Learning with Class Skews and Small Disjuncts

Ronaldo C. Prati, Gustavo E. A. P. A. Batista, and Maria C. Monard

Institute of Mathematics and Computer Science at University of São Paulo
P. O. Box 668, ZIP Code 13560-970, São Carlos, SP, Brazil
{prati,gbatista,mcmonard}@icmc.usp.br

Abstract. One of the main objectives of a Machine Learning – ML – system is to induce a classifier that minimizes classification errors. Two relevant topics in ML are the understanding of which domain characteristics and inducer limitations might cause an increase in misclassification. In this sense, this work analyzes two important issues that might influence the performance of ML systems: class imbalance and error-prone small disjuncts. Our main objective is to investigate how these two important aspects are related to each other. Aiming at overcoming both problems we analyzed the behavior of two over-sampling methods we have proposed, namely Smote + Tomek links and Smote + ENN. Our results suggest that these methods are effective for dealing with class imbalance and, in some cases, might help in ruling out some undesirable disjuncts. However, in some cases a simpler method, Random over-sampling, provides compatible results requiring less computational resources.

1 Introduction

This paper aims to investigate the relationship between two important topics in recent ML research: learning with *class imbalance* (class skews) and *small disjuncts*. Symbolic ML algorithms usually express the induced concept as a set of rules. Besides a small overlap within some rules, a set of rules might be understood as a disjunctive concept definition. The size of a disjunct is defined as the number of training examples it correctly classifies. Small disjuncts are those disjuncts that correctly cover only few training cases. In addition, class imbalance occurs in domains where the number of examples belonging to some classes heavily outnumber the number of examples in the other classes. Class imbalance has often been reported in the ML literature as an obstacle for the induction of good classifiers, due to the poor representation of the minority class. On the other hand, small disjuncts have often been reported as having higher misclassification rates than large disjuncts. These problems frequently arise in applications of learning algorithms in real world data, and several research papers have been published aiming to overcome such problems. However, these efforts have produced only marginal improvements and both problems still remain open. A better understanding of how class imbalance influences small disjuncts (and of course, the inverse problem) may be required before meaningful results might be obtained.

Weiss [1] suggests that there is a relation between the problem of small disjuncts and class imbalance, stating that one of the reasons why small disjuncts have a higher error rate than large disjuncts is due to class imbalance. Furthermore, Japkowicz [2] enhances this hypothesis stating that the problem of learning with class imbalance is potentiated when it yields small disjuncts. Even though these papers point out a connection between such problems, the true relationship between them is not yet well-established. In this work, we aim to further investigate this relationship.

This work is organized as follows: Section 2 reports some related work and points out some connections between class imbalance and small disjuncts. Section 3 describes some metrics for measuring the performance of ML algorithms regarding small disjuncts and class skews. Section 4 discusses the experimental results of our work and, finally, Section 5 presents our concluding remarks and outlines future research directions.

2 Related Work

Holt et al. [3] report two main problems when small disjuncts arise in a concept definition: (a) the difficulty in reliably eliminating the error-prone small disjuncts without producing an undesirable net effect on larger disjuncts and; (b) the algorithm maximum generality bias that tends to favor the induction of good large disjuncts and poor small disjuncts.

Several research papers have been published in the ML literature aiming to overcome such problems. Those papers often advocate the use of pruning to draw small disjuncts off the concept definition [3, 4] or the use of alternative learning bias, generally using hybrid approaches, for coping with the problem of small disjuncts [5]. Similarly, class imbalance has been often reported as an obstacle for the induction of good classifiers, and several approaches have been reported in the literature with the purpose of dealing with skewed class distributions. These papers often use sampling schemas, where examples of the majority class are removed from the training set [6] or examples of the minority class are added to the training set [7] in order to obtain a more balanced class distribution. However, in some domains standard ML algorithms induce good classifiers even using highly imbalanced training sets. This indicates that class imbalance is not solely accountable for the decrease in performance of learning algorithms. In [8] we conjecture that the problem is not only caused by class skews, but is also related to the degree of data overlapping among the classes.

