The class imbalance problem: A systematic study[1]

Nathalie Japkowicz and Shaju Stephen
School of Information Technology and Engineering, University of Ottawa, 150 Louis Pasteur, P.O. Box 450 Stn. A, Ottawa, Ontario, Canada, B3H 1W5

Received 25 September 2001
Revised 11 November 2001
Accepted 17 February 2002

Abstract. In machine learning problems, differences in prior class probabilities – or class imbalances – have been reported to hinder the performance of some standard classifiers, such as decision trees. This paper presents a systematic study aimed at answering three different questions. First, we attempt to understand the nature of the class imbalance problem by establishing a relationship between concept complexity, size of the training set and class imbalance level. Second, we discuss several basic re-sampling or cost-modifying methods previously proposed to deal with the class imbalance problem and compare their effectiveness. The results obtained by such methods on artificial domains are linked to results in real-world domains. Finally, we investigate the assumption that the class imbalance problem does not only affect decision tree systems but also affects other classification systems such as Neural Networks and Support Vector Machines.

Keywords: Concept learning, class imbalances, re-sampling, misclassification costs, C5.0, Multi-Layer Perceptrons, Support Vector Machines

1. Introduction

As the field of machine learning makes a rapid transition from the status of “academic discipline” to that of “applied science”, a myriad of new issues, not previously considered by the machine learning community, is now coming into light. One such issue is the *class imbalance* problem. The class imbalance problem corresponds to the problem encountered by inductive learning systems on domains for which one class is represented by a large number of examples while the other is represented by only a few.[2]

The class imbalance problem is of crucial importance since it is encountered by a large number of domains of great environmental, vital or commercial importance, and was shown, in certain cases, to cause a significant bottleneck in the performance attainable by standard learning methods which assume a balanced class distribution. For example, the problem occurs and hinders classification in applications

[1] This research was supported by an NSERC grant and a grant from the University of Ottawa. We would like to thank the anonymous reviewers for their thoughtful comments as well as the TAMALE Seminar audience at the University of Ottawa, especially Chris Drummond, Rob Holte and Andrew McPherson who suggested many useful experiments during earlier presentations of this work.

[2] In this paper, we only consider the case of concept-learning. However, the discussion also applies to multi-class problems.

as diverse as the detection of oil spills in satellite radar images [15], the detection of fraudulent telephone calls [6], in-flight helicopter gearbox fault monitoring [8], information retrieval and filtering [17] and diagnoses of rare medical conditions such as thyroid diseases [16].

To this point, there have been a number of attempts at dealing with the class imbalance problem [1–3,6, 8,14,15,18,20]; However, these attempts were mostly conducted in isolation. In particular, there has not been, to date, much systematic strive to link specific types of imbalances to the degree of inadequacy of standard classifiers nor have there been many comparisons of the various methods proposed to remedy the problem. Furthermore, no comparison of the performance of different types of classifiers on imbalanced data sets has yet been performed.[3]

The purpose of this paper is to address these three concerns in an attempt to unify the research conducted on this problem. In a first part, the paper concentrates on explaining what the class imbalance problem is by establishing a relationship between concept complexity, size of the training set and class imbalance level. In doing so, we also identify the class imbalance situations that are most damaging for a standard classifier that expects balanced class distributions. The second part of the paper turns to the question of how to deal with the class imbalance problem. In this part we look at five different methods previously proposed to deal with this problem and which are all assumed to be more or less equivalent to each other. We attempt to establish to what extent these methods are, indeed, equivalent and to what extent they differ. The first two parts of our study were conducted using the C5.0 decision tree induction system. In the third part, we set out to find out whether or not the problems encountered by C5.0 when trained on imbalanced data sets are specific to C5.0. In particular, we attempt to find out whether or not the same pattern of hindrance is encountered by Neural Networks and Support Vector Machines and whether similar remedies can apply.

The remainder of the paper is divided into six sections. Section 2 is an overview of the paper explaining why the questions we set out to answer are important and how they will advance our understanding of the class imbalance problem. Section 3 describes the part of the study focusing on understanding the nature of the class imbalance problem and finding out what types of class imbalance problems create greater difficulties for a standard classifier. Section 4 describes the part of the study designed to compare the five main types of approaches previously attempted to deal with the class imbalance problem. Section 5 addresses the question of what effect class imbalances have on classifiers other than C5.0. Sections 6 and 7 conclude the paper.

2. Overview of the paper

As mentioned in the previous section, the study presented in this paper investigates the following three series of questions:

Question 1: What is the nature of the class imbalance problem? i.e., in what domains do class imbalances most hinder the accuracy performance of a standard classifier such as C5.0?

