortalities (class 'Die').

2.1 Identify Seed Examples

The aim of the seed selection process is to identify hard examples for both the majority and minority classes. These examples are used as input for the data generation process as discussed in Section 2.2.

The seed examples are selected as follows. Firstly, the examples in the training set (E_{train}) are sorted in descending order, based on their weights. The original training set E_{train} contains N_{maj} examples from the majority class and N_{min} examples from the minority class. The number of examples that is considered to be hard (denoted by N_s) is calculated as (E_{train} *x Err)*, where *Err* is the error rate of the currently trained classifier. Next, the set E_s, which contains the N_s examples with the highest weights in E_{train}, is created. The set E_s consists of two subset of examples E_{smin} and $E_{smaj,}$ i.e. examples from the minority and majority classes, respectively. Here, E_{smin} and E_{smaj} contain N_{smin} and N_{smaj} examples, where $N_{smin} < N_{min}$ and $N_{smaj} < N_{maj}$. We select a number of seed examples of the majority class in E_{smaj} by calculating M_L, which is equal to *min* $(N_{maj}/N_{min,}, N_{smaj})$. Correspondingly, a subset M_S of the minority class examples in $E_{smin,}$ is selected as seeds, where M_S is calculated as *min* $((N_{maj} \text{ x } M_L) / N_{min}, N_{smin})$. These values of M_l and M_s were found, by inspection, to produce data generation set sizes which augment the original training set well.

The final sets of seed examples are placed in sets E_{maj} and E_{min}. Note that, when considering an imbalanced data set, our experimental results against seventeen data sets indicate that a very high percentage of minority class examples are hard examples with high weights. Due to this fact, experimental results show that for the seed examples, the number of higher weighted examples from the minority class is more**.**

For example, for the illustrative Hepatitis data set, assume that, in the fifth iteration of the boosting, the current trained classifier's error rate is 18%. The set E_s will consist of the 27 examples with the highest weights as selected from the sorted E_{train}. Of these 27 hard examples, 2 correspond to the majority class 'Live', and 25 examples are of the class 'Die'. That is, the high occurrence of examples from the minority class is due to the fact that, for imbalanced data sets, the minority class is harder to learn. M_L is equal to 2, calculated as M_L = min (2, 3) and E_{mai} will thus contain both hard examples of the majority class 'Live'. M_s is equal to 8 and the set E_{min} will consist of the 8 highest weighted examples of class 'Die'. The output of this step is shown in the Table 1.

Table 1: Seed examples and their weights of the Hepatitis Data set

SEED examples and their weights for the majority class (stored in E_{maj}):	
2,female,y,y,y,n,n,y,y,n,y,n,n,1,59,249,3.7,54,n,LIVE	{.98}
41,female,y,y,y,n,n,y,y,n,n,n,n,0.9,8,60,3.9,52,n,LIVE	{.96}
SEED examples and their weights for the minority class (stored in E_{minj}):	
44,female,n,n,y,y,n,y,n,y,n,n,y,0.9,135,55,?,41,y,DIE	{.43}
43,female,y,n,y,n,n,y,n,y,y,y,n,1,100,19,3.1,42,y,DIE	{.43}
31,female,n,n,y,y,y,y,n,y,n,n,n,8,?,101,2.2,?,y,DIE	{.34}
38,female,n,n,n,n,n,y,y,n,n,n,n,0.4,243,49,3.8,90,y,DIE	{.32}
46,female,y,n,y,y,y,y,n,n,n,n,y,7.6,?,42,3.3,50,y,DIE	{.32}
33,female,n,n,y,y,n,y,n,n,n,y,n,0.7,63,80,3,31,y,DIE	{.23}
37,female,y,n,y,n,n,y,n,n,y,n,n,0.6,67,28,4.2,?,n,DIE	{.23}
34,female,n,n,y,y,n,n,y,n,y,n,n,2.8,127,182,?,?,n,DIE	{.20}

2.2 Data Generation and Class Frequency Balancing

The aim of the data generation process is to generate additional synthesis instances to *add to* the original training set E_{train}. The data generation process extends our earlier work, as presented in [13, 17, 18], by generating data for the majority and minority classes separately. That is, the data generation process generates two sets of data. Firstly a total of M_L sets of new majority class examples, based on each seed instance in $E_{maj,}$ are generated. For each attribute included in the synthetic example, a new value is generated based on the following constraints [13, 17, 18].

- For ***Nominal*** attribute, the data generation produces a total of N_{maj} attribute values for each seed in E_{maj}. The values are chosen to reflect the distribution of values contained in the original training attribute with respect to the particular class. This is achieved by considering, for each class, the number of occurrences of different attribute values in the original data set. For example, the attribute 'GENDER' in the Hepatitis data set has a value of either 'MALE' or 'FEMALE'. Assume that for the class 'Live', the number of occurrences of 'MALE' is 16 and 'FEMALE' is 107. The data generation creates 16 occurrences of 'MALE' and 107 occurrences of 'FEMALE'. These 123 values are randomly assigned to the 123 examples created during data generation.
- For ***Continuous*** attribute, the data generation produces a total of N_{maj} attribute values. The values are chosen by considering the range [*min, max*] of the original attribute values with respect to the seed class. Also, the distribution of original attribute values, in terms of the deviation and the mean, is used during data generation. For example, assume that, for the 'ALBUMIN' attribute in the Hepatitis data set, the 123 values for class 'Live' lies between 2.1 and 6.4, and the mean and deviation values are 3.817 and 0.652. The data generation randomly generates a total of 123 values between 2.1 and 6.4, following a mean value of 3.817 and a deviation value of 0.652. Again, the 123 values are randomly assigned to the 123 examples generated.

  Similarly, M_s different sets of new minority class examples, each based on a seed instance in E_{min}, are constructed. These sets of instances are added to the original training set.

Table 2: The number of synthetic examples generated and the number of their seeds

	Total cases	Synthetic cases generated	Original cases	Seeds
Majority Class	369	246	123	2
Minority Class	288	256	32	8

For the Hepatitis example, recall from Table 1 that E_{maj} contains 2 examples for the class 'Live' and E_{min} contains 8 instances for the class 'Die'. Followed the above-mentioned approach, the data generation process generates two sets of examples for the class 'Live', each set contains 123 synthetic examples for each one of the seeds in $E_{maj,}$. Eight sets of examples containing a total of 32 synthetic examples of the class 'Die', based on the 8 seed examples in $E_{min,}$ will also be created. A total set consisting of 246 synthetic examples of 'Live' and 256 instances of 'Die' is thus newly generated, as shown in Table 2. These instances are added to the original training set, leading to a final training set containing 369 instances of the class 'Live' and 288 instances of the class 'Die'.