A straightforward connection between both themes can be traced by observing that minority classes may lead to small disjuncts, since there are fewer examples in these classes than in the others, and the rules induced from them tend to cover fewer examples. Moreover, disjuncts induced to cover rare cases are likely to have higher error rates than disjuncts that cover common cases, as rare cases are less likely to be found in the test set. Conversely, as the algorithm tries to generalize from the data, minority classes may yield some small disjuncts to

Table 1. Confusion matrix for a two-class problem.

	Positive Prediction	*Negative Prediction*
Positive Class	True Positive (TP)	False Negative (FN)
Negative Class	False Positive (FP)	True Negative (TN)

be ruled out from the set of rules. When the algorithm is generalizing, common cases can "overwhelm" a rare case, favoring the induction of larger disjuncts.

Nevertheless, it is worth noticing the differences between class imbalance and small disjuncts. Rare cases exist in the underlying population from which training examples are drawn, while small disjuncts might also be a consequence of the learning algorithm bias. In fact, as we stated before, rare cases might have a dual role regarding small disjuncts, either leading to undesirable small disjuncts or not allowing the formation of desirable ones, but rather small disjuncts might be formed even though the number of examples in each class is naturally equally balanced. In a nutshell, class imbalance is a characteristic of a domain while small disjuncts are not [9].

As we mentioned before, Weiss [1] and Japkowicz [2] have suggested that there is a relation between both problems. However, Japkowicz performed her analysis on artificially generated data sets and Weiss only considers one aspect of the interaction between small disjuncts and class imbalances.

3 Evaluating Classifiers with Small Disjuncts and Imbalanced Domains

From hereafter, in order to facilitate our analysis, we constrain our discussion to binary class problems where, by convention, the minority is called **positive class** and the majority is called **negative class**. The most straightforward way to evaluate the performance of classifiers is based on the confusion matrix analysis. Table 1 illustrates a confusion matrix for a two-class problem. A number of widely used metrics for measuring the performance of learning systems can be extracted from such a matrix, such as error rate and accuracy. However, when the prior class probabilities are very different, the use of such measures might produce misleading conclusions since those measures do not take into consideration misclassification costs, are strongly biased to favor the majority class and are sensitive to class skews.

Thus, it is more interesting to use a performance metric that disassociates the errors (or hits) that occur in each class. Four performance metrics that directly measure the classification performance on positive and negative classes independently can be derived from Table 1, namely true positive rate – $TP_{rate} = \frac{TP}{TP+FN}$ – (the percentage of correctly classified positive examples), false positive rate – $FP_{rate} = \frac{FP}{FP+TN}$ – (the percentage of incorrectly classified positive examples), true negative rate – $TN_{rate} = \frac{TN}{FP+TN}$ – (the percentage of correctly classified negative examples) and false negative rate – $FN_{rate} = \frac{FN}{TP+FN}$ – (the percentage of incorrectly classified negative examples). These four performance metrics have the advantage of being independent of class

costs and prior probabilities. The aim of a classifier is to minimize the false positive and negative rates or, similarly, to maximize the true negative and positive rates. Unfortunately, for most real world applications there is a tradeoff between FN_{rate} and FP_{rate}, and similarly between TN_{rate} and TP_{rate}.

ROC (Receiver Operating Characteristic) analysis enables one to compare different classifiers regarding their true positive rate and false positive rate. The basic idea is to plot the classifiers performance in a two-dimensional space, one dimension for each of these two measurements. Some classifiers, such as the Naïve Bayes classifier and some Neural Networks, yield a score that represents the degree to which an example is a member of a class. For decision trees, the class distributions on each leaf can be used as a score. Such ranking can be used to produce several classifiers by varying the threshold of an example to be classified into a class. Each threshold value produces a different point in the ROC space. These points are linked by tracing straight lines through two consecutive points to produce a ROC curve. The area under the ROC curve (AUC) represents the expected performance as a single scalar. In this work, we use a decision tree inducer and the method proposed in [10] with Laplace correction for measuring the leaf accuracy to produce ROC curves.