[3] Two studies attempting to systematize research on the class imbalance problem are worth mentioning, nonetheless: one, currently in progress at AT&T Lab [26], links different degrees of imbalances to the performance of C4.5, a decision tree learning system on a large number of real-world data sets. However, it does not study the effect of concept complexity nor training set size in the context of their relationship with class imbalances, nor does it look at ways to remedy the class imbalance problem or the effect of class imbalances on classifiers other than C4.5. The second study is that by [13], which does not study the effect of class imbalances on classifiers' performance but which compares a number of specific approaches proposed to deal with class imbalances in the context of Neural Networks and on a few real-world data sets. In their study, no classifiers other than Neural Networks were considered and no systematic study, conducted.

Question 2: How do the different approaches proposed for dealing with the class imbalance problem compare?

Question 3: Does the class imbalance problem hinder the accuracy performance of classifiers other than C5.0?

These questions are important since their answers may put to rest currently assumed but unproven facts, dispel other unproven beliefs as well as suggest fruitful directions for future research. In particular, they may help researchers focus their inquiry onto the particular type of solution found most promising, given the particular characteristics identified in their application domain.

Question 1 raises the issue of when class imbalances are damaging. While the studies previously mentioned identified specific domains for which an imbalance was shown to hurt the performance of certain standard classifiers, they did not discuss the questions of whether imbalances are always damaging and to what extent different types of imbalances affect the classification performance. This paper takes a global stance and answers these questions in the context of the C5.0 tree induction system on a series of artificial domains spanning a large combination of characteristics.[4]

Question 2 considers five related approaches previously proposed by independent researchers for tackling the class imbalance problem:[5]

1. Upsizing the small class at random.
2. Upsizing the small class at "focused" random.
3. Downsizing the large class at random.
4. Downsizing the large class at "focused" random.
5. Altering the relative costs of misclassifying the small and the large classes.

In more detail, Methods 1 and 2 consist of re-sampling patterns of the small class (either completely randomly or randomly but within parts of the input space close to the boundaries with the other class) until there are equally many data from the small class as from the large one.[6] Methods 3 and 4 consists of eliminating data from the large class (either completely randomly or, randomly but within parts of the input space far away from the boundaries with the large class) until there are as many data in both classes. Finally, method 5 consists of reducing the relative misclassification cost of the large class (or, equivalently, increasing that of the small one) to make it correspond to the size of the small class.

These methods were previously proposed by [1–3,14,18] but were not systematically compared before. Here, we compare the five methods on the data sets used in the previous part of the paper. This was done to see whether or not the five approaches for dealing with class imbalances respond to different domain characteristics in the same way.

[4] The paper, however, concentrates on domains that present a "between-class imbalance" in that the imbalance affects each subcluster of the small class to the same extent. Because of lack of space, the interesting issue of "within-class imbalances" – which are special cases of the problem of small disjuncts [7,25] – has been omitted here. This very important question is dealt with elsewhere [10,19].

[5] In this study, we focus on discrimination-based approaches to the problem which base their decisions on both the positive and negative data. The study of recognition-based approaches which base their decision on one of the two classes but not both has been attempted in [8,9,11] but did not seem to do as well as discrimination-based methods (this might be linked, however, to the fact that the recognition threshold was not chosen very carefully. Nonetheless, we leave it to future work to determine whether or not this is truly the case).

[6] [4,5,26] show that, in fact, the optimal amount of re-sampling is not necessarily that which yields the same number of data in each class. The optimal amount seems to depend upon the input domain and does not seem easy to estimate a priori. In order to simplify our study, here, we decided to re-sample until the two classes are of the same size. This decision will not alter our results, however, since we are interested in the *relative* performance of the different remedial approaches we consider.

Question 3, finally, asks whether the observations made in answering the previous questions for C5.0 also hold for other classifiers. In particular, we study the effect of class imbalances on Multi-Layer Perceptrons (MLPs), which could be thought to be capable of more flexible learning than C5.0, and thus, less sensitive to class imbalances. We then repeat this study with Support Vector Machines (SVMs) which could be believed not to be affected by the class imbalance problem given that they base their classification on a small number of support vectors and, thus, may not be sensitive to the number of data representing each class. We look at the performance of MLPs and SVMs on a subset of the series of domains used in the previous part of the paper so as to see whether the three approaches are affected by different domain characteristics in the same ways.

3. Question 1: What is the nature of the class imbalance problem?

In order to answer Question 1, a series of artificial concept-learning domains was generated that varies along three different dimensions: the degree of *concept complexity*, the *size* of the training set, and the level of *imbalance* between the two classes. The standard classifier system tested on this domain in this section was the C5.0 decision tree induction system [21]. This classifier has previously been shown to suffer from the class imbalance problem (e.g. [15]), but not in a completely systematic fashion. The study in this section aims at answering the question of what different faces a class imbalance can take and which of these faces hinders C5.0 most.