Note that a detailed description of the data generation process falls beyond the scope of this paper. Interested readers are referred to [13, 17, 18] for a description of this process and its evolution.

2.3 Balancing the Training Weights

In the final step prior to re-training, the total weights of the examples in the different classes are rebalanced. Recall that boosting aims to, during each of the iterations, produce new classifiers that are better able to predict examples for which the precious classifier's performance is poor. This is achieved by concentrating on classifying the examples with high weights correctly. In an imbalanced data set, the difference of the total weights between the different classes is large. By rebalancing the total weights of the different classes, boosting is forced to focus on hard as well as rare examples.

Recall that the data generation process generates sets of synthetic examples based on seed examples E_{maj} and E_{min} corresponding to

the majority and minority classes. Before the generated data are added to the original data set, each of the synthetic examples is assigned an initial weight. The initial weight of each example is calculated by dividing the weight of the seed example by the number of instances generated from it. In this way, the very high weights associated with the hard examples are balanced out. Rebalancing ensures that the boosting algorithm focuses on hard as well as minority class examples.

When the new training set is formed, the total weights of the majority class examples (denoted by W_{maj}) and the minority class examples (denote by W_{min}) in the new training data are rebalanced as follows. If $W_{maj} > W_{min,}$ the weight of each instance in the *minority* class is multiplied by $W_{maj} / W_{min,}$ Otherwise, the weight of each instance in the *majority* class is multiplied by $W_{min} / W_{maj.}$ In this way, the total weight of the majority and minority classes will be balanced. Note that, prior to training, the weights of the new training set will be renormalized, following the AdaBoostM1 method, so that their sum equals one [8, 9, 10].

For the Hepatitis example, assume that seed example x in E_{maj} has a weight of 9.86 and seed example y has a weight of 9.62. This implies that each of the 123 synthetic examples generated based x is assigned a weight of 9.86/123 and those based on y are assigned weights of 9.62/123. Similarly, an initial weight are assigned to each of the synthetic examples generated based on the seed examples from E_{min}. After adding the synthetic data to the original data set, the new training data set contains 369 examples of class 'Live' and 288 cases of the class 'Live'. Assume that W_{maj} is equal to 122.51 and W_{min} equals 69.83. Since $W_{maj} > W_{min}$, each of the 288 examples describing the minority class is multiplied by a constant equal to 122.51/69.83. As a result, the total weights of the majority and minority classes in the new training set are equal to 122.51, thus equally distributing the balance of the two classes.

3. EXPERIMENTAL RESULTS

This section describes the results of evaluating the performance of the DataBoost-IM algorithm, in comparison with the C4.5 decision tree [20], AdaBoostM1[8, 9], DataBoost [13], AdaCost [21], CSB2 [22] and SMOTEBoost [7] boosting algorithms. The C4.5 algorithm, which has become a de facto standard against which new algorithms are being judged, is used as base classifier [23].

Table 3: Confusion Matrix

	Predicted Negative	Predicted Positive
Actual Negative	*TN* (the number of True Negatives)	*FP*(the number of False Positives)
Actual Positive	*FN* (the number of False Negatives)	*TP*(the number of True Positives)

Traditionally, the performance of a classifier is evaluated by considering the overall accuracy against test cases [16]. However, when learning from imbalanced data sets, this measure is often not sufficient [16]. Following [3, 4, 5, 7], we employ the *overall accuracy*, *G-Mean* [5] and *F-Measures* [14] metrics to evaluate our DataBoost-IM method. The confusion matrix, as shown in Table 3, represents the typical metrics for evaluating the performance of machine learning algorithms on skew class problems. In Table 3, the *TP Rate* and *FP Rate* are calculated as *TP/(FN+TP)* and *FP/(FP+TN)*. The *Precision* and *Recall* are calculated as *TP / (TP + FP)* and *TP / (TP + FN)*. The *F-measure* is defined as

$$((1+b^2)\times Recall\times Precision)\,/\,(b^2\times Recall + Precision) \qquad (1)$$

where ß corresponds to the relative importance of precision versus the recall and it is usually set to 1. The *F-measure* incorporates the *recall* and *precision* into a single number. It follows that the *F-measure* is high when both the *recall* and *precision* are high [6]. This implies that the *F-measure* is able to measure the "goodness" of a learning algorithm on the current class of interest. Note that we also use this measure for the majority class, since we are interested in measuring the performance of both classes. The *ROC* curve is a technique for summarizing a classifier's performance over a range, by considering the tradeoffs between *TP Rate* and *FP Rate* [15]. Another criteria used to evaluate a classifier's performance on skew data is the *G-mean* [4, 5, 7]. The *G-mean* is defined as

$$\sqrt{PositiveAc\ curacy \times NegativeAc\ curacy} \qquad (2)$$

where *Positive Accuracy* and *Negative Accuracy* are calculated as *TP/(FN+TP)* and *TN/(TN+FP)*. This measure relates to a point on the *ROC* curve and the idea is to maximize the accuracy on each of the two classes while keeping these accuracies balanced [5]. For instance, a high *Positive accuracy* by a low *Negative accuracy* will result in poor *G-Mean* [5].

Table 4: Summary of the data sets used in this paper. Shown are the number of examples in the data set; the number of minority class; the number of majority class; the class distribution; the number of continous, and the number of discrete input features.

