Predicting Rare Classes: Can Boosting Make Any Weak Learner Strong?

Mahesh V. Joshi*
IBM T. J. Watson Research Center and University of Minnesota, Minneapolis
joshim@us.ibm.com

Ramesh C. Agarwal
IBM Almaden Research Center
650 Harry Road
San Jose, CA
ragarwal@us.ibm.com

Vipin Kumar†
University of Minnesota
Department of Computer Science
Minneapolis, MN 55455
kumar@cs.umn.edu

ABSTRACT

Boosting is a strong ensemble-based learning algorithm with the promise of iteratively improving the classification accuracy using any base learner, as long as it satisfies the condition of yielding weighted accuracy > 0.5. In this paper, we analyze boosting with respect to this basic condition on the base learner, to see if boosting ensures prediction of rarely occurring events with high recall and precision. First we show that a base learner can satisfy the required condition even for poor recall or precision levels, especially for very rare classes. Furthermore, we show that the intelligent weight updating mechanism in boosting, even in its strong cost-sensitive form, does not prevent cases where the base learner always achieves high precision but poor recall or high recall but poor precision, when mapped to the original distribution. In either of these cases, we show that the voting mechanism of boosting fails to achieve good overall recall and precision for the ensemble. In effect, our analysis indicates that one cannot be blind to the base learner performance, and just rely on the boosting mechanism to take care of its weakness. We validate our arguments empirically on variety of real and synthetic rare class problems. In particular, using AdaCost as the boosting algorithm, and variations of PNrule and RIPPER as the base learners, we show that if algorithm A achieves better recall-precision balance than algorithm B, then using A as the base learner in AdaCost yields significantly better performance than using B as the base learner.

*Contact Author

†This work was supported in part by the Army High Performance Computing Research Center cooperative agreement number DAAD19-01-2-0014, the content of which does not necessarily reflect the position or the policy of the government, and no official endorsement should be inferred.

SIGKDD '02 Edmonton, Alberta, Canada

1. INTRODUCTION AND MOTIVATION

In many domains such as fraud detection, network intrusion detection, text categorization, and web mining, it is important to be able to learn effective predictive models for some critical events that usually occur very rarely. For example, in network intrusion detection domain, the number of past attacks on a server or a network is very small as compared to the number of normal connections. It is crucial to build good models which capture large number of the occurrences of the future attacks with small false positive rate. In another example, it is important to predict the successful or actionable web sessions, that usually occur in relatively much smaller proportion in the sea of thousands of sessions occurring at a web-site on a given day. In many domains, past experience is usually available in the form of a set of records labeled with the category of that occurrence. In such cases, classification is a natural and promising choice of method.

Some work has started to emerge in the field of building descriptive and predictive classifier models for rare events [8, 3, 9]. Boosting [12, 4, 13, 6] is a promising meta-technique with a theoretically justified ability to improve the performance of any weak classification algorithm[1]. All boosting algorithms work in iterations. In each iteration, a classifier model is learned via a weak learning algorithm on a weighted distribution of training records. After each iteration, the weights of all examples are updated in a manner that forces the weak classifier to focus more on the incorrectly classified examples in the next iteration. The precise form of weight update differs across algorithms [10]. The algorithm stops either after a number of iterations computed via cross-validation or after some metric of quality starts deteriorating. Usually, former method is chosen. The process of classifying a new example uses the ensemble of all the classifier models learned in all the iterations. Each model votes for a class. The vote is monotonic with the model's accuracy. The example is predicted to belong to a class with the largest vote.

We assume a binary classification problem for illustrating the concepts of this paper. The boosting theory imposes a

[1]In terms of PAC learning theory [11], weak learner is defined as an algorithm that, given $\epsilon \leq 1/2 - \gamma$ ($\gamma > 0$) and $\delta > 0$, can achieve an error rate of ϵ, that is at least slightly better than than random guessing, with a probability of at least $(1 - \delta)$.

simple restriction on the accuracy of the base learner; that it should be better than 50% on any weighted distribution of training examples presented to it. Then, using the intelligent weight adaptation and the ensemble nature, boosting promises to achieve lower overall error rates as it progresses. Achieving better overall accuracy is the traditional goal of boosting. However, for rare classes, overall accuracy is not a meaningful metric, because for a problem with only 1% population of rare class, one can achieve good accuracy by just predicting everything to be of the non-rare class. It is more meaningful to achieve high recall and high precision[2] for the rare class of interest. Our primary goal in this paper is to evaluate boosting mechanism with respect to this objective. There are four components to boosting: the accuracy condition, the ensemble-based voting, the process of updating weights after each iteration, and the base learner algorithm. The effect of the voting process was analyzed in [2] with respect to the traditional goal of accuracy, to show that the key to the boosting performance is the intelligent weight update mechanism and not any specific form of voting (weighted vs. unweighted). The effect of various weight update mechanisms in the context of rare classes was compared in [10]. It was shown that the difference in the weight update mechanism significantly affects the performance. However, the choice of the base learner was kept fixed in that work. The question we ask in this paper is: for a given weight update mechanism and a simple unweighted form of the voting process, can boosting improve the performance of *any* base learner for the rare class, provided that the base learner satisfies the condition on weighted accuracy in every iteration? The insight we obtain will be useful towards extracting the best performance from a given boosting algorithm.

We present detailed analysis to show three things. First, the accuracy condition can be satisfied despite having the recall and precision of the rare class vary in a wide range. Secondly, we show that the ensemble-based voting nature of boosting can fail to achieve overall good recall and precision levels if the base learner always achieves poor recall or poor precision with respect to the original distribution of the training data. This is related to the nature of the voting process, which requires that for an example to get a correct prediction from the ensemble, total number of the base learners predicting that example to be correct should be more than the total number of base learners making incorrect predictions. For example, if each base learner achieves poor recall, then the total number of classifiers predicting a class C example to be of class C, will be less, which indicates that the recall of the ensemble will also be poor. On the other hand, if each base learner achieve poor precision, then many base learners will erroneously predict an example not belonging to C to belong to C, thus reducing the precision of the ensemble also.

The only safeguard that boosting provides against such cases, is the weight update mechanism. Third and more important thing we show, via detailed mathematical analysis, is that the weight update mechanism does not provide any guarantees of preventing the base learner from always achieving poor recall or poor precision, especially as the target class becomes rare. In particular, we perform an incremental analysis of a generic boosting mechanism to show that the weight update mechanism does not force two consecutive base learners to have significant overlap for low recall or low precision values, under the condition that the weighted accuracy of both learners remains > 50%.

Once we show these things, it follows that the performance achieved by the boosting algorithm is strongly tied to the choice of the base learning algorithm. Thus, if the chosen base learner always achieves poor recall or poor precision, then boosting performs poorly. More interesting outcome of our analysis is that, if an appropriate algorithm is chosen as the base learner, especially the one that always tries to achieve a good tradeoff between recall and precision on a given weighted distribution, then boosting performance can be significantly improved.