This part of the paper first discusses the domain generation process followed by a report of the results obtained by C5.0 on the various domains. Some modifications to the domain generation and classification processes are then proposed that allow us to further refine our conclusions.

3.1. Domain generation

For the first experiments of this section, 125 domains were created with various combinations of *concept complexity*, *training set size*, and *degree of imbalance*. The generation method used was inspired by Schaffer who designed a similar framework for testing the effect of overfitting avoidance in sparse data sets [23]. From Schaffer's study, it was clear that the complexity of the concept at hand was an important part of the data overfitting problem and, given the relationship between the problem of overfitting the data and dealing with class imbalances (see [15]),[7] it seems reasonable to assume that, here again, concept complexity is an important piece of the puzzle. Similarly, the training set size should also be a factor in a classifier's ability to deal with imbalanced domains given the relationship between the data overfitting problem and the size of the training set. Finally, the degree of imbalance is the obvious other parameter expected to influence a classifier's ability to classify imbalanced domains.

The 125 generated domains of our study were generated in the following way: each of the domains is one-dimensional with inputs in the [0, 1] range associated with one of the two classes (1 or 0). The input range is divided into a number of regular intervals (i.e., intervals of the same size), each associated with a different class value. Contiguous intervals have opposite class values and the degree of concept complexity corresponds to the number of alternating intervals present in the domain. Actual training sets are generated from these backbone models by sampling points at random (using a uniform distribution), from each of the intervals. The number of points sampled from each interval depends on the size of the domain as well as on its degree of imbalance. An example of a backbone model is shown in Fig. 1.

[7] In the class imbalance problem, some subclusters of the data can contain only a few examples. In such cases, a classifier may be able to learn a hypothesis that fits these examples well, but that fits them too specifically. This is called overfitting and

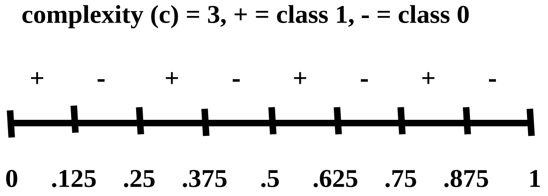


Fig. 1. A backbone model of complexity 3.

Five different complexity levels were considered ($c = 1 \ldots 5$) where each level, c, corresponds to a backbone model composed of 2^c regular intervals. For example, the domains generated at complexity level $c = 1$ are such that every point whose input is in range [0, 0.5) is associated with a class value of 1, while every point whose input is in range (0.5, 1] is associated with a class value of 0; At complexity level $c = 2$, points in intervals [0, 0.25) and (0.5, 0.75) are associated with class value 1 while those in intervals (0.25, 0.5) and (0.75, 1] are associated with class value 0; etc., regardless of the size of the training set and its degree of imbalance.[8]

Five training set sizes were considered ($s = 1 \ldots 5$) where each size, s, corresponds to a training set of size round$((5000/32) * 2^s)$. Since this training set size includes all the regular intervals in the domain, each regular interval is, in fact, represented by round$(((5000/32) * 2^s)/2^c)$ training points (before the imbalance factor is considered). For example, at a size level of $s = 1$ and at a complexity level of $c = 1$ and before any imbalance is taken into consideration, intervals [0, 0.5) and (0.5, 1] are each represented by 157 examples; If the size is the same, but the complexity level is $c = 2$, then each of intervals [0, 0.25), (0.25, 0.5), (0.5, 0.75) and (0.75, 1] contains 78 training examples; etc.

Finally, five levels of class imbalance were also considered ($i = 1 \ldots 5$) where each level, i, corresponds to the situation where each sub-interval of class 1 is represented by all the data it is normally entitled to (given c and s), but each sub-interval of class 0 contains only $1/(32/2^i)$th (rounded) of all its normally entitled data. This means that each of the sub-intervals of class 0 are represented by round$((((5000/32) * 2^s)/2^c)/(32/2^i))$ training examples. For example, for $c = 1$, $s = 1$, and $i = 2$, interval [0, 0.5) is represented by 157 examples and (0.5, 1] is represented by 79; If $c = 2$, $s = 1$ and $i = 3$, then [0, 0.25) and (0.5, 0.75) are each represented by 78 examples while (0.25, 0.5) and (0.75, 1] are each represented by 20; etc.[9]

The number of testing points representing each sub-interval was kept fixed (at 50). This means that all domains of complexity level $c = 1$ are tested on 50 positive and 50 negative examples; all domains of complexity level $c = 2$ are tested on 100 positive and 100 negative examples; etc.