Data set	Case	Min. Class	Maj. Class	Class Dist.	Feature Cont.	Feature Disc.
SONAR	208	97	111	0.47:0.53	60	0
MONK2	169	64	105	0.37:0.63	0	6
IONOSPHERE	351	126	225	0.35:0.65	34	0
BREAST-W	699	241	458	0.34:0.66	9	0
BREAST-CANCER	286	85	201	0.30:0.70	0	9
PHONEME	5484	1586	3818	0.29:0.71	5	0
VEHICLE	846	199	647	0.23:0.77	18	0
HEPATITIS	155	32	123	0.20:0.80	6	13
SEGMENT	2310	330	1980	0.14:0.86	19	0
GLASS	214	29	185	0.13:0.87	9	0
SATIMAGE	6435	626	5809	0.09:0.91	33	0
VOWEL	990	90	900	0.09:0.91	10	3
SICK	3772	231	3541	0.06:0.94	6	23
ABALONE	731	42	689	0.06:0.94	7	1
YEAST	483	20	463	0.04:0.96	8	0
PRIMARY-TUMOR	339	14	325	0.04:0.96	0	17
OIL	937	41	896	0.04:0.96	49	0

3.1 Data Sets

To evaluate the performance of the DataBoost-IM, we obtained sixteen data sets from the UCI data repository [12] as well as the Oil Spill data set [4, 5]. These data sets were carefully selected to ensure that they (a) are based on real-world problems, (b) varied in feature characteristics, and (c) vary extensively in size and class distribution. Table 4 presents the characteristics of the data sets

used for the experiments. Shown are the number of cases, the number of the majority and minority classes, the class distribution, and the type of the features. For the Glass, Vowel, Vehicle, Satimage, and Primary-tumor data sets, we increased the degree of skew by converting all but the smallest class into a single class. For the Sick data sets, we deleted the 'TBG' attribute due to the high number of missing values. For the Abalone and Yeast data sets, we respectively learned the classes '9' versus '18' in the Abalone, and the classes 'CYT' versus 'POX' in the Yeast in order to present the DataBoost-IM algorithm highly imbalanced problems.

3.2 Methodology and Experimental Results

We implemented the experiments using Weka [19], a Java-based knowledge learning and analysis environment developed at the University of Waikato in New Zealand.

Results for the above data sets, as shown in Table 5 and appendix A, were averaged over five standard 10-fold cross validation experiments. For each 10-fold cross validation the data set was first partitioned into 10 equal sized sets and each set was then in turn used as the test set while the classifier trains on the other nine sets. A stratified sampling technique was applied here to ensure that each of the sets had the same proportion of different classes. For each fold an ensemble of *ten* component classifiers was created. In the experiments, the C4.5 decision trees were pruned [20].

The experimental results for the eigth most highly imbalanced data sets, as described in Table 4, are presented in Table 5 . We also include the results for the nine moderately imbalanced data sets in Appendix A. For each data set, we present the results achieved when using the C4.5, AdaBoostM1, DataBoost, AdaCost [21], CSB2 [22], SMOTEBoost [7] and DataBoost-IM methods. Also, for each algorithm, the table presents the results in terms of the *G-mean*, *overall accuracy rates, TP rates* and *F-measures* against the majority and minority classes respectively. Following the work of Fan et al. [21], Ting [22], and Chawla et al. [7], the cost-adjustment functions from the AdaCost and CSB2 algorithms were chosen as follows: $b_- = 0.5 * c + 0.5$ and $b_+ = 0.5 * c + 0.5$, where b_+ and b_- are the functions for correctly and mislabeled labeled examples, respectively. Also, the three cost factors used in these two algorithms are 2, 5, and 9. The parameter N, which specifies the amount of synthetically generated examples in the SMOTEBoost was set to 100, 300 and 500, respectively.

Table 5: Results against eigth highly imbalanced data sets: *F-measure* of minority class, *F-measure* of majority class, *overall accuracy* , *G-mean*, *true positive rate* of minority class, and *true positive rate* of majority class for the data sets using (1) the C4.5 classifier (2) AdaBoostM1, (3) DataBoost, (4) AdaCost, (5) CSB2, (6) SMOTEBoost, and (7) DataBoost-IM ensembles.

Data Set Name	Methods		F-measure of min. class	F-measure of maj. class	Overall Accuracy	G-Mean	TP rate of min. class	TP rate of Maj. Class
GLASS	C4.5		78.5	96.7	94.3	85.9	75.8	97.2
	AdaBoostM1		81.3	97.0	94.8	89.4	82.7	96.7
	DataBoost		86.2	97.8	96.3	84.3	86.2	97.8
	AdaCost	(Cost Factor: 2)	84.4	97.3	95.3	94.3	93.1	95.7
		(Cost Factor: 5)	77.6	95.8	92.9	91.5	89.7	93.5
		(Cost Factor: 9)	73.5	95.0	91.5	89.2	86.2	92.4
	CSB2	(Cost Factor: 2)	76.9	95.9	55.6	90.9	88.2	93.7
		(Cost Factor: 5)	59.3	89.2	50.0	87.4	94.1	81.3
		(Cost Factor: 9)	36.5	65.6	38.3	68.3	95.0	49.2
	SMOTEBoost	(N=100)	84.0	97.4	95.6	91.1	85.5	97.1
		(N=300)	82.5	97.1	95.0	91.1	86.2	96.4
		(N=500)	81.4	96.8	94.5	91.5	87.5	95.6
	DataBoost-IM		89.2	98.3	97.1	92.3	86.2	98.9
SATIMAGE	C4.5		56.4	95.4	91.7	72.7	55.2	95.6
	AdaBoostM1		66.7	96.7	94.1	77.0	60.7	97.7
	DataBoost		66.3	96.7	93.9	77.3	61.3	97.5
	AdaCost	(Cost Factor: 2)	64.9	95.8	92.4	82.3	71.6	94.7
		(Cost Factor: 5)	58.6	92.9	87.8	88.1	88.5	87.8
		(Cost Factor: 9)	51.4	89.6	82.9	87.2	93.0	81.8
	CSB2	(Cost Factor: 2)	64.7	95.4	91.8	84.5	76.4	93.6
		(Cost Factor: 5)	47.7	87.5	79.7	86.0	94.7	78.2
		(Cost Factor: 9)	30.5	68.2	56.3	71.4	98.6	51.8
	SMOTEBoost	(N=100)	64.5	96.5	93.7	75.6	58.6	97.5
		(N=300)	65.3	96.6	93.8	76.0	59.3	97.6
		(N=500)	65.2	96.6	93.8	76.1	59.4	97.5
	DataBoost-IM		68.8	96.7	94.1	80.4	66.6	97.1
VOWEL	C4.5		93.7	99.3	98.8	95.8	92.2	99.5
	AdaBoostM1		97.1	99.7	99.4	97.6	95.5	99.8
	DataBoost		95.5	99.6	99.2	94.1	94.4	99.7
	AdaCost	(Cost Factor: 2)	96.1	99.6	99.2	98.1	96.7	99.6
		(Cost Factor: 5)	96.7	99.7	99.3	98.7	97.8	99.6
		(Cost Factor: 9)	88.8	98.8	97.7	97.3	96.7	97.9
	CSB2	(Cost Factor: 2)	96.6	99.7	79.4	99.6	100	99.3
		(Cost Factor: 5)	78.8	97.3	95.1	96.8	98.9	94.8
		(Cost Factor: 9)	41.3	83.5	66.7	84.6	100	71.6
	SMOTEBoost	(N=100)	97.3	99.7	99.5	98.7	97.7	99.6
		(N=300)	96.1	99.6	99.2	97.8	96.2	99.5
		(N=500)	96.3	99.6	99.3	97.9	96.2	99.6
	DataBoost-IM		98.8	99.8	99.7	99.3	98.8	99.8