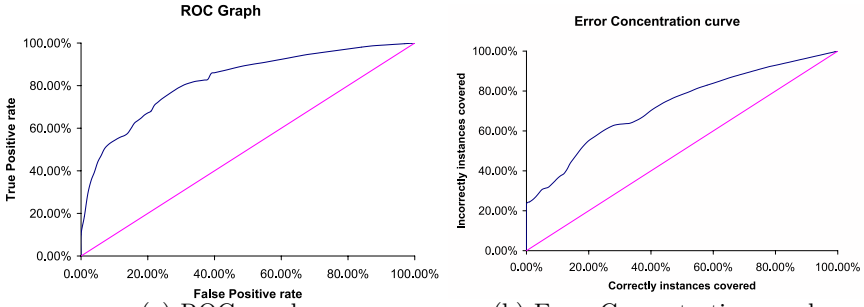
In order to measure the degree to which errors are concentrated towards smaller disjuncts, Weiss [1] introduced the Error Concentration (EC) curve. The EC curve is plotted starting with the smallest disjunct from the classifier and progressively adding larger disjuncts. For each iteration where a larger disjunct is added, the percentage of test errors *versus* the percentage of correctly classified examples is plotted. The line $Y = X$ corresponds to classifiers having errors equally distributed towards all disjuncts. Error Concentration is defined as the percentage of the total area above the line $Y = X$ that falls under the EC curve. EC may take values from between 100%, which indicates that the smallest disjunct(s) covers all test errors before even a single correctly classified test example is covered, to -100%, which indicates that the largest disjunct(s) covers all test errors after all correctly classified test examples have been covered.

In order to illustrate these two metrics Figure 1 shows the ROC (Fig. 2(a)) and the EC (Fig. 2(b)) graphs for the pima data set and pruned trees – see Table 3. The AUC for the ROC graph is 81.53% and the EC measure from the EC graph is 42.03%. The graphs might be interpreted as follows: from the ROC graph, considering for instance a false positive rate of 20%, one might expect a true positive rate of nearly 65%; and from the EC graph, the smaller disjuncts that correctly cover 20% of the examples are responsible for more than 55% of the misclassifications.

4 Experimental Evaluation

The aim of our research is to provide some insights into the relationship between class imbalances and small disjuncts. To this end, we performed a broad experimental evaluation using ten data sets from UCI [11] having minority class distribution spanning from 46.37% to 7.94%, *i.e.*, from nearly balanced to skewed

Fig. 1. ROC and EC graphs for the pima data set and pruned trees.



(a) ROC graph (b) Error Concentration graph

distributions. Table 2 summarizes the data sets employed in this study. It shows, for each data set, the number of examples (#Examples), number of attributes (#Attributes), number of quantitative and qualitative attributes and class distribution. For data sets having more than two classes, we chose the class with fewer examples as the positive class, and collapsed the remainder classes as the negative class.

Table 2. Data sets summary descriptions.

Data sets	#Examples	#Attributes (quanti., quali.)	Classes (min., maj.)	Classes % (min., maj.)
Sonar	207	60 (60, 0)	(R, M)	(46.37%, 53.63%)
Bupa	345	6 (6, 0)	(1, 2)	(42.03%, 57.97%)
Pima	768	8 (8, 0)	(1, 0)	(34.77%, 65.23%)
German	1000	20 (7, 13)	(Bad, Good)	(30.00%, 70.00%)
Haberman	306	3 (3, 0)	(Die, Survive)	(26.47%, 73.53%)
New-thyroid	215	5 (5, 0)	(hypo, remainder)	(16.28%, 83.72%)
E-coli	336	7 (7, 0)	(iMU, remainder)	(10.42%, 89.58%)
Satimage	6435	36 (36, 0)	(4, remainder)	(9.73%, 90.27%)
Flag	194	28 (10, 18)	(white, remainder)	(8.76%, 91.24%)
Glass	214	9 (9, 0)	(Ve-win-float-proc, remainder)	(7.94%, 92.06%)

In our experiments we used the release 8 of the C4.5 symbolic learning algorithm to induce decision trees [12]. Firstly, we ran C4.5 over the data sets and calculated the AUC and EC for pruned (default parameters settings) and unpruned trees induced for each data set using 10-fold stratified cross-validation. Table 3 summarizes these results, reporting mean value results and their respective standard deviations. It should be observed that for two data sets, Sonar and Glass, C4.5 was not able to prune the induced trees. Furthermore, for data set Flag and pruned trees, the default model was induced.