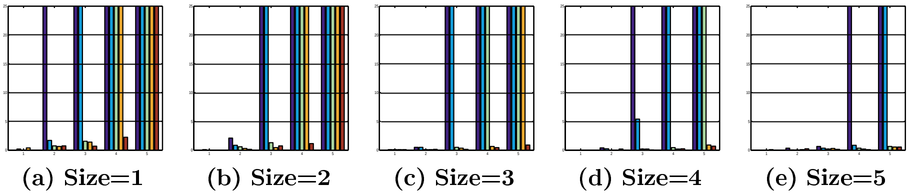
3.2. Results for Question 1

We first collect results on the artificial domains of the type just discussed. We then extend our study by considering a number of shortcomings of our chosen domain generation and classification settings and addressing them.

it is the same problem as the one considered by [23].

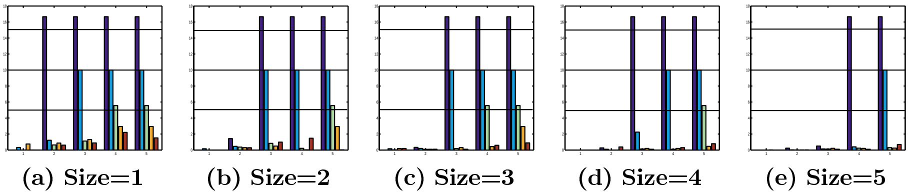
[8] In this paper, complexity is varied along a single very simple dimension. Other more sophisticated models could be used in order to obtain finer-grained results. In [4], for example, a k-DNF model using several dimensions was used to generate a few artificial domains presenting class imbalances. The study was less systematic than the one in this paper, but it yielded results corroborating those of this section.

[9] Note that throughout the paper, high values of c and s indicate high complexity and large training set sizes, rspectively, but that high values of i corresponds to low levels of imbalance (while low values of i correspond to high levels of imbalance).



(a) Size=1 (b) Size=2 (c) Size=3 (d) Size=4 (e) Size=5

Fig. 2. C5.0 and the class imbalance problem – corrected.



(a) Size=1 (b) Size=2 (c) Size=3 (d) Size=4 (e) Size=5

Fig. 3. C5.0 and the class imbalance problem – uncorrected.

3.2.1. Results with the original domain generation settings

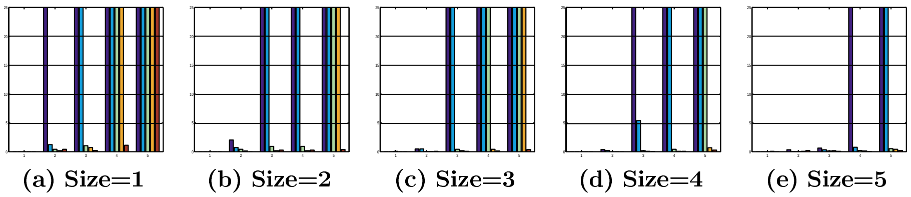
The results for C5.0 are displayed in Figs 2, 3, 4 and 5, which plot the error C5.0 obtained for each combination of concept complexity, training set size, and imbalance level, on the entire testing set. For each experiment, we reported four types of results: 1) the *corrected results* in which no matter what degree of class imbalance is present in the training set, the contribution of the false positive error rate is the same as that of the false negative one in the overall report.[10] 2) the *uncorrected results* in which the reported error rate reflects the same imbalance as the one present in the training set.[11] 3) the *false positive* error rate; and 4) the *false negative* error rate. The corrected and uncorrected results are provided so as to take into consideration two out of any possible number of situations: one in which, despite the presence of an imbalance, the costs of misclassifying the data of one class is the same as that of classifying those of the other class (the corrected version); the other situation is the one where the relative cost of misclassifying the two classes correspond to the class imbalance.[12]

Each plot in each of these figures represents the error obtained at a different training set size. The leftmost plot corresponds to the smallest size ($s = 1$) and progresses until the rightmost plot which corresponds to the largest ($s = 5$). Within each of these plots, each cluster of five bars represent the concept complexity level. The leftmost cluster corresponds to the simplest concept ($c = 1$) and progresses until the rightmost one which corresponds to the most complex ($c = 5$). Within each cluster, finally, each bar corresponds to a particular imbalance level. The leftmost bar corresponds to the most imbalanced level ($i = 1$) and progresses until the rightmost bar which corresponds to the most balanced level ($i = 5$, or no imbalance). The height of each bar represents the average percent error rate obtained

[10] For this set of results, we simply report the error rate obtained on the testing set corresponding to the experiment at hand.