Data Set Name	Methods	F-measure of min. class	F-measure of maj. class	Overall Accuracy	G-Mean	TP rate of min. class	TP rate of Maj. Class
	C4.5	89.1	99.3	98.7	93.0	87.0	99.4
	AdaBoostM1	91.1	99.4	98.9	94.2	89.1	99.5
	DataBoost	90.3	99.3	98.8	96.1	93.1	99.2
	AdaCost (Cost Factor: 2)	91.8	99.5	98.9	95.9	92.6	99.4
SICK	(Cost Factor: 5)	90.8	99.4	98.8	97.3	95.7	99.0
	(Cost Factor: 9)	86.7	99.0	98.1	97.6	97.0	98.2
	CSB2 (Cost Factor: 2)	91.1	99.4	98.8	97.3	95.7	99.1
	(Cost Factor: 5)	73.3	97.6	95.5	97.1	99.1	95.3
	(Cost Factor: 9)	27.1	79.0	67.4	80.4	99.1	65.3
	SMOTEBoost (N=100)	89.6	99.3	98.6	95.5	92.1	99.1
	(N=300)	88.5	99.2	98.5	95.0	91.2	99.0
	(N=500)	87.9	99.1	98.4	94.9	91.0	98.9
	DataBoost-IM	91.8	99.4	98.9	95.9	92.6	99.4
	C4.5	36.0	97.2	94.6	50.8	26.1	98.8
	AdaBoostM1	41.0	96.9	94.1	59.0	35.7	97.6
	DataBoost	39.7	96.9	94.2	32.6	33.3	97.9
	AdaCost (Cost Factor: 2)	29.8	95.2	90.9	56.1	33.3	94.5
ABALONE	(Cost Factor: 5)	28.8	93.3	87.8	62.3	42.9	90.6
	(Cost Factor: 9)	26.3	90.9	83.8	65.5	50.0	85.9
	CSB2 (Cost Factor: 2)	39.6	95.8	92.0	65.4	45.2	94.9
	(Cost Factor: 5)	23.4	81.7	70.4	74.1	78.6	70.0
	(Cost Factor: 9)	15.7	60.1	45.8	61.7	88.1	43.3
	SMOTEBoost (N=100)	37.6	96.6	93.6	56.9	33.3	97.3
	(N=300)	39.0	96.7	93.7	58.1	34.7	97.3
	(N=500)	37.4	96.6	93.5	56.9	33.3	97.2
	DataBoost-IM	45.0	97.1	94.6	61.1	38.0	98.1
	C4.5	9.5	97.9	96.0	22.3	5.0	100
	AdaBoostM1	51.4	98.1	96.4	66.6	45.0	98.7
	DataBoost	54.1	98.2	96.5	70.2	50.0	98.5
	AdaCost (Cost Factor: 2)	40.9	97.2	94.6	66.0	45.0	96.8
YEAST	(Cost Factor: 5)	47.1	97.0	94.4	75.8	60.0	95.9
	(Cost Factor: 9)	43.8	96.0	92.5	80.9	70.0	93.5
	CSB2 (Cost Factor: 2)	55.6	98.3	96.6	70.2	50.0	98.7
	(Cost Factor: 5)	25.5	90.4	83.0	76.5	70.0	83.6
	(Cost Factor: 9)	14.5	74.6	60.8	69.2	80.0	60.0
	SMOTEBoost (N=100)	57.6	98.5	97.1	67.5	46.0	99.3
	(N=300)	53.0	98.2	96.5	68.0	47.0	98.7
	(N=500)	46.7	97.6	95.5	67.7	47.0	97.6
	DataBoost-IM	58.0	98.6	97.3	66.9	45.0	99.5
	C4.5	0.00	97.8	95.8	0.00	0.00	100
	AdaBoostM1	19.0	97.4	94.9	37.5	14.2	98.4
	DataBoost	17.3	97.1	94.4	14.0	14.3	97.8
	AdaCost (Cost Factor: 2)	12.0	93.0	87.0	43.8	21.4	89.8
PRIMARY-	(Cost Factor: 5)	11.0	89.3	80.8	48.7	28.6	83.1
TUMOR	(Cost Factor: 9)	14.6	88.3	79.3	58.9	42.9	80.9
	CSB2 (Cost Factor: 2)	16.3	93.5	87.9	50.8	28.6	90.5
	(Cost Factor: 5)	13.8	66.9	52.2	68.4	92.9	50.5
	(Cost Factor: 9)	8.1	3.6	5.8	13.4	100	1.8
	SMOTEBoost (N=100)	16.4	96.8	93.9	37.3	14.2	97.4
	(N=300)	21.4	97.0	94.3	42.3	18.5	97.6
	(N=500)	24.3	97.1	94.5	45.7	21.4	97.6
	DataBoost-IM	28.5	96.9	94.1	52.6	28.5	96.9
	C4.5	37.6	97.6	95.4	55.8	31.7	98.3
	AdaBoostM1	38.8	97.7	95.6	55.8	31.7	98.5
	DataBoost	45.5	98.0	96.2	36.2	36.6	98.9
	AdaCost (Cost Factor: 2)	42.2	97.1	94.4	66.9	46.3	96.7
OIL	(Cost Factor: 5)	35.8	95.5	91.5	70.7	53.7	93.3
	(Cost Factor: 9)	32.1	93.8	88.6	74.0	61.0	90.0
	CSB2 (Cost Factor: 2)	50.6	97.6	95.4	72.2	53.7	97.3
	(Cost Factor: 5)	33.3	91.9	85.4	84.2	82.9	85.6
	(Cost Factor: 9)	20.1	81.1	69.3	77.5	87.8	68.5
	SMOTEBoost (N=100)	53.7	98.1	96.5	67.5	46.3	98.8
	(N=300)	49.4	98.0	96.2	63.9	41.4	98.7
	(N=500)	52.7	98.1	96.4	66.8	45.3	98.7
	DataBoost-IM	55.0	98.2	96.6	67.7	46.3	98.9

The results, as shown in Table 5 indicate that the DataBoost-IM algorithm performs well against highly imbalanced data sets, in terms of the *F-measures* of both the minority and majority classes. In many cases, our results are comparable or slightly higher than that produced by the other algorithms. Our approach also yields good results in terms of the *G-mean* and *overall accuracy*, when compared to the other approaches.