We consider the results obtained for both pruned and unpruned trees because we aim to analyze whether pruning is effective for coping with small disjuncts in the presence of class skews. Pruning is often reported in the ML literature as a rule of thumb for dealing with the small disjuncts problem. The conventional wisdom beneath pruning is to perform significance and/or error rate tests aiming to reliably eliminate undesirable disjuncts. The main reason for verifying the effectiveness of pruning is that several research papers indicate that pruning

Table 3. AUC and EC results for pruned and unpruned decision trees.

Data set	Pruned Trees		Unpruned Trees	
	AUC	EC	AUC	EC
Sonar	86.71(6.71)	61.51(19.03)	86.71(6.71)	61.51(19.03)
Bupa	79.44(4.51)	66.03(12.36)	79.93(5.02)	65.80(14.04)
Pima	81.53(5.11)	42.03(11.34)	82.33(5.70)	45.41(8.52)
German	78.49(7.75)	52.92(17.22)	85.67(4.37)	87.61(7.72)
Haberman	58.25(12.26)	29.33(22.51)	67.91(13.76)	36.25(20.06)
New-thyroid	94.73(9.24)	33.54(41.78)	94.98(9.38)	33.13(42.64)
E-coli	87.64(15.75)	55.13(36.68)	92.50(7.71)	71.97(26.93)
Satimage	93.73(1.91)	80.97(4.19)	94.82(1.18)	83.75(5.21)
Flag	45.00(15.81)	0.00(0.00)	76.65(27.34)	61.82(39.01)
Glass	88.16(12.28)	56.53(57.38)	88.16(12.28)	56.53(57.38)

should be avoided when target misclassification costs or class distributions are unknown [13, 14]. One reason to avoid pruning is that most pruning schemes, including the one used by C4.5, attempt to minimize the overall error rate. These pruning schemes can be detrimental to the minority class, since reducing the error rate on the majority class, which stands for most of the examples, would result in a greater impact over the overall error rate. Another fact is that significance tests are mainly based on coverage estimation. As skewed class distributions are more likely to include rare or exceptional cases, it is desirable for the induced concepts to cover these cases, even if they can only be covered by augmenting the number of small disjuncts in a concept.

Table 3 results indicate that the decision of not pruning the decision trees systematically increases the AUC values. For all data sets in which the algorithm was able to prune the induced trees, there is an increase in the AUC values. However, the EC values also increase in almost all unpruned trees. As stated before, this increase in EC values generally means that the errors are more concentrated towards small disjuncts. Furthermore, pruning removes most branches responsible for covering the minority class, thus not pruning is beneficial for learning with imbalanced classes. However, the decision of not pruning also leaves these small disjuncts in the learned concept. As these disjuncts are error-prone, since pruning would remove them, the overall error tends to concentrate on these disjuncts, increasing the EC values. Thus, concerning the problem of pruning or not pruning, a trade-off between the increase we are looking for in the AUC values and the undesirable raise in the EC values seems to exist.

We have also investigated how sampling strategies behave with respect to small disjuncts and class imbalances. We decided to apply the sampling methods until a balanced distribution was reached. This decision is motivated by the results presented in [15], in which it is shown that when AUC is used as performance measure, the best class distribution for learning tends to be near the balanced class distribution. Moreover, Weiss [1] also investigates the relationship between sampling strategies and small disjuncts using a Random under-sampling method to artificially balance training sets. Weiss' results show that the trees induced using balanced data sets seem to systematically outperform the trees induced using the original stratified class distribution from the data sets, not only increasing the AUC values but also decreasing the EC values. In our view, the decrease in the EC values might be explained by the reduction in the num-

Table 4. AUC and EC results for over-sampled data and unpruned decision trees.