We validate our arguments on various synthetic and real data sets, using AdaCost as the boosting algorithm, and variations of RIPPER [5] and PNrule [9] as the base learners. In particular, we show that if the base learner of one kind performs better than the other on its own, then AdaCost using the better base learner performs better. Sometimes, the performance improvement achieved by using an appropriate base learner is seen to be better than factor of two.

Contributions:

- Detailed generic analysis of boosting mechanism with respect to the accuracy condition imposed by boosting, in the context of achieving higher recall-precision balance for the rare classes.
- An insight that choice of the appropriate base learner is crucial when it comes to predicting rare classes effectively.
- Use of the above insight to achieve significantly better performance from boosting.

The rest of the paper is organized as follows. Section 2 gives a brief overview of a generic boosting algorithm. Section 3 analyzes the accuracy condition in context of rare classes, and shows the inadequacy of boosting to guarantee good recall-precision performance. Section 4 presents various supporting results, and Section 5 concludes the paper with the summary and some remarks.

2. BOOSTING OVERVIEW

Boosting algorithms work in iterations. In each iteration, base learning algorithm is invoked on a weighted distribution of the training data. The key idea in boosting is to modify the weights on the examples so that the weak learner in the next iteration focuses on correctly classifying the examples that were misclassified by most of the earlier weak learners. To achieve this, boosting algorithm increases the weights of the incorrect decisions made by the weak learner of the present iteration, and suppresses the weights of the correct decisions. The factors used for boosting or suppressing the weights depend on the accuracy of the weak learner in general and differ across algorithms. However, the boosting theory ensures that all these factors are > 1 (i.e. factors used for boosting weights indeed increase the weights and factors

[2]Given that a classifier detects total m examples to be of class C, out of which l are indeed of class C, then precision of the classifier for class C is l/m. If there are total n examples of class C in the set, then the classifier has a recall of l/n for class C.

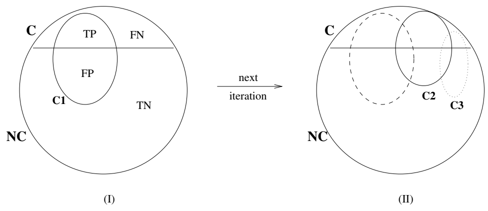


Figure 1: Consecutive iterations of a boosting algorithm. TP and TN denote true positive and negative predictions with respect to C for classifier C1. FP and FN denote false positive and negative predictions.

used for suppressing weights indeed decrease the weights), by imposing a condition that the weak learner should do better than random guessing; i.e. its accuracy on the weighted distribution from which it learns should be > 0.5.

The boosting process is illustrated in Figure 1, which shows consecutive iterations of a boosting algorithm operating on a binary classification problem with rare class C and non-rare class NC. The net effect of the weight update mechanism is that the next weak learner for a class, say C, tries to predict more of the false negative predictions of C and less of the false positive predictions of C as belonging to C. However, because of the reduction in weight on true predictions, some of them will get predicted incorrectly by the next base learner. Overall, if one looks at the models learned in the consecutive iterations, the picture will look similar to that shown in Figure 1. Here the circles indicated by C1 and C2 indicate the boundary of the classifier for class C; i.e. everything within these circles is predicted as C and everything outside is predicted as NC. The key thing to observe is how C2 is shifted from C1 to cover more false negatives of C1 and less false positives of C1. If the process continues beyond C2, then a boundary similar to C3 will be learned.

This process is common across all algorithms, what is different is the factors by which each of the four types of examples, TP, FP, TN, FN, shown in left part of the Figure, get updated to form the distribution from which C2 is learned. We give details of these factors for AdaCost later in Section 3.3. A comprehensive comparison can be found in [10].

3. EFFECT OF BASE LEARNER

The goal of this paper is to formally analyze the inadequacy of the only condition imposed by boosting algorithms on the weak learner; i.e. its weighted accuracy should be $> 50\%$, when it comes to rare classes, and make a case that the nature of weakness of the base learner matters when it comes to predicting rare classes effectively. We do this step by step in following subsections.

3.1 Analyzing the Weakness Condition

The first step in our analysis will show how a weak learner accuracy condition translates into the conditions on recall and precision of the base learner with respect to the rarer class. Let us first show a relationship between overall accuracy and recall, precision values of the rare class. We will assume the confusion matrix of Table 1 for all our discussions. Note that the total weight on all the examples in the training set is assumed to be 1, at the beginning of each

	predict C	predict NC	Total
actual C	τ (TP)	$\rho - \tau$ (FN)	ρ
actual NC	ϕ (FP)	$1 - \rho - \phi$ (TN)	$1 - \rho$

Table 1: Confusion Matrix in terms of the true positive predictions (τ), false positive predictions (ϕ), and the proportion of C's examples in the training set (ρ).

iteration. In fact, boosting has a step of normalizing the weights after they are modified at the end of every iteration. Let us denote the sum of the weights on all the true positives (TPs), true negatives (TNs), false positives (FPs), and false negatives (FNs) by τ, τ_n, ϕ, and ϕ_n, respectively.

For all the discussion, let C be the original rare class and let NC represent all the other examples (may include all the classes other than C, when a multi-class problem is translated to binary problem for modeling C).

Recall of C, denoted by R, is defined as how many of the total C's examples are predicted correctly; i.e., $R = TP/(TP + FN)$. For our confusion matrix, $R = \tau/\rho$. Precision with respect to C, denoted by P, is defined as how many of the examples predicted as C indeed belong to C; i.e., $P = TP/(TP + FP)$. For our confusion matrix, $P = \tau/(\tau + \phi)$.

Overall accuracy, denoted by ν, is $\nu = \tau + \tau_n$. For our confusion matrix, $\tau_n = 1 - \rho - \phi$. Hence, $\nu = 1 + \tau - \rho - \phi$. Using substitutions $\tau = R\rho$ and $\phi = \tau((1/P) - 1) = R\rho((1/P) - 1)$, ν can be expressed in terms of R and P as $\nu = 1 + R\rho - \rho - R\rho((1/P) - 1)$.