The DataBoost-IM approach achieved promising results when considering the minority class *F-measures*. In some cases, such as the Yeast and Sick data sets, the value obtained is the same as, or slightly higher than, the best value produced by the other algorithms. However, for data sets such as the Abalone, Primary-Tumor, Glass, and Satimage the minority class *F-measure* surpasses that of all the other techniques. Also, the majority class

F-measure against five of the eight data sets as produced by the DataBoost-IM algorithm is the highest, or the same as, that obtained by the other algorithms. For the other three data sets, namely the Sick, Abalone and Primary-Tumor data sets, the values obtained are lower by only 0.1, 0.1 and 0.9, when compared to the best performing algorithm.

Our results when comparing the DataBoost-IM method to the base-line C4.5 and AdaBoostM1 methods, shows that DataBoost-IM performs well in terms of both *F-Measures*. In some cases such as the Glass, Abalone, Yeast, Primary-Tumor, and Oil spill data sets, there are large improvements. For example, in the Oil spill data set, the C4.5 and AdaBoostM1 algorithms produced minority class *F-measures* of 37.6 and 38.8, respectively. The DataBoost-IM approach achieved a minority class *F-measure* of 55.0. In the Primary data set, the C4.5 and AdaBoostM1 algorithms produced minority class *F-measures* of 0 and 19.0 respectively, whereas the DataBoost-IM approach achieved a minority class *F-measure* of 28.5. Also, the *G-mean* of the DataBoost-IM method are, for all eight data sets, the same of slightly higher than that of the other two approach. Similar results holds for the *overall accuracy*, where DataBoost-IM performs slightly lower in only one case.

When considering the DataBoost-IM and the original DataBoost approaches, the values shown in Table 5 confirm that the DataBoost-IM approach benefits from generating synthetic examples for different classes separately and rebalancing the class frequencies and the total weights from the different classes. The DataBoost-IM approach achieved higher *F-measures* against both classes, except for the Primary-Tumor data set, in which the value against the majority class is lower by only 0.2, and the Satimage data set, in which the values against the majority class were the same. In many cases the improvement for the minority class was quite significant. For example, in the Primary-Tumor data set, the DataBoost method produced a minority class *F-measure* of 17.3, while The DataBoost-IM approach achieved a value of 28.5. Also, for the highly imbalanced Oil spill and Abalone data sets, the improvements achieved by the DataBoost-IM method over DataBoost algorithm were promising, i.e. 9.5 and 5.3 respectively. For the *G-mean* and *overall accuracy*, the DataBoost-IM method consistently produced similar or higher results than that of the DataBoost algorithm.

When comparing the DataBoost-IM method with the AdaCost, CSB2, and SMOTEBoost algorithms, the results shows that the DataBoost-IM method produced results which compare well, in terms of the *G-mean*, *overall accuracy* and *F-measures*. The DataBoost-IM method achieved similar or slightly better minority and majority class *F-measures* against the eight data sets. Importantly, our results indicate that our approach does not sacrifice one class, but produces high predictions against both. For example, for the Primary-Tumor data set, where the majority class *F-measure* values were lower by 0.2, when compared to the best values obtained by the other algorithms, the improvement of minority class *F-measure* were 4.2. Similarly, for the Abalone and Glass data sets, the improvements in minority class *F-measures* were 5.3 and 4.8, respectively.

Further analysis of the moderately imbalanced data sets, as shown in Appendix A, shows that the DataBoost-IM approach obtained the highest values against six of the nine data sets in terms of the minority class *F-measures*. Also, against seven of the nine data sets the majority class *F-measure* results are slightly higher. However, the improvements in term of *F-measures* are less significant than those against the highly imbalanced data sets. For the Monk2 and Breast-Cancer data sets, the majority class *F-measures* decreased by more than 40.0, when compared to the values obtained by the DataBoost-IM algorithm. The same results hold for the *overall accuracy* and *G-mean*, where DataBoost-IM produces comparable and slightly higher results in six of the nine data sets.

In conclusion, the results, as shown in Table 5 and Appendix A, indicate that the results obtained by the DataBoost-IM approach are comparable to that of the other techniques, when evaluated in terms of *overall accuracy*, *G-mean* and *F-measures*. In particular, the results against highly imbalanced data sets are promising. Importantly, for some highly imbalanced data sets, the DataBoost-IM approach produces the highest results in terms of both minority and majority class *F-measures*. Results indicate that the DataBoost-IM technique does not sacrifice the one class to favor the other. Rather, it aims at producing an ensemble which produces high values against both.

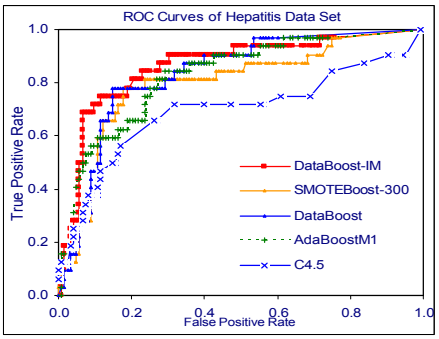

Figure 3: ROC Curve of the Hepatitis Data Set

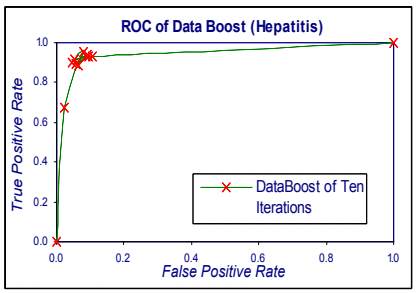

Figure 4: ROC Curve of ten iteration of the DataBoost-IM algorithm

To better understand the achievements of the DataBoost-IM method, we also present the *ROC* analysis results of the Hepatitis data set. This data set was chosen since it contains both continuous and discrete features, and has a moderately imbalance degree and instance size. We repeated the previous experiment against the Hepatitis data set, using the optimal parameter values for all algorithms as shown in Table 5. We varied the decision thresholds of the C4.5 algorithm, by varying the proportion of instances at the leaf node of the decision tree for labeling a class, to obtain the *ROC* curve. We produced an average *ROC* curve, as shown in Figure 3, by averaging the *TP* and *FP* rates over ten runs [15]. Also, we drew the *ROC* curves of the ten iterations of the