Data set	Random		Smote	
	AUC	EC	AUC	EC
Sonar	86.52(4.69)	47.29(27.24)	86.74(8.91)	52.07(24.63)
Bupa	80.06(3.48)	33.14(26.01)	72.81(9.13)	40.47(23.94)
Pima	86.03(4.14)	57.59(17.65)	85.97(5.82)	52.62(13.18)
German	85.03(4.91)	84.07(4.55)	84.19(5.54)	81.95(12.18)
Haberman	73.58(14.22)	54.66(22.37)	75.45(11.02)	43.15(25.55)
New-thyroid	98.89(2.68)	15.71(40.35)	98.91(1.84)	23.83(38.53)
E-coli	93.55(6.89)	81.93(13.09)	95.49(4.30)	91.48(16.12)
Satimage	95.52(1.12)	86.81(3.23)	95.69(1.28)	90.35(3.02)
Flag	79.78(28.98)	85.47(16.41)	73.87(30.34)	54.73(44.75)
Glass	92.07(12.09)	81.48(22.96)	91.27(8.38)	78.17(30.85)

ber of induced disjuncts in the concept description, which is a characteristic of under-sampling methods. We believe this approach might rule out some interesting disjuncts from the concept. Moreover, in previous work [16] we showed that over-sampling methods seem to perform better than under-sampling methods, resulting in classifiers with higher AUC values. Table 4 shows the AUC and EC values for two over-sampling methods proposed in the literature: Random over-sampling and Smote [7]. Random over-sampling randomly duplicates examples from the minority class while Smote introduces artificially generated examples by interpolating two examples drawn from the minority class that lie together.

Table 4 reports results regarding unpruned trees. Besides our previous comments concerning pruning and class imbalance, whether pruning can lead to a performance improvement for decision trees grown over artificially balanced data sets still seems to be an open question. Another argument against pruning is that if pruning is allowed to execute under such conditions, the learning system would prune based on false assumption, *i.e.*, that the test set distribution matches the training set distribution.

The results in Table 4 show that, in general, the best AUC result obtained by an unpruned over-sampled data set is similar (less than 1% difference) or higher than those obtained by pruned and unpruned trees grown over the original data sets. Moreover, unpruned over-sampled data sets also tend to produce higher EC values than pruned and unpruned trees grown over the original data sets. It is also worth noticing that Random over-sampling, which can be considered the simplest method, produced similar results to Smote (with a difference of less than 1% in AUC) in six data sets (Sonar, Pima German, New-thyroid, Satimage and Glass); Random over-sampling beats Smote (with a difference greater than 1%) in two data sets (Bupa and Flag) and Smote beats Random over-sampling in the other two (Haberman and E-coli). Another interesting point is that both over-sampling methods produced lower EC values than unpruned trees grown over the original data for four data sets (Sonar, Bupa, German and New-thyroid), and Smote itself produced lower EC values for another one (Flag). Moreover, in three data sets (Sonar, Bupa and New-thyroid) Smote produced lower EC values even if compared with pruned trees grown over the original data.

These results might be explained observing that by using an interpolation method, Smote might help in the definition of the decision border of each class. However, as a side effect, by introducing artificially generated examples Smote

Table 5. AUC and EC results for over-sampled data: Smote + ENN and Smote + Tomek links and unpruned decision trees.

Data set	Smote + ENN		Smote + Tomek	
	AUC	EC	AUC	EC
Sonar	85.31(11.09)	52.56(28.21)	86.90(9.62)	49.77(17.24)
Bupa	78.84(5.37)	41.72(14.68)	75.35(10.65)	38.39(18.71)
Pima	83.64(5.35)	54.07(19.65)	85.56(6.02)	47.54(21.06)
German	82.76(5.93)	82.21(10.52)	84.40(6.39)	88.53(6.54)
Haberman	77.01(5.10)	62.18(19.08)	78.41(7.11)	43.26(29.39)
New-thyroid	99.22(1.72)	27.39(44.34)	98.91(1.84)	23.83(38.53)
E-coli	95.29(3.79)	87.58(18.36)	95.98(4.21)	90.92(16.17)
Satimage	96.06(1.20)	88.56(3.31)	95.69(1.28)	90.35(3.02)
Flag	78.56(28.79)	78.78(20.59)	82.06(29.52)	70.55(38.54)
Glass	93.40(7.61)	80.14(30.72)	91.27(8.38)	78.17(30.85)