For the accuracy to be greater than 50%; i.e., for $\nu > 0.5$ to hold, following limits on R must hold, since C is the rare class (i.e. $\rho < 0.5$):

$$\begin{aligned} R &< ((0.5/\rho) - 1)/((1/P) - 2), \text{ for } P < 0.5 \\ &> ((0.5/\rho) - 1)/((1/P) - 2) = -\delta, \text{ for } P > 0.5, \end{aligned}$$

where $\delta > 0$. As these limits indicate, for $P > 0.5$, any R value can satisfy the accuracy constraint, provided $\rho < 0.5$; i.e. recall of the rarer class can assume any value provided that its precision is $> 50\%$. For $P < 0.5$, one can make following observations on the upper limit on recall. As both ρ and P tend towards 0.5, the upper limit on recall tends towards 0.5. This indicates that for similar distributions of classes, if precision is closer to 50%, so should be the recall. In fact, for similar distributions of classes, recall and precision are closely coupled and there is not much freedom in choosing value of one for a fixed value of the other, especially as the precision starts getting poorer. However, as the rarity increases, upper limit on recall tends towards 1 as long as $P > 1/((0.5/\rho) + 1)$. As an example, if $\rho = 0.05$ (5% population of C), for any value of precision $> 9.1\%$, recall can assume any value (from 0 to 100%) still maintaining the accuracy value to be > 0.5. Similarly, for $\rho = 0.01$ (1% population of C), for any value of precision $> 1.96\%$, recall can range from 0% to 100%. Thus both precision and recall can basically assume almost any values as the target class proportion decreases significantly. This discussion shows that when it comes to rare classes, the condition on accuracy imposed by the boosting algorithms, allows one to choose a wide range of learning algorithms as the base learner, irrespective of their recall-precision performance on the rare

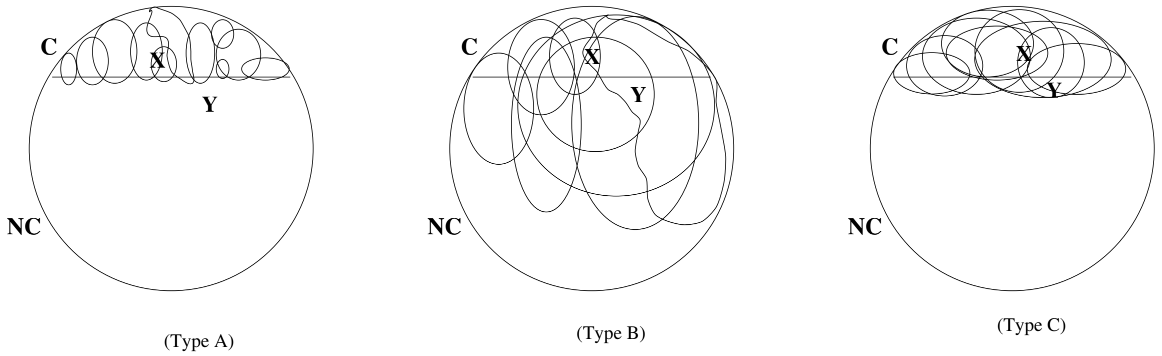


Figure 2: ***Three types of weak learners: A. Weak learner with over-emphasis on precision (e.g. RIPPER0 with unit weight). B. Weak learner with over-emphasis on recall (e.g. RIPPER0 with weight stratification). C. Weak learner with similar emphasis on recall as well as precision (e.g. PNrule). Example X belongs to C, and Y belongs to NC. Refer to text for how X and Y's prediction is affected by boosting a given type of the base learner.***

class.

So, the natural question is, does the boosting performance as a whole depend on the choice of the base learner, when it comes to effectively predicting rare classes? Or does the boosting mechanism have an ability to overcome the poor performance of the weak learner to achieve an overall good performance for the rare class? In the next two subsections, our attempt is to show qualitatively and formally that boosting mechanism does not offer any guarantee of such ability.

3.2 Effect of the Ensemble Voting

The second step of our analysis makes a case that there can exist at least three different types of base learners, for which the effect of ensemble-based voting can be significantly different. In particular, we consider three types of classifiers: **A.** the one geared toward achieving high precision irrespective of how much recall it achieves, **B.** the one that is geared towards achieving high recall without caring much for the precision, and **C.** the one that can achieve a good tradeoff between recall and precision simultaneously. For these three types of classifiers, Figure 2 shows how the classifier boundaries for C in different boosting iterations will look like. A point to note is that each classifier learned from a different distribution, so to put them all in single perspective, the Figure is drawn in the original weight space. The figure may look cluttered, but focusing on how many classifiers predict the indicated examples X and Y as belonging to C, will help in understanding the following discussion.

The type A classifier tries to achieve good precision in every iteration, but in the process may achieve very poor recall. The weight update scheme shifts the classifier focus to capturing less of correctly predicted examples and more of incorrectly predicted examples. Thus its primary effect to keep shifting the classifier boundary all over the region of C, such that every positive example of C gets covered by at least some classifier. If the recall in each iteration is limited because of the type of the weak learner, then in a given number of iterations, each X example may get predicted as C only by a few number of classifiers. Thus, the ensemble-based prediction may assign it a class of NC, thus achieving a overall poor recall.

The problem with type B classifiers is the opposite, as shown in Part II of Figure 2. If every base learner focuses only on the recall, and inherently achieves poor precision in the process, then many X examples will get covered by many classifiers. Thus, the overall ensemble can achieve very good recall levels. However, in the process, sufficiently large number of Y examples can also get covered within C's boundaries. Remember that the total number of X examples is rare, so even if a small fraction of total number of Y examples get captured by sufficiently many classifiers, the precision will suffer. This is shown in the figure by various regions in the NC part that fall within the boundaries of sufficiently many base learners. All these will be predicted as C's, thus causing false positive rate to go up and suffering on precision.

Finally, the type C classifiers try to achieve the best features of types A and B (higher recall and higher precision) in *every* iteration of the boosting algorithm. An example of it is depicted part III for Figure 2. Thus, the final ensemble can achieve good recall by having many classifiers covering any given X example, and good precision as well by *not* having many classifiers cover many Y examples.

3.3 Effect of Weight Update

The final step of our analysis is to basically find the conditions under which all the scenarios presented in Figure 2 are possible. In other words, we want to find out if the undesirable situations, such as those of types A and B base learners, can be avoided. After the analysis of voting process and the accuracy condition presented so far, the only mechanism boosting offers is its intelligent weight updating process. We want to see if this mechanism provides any safeguard against type A and B situations. For example, scenario A can be avoided in couple of ways. One is by ensuring that each weak learner progressively covers more and more part of the target class (i.e. achieves higher recall) in the original weight space. Second is by forcing two consecutive weak learners to have significant overlap in their true positives. In either of the cases, there will be enough overlap among classifiers for a given example to achieve overall good recall. Scenario B can be avoided by forcing each base learner to have higher precision; i.e. less number of false positives, for a given recall, so examples of Y kind will have less number of classifiers covering them, achieving overall good precision. Here is the approach we take to show that

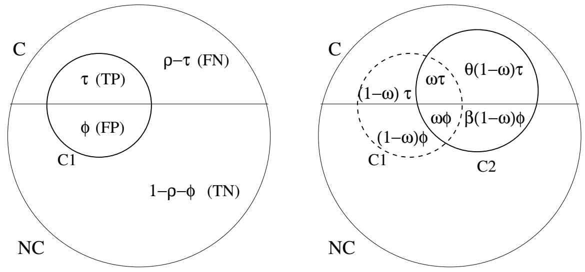


Figure 3: Two consecutive iterations of a boosting algorithm. The figure also embodies the parameters and assumptions used in our analysis.

none of these solutions can be forced by boosting.