DataBoost-IM algorithm as shown in Figure 4. From the analysis of the *ROC* curves as shown in Figure 3, we conclude that the DataBoost-IM ensemble's *ROC* curve is of a high quality. This result indicates that the DataBoost-IM method achieved a high outcome, which compares well to that of the C4.5, AdaBoostM1, DataBoost and SMOTEBoost algorithms over most of the threshold values of the *ROC* space. Further analysis of Figure 4 shows us an essential fact of the DataBoost-IM approach, namely that each component classifier in the DataBoost-IM ensembles was pursuing a point with both a high *TP rate* and a low *FP rate* in the *ROC* space. This implies that the DataBoost-IM was able to produce a series of high quality classifiers, and each of them will be better able to predict examples for which the previous classifier's performance is poor.

4. CONCLUSIONS

This paper introduced a novel approach for learning from imbalanced data sets through combining boosting and data generation. In this approach, the class frequencies and the total weights against different the classes within the ensemble's training set, which consist of both the synthesis and the original training data, are rebalanced during all iterations of the boosting algorithm. The DataBoost-IM algorithm was illustrated by means of seventeen data sets with various features, degrees of imbalance and sizes. The results obtained indicate that the DataBoost-IM approach performs well against imbalanced data sets. In particular, the DataBoost-IM algorithm achieved comparable and slightly better predictions, in terms of the *G-mean* and *F-measures metrics*, against both the minority and majority classes, when compared with a component classifier as well as four other boosting algorithms. Importantly, our method does not sacrifice one class for the other, but produce high predictive accuracy against both the majority and the minority class.

In conclusion, these results indicate four reasons for the performance improvements achieved by the DataBoost-IM algorithm. The first is that the additional synthetic data provide complementary knowledge for the learning process. The second is that rebalancing the class frequencies alleviates the classifiers' learning bias toward the majority class. The third one is that rebalancing the total weight distribution of different classes forces the boosting algorithm to focus on the hard examples as well as rare examples**.** The last one is that the synthetic data prevent boosting from over-emphasize the hard examples. This property is especially important when considering the minority class which contains few examples.

Our future research will address some issues to extend the DataBoost-IM approach, including further investigating the optimal number of new seed examples to generate, experimenting with other component classifiers and considering the performance against noisy data**.** Also, other weight-assignment methods will be further investigated. Future work will also include studying the voting mechanism of the boosting algorithm using different metrics such as the *ROC* curve. Although the DataBoost-IM and the experiments addressed only two-class problems, we believe that a similar approach can be used in the frame of multi-class learning problems. This will be further investigated.

5. ACKNOWLEDGMENTS

Thanks are due to R.Holte for allowing us to use the Oil Spill data set and to N. Chawla for supplying the SMOTE script. We also wish to thank the anonymous reviewers for their comments. This paper extends our earlier work as reported in [24].

6. REFERENCES

[1] N. Japkowicz. Learning from imbalanced data sets: A comparison of various strategies, Learning from imbalanced data sets: The AAAI Workshop 10-15. Menlo Park, CA: AAAI Press. Technical Report WS-00-05, 2000.

[2] N. Chawla, K. Bowyer, L. Hall and W. Kegelmeyer. SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research, 16, 321-357, 2002.

[3] M.A. Maloof . Learning when data sets are Imbalanced and when costs are unequal and unknown, ICML-2003 Workshop on Learning from Imbalanced Data Sets II, 2003.

[4] M. Kubat, R. Holte and S. Matwin. Machine Learning for the Detection of Oil Spills in Satellite Radar Images. Machine Learning, 30, 195–215, 1998.

[5] M. Kubat and S. Matwin. Addressing the curse of imbalanced training sets: One-sided selection. Proceedings of the Fourteenth International Conference on Machine Learning San Francisco, CA, Morgan Kaufmann, 179-186,1997

[6] M. Joshi, V. Kumar and R. Agarwal. Evaluating boosting algorithms to classify rare classes: comparison and improvements. Technical Report RC-22147, IBM Research Division, 2001.

[7] N. Chawla, A. Lazarevic, L. Hall and K. Bowyer. SMOTEBoost: improving prediction of the minority class in boosting. 7th European Conference on Principles and Practice of Knowledge Discovery in Databases, Cavtat-Dubrovnik, Croatia , 107-119, 2003.

[8] Y. Freund and R. Schapire. Experiments with a new boosting algorithm. the Proceedings of the Thirteenth International Conference on Machine Learning, Bari, Italy, 148-156, 1996

[9] Y. Freund and R.E.Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences, 55(1), 119-139, 1997.

[10] H.Schwenk and Y. Bengio. AdaBoosting Neural Networks: Application to On-line Character Recognition, International Conference on Artificial Neural Networks (ICANN'97), Springer-Verlag, 969-972, 1997.

[11] T.G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. Machine Learning 40, 139-157, 2000.

[12] C.L. Blake and C. J. Merz. UCI Repository of Machine Learning Databases [http://www.ics.uci.edu/~mlearn/MLRepository.html], Department of Information and Computer Science, University of California, Irvine, CA, 1998.

[13] H Guo and HL Viktor. Boosting with data generation: Improving the Classification of Hard to Learn Examples, to be presented at the 17th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems (IEA/AIE), Ottawa, Canada, May 17-20, 2004.

[14] C. J. van Rijsbergen. Information Retrieval. Butterworths, London, 1979.

[15] F. Provost, T. Fawcett, and R. Kohavi. The case against accuracy estimation for comparing induction algorithms, Proceedings of the Fifteenth International Conference on Machine Learning, San Francisco, CA: Morgan Kaufmann, 445-453, 1998.