might introduce noise in the training set. Although Smote might help in overcoming the class imbalance problem, in some cases it might be detrimental regarding the problem of small disjuncts. This observation, allied to the results we obtained in a previous study that poses class overlapping as a complicating factor for dealing with class imbalance [8] motivated us to propose two new methods to deal with the problem of learning in the presence of class imbalance [16]. These methods ally Smote [7] with two data cleaning methods: Tomek links [17] and Wilson's Edited Nearest Neighbor Rule (ENN) [18]. The main motivation behind these methods is to pick up the best of the two worlds. We not only balance the training data aiming at increasing the AUC values, but also remove noisy examples lying in the wrong side of the decision border. The removal of noisy examples might aid in finding better-defined class clusters, allowing the creation of simpler models with better generalization capabilities. As a net effect, these methods might also remove some undesirable small disjuncts, improving the classifier performance. In this matter, these data cleaning methods might be understood as an alternative for pruning.

Table 5 shows the results of our proposed methods on the same data sets. Comparing these two methods it can be observed that Smote + Tomek produced the higher AUC values for four data sets (Sonar, Pima, German and Haberman) while Smote+ENN is better in two data sets (Bupa and Glass). For the other four data sets they produced compatible AUC results (with a difference lower than 1%). However, it should be observed that for three data sets (New-thyroid, Satimage and Glass) Smote+Tomek obtained results identical to Smote – Table 4. This occurs when no Tomek links or just a few of them are found in the data sets.

Table 6 shows a ranking of the AUC and EC results obtained in all experiments for unpruned decision trees, where: O indicates the original data set (Table 3) R and S stand respectively for Random and Smote over-sampling (Table 4) while S+E and S+T stand for Smote + ENN and Smote + Tomek (Table 5). $\surd_1$ indicates that the method is ranked among the best and $\surd_2$ among the second best for the corresponding data set. Observe that results having a difference lower than 1% are ranked together. Although the proposed conjugated over-sampling methods obtained just one EC value ranked in the first place (Smote + ENN on data set German) these methods provided the highest AUC

Table 6. AUC and EC ranking results for unpruned decision trees.

Data sets	AUC					EC				
	O	R	S	S+E	S+T	O	R	S	S+E	S+T
Sonar	$\surd_1$	$\surd_1$	$\surd_1$	$\surd_2$	$\surd_1$		$\surd_1$			$\surd_2$
Bupa	$\surd_1$	$\surd_1$		$\surd_2$			$\surd_1$			$\surd_2$
Pima		$\surd_1$	$\surd_1$	$\surd_2$	$\surd_1$	$\surd_1$				$\surd_2$
German	$\surd_1$	$\surd_1$		$\surd_2$	$\surd_2$		$\surd_2$	$\surd_1$	$\surd_1$	
Haberman				$\surd_2$	$\surd_1$	$\surd_1$		$\surd_2$		$\surd_2$
New-thyroid	$\surd_2$	$\surd_1$	$\surd_1$	$\surd_1$	$\surd_1$		$\surd_1$	$\surd_2$		$\surd_2$
E-coli		$\surd_2$	$\surd_1$	$\surd_1$	$\surd_1$	$\surd_1$	$\surd_2$			
Satimage	$\surd_2$	$\surd_1$	$\surd_1$	$\surd_1$	$\surd_1$	$\surd_1$	$\surd_2$			
Flag		$\surd_2$			$\surd_1$	$\surd_2$		$\surd_1$		
Glass		$\surd_2$	$\surd_2$	$\surd_1$	$\surd_2$	$\surd_1$		$\surd_2$		$\surd_2$

values in seven data sets. Smote + Tomek produced the highest AUC values in four data sets (Sonar, Haberman, Ecoli and Flag), and the Smote + ENN method produced the highest AUC values in another three data sets (Satimage, New-thyroid and Glass). If we analyze both measures together, in four data sets where Smote + Tomek produced results among the top ranked AUC values, it is also in second place with regard to lower EC values (Sonar, Pima, Haberman and New-thyroid). However, it is worth noticing in Table 6 that simpler methods, such as the Random over-sampling approach (R) or taking only the unpruned tree (O), have also produced interesting results in some data sets. In the New-thyroid data set, Random over-sampling produced one of the highest AUC values and the lowest EC value. In the German data set, the unpruned tree produced the highest AUC value, and the EC value is almost the same as in the other methods that produced high AUC values. Nevertheless, the results we report suggest that the methods we propose in [16] might be useful, specially if we aim to further analyze the induced disjuncts that compound the concept description.