Our analysis is similar to the proof-by-contradiction strategy. We first enforce that the recall and precision of the base learner follow a fixed pattern from one iteration to the next. Then we show that a particular case where each weak learner achieves identical recall and precision in the original weight space is allowed by weight update mechanism, especially for the combinations of low recall, high precision (type A learner) and low precision, high recall (type B learner). What this effectively means is that the weight update does not enforce the weak learner to achieve progressively higher recall or precision levels. Also, if we can show that the overlap among true and false positives allowed between two consecutive base learners is not restricted especially for these recall-precision combinations, then the second solution above to avoid situation A is not enforced. Moreover, we want to show that these issues become relevant especially for rare classes.

We show all these points by presenting a generic analysis of two consecutive iterations of boosting. We parameterize our requirement that recall and precision follow a fixed pattern and also use a parameter to describe the overlap between two consecutive learners. The key is to find what are the conditions required to assure that the next base learner achieves a weighted accuracy of $> 50\%$ given that the current base learner has weighted accuracy of $> 50\%$. We keep our analysis very general in the sense that it can be applied to most boosting algorithms.

Let us start with Figure 3 to illustrate various parameters and assumptions of our analysis. The first part of the figure shows an arbitrary iteration (t) and uses the same notation as given in Table 1. The second part illustrates the next iteration $(t+1)$ and embodies different parameters and assumptions that we describe next.

Let D^t denote the distribution from which C1 is learned, and D^{t+1} denote the weighted distribution from which C2 is learned. Figure 3 is drawn in the D^t space and all the parameters are also defined with respect to the D^t distribution. The key to our analysis, as will be clear later, is to map all the weights in the D^{t+1} distribution to those in the D^t distribution.

The two consecutive classifiers are assumed to have an overlap of ω $(0 \leq \omega \leq 1)$ in their TP and FP predictions. Ideally, one can assume different overlap factors for each, but assuming them equal simplifies the analysis without much loss of generality, especially for smaller values of ω.

The non-overlapping part of TPs of C2 is assumed to be θ times the non-overlapping part of the TPs of C1; hence, the figure shows a total weight of $\theta(1-\omega)\tau$ for C2's non-overlapping TPs. Similarly, non-overlapping part of the FPs of C2 is assumed to be β times the non-overlapping part of the FPs of C1; hence the total weight of $\beta(1-\omega)\phi$ for C2's non-overlapping FPs.

In fact, θ, β, and ω parameters together allow us to enforce a pattern on the recall and precision of the base learners from iteration to iteration, because recall and precision of C2 (denoted by R_{C2} and P_{C2}, resp.) depend on the recall and precision of C1 (R and P, resp.) and these parameters. The exact expressions are presented later.

One more set of parameters is the weight update factors. Let $\gamma > 1$ be the factor by which weight on each TP or TN example is reduced, $\gamma_p > 1$ be the factor by which FPs are boosted, and $\gamma_n > 1$ be the factor by which FNs are boosted. In general, the factors for TP and TN can also be different, but for simplifying the presentation of our analysis we assume them to be identical. In fact for many boosting algorithms including AdaBoost [12] and AdaCost [6], this is true [10].

Using these parameters and the notations of Table 1 and Figure 3, our goal is to express the accuracy of C2, ν_{C2}, in terms of weights of distribution D^t. We start with the conjecture that the weighted accuracy of C1 is $\nu_{C1} = 0.5+\delta$, where $\delta > 0$.

Note that boosting ensures that each iteration starts on a *distribution* of weights; i.e. the sum of all weights is equal to 1. This allows us to express accuracy of C1 or C2 as the sum of their respective true positives and true negatives. Using notations of section 3.1, $\nu_{C1} = \tau + \tau_n$. Let us denote the sum of weights on the true positives, true negatives, and false positives of C2 with respect to the D^{t+1} distribution, by τ', τ'_n, and ϕ', respectively[3]. Hence, $\nu_{C2} = \tau' + \tau'_n$.

We now express τ' and τ'_n in terms of the parameters of C1 and weights of distribution D^t. One can decompose τ' as $\tau'_{p,o}+\tau'_{p,no}$, where $\tau'_{p,o}$ corresponds to the true positives of C2 that overlap with true positive predictions of C1, and $\tau'_{p,no}$ corresponds to the non-overlapping part. The reason for this decomposition is that the weights on each of these parts were updated with different factors. Now, using the parameters of weight update mechanism, we can write $\tau'_{p,o} = \tau_{p,o}/(\gamma z)$, where $\tau_{p,o}$ is the overlapping part of the TPs of C1. This expression comes from the fact that the weights of TPs of C1 were suppressed by a factor γ and normalized by the z factor to achieve a sum of 1 on weights of distribution D^{t+1} (it can be verified that $z = ((\tau + \tau_n)/\gamma) + \gamma_p\phi + \gamma_n\phi_n)$. Using Figure 3, we can further conclude that $\tau_{p,o} = \omega\tau$. Thus, $\tau'_{p,o} = \tau\omega/(\gamma z)$. The non-overlapping part of τ' corresponds to the false negative predictions of C1, which were boosted by a factor of γ_n and normalized by z; hence, it can be expressed as $\tau'_{p,no} = \phi_{n,o}\gamma_n/z$, where $\phi_{n,o}$ is the part of C1's FNs that overlap with C2's TPs. Again, using Figure 3, we can write $\tau'_{p,no} = \tau\theta(1-\omega)\gamma_n/z$. Overall, one can express

$$\tau' = \tau(\theta(1-\omega)\gamma_n + \frac{\omega}{\gamma})/z. \quad (1)$$

Using same arguments as for τ', it can be verified that false positives of C2, $\phi' = \phi(\frac{\beta(1-\omega)}{\gamma} + \omega\gamma_p)/z$. This uses the fact that the weight on overlapping false positives in-

[3]Parameter for false negatives of C2 is not specifically defined because it is not explicitly required in our analysis. Also, given that sum of all weights is 1, it is not a free parameter.

creases by a factor of $\gamma_p z$, and non-overlapping part weight decreases by a factor of γz, after C1 is learned.