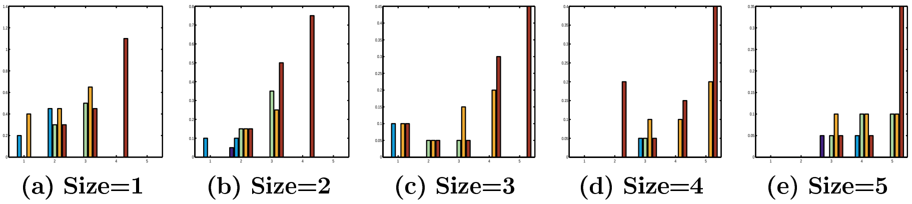
[11] For this set of results, we modify the ratio of false positive to false negative error obtained on the original testing set to make it correspond to the ratio of positive to negative examples in the training set.

[12] A more complete set of results could have involved comparisons at other relative costs as well. However, given our large number of experiments, this would have been unmanageable. We thus decided to focus on two meaningful and important cases only. Similarly, and for the same reasons, we decided not to vary C5.0's decision threshold across the ROC space [24]. Since we are seeking to establish the relative performance of several classification approaches we believe that all the results obtained using the same decision threshold are representative of what would have happened along the ROC curves. We leave it to future work, however, to verify this assumption.



(a) Size=1 (b) Size=2 (c) Size=3 (d) Size=4 (e) Size=5

Fig. 4. C5.0 and the class imbalance problem – false positive error rate.



(a) Size=1 (b) Size=2 (c) Size=3 (d) Size=4 (e) Size=5

Fig. 5. C5.0 and the class imbalance problem – false negative error rate: Very close to 0.

by C5.0 (over five runs on different random data sets generated from the same backbone model) on the complexity, class size and imbalance level this bar represents. To make the comparisons easy, horizontal lines were drawn at every 5% marks. If a graph does not display any horizontal line, it is because all the bars represent an average percent error below 5%, and we consider the error negligible in such cases.

Our results reveal several points of interest: first, no matter what the size of the training set is, linearly separable domains (domains of complexity level $c = 1$) do not appear sensitive to any amount of imbalance. As a matter of fact, as the degree of concept complexity increases, so does the system's sensitivity to imbalances. Indeed, we can clearly see both in Fig. 2 (the corrected results) and Fig. 3 (the uncorrected results) that as the degree of complexity increases, high error rates are caused by lower and lower degrees of imbalance. Although the error rates reported in the corrected cases are higher than those reported in the uncorrected cases, the effect of concept complexity on class imbalances is clearly visible in both situations.

A look at Figs 4 and 5 explains the difference between Figs 2 and 3 since it reveals that most of the error represented in these graphs actually occurs on the negative testing set (i.e., most of the errors are false positive errors). Indeed, none of the average percents of false negative errors over all degrees of concept complexity and levels of imbalance ever exceed 5%. This is not surprising since we had expected the classifier to neglect the minority class, but the extent to which it does so might seem a bit surprising.

As could be expected, imbalance rates are also a factor in the performance of C5.0 and, perhaps more surprisingly, so is the training set size. Indeed, as the size of the training set increases, the degree of imbalance yielding a large error rate decreases. This suggests that in very large domains, the class imbalance problem may not be a hindrance to a classification system. Specifically, the issue of relative cardinality of the two classes – which is often assumed to be the problem underlying domains with class imbalanced – may in fact be easily overridden by the use of a large enough data set (if, of course, such a data set is available and its size does not prevent the classifier from learning the domain in an acceptable time frame).[13]

[13] This question is considered in more detail in Section 3.2.2.

All in all, our study suggests that the imbalance problem is a *relative* problem depending on both the complexity of the concept[14] represented by the data in which the imbalance occurs and the overall size of the training set, in addition to the degree of class imbalance present in the data. In other words, a huge class imbalance will not hinder classification of a domain whose concept is very easy to learn nor will we see a problem if the training set is very large. Conversely, a small class imbalance can greatly harm a very small data set or one representing a very complex concept.

Though these results shed some light onto the nature of the class imbalance problem, we now question whether we can refine these conclusions further. In particular, we address a number of issues that were not fully considered in the experiments just reported.

3.2.2. Additional results

There are three important issues that were not fully considered by our experiments of the previous section. The first two are concerned with the number and distribution of examples contained in the training sets while the third one is concerned with simple ways to avoid overfitting.