[16] F. Provost and T. Fawcett. Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In proceedings of the Third international conference on Knowledge discovery and data mining, Menlo park, CS. AAAI Press, 43-48, 1997.

[17] HL Viktor. The CILT multi-agent learning system, South African Computer Journal (SACJ), 24, 171-181, 1999.

[18] HL Viktor and I Skrypnik. Improving the Competency of Ensembles of Classifiers through Data Generation, ICANNGA'2001, Prague: Czech Republic, April 21-25, 59-62, 2001.

[19] I. Witten, E.Frank. Data Mining: Practical Machine Learning tools and Techniques with Java Implementations, Chapter 8, Morgan Kaufmann Publishers, 2000.

[20] JR Quinlan, C4.5. Programs for Machine Learning, Morgan Kaufmann, California: USA, 1994.

[21] W. Fan, S. Stolfo, J. Zhang, P. Chan, AdaCost: Misclassification Cost-Sensitive Boosting, Proceedings of 16th International Conference on Machine Learning, Slovenia, 1999.

[22] K. Ting, A Comparative Study of Cost-Sensitive Boosting Algorithms, Proceedings of 17th International Conference on Machine Learning, 983-990, Stanford, CA, 2000.

[23] C. Drummond and R. Holte. C4.5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling, Workshop on Learning from Imbalanced Data sets II held in conjunction with ICML'2003, 2003.

[24] H.L. Viktor and H. Guo, Multiple Classifier Prediction Improvements against Imbalanced Datasets through Added Synthetic Examples, *to be presented at the* 10th International Workshop on Statistical Pattern Recognition, Lisbon, Portugal, August 18-20, 2004.

APPENDIX A

Experimental results against nine moderately imbalanced data sets: *F-measure* of minority class, *F-measure* of majority class, *overall accuracy* , *G-mean*, *true positive rate* of minority class, and *true positive rate* of majority class for the data sets using (1) C4.5, (2) AdaBoostM1 ensembles, (3) DataBoost, (4) AdaCost, (5) CSB2, (6) SMOTEBoost, and (7) DataBoost-IM ensembles.

Data Set Name	Methods		F-measure of min. class	F-measure of maj. class	Overall Accuracy	G-Mean	TP rate of min. class	TP rate of Maj. Class
	C4.5		71.6	73.4	72.5	72.6	74.2	71.1
	AdaBoostM1		81.2	83.9	82.6	82.5	80.4	84.6
	DataBoost		81.4	83.8	82.7	68.2	81.4	83.8
	AdaCost	(Cost Factor: 2)	73.7	71.4	72.5	72.6	82.5	64.0
SONAR		(Cost Factor: 5)	76.5	67.1	72.5	70.8	95.9	52.3
		(Cost Factor: 9)	72.7	52.6	65.3	59.7	99.0	36.0
	CSB2	(Cost Factor: 2)	79.7	73.3	76.9	75.9	96.9	59.5
		(Cost Factor: 5)	67.1	25.2	54.3	37.9	100	14.4
		(Cost Factor: 9)	64.2	5.3	48.0	16.4	100	2.7
	SMOTEBoost	(N=100)	78.6	79.9	79.3	79.3	81.6	77.2
		(N=300)	76.9	80.4	78.8	78.5	75.6	81.6
		(N=500)	75.8	79.2	77.6	77.4	75.2	79.8
	DataBoost-IM		82.1	84.9	83.6	83.3	80.4	86.4
	C4.5		29.2	74.6	62.7	42.4	20.3	88.5
	AdaBoostM1		45.9	69.4	60.9	55.9	43.7	71.4
	DataBoost		41.0	68.8	59.2	27.1	37.5	72.4
	AdaCost	(Cost Factor: 2)	53.8	35.5	46.1	44.3	82.8	23.8
MONK2		(Cost Factor: 5)	54.7	12.7	40.3	25.8	95.1	7.0
		(Cost Factor: 9)	54.9	0	37.8	0.0	100	0
	CSB2	(Cost Factor: 2)	56.2	28.1	45.5	39.7	92.2	17.1
		(Cost Factor: 5)	54.9	0	37.8	0.0	100	0
		(Cost Factor: 9)	54.9	0	37.8	0.0	100	0
	SMOTEBoost	(N=100)	46.8	62.4	55.9	54.9	51.2	58.8
		(N=300)	54.3	60.2	57.5	58.8	66.8	51.8
		(N=500)	53.8	57.6	55.8	57.3	68.1	48.3
	DataBoost-IM		52.7	70.8	63.9	61.1	53.1	70.4
	C4.5		83.3	91.3	88.6	86.2	79.3	93.7
	AdaBoostM1		87.2	93.5	91.4	88.9	81.7	96.8
	DataBoost		89.3	94.3	92.6	82.8	86.0	96.3
	AdaCost	(Cost Factor: 2)	88.4	93.6	91.7	90.6	87.3	94.2
Ionosphere		(Cost Factor: 5)	84.7	90.2	88.0	88.8	92.1	85.8
		(Cost Factor: 9)	79.6	84.9	82.6	84.7	94.4	76.0
	CSB2	(Cost Factor: 2)	89.7	93.6	82.9	93.0	96.5	89.7
		(Cost Factor: 5)	68.9	67.1	61.2	70.8	99.1	50.7
		(Cost Factor: 9)	54.3	12.0	35.8	25.3	100	6.4
	SMOTEBoost	(N=100)	90.2	94.7	93.1	92.0	88.4	95.8
		(N=300)	89.4	94.4	92.7	91.2	86.5	96.1
		(N=500)	88.6	94.0	92.1	90.5	85.3	96.0
	DataBoost-IM		91.2	95.4	94.0	92.3	87.3	97.7