Now, τ'_n can be expressed as $n' - \phi'$, where n' is the total weight of NC's examples in distribution D^{t+1}, which is $n' = ((\tau_n/\gamma) + \phi\gamma_p)/z$, following the weight update mechanism. Using $\nu_{C1} = \tau_n + \tau = 0.5 + \delta(\delta > 0)$, and expression for ϕ' above, we can express τ'_n as

$$\tau'_n = \frac{1}{z}\left(\frac{0.5+\delta-\tau}{\gamma} + \phi\left(\gamma_p - \frac{\beta(1-\omega)}{\gamma} - \omega\gamma_p\right)\right). \quad (2)$$

From the formulae 1 and 2, $\nu_{C2} = \tau' + \tau'_n$ can be expressed as,

$$\nu_{C2} = \frac{1}{z}\left(\frac{0.5+\delta}{\gamma} + (1-\omega)\left(\tau\left(\theta\gamma_n - \frac{1}{\gamma}\right) + \phi\left(\gamma_p - \frac{\beta}{\gamma}\right)\right)\right).$$

Using this, and expressing τ and ϕ in terms of recall (R) and precision (P) of C1 as $\tau = R\rho$ and $\phi = ((1/P)-1)R\rho$, some simple algebraic manipulations prove the following theorem:

THEOREM 1. *For the weighted accuracy of C2 to be > 0.5, under the assumptions that C1's accuracy is $0.5+\delta$ ($\delta > 0$), the overlap factor between C1 and C2's positive predictions must satisfy*

$$\omega < 1 - \frac{0.5(\gamma z - 1) - \delta}{Q\gamma}, \text{ if } \gamma z > 1 + 2\delta, Q > 0, \quad (3)$$

or

$$\omega > 1 - \frac{0.5(\gamma z - 1) - \delta}{Q\gamma}, \text{ if } \gamma z < 1 + 2\delta, Q < 0, \quad (4)$$

where Q is defined as

$$Q = R\rho\left(\theta\gamma_n - \frac{1}{\gamma} + \left(\frac{1}{P} - 1\right)\left(\gamma_p - \frac{\beta}{\gamma}\right)\right), \quad (5)$$

and z is defined as

$$z = \frac{0.5+\delta}{\gamma} + R\rho\left(\gamma_p\left(\frac{1}{P} - 1\right) + \gamma_n\left(\frac{1}{R} - 1\right)\right). \quad (6)$$

If the conditions $\gamma z > 1 + 2\delta$ and $Q < 0$ are both satisfied simultaneously, no solution exists for ω; implying some parameter values are infeasible.

If $\gamma z < 1 + 2\delta$ and $Q > 0$, then every possible value of ω can make weighted accuracy of C2 > 0.5. □

This theorem essentially offers a tool to analyze the tolerance allowed on the overlap between the two consecutive classifiers, for given values of ρ, R, and P, in order to achieve the condition on base learner's weighted accuracy imposed by the boosting mechanism. Note that ρ corresponds to the proportion of class C according to D^t distribution.

At first sight, it may seem that there are too many free parameters in the theorem to make it practical. However, one can make use of some more information to reduce the number of free parameters.

First thing to note is that the range of ω is also naturally limited by the choice of θ and β. Because, from Figure 3, $0 \leq \theta(1-\omega)\tau \leq (\rho - \tau)$ and $0 \leq \beta(1-\omega)\phi \leq 1 - \rho - \phi$ hold. Thus, choice of θ and β values impose following limits on ω:

$$\omega \geq \omega_\theta; \omega_\theta = 1 - \left(\frac{\rho - \tau}{\theta\tau}\right) \quad (7)$$

$$\omega \geq \omega_\beta; \omega_\beta = 1 - \left(\frac{1 - \rho - \phi}{\beta\phi}\right) \quad (8)$$

If Theorem 1 imposes a lower limit of ω_{th} on ω, then ultimately we can verify that $\omega \geq max(\omega_{th}, \omega_\theta, \omega_\beta)$.

The values of δ, R, P, and ρ are interrelated via different equations: $\delta = 0.5 - R\rho((1/R) + (1/P) - 2)$, $\phi \leq 1 - \rho$, and $P \geq 1/(1 + (1-\rho)/(R\rho))$. A specific boosting algorithm can be assumed, such as AdaCost [6], to determine the values of the parameters γ, γ_p, and γ_n. For AdaCost, if the cost factor associated with FNs is f, the parameters are defined as follows [10]: $\gamma_{Acst} = e^{\alpha_{Acst}}$, $\gamma_{n,Acst} = \gamma^2_{Acst}$, and $\gamma_{p,Acst} = \gamma^{(f+1)/f}_{Acst}$, where $\alpha_{Acst} = 1/2 \ln((1 + r_{Acst})/(1 - r_{Acst}))$. r_t is the cost- and sign-weighted sum of all the examples; i.e., $r_{Acst} = 0.5(\tau + \tau_n) - 0.5\phi_n(1+f) - \phi$.

Finally, we show a relationship between recall and precisions of C2 in terms of the recall (R) and precision (P) of C1. Using Figure 3, the recall and precision values of C2 with respect to the distribution D^t, are:

$$R_{C2} = R(\omega + \theta(1-\omega)) \quad (9)$$

$$P_{C2} = P\left(\frac{\omega + \theta(1-\omega)}{\omega + (1-\omega)(\beta + P(\theta - \beta))}\right) \quad (10)$$

From these, one can see how the choice of θ, β, and ω affect the relationships between R (resp. P) and R_{C2} (resp. P_{C2}). For example, if $\theta = \beta$, then $P_{C2} = P$; i.e. we are seeking the existence of weak learners that maintain their precision across iterations with respect to the original distribution. If $\theta = \beta = 1$, then $R_{C2} = R$ as well as $P_{C2} = P$; i.e. we are seeking the existence of weak learners that maintain identical recall and precision throughout the boosting iterations. In general, the choice of θ and β determines the class of weak learners allowed to be generated in the boosting process.

From this point on, we deviate from the rigorous mathematical analysis, and conduct some simulation experiments to show what we started off to show; i.e., the situations of type A and B (Figure 2), cannot be avoided for a specific case of keeping recall and precision identical in all iterations; i.e. the case of $\theta = \beta = 1$.

3.3.1 Analysis for AdaCost for $\theta = \beta = 1$

Let us see how the overlap allowed between two consecutive classifiers varies with the variation in rarity, recall, and precision numbers of a given classifier. We present simulation results for AdaCost, one of the best performing algorithms from recall-precision perspective [10].

The variation in ω_{min}, minimum limit on ω, for various values of R, P, and ρ is shown in Figure 4(a). The figure is for $f = 1$ (cost-factor in AdaCost). Each graph shows for a given value of P and ρ, how ω_{min} varies with R. A point with $\omega_{min} = 1$ implies that combination of R, P, and ρ corresponding to that point is infeasible to satisfy the requirement that $\nu_{C2} > 0.5$. Such points correspond to the failure of one of the requirements: $\delta > 0$, $\gamma > 1$, $\phi \leq (1-\rho)$, or the one stated in Theorem 1.

The key observation from this figure is this: for a wide range of combinations of R and P, the ω value is unrestricted, especially for lower ρ values; i.e. the more rare classes. This implies that one can have any kind of base learner, and still maintain that each of them achieves an accuracy better than random guessing. Especially a base learner that has low recall (much less than 50%) and high precision (e.g., plot for $P = 0.8, \rho = 0.01$), is allowed to be used by the weight update mechanism. Note that this is the precise description of the type A classifiers of Figure 2. Base learner that has sufficiently high recall and low precision is also allowed, as is evident from the plot for $P = 0.1, \rho = 0.01$. This corresponds to the type B classifiers of Figure 2. Note that a recall of around 50% is also very high when looking at each individual base learner. We can see that ω_{min} is 0 (i.e. ω can range from 0 to 1) especially for $R \leq 0.5$ and lower values of ρ. So, within this range all the three types of classifiers are allowed (plot for $P = 0.4, \rho = 0.01$ can be thought of as corresponding to the type C learner).