In more detail, the first issue concerns the overall training set size of our data sets. Although each of the five graphs in each figure in the previous section are labeled as pertaining to a given size ($s = 1$, 2, 3, 4 or 5), it is important to notice that not all the results presented within one of these graphs were obtained using data sets of the same overall size. Indeed, the domain generation procedure outlined in Section 3.1 is such that, at every imbalance level, data from class 0 are taken away, reducing the overall training set size as the imbalance level increases. For example, for size $s = 1$, the overall training set size for parameters $s = 1$, $c = 1$, $i = 5$ (i.e., no imbalance) is 312 examples. It is 234 examples for $s = 1$, $c = 1$, and $i = 4$, etc. $\ldots$ until parameters $s = 1$, $c = 1$, and $i = 1$ where 166 examples are used in the training set.

This observation leads us to wonder whether the results obtained for different imbalance levels are really caused by the increased imbalance or whether they are caused by a decrease in overall training set size. To answer this question, we computed results obtained in cases where a class imbalance does not entail a decrease in overall training set size. In particular, we generated new domains that obey the following equation:

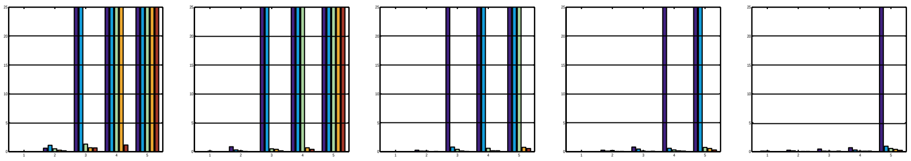
$$((x \times 2^s)/2^c) + (((x \times 2^s)/2^c)/(32/2^i)) = \text{size}_x \tag{1}$$

and where term $(x \times 2^s)/2^c$ represents the number of examples contained in class 1, term $((x \times 2^s)/2^c)/(32/2^i)$ represents the number of examples contained in class 0, and size$_x$ corresponds to the overall training set size and remains fixed for all values of c and i for a given value of s. After simplification, Eq. (1) is equivalent to

$$x \times (2^{s-c} + 2^{s-c+1-5}) = \text{size}_x \tag{2}$$

We fix $size_x$ successively to 312, 625, 1250, 2500, and 5,000, which correspond, respectively to $s =$ 1, 2, 3, 4, and 5, and solve Eq. (2) for x given each combination of s, c, and i values considered. Each x is then used in each term of Eq. (1) to find out how many class 1 and class 0 examples, respectively, need to be generated. Once the various training sets are created, C5.0 is run and its results recorded in the same manner as in the experiments of Section 3.2.1.

[14] Where “concept complexity” corresponds to the number of subclusters into which the classes are subdivided.



(a) Size=1 (b) Size=2 (c) Size=3 (d) Size=4 (e) Size=5

Fig. 6. C5.0 and the class imbalance problem with adjusted training sizes – corrected.

The data generation process used here differs from the one outlined in Section 3.1 in that the constant value 5,000/32 used in Section 3.1 is replaced by the variable x which varies as a function of s and c. The effect of this modification is that creating high degrees of imbalance does not cause a decrease in overall training set size anymore, and all the results reported for a single size now do, actually, contain training sets of the same size.

The results obtained on this new set of domains are reported in Fig. 6.[15] They show that though keeping the same number of overall training examples in all the experiments does improve the results of C5.0 a little bit (please compare the graphs of Fig. 6 with those of Fig. 2), it does not do so enough to mitigate our previous general observations. In other words, even after these results, we can still conclude that it is the class imbalance and not the decrease in overall training set size that caused C5.0 a decrease in classification accuracy.

The second issue that we consider continues the same line of investigation as that of issue 1 since it relates to the number of training examples included in each subinterval of class 0, the class from which data are subtracted when class imbalances are implemented. This study stems from the observation that, in the original domain generation process of Section 3.1, in cases where c is high (high concept complexity), i is low (large imbalance) and s is small (small training set size), class 0 contains a very small number of training examples. For example, for $c = 5$ and $i = s = 1$, there is only 1 example contained in each class 0 interval. In contrast, class 1 (which is not affected by the imbalance) contains 156 examples per interval. Even at size $s = 5$, for $c = 5$ and $i = 1$, class 0's intervals contain 10 training examples while class 1's intervals contain 156 training examples. In this part of the study, we are interested in changing this situation by guaranteeing a reasonable number of examples per interval of class 0 and generating domains of different c and i values. This means that we now completely disregard the value of s and only focus on the values of c and i. In particular, we set the number of training examples in each class 0 interval to 50. Based on these rules, we generate domains containing the following numbers of training examples per interval and per class:

- Number of Training examples per class 1 intervals: $50 \times 2^{5-i}$
- Number of Training examples per class 0 intervals: 50
- Number of intervals in each class: 2^{c-1}