5 Conclusion

In this work we discuss results related to some aspects of the interaction between learning with class imbalances and small disjuncts. Our results suggest that pruning might not be effective for dealing with small disjuncts in the presence of class skews. Moreover, artificially balancing class distributions with over-sampling methods seems to increase the number of error-prone small disjuncts. Our proposed methods, which ally over sampling with data cleaning methods produced meaningful results in some cases. Conversely, in some cases, Random over-sampling, a very simple over-sampling method, also achieved compatible results. Although our results are not conclusive with respect to a general approach for dealing with both problems, further investigation into this relationship might help to produce insights on how ML algorithms behave in the presence of such conditions. In order to investigate this relationship in more depth, several further approaches might be taken. A natural extension of this work is to individually analyze the disjuncts that compound each description assessing their quality concerning some objective or subjective criterium. Another interesting topic is

to analyze the ROC and EC graphs obtained for each data set and method. This might provide us with a more in depth understanding of the behavior of pruning and balancing methods. Last but not least, another interesting point to investigate is how alternative learning bias behaves in the presence of class skews.

Acknowledgements. We wish to thank the anonymous reviewers for their helpful comments. This research was partially supported by the Brazilian Research Councils CAPES and FAPESP.

References

1. Weiss, G.M.: The Effect of Small Disjuncts and Class Distribution on Decision Tree Learning. PhD thesis, Rutgers University (2003)
2. Japkowicz, N.: Class Imbalances: Are we Focusing on the Right Issue? In: ICML Workshop on Learning from Imbalanced Data Sets. (2003)
3. Holte, R.C., Acker, L.E., Porter, B.W.: Concept Learning and the Problem of Small Disjuncts. In: IJCAI. (1989) 813–818
4. Weiss, G.M.: The problem with Noise and Small Disjuncts. In: ICML. (1988) 574–578
5. Carvalho, D.R., Freitas, A.A.: A Hybrid Decision Tree/Genetic Algorithm for Coping with the Problem of Small Disjuncts in Data Mining. In: Genetic and Evolutionary Computation Conference. (2000) 1061–1068
6. Kubat, M., Matwin, S.: Addressing the Course of Imbalanced Training Sets: One-Sided Selection. In: ICML. (1997) 179–186
7. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: Synthetic Minority Over-sampling Technique. JAIR **16** (2002) 321–357
8. Prati, R.C., Batista, G.E.A.P.A., Monard, M.C.: Class Imbalances *versus* Class Overlapping: an Analysis of a Learning System Behavior. In: MICAI. (2004) 312–321 Springer-Verlag, LNAI 2972.
9. Weiss, G.M.: Learning with Rare Cases and Small Disjucts. In: ICML. (1995) 558–565
10. Ferri, C., Flach, P., Hernndez-Orallo, J.: Learning Decision Trees Using the Area Under the ROC Curve. In: ICML. (2002) 139–146
11. Blake, C., Merz, C.: UCI Repository of Machine Learning Databases (1998) http://www.ics.uci.edu/∼mlearn/MLRepository.html.
12. Quinlan, J.R.: C4.5 Programs for Machine Learning. Morgan Kaufmann (1993)
13. Zadrozny, B., Elkan, C.: Learning and Making Decisions When Costs and Probabilities are Both Unknown. In: KDD. (2001) 204–213
14. Bauer, E., Kohavi, R.: An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants. Machine Learning **36** (1999) 105–139
15. Weiss, G.M., Provost, F.: Learning When Training Data are Costly: The Effect of Class Distribution on Tree Induction. JAIR **19** (2003) 315–354
16. Batista, G.E.A.P.A., Prati, R.C., Monard, M.C.: A Study of the Behavior of Several Methods for Balancing Machine Learning Training Data. SIGKDD Explorations **6** (2004) (to appear).
17. Tomek, I.: Two Modifications of CNN. IEEE Transactions on Systems Man and Communications **SMC-6** (1976) 769–772
18. Wilson, D.L.: Asymptotic Properties of Nearest Neighbor Rules Using Edited Data. IEEE Transactions on Systems, Man, and Communications **2** (1972) 408–421