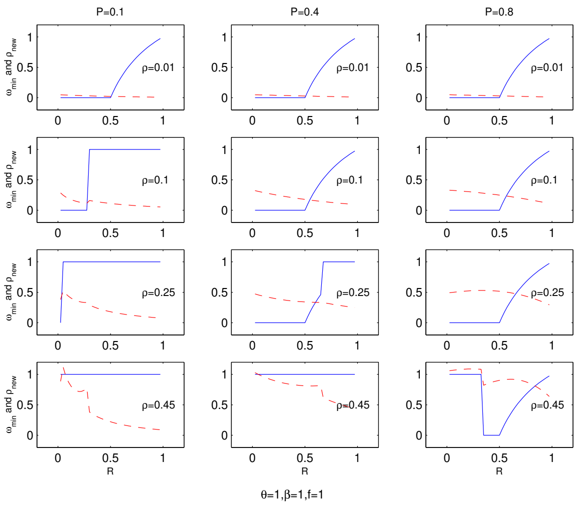
Even for $R > 0.5$, it can be verified that the value of ω becomes unrestricted within the *naturally* allowed range, as ρ becomes smaller and precision becomes higher. This natural limitation is imposed by the specific choice of θ and β. For example, for $\rho = 0.01$, the curves for $R > 0.5$ are identical across all precision levels, and this curve is precisely the one dictated by equations 7 and 8. This essentially indicates that type B and C classifiers are allowed even for their recall values of > 0.5.

Second observation from this plots is that as the rarity of the class decreases from top to bottom, ω_{min} starts getting closer to 1; i.e., less and less variation is allowed in ω. This is especially seen for lower values of P, which means that type B classifiers (high recall, low precision) will not be allowed for very low values of P (e.g. plot for $P = 0.1$, $\rho = 0.1$). Also, type A classifiers (low recall, high precision) will not be allowed (e.g. plot for $P = 0.1$, $\rho = 0.25$). However for relatively higher values of P, both types of classifiers, especially those of type A (plot for $P = 0.8$ and $\rho = 0.25$), are allowed for not-so-rare classes also.

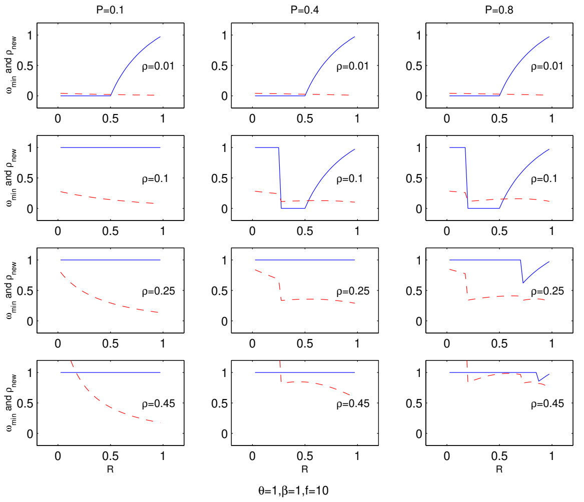
Third observation to make is regarding the validity of incremental analysis for more than just two iterations for which the plots are drawn. Each graph also shows in dashed line the ρ_{new} value; i.e. the proportion of C in distribution D^{t+1}. The proportion does not change much from ρ, especially for lower values of ρ. This illustrates that this incremental analysis is also valid if one does it going from the $(t+1)^{st}$ to $(t+2)^{nd}$ iteration, and most probably for many iterations after that.

Final observation is that as the f value increases in AdaCost, as shown in Figure 4(b); the range of overlap starts getting smaller, but still for rare classes (e.g. $\rho = 0.01$) and higher precision values, there is a wide choice of ω allowed. Thus, type A classifiers (high precision) can exist for a wider range of f values also. Type B classifiers (low precision, high recall) can also exist if the precision becomes lower and lower.

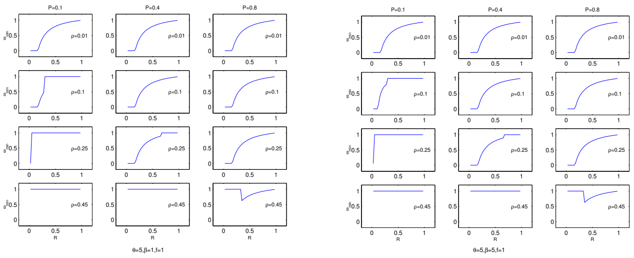
We also analyzed the more general cases. $\theta > 1, \beta = 1$ allows recall to increase progressively (left part of Figure 4(c)). As θ increases, the threshold on R beyond which the natural limitation starts applying decreases (its around 0.33 for $\theta = 2$ and 0.17 for $\theta = 5$, as against 0.5 for $\theta = 1$), but within this naturally limited range, ω can assume any value. The effect of $\theta = 1, \beta > 1$ is shown in the right part of Figure 4(c). As β value increases, ω_{min} increases more sharply after the R threshold. This implies that the ω value gets more restricted. For $\theta = \beta > 1$ (that allows constant precision with variable recall), the sharper rise in ω_{min} curve is accompanied by the lowering of threshold. There is still a wide variation allowed in the ω values, especially as ρ becomes smaller; i.e. for rarer classes.



(a)



(b)



(c)

Figure 4: How ω_{min} (solid line) and ρ_{new} (dashed line) vary with R, P, and ρ. (a) $\theta = 1, \beta = 1, f = 1$, (b) $\theta = 1, \beta = 1, f = 10$, (c) Left: $\theta = 5, \beta = 1, f = 1$. Right: $\theta = 5, \beta = 5, f = 1$. P increases from left to right, ρ increases from top to bottom, and R varies within each graph. In some PDF viewers, plots may show w in place of ω, r for ρ, q for θ, and b for β.

3.3.2 Summary

From the simulation results shown above, and the theoretical tools developed earlier, we can conclude that weight update mechanism does not provide any safeguard against making a base learner achieve identical recall and precision values in many iterations, and thereby having situations of types A, B, C of Figure 2, when the class becomes more and more rare. This argument can be extended for other patterns of recall-precision values also, such as precision remaining the same with recall increasing steadily. The general understanding is that the three types of classifiers can exist under different patterns, especially for rare classes.

4. RESULTS

In this section, we attempt to validate the arguments presented in the previous sections further. In particular, we illustrate how the choice of different base learning algorithms affects the performance of AdaCost [6], a cost-sensitive boosting algorithm. The experimental setting is as follows.

We use three different forms of base learners: RIP0-S, RIP0-M, and PN-S. RIP0-S is the RIPPER0 (= IREP* [5]) algorithm which builds a single model for the given rare class. RIP0-M learns two models with RIPPER0, one for the rare class and other for the non-rare class. PN-S is the PNrule algorithm [9] that builds single model for the rare class. We use PNrule parameters of $rp = 0.7$ and $rn = 0$. Details of AdaCost, RIPPER, and PNrule can be found in the respective papers: [6], [5], and [9], but it suffices to know that we are using a strong boosting algorithm and weak forms of base learner algorithms.