The purpose of these experiments is to draw a contrast between the class imbalance problem and the small sample size problem. In particular, we want to find out whether it is 1) the class imbalance problem or 2) the fact that small subclusters are very poorly represented in domains of high concept complexity, small size and high imbalance levels that cause a sharp decrease in C5.0's classification accuracy. The results of our experiments are reported in Fig. 7.[16]

[15] Note that Fig. 6 only reports the "corrected" results as will all the graphs in the remainder of the paper. This was done to

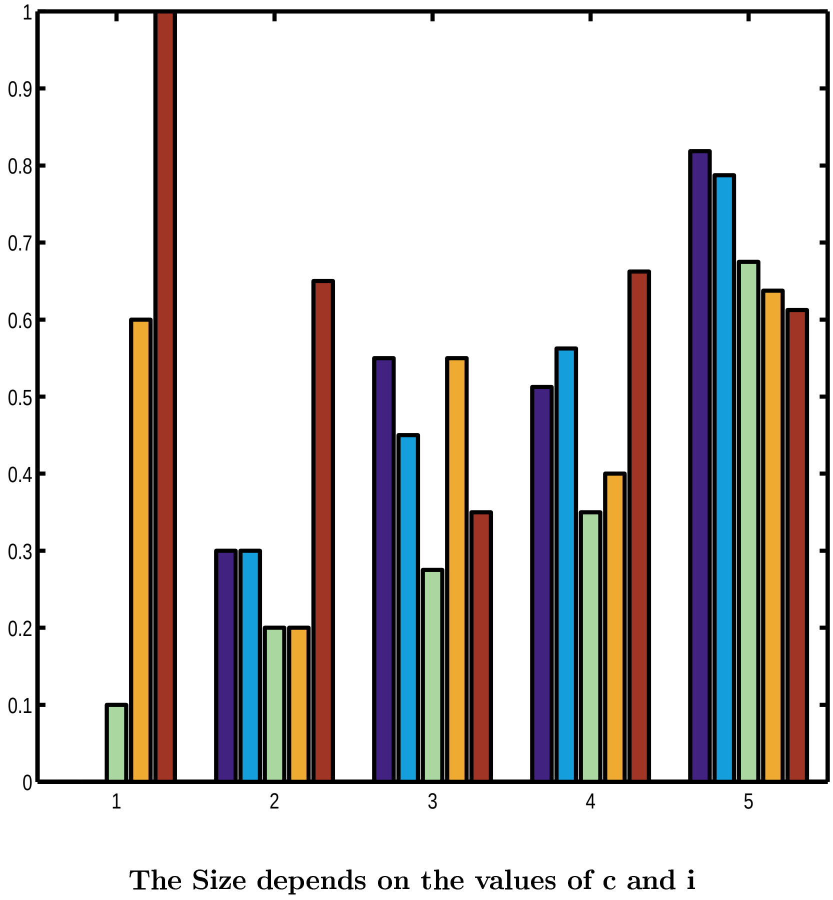


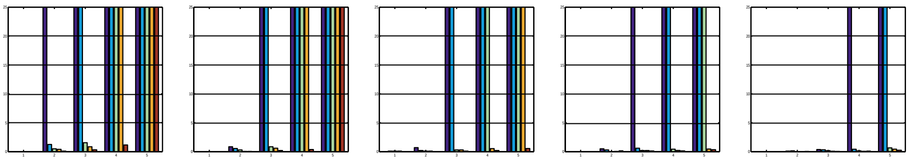
Fig. 7. C5.0 and the class imbalance problem without the small sample problem – corrected.

The results of Fig. 7 show that all the errors obtained in our experiments are at or below 1%. We, thus, consider them negligible. As well, the pattern of error does not follow the pattern we noticed in the previous graphs. This suggest that it is not the presence of a class imbalance, of a high concept complexity or of a small training set, per se, that cause C5.0 a loss of accuracy. Rather, these phenomena cause the subclusters of the small class to be very sparse and it is this condition which, in turn, causes C5.0's decrease in accuracy. This conclusion points to the true nature of the class imbalance problem which turns out not to be a problem in itself, but rather to cause a serious condition that can be thought of as the small sample problem. More specifically, rather than being a problem because of the relative size of the large and the small class, the class imbalance problem is only a problem when the size of its small class is very small with respect to the concept complexity; i.e., when it contains very small subclusters.