Data Set Name	Methods	F-measure of min_class	F-measure of maj_class	Overall Accuracy	G-Mean	TP rate of min_class	TP rate of Maj_Class
	C4.5	92.4	95.9	94.7	94.3	93.3	95.4
	AdaBoostM1	93.9	96.8	95.8	95.4	94.1	96.7
	DataBoost	93.7	96.7	95.6	90.8	94.2	96.4
	AdaCost (Cost Factor: 2)	93.2	96.2	95.1	95.4	96.7	94.3
BREAST-W	(Cost Factor: 5)	93.5	96.3	95.2	96.0	98.8	93.4
	(Cost Factor: 9)	92.2	95.5	94.2	95.2	98.8	91.9
	CSB2 (Cost Factor: 2)	93.6	96.4	95.4	95.7	96.7	94.8
	(Cost Factor: 5)	87.9	92.3	90.5	92.4	99.6	85.8
	(Cost Factor: 9)	68.3	67.6	67.9	71.4	100	51.1
	SMOTEBoost (N=100)	94.1	96.8	95.9	95.8	95.6	96.1
	(N=300)	94.3	96.9	96.0	95.9	95.6	96.1
	(N=500)	94.5	97.0	96.1	96.2	96.3	96.0
	DataBoost-IM	95.2	97.4	96.7	96.4	95.4	97.3
	C4.5	39.3	84.3	75.1	50.8	27.0	95.5
	AdaBoostM1	44.0	74.9	65.3	58.1	45.8	73.6
	DataBoost	40.2	75.9	65.7	29.9	38.8	77.1
	AdaCost (Cost Factor: 2)	46.5	56.1	51.7	55.6	70.6	43.8
BREAST-	(Cost Factor: 5)	47.3	12.0	34.0	24.7	93.0	6.6
CANCER	(Cost Factor: 9)	46.1	0	29.9	0.00	100	0
	CSB2 (Cost Factor: 2)	42.4	41.5	41.9	45.9	71.8	29.4
	(Cost Factor: 5)	45.8	0	29.7	0.00	100	0
	(Cost Factor: 9)	45.8	0	29.7	0.00	100	0
	SMOTEBoost (N=100)	44.9	75.6	66.2	58.8	46.3	74.6
	(N=300)	45.9	73.9	64.8	59.7	50.3	70.9
	(N=500)	46.1	72.8	63.9	59.8	52.0	68.9
	DataBoost-IM	46.5	77.0	67.8	60.0	47.0	76.6
	C4.5	77.2	90.0	86.1	84.2	79.8	88.7
	AdaBoostM1	81.8	92.6	89.4	86.7	80.8	93.0
	DataBoost	81.8	92.7	89.6	74.9	80.1	93.5
	AdaCost (Cost Factor: 2)	78.8	88.9	85.4	87.3	92.3	82.6
PHONEME	(Cost Factor: 5)	71.9	81.5	77.7	82.2	97.3	69.6
	(Cost Factor: 9)	67.2	75.3	71.7	77.3	98.4	60.8
	CSB2 (Cost Factor: 2)	80.4	89.8	86.6	88.4	93.4	83.8
	(Cost Factor: 5)	58.0	57.4	57.7	63.4	99.5	40.4
	(Cost Factor: 9)	46.2	6.0	31.5	17.6	100	3.1
	SMOTEBoost (N=100)	82.4	92.4	89.4	87.8	84.3	91.5
	(N=300)	80.5	91.3	88.0	86.9	84.4	89.4
	(N=500)	79.8	90.9	87.5	86.4	84.0	89.0
	DataBoost-IM	83.8	93.2	90.5	88.4	83.7	93.3
	C4.5	87.9	96.1	94.2	92.5	89.4	95.6
	AdaBoostM1	92.5	97.6	96.4	95.5	93.9	97.2
	DataBoost	91.6	97.5	96.1	88.8	91.0	97.7
	AdaCost (Cost Factor: 2)	88.3	96.2	94.2	93.8	93.0	94.6
VEHICLE	(Cost Factor: 5)	86.2	95.0	92.6	94.3	97.5	91.2
	(Cost Factor: 9)	85.1	94.4	91.8	94.1	99.0	89.6
	CSB2 (Cost Factor: 2)	90.5	96.7	95.1	96.1	98.0	94.3
	(Cost Factor: 5)	73.1	87.3	82.7	87.8	99.5	77.6
	(Cost Factor: 9)	45.0	39.7	42.4	49.7	100	24.7
	SMOTEBoost (N=100)	92.4	97.6	96.4	95.3	93.4	97.3
	(N=300)	92.1	97.5	96.2	95.5	94.3	96.8
	(N=500)	91.0	97.1	95.6	95.1	94.1	96.1
	DataBoost-IM	93.7	98.0	97.0	95.7	93.4	98.1
	C4.5	42.1	86.9	78.7	57.9	37.5	89.4
	AdaBoostM1	52.4	88.3	81.2	66.8	50.0	89.4
	DataBoost	58.3	89.2	83.9	50.0	54.7	91.5
	AdaCost (Cost Factor: 2)	59.5	87.3	80.6	75.8	68.8	83.7
HEPATITIS	(Cost Factor: 5)	50.9	73.0	65.1	72.0	87.5	59.3
	(Cost Factor: 9)	48.8	66.3	59.3	68.7	93.8	50.4
	CSB2 (Cost Factor: 2)	63.4	86.8	80.6	80.9	81.3	80.5
	(Cost Factor: 5)	42.3	51.2	47.0	57.3	93.8	35.0
	(Cost Factor: 9)	36.0	13.6	26.4	27.0	100	7.3
	SMOTEBoost (N=100)	58.9	89.3	83.0	72.6	59.3	89.2
	(N=300)	59.2	88.4	82.0	74.0	63.1	86.9
	(N=500)	58.4	87.8	81.1	74.1	64.3	85.5
	DataBoost-IM	62.6	89.7	83.8	76.2	65.6	88.6
	C4.5	87.2	97.8	96.3	92.2	86.9	97.9
	AdaBoostM1	93.5	98.9	98.1	95.9	93.0	99.0
	DataBoost	93.9	99.0	98.3	92.5	93.3	99.1
	AdaCost (Cost Factor: 2)	90.7	98.4	97.2	96.0	94.5	97.7
SEGMENT	(Cost Factor: 5)	86.4	97.4	95.6	96.1	96.7	95.5
	(Cost Factor: 9)	84.3	96.9	94.7	96.2	98.5	94.1
	CSB2 (Cost Factor: 2)	91.7	98.5	97.4	97.4	97.3	97.5
	(Cost Factor: 5)	75.2	94.2	90.6	94.0	99.1	89.2
	(Cost Factor: 9)	44.7	74.1	64.7	76.6	100	58.8
	SMOTEBoost (N=100)	94.2	99.0	98.3	96.2	93.4	99.1
	(N=300)	95.4	99.2	98.6	97.2	95.2	99.2
	(N=500)	94.1	99.0	98.3	96.4	94.0	99.0
	DataBoost-IM	95.5	99.2	98.7	97.3	95.4	99.2