Our performance criterion is based on F_1-measure [14], which for a class C is $2*(R*P)/(R+P)$, where R and P are recall and precision of C. F_1 $(0 \leq F_1 \leq 1)$ is high if both R and P are high, and is dominated by the smaller of the two. Given no a-priori knowledge of costs associated with R and P, use of F_1 seeks a balance between the two.

For AdaCost, we use various values for f: 1, 2, 5, 10. For each f value, we run AdaCost for 25 iterations with RIP0-M and PN-S, and 50 iterations with RIP0-S. After every iteration, we monitor recall, precision, and F_1-measure on the validation data which is a one-third random sample of the training data. We choose parameters f and number of iterations, that give the best F_1 value on this data as the optimal values, and report results of AdaCost using these values on the test data.

4.1 Datasets Used

We have conducted experiments on many different synthetic and real datasets. Here, we just present the key results.

We generate different synthetic datasets from the model described in Figure 5, which shows class distributions over different types of attributes. For example, attributes of type AC_b have distinguishing signature peaks for the subclasses of type C_b. C_a and C_b are the subclasses of the rare class C, and all subclasses of types NC_a and NC_b form non-rare class NC. Note that there is a correlation between in the attributes $ACORR_a$ and ANC_a for every subclass of type NC_a. Different parameters of the models are the number of subclasses of each type, the widths of the signature peaks, number of signature peaks for each type of attribute, and of course, the proportion of C vs. NC (the rarity). We vary these parameters to vary the learning difficulty in terms of achieving high recall and precision values. For example, if the peaks of AC_b are wider, number of false positives captured (i.e., the examples of NC_a and NC_b) will be more. So, an algorithm's ability to achieve higher precision for a given recall will be tested. Also, the difficulty increases as the peaks of ANC_a and ANC_b becomes wider, because precision of C suffers (especially for algorithms PN-S and RIP0-M, which learn explicit models for NC).

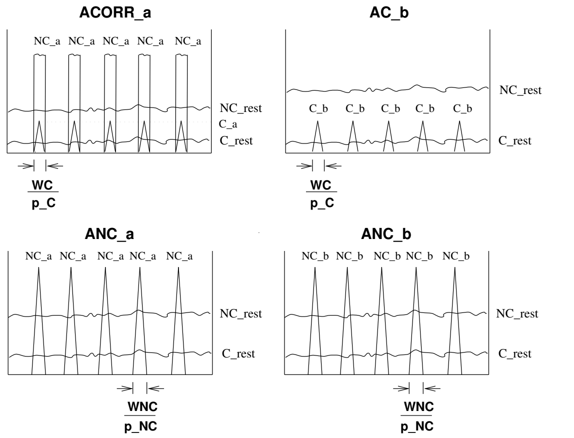


Figure 5: *Description of the model generating synthetic datasets with correlations among attributes.*

We generate two main datasets from this model: sc-0 and sc-1. Each one has a 10% population of class C out of 22,000 total examples. sc-0 has parameters $n_{ACORR_a} = 0$, $n_{AC_b} = 3$, $n_{ANC_a} = 0$, $n_{ANC_b} = 7$, whereas sc-1 has parameters $n_{ACORR_a} = 3$, $n_{AC_b} = 0$, $n_{ANC_a} = 3$, $n_{ANC_b} = 4$. Here n_A indicates number of attributes of type A. These Both these datasets have 4 signature peaks in subclasses of C and 8 peaks in subclasses of NC. The widths as defined in figure are $W_C = 0.2$, $W_{NC} = 2$, which should be compared against the attribute value range of 50.

We formulate a set of real-world problems from the OHSU Medline document corpus of medical abstracts from years 1990 and 1991 [7]. Each document has multiple topics assigned to it such as Infant, Infection, Cell_Line, etc. There are around 148,000 documents with 73,700 words remaining after stemming and stop-word removal. We select a few of the topics which have 300 more documents ($> 0.2\%$ population). For each topic, we select top 75 distinguishing words according to two different evaluation metrics, and take their intersection to choose the final set of feature words. This way, we form a different binary classification problem for each topic.

We form the final set of problems using the king-rook-king dataset of the UCI machine learning repository [1]. In particular we form binary problems for different classes appearing in the data-set. Almost all the classes have a proportion $< 15\%$.

4.2 Comparing Base Learner Effect

Here, we present results that compare the effect of base learner on the F_1 value. We use ACst-RS, ACst-RM, and ACst-PN to denote the AdaCost algorithm formed by using RIP0-S, RIP0-M, and PN-S as the base learners. We want to see if the relative performance of base learners reflects in the same relative performance of AdaCost based on them.

The results shown in Table 2 show the effect of having correlations in the synthetic model (sc-0 vs. sc-1).

We also varied various parameters of sc-1 dataset to see how the relative performance is affected (Table 3). In most cases, for this data-set, PN-S performs better than the other two base learners. As the results for ACst indicates, ACst-PN is also significantly better than ACst-RS and ACst-RM. Sometimes, the performance improvement obtained by using the right base learner is more than factor of 2 (e.g. datasets with $p_{NC} = 20$, $W_C = 0.001$, $W_C = 0.04$). From the maximum achievable F_1 value in all these datasets, one can see that the relative performance of base learners and their effect on boosting is valid for a wide range of datasets.

We also show the effect of varying the target class proportion (Table 4). As the rarity decreases, the effect of choosing a particular base learner diminishes. For example, for 50% population of the target class, all the variations of ACst are equally well.

On the OHSU Medline corpus, the results are shown in Table 5. Though not as dramatic as synthetic datasets, these datasets also show that base learner choice matters, sometimes significantly (HIV, Colon, Adenocarcinoma). A point to observe is that: when two base learners are comparable by themselves, boosting performance based on them may either be comparable or significantly better for one (Anoxia). This still supports our point that choice of the base learner matters.

Table 6 shows results on the various binary problems formed from the king-rook-king dataset. The base learner's effect here is again quite significant for many cases (observe classes nine, ten, eleven, thirteen, fourteen). When base learners are comparable, boosting performance is either comparable (class eight) or significantly better (class five).

Overall, our results indicate: if one chooses a base learner that performs better in a stand-alone comparison among such learners, the the boosting algorithm based on it will in general give comparable or significantly better (but very rarely poor) performance.

5. CONCLUSION

Boosting has emerged as a strong classification technique is recent years. It is claimed that it can improve the performance of any weak learner. In this paper, we attempt to critically evaluate this claim in the context of an important data mining problem of achieving better recall and precision balance for the given *rare* or infrequently occurring classes. Using strong mathematical and qualitative arguments, we show that for rare class scenarios, the ability of a boosting algorithm is critically dependent on the abilities of its base learner. We analyze each component of the boosting: the accuracy condition, the voting process, and the weight update mechanism, and also show empirical evidence on different synthetic, real, and benchmark data sets, in order to support our claim. We show that in many cases, the performance improvement obtained with the right base learner, can be dramatic, such as better than factor of two. We also demonstrate that as the rarity disappears, boosting performance depends less on the choice of the base learner. A very practical outcome of our analysis and results is that for rare class problems, one can decide which base learner to use by just comparing their stand-alone performance. Using the best base learner extracts the best or comparable to the best performance out of boosting. This we believe is a very valuable insight.

Acknowledgments: The authors would like to thank Aleksandar Lazarevic and Pusheng (Alex) Zhang of Army HPC Center, University of Minnesota, for their valuable comments.

6. REFERENCES

[1] C. Blake and C. Merz. UCI repository of machine learning databases, 1998. http://www.ics.uci.edu/~mlearn/MLRepository.html.

[2] L. Breiman. Arcing classifiers. *The Annals of Statistics*, 26(3):801–849, 1998.

[3] P. Chan and S. Stolfo. Towards scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection. In *Proc. of Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98)*, pages 164–168, New York City, 1998.

[4] W. Cohen and Y. Singer. A simple, fast, and effective rule learner. In *Proc. of Annual Conference of American Association for Artificial Intelligence*, pages 335–342, 1999.

[5] W. W. Cohen. Fast effective rule induction. In *Proc. of Twelfth International Conference on Machine Learning*, Lake Tahoe, California, 1995.

[6] W. Fan, S. J. Stolfo, J. Zhang, and P. K. Chan. AdaCost: Misclassification cost-sensitive boosting. In *Proc. of Sixth International Conference on Machine Learning (ICML-99)*, Bled, Slovenia, 1999.

[7] W. Hersh, C. Buckley, T. Leone, and D. Hickam. Ohsumed: An interactive retrieval evaluation and new large test collection for research. In *In Proceedings of the 17th Annual ACM SIGIR Conference*, pages 192–201, 1994.

[8] R. C. Holte, N. Japkowicz, C. X. Ling, and S. M. (eds.). Learning from imbalanced data sets (papers from aaai workshop). Technical Report WS-00-05, AAAI Press, Menlo Park, CA, 2000.

[9] M. V. Joshi, R. C. Agarwal, and V. Kumar. Mining needles in a haystack: Classifying rare classes via two-phase rule induction. In *Proc. of ACM SIGMOD Conference*, pages 91–102, Santa Barbara, CA, 2001.

[10] M. V. Joshi, V. Kumar, and R. C. Agarwal. Evaluating boosting algorithms to classify rare classes: Comparison and improvements. In *Proc. of The First IEEE International Conference on Data Mining (ICDM)*, San Jose, CA, Nov 2001.

[11] M. J. Kearns and U. V. Vazirani. *An Introduction to Computational Learning Theory*. The MIT Press, 1994.

[12] R. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.

[13] K. M. Ting. A comparative study of cost-sensitive boosting algorithms. In *Proc. of 17th International Conf. on Machine Learning*, pages 983–990, Stanford University, CA, 2000.

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

DS	ρ	RIP0-S	RIP0-M	PN-S	ACst-RS	ACst-RM	ACst-PN
sc-0	9.137 %	85.45	50.48	88.45	88.93	89.08	88.45
sc-1	9.137 %	2.33	41.65	59.55	35.86	41.65	77.58

Table 2: Comparing classifiers on synthetic datasets

DS	ρ	RIP0-S	RIP0-M	PN-S	ACst-RS	ACst-RM	ACst-PN
$p_{NC} = 2$	9.149 %	0.00	67.75	61.45	54.56	72.00	86.29
$p_{NC} = 4$	9.137 %	7.69	32.11	51.53	60.96	58.23	81.25
$p_{NC} = 8$	9.137 %	2.33	41.65	59.55	35.86	41.65	77.58
$p_{NC} = 12$	13.043 %	41.17	19.94	53.94	50.13	47.40	71.07
$p_{NC} = 20$	9.091 %	2.12	3.90	34.10	27.94	28.57	60.68
$W_C = 0.001$	9.137 %	8.01	1.77	47.78	38.69	33.70	83.28
$W_C = 0.004$	9.137 %	2.33	41.65	59.55	35.86	41.65	77.58
$W_C = 0.04$	9.137 %	0.20	0.00	44.74	24.33	25.58	49.52

Table 3: Comparing classifiers on variations of sc-1 parameters

ρ	RIP0-S	RIP0-M	PN-S	ACst-RS	ACst-RM	ACst-PN
2.452 %	0.00	0.00	12.60	2.45	0.76	51.02
4.787 %	1.55	0.00	20.18	15.38	11.44	48.48
9.137 %	2.33	41.65	59.55	35.86	41.65	77.58
16.667 %	3.22	0.39	79.73	45.80	44.56	81.98
28.571 %	2.70	23.09	80.48	60.95	59.89	84.17
50.000 %	65.01	63.70	78.53	78.44	78.44	78.53

Table 4: Comparing classifiers when target class proportion changes in sc-1 dataset

DS	ρ	RIP0-S	RIP0-M	PN-S	ACst-RS	ACst-RM	ACst-PN
Anoxia	1.252	40.29	40.58	47.93	47.06	43.71	51.71
HIV	1.376	0.00	4.05	15.01	3.51	12.55	15.53
Internship_and_Residency	2.178	57.29	55.91	53.98	57.29	55.91	54.07
Adenocarcinoma	2.267	43.72	51.92	53.75	52.65	55.61	56.93
Heart	2.748	2.11	1.28	31.84	34.83	33.91	36.39
Colon	4.878	7.92	7.92	37.50	42.39	37.58	45.56
Molecular_Sequence_Data	7.136	60.43	59.91	66.06	67.52	67.54	69.35

Table 5: Comparing classifiers on some topics from OHSU Medline corpus

DS	ρ	RIP0-S	RIP0-M	PN-S	ACst-RS	ACst-RM	ACst-PN
krkopt-1_sixteen	1.404	25.71	50.92	56.35	47.87	66.67	70.17
krkopt-1_five	1.861	2.26	2.26	63.48	56.01	64.24	65.84
krkopt-1_eight	5.182	5.51	5.05	52.74	52.34	52.97	61.84
krkopt-1_nine	5.988	4.15	31.48	43.35	48.88	51.31	59.11
krkopt-1_ten	6.808	0.21	0.00	42.08	44.92	48.79	54.63
krkopt-1_fifteen	7.685	17.26	61.89	66.07	53.26	66.03	72.09
krkopt-1_eleven	10.372	0.82	0.00	49.00	46.17	49.59	58.62
krkopt-1_thirteen	15.227	12.38	44.54	58.51	48.80	52.27	61.56
krkopt-1_fourteen	16.317	26.66	61.73	61.73	51.76	64.47	72.90

Table 6: Comparing classifiers on king-rook-king problem from UCI Repository