Practically speaking, this means that in very large domains in which there is a good chance that the subclusters of each class are represented by a reasonable number of examples, the class imbalance

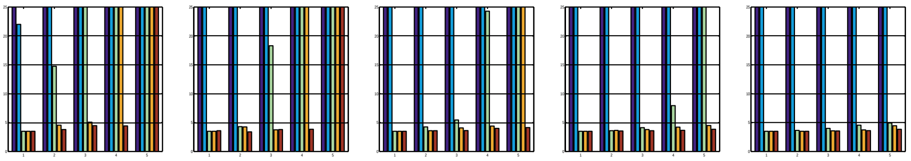
shorten the paper and improve its readability.

[16] Figure 7 is composed of a single graph because the notion of training set size, s, is meaningless in our new domain generation procedure.



(a) Size=1 (b) Size=2 (c) Size=3 (d) Size=4 (e) Size=5

Fig. 8. C5.0 and the class imbalance problem with no pruning and no class noise – corrected.



(a) Size=1 (b) Size=2 (c) Size=3 (d) Size=4 (e) Size=5

Fig. 9. C5.0 and the class imbalance problem with no pruning and class noise – corrected.

problem may be of no consequence. In smaller domains, however, because small subclusters will necessarily be present, it will need to be addressed as will be discussed in Sections 4 and 5.

The third and last issue that needs to be discussed here departs from training set size considerations. Instead, it focuses on the classification methodology used. When faced with small domains of high concept complexity, Schaffer [23] stipulated that pruning may, actually, hurt rather than help the performance of decision tree classifiers. He suggested that better results may be obtained by disabling the pruning option, usually used as a default. Because our study leads to questions of overfitting, we too decided to consider the relationship between pruning and class imbalances. In all our previous experiments, C5.0 was used with its default parameters, meaning that pruning was invoked whenever the system deemed it necessary. In the following set of experiments, we set C5.0's parameters such that pruning is always disabled. This will allow more overfitting, and thus, may allow for a better representation of the small class, but on the other hand, if noise is present, disabling C5.0 could hurt its overall performance. To verify this assumption, we, first, took the data sets used as a basis in Figs 2–5 (those generated from the methodology laid out in Section 3.1), and we ran C5.0 with pruning disabled on these sets. We then injected 15% random class noise in all the data sets and tested C5.0 with pruning disabled, again, on all these sets. The results are presented in Figs 8 and 9.

Comparing Fig. 8 with Fig. 2, we see that, unlike we expected, disabling pruning has no effect whatsoever on the accuracy of C5.0 in class imbalance domains. There is, thus, no advantage to disabling pruning in such cases, especially since, as demonstrated by the results of Fig. 9, disruption to C5.0's accuracy is introduced by a lack of pruning in case the data set contains class noise.

This concludes our investigation into the nature of the class imbalance problem. In particular, we discovered that, as long as all the subclusters of a domain have a reasonable size, the class imbalance problem is of no consequence for C5.0. If, however, some subclusters are very small, then C5.0's performance will deteriorate. In such cases, disabling pruning is not useful and other methods must be found to alleviate the problem. In the next part of the paper, we discuss and compare different ways to deal with the problem where a class imbalance yields the presence of small disjuncts, and, thus, low accuracy in the small class.

4. Question 2: A comparison of various strategies

Having identified the domains for which a class imbalance, by ways of the small sample problem, does impair the accuracy of a regular classifier such as C5.0, this section now proposes to compare the main methodologies that have been proposed to deal with this problem. First, the various schemes used for this comparison are described, followed by a comparative report on their performance. In all the experiments of this section, once again, C5.0 is used as our standard classifier. [17]

4.1. Schemes for dealing with class imbalances

Over-Sampling

Two oversampling methods were considered in this category. The first one, *random oversampling*, consists of oversampling the small class at random until it contains as many examples as the other class. The second method, *focused oversampling*, consists of oversampling the small class only with data occurring close to the boundaries between the concept and its negation. A factor of $\alpha = 0.25$ was chosen to represent closeness to the boundaries. [18]

Under-Sampling

Two under-sampling methods, closely related to the over-sampling methods were considered in this category. The first one, *random undersampling*, consists of eliminating, at random, elements of the over-sized class until it matches the size of the other class. The second one, *focused undersampling*, consists of eliminating only elements further away (where, again, $\alpha = .25$ represents closeness to the boundaries).

Cost-Modifying

The cost-modifying method used in this study consists of modifying the relative cost associated to misclassifying the positive and the negative class so that it compensates for the imbalance ratio of the two classes. For example, if the data presents a 1:9 class imbalance in favour of the negative class, the cost of misclassifying a positive example will be set to 9 times that of misclassifying a negative one.

4.2. Results for Question 2

We begin this section by discussing the results obtained on our artificial domains and we continue by discussing whether our results can be mapped to real-world domains.

[17] Although the results obtained for issue 2 of Section 3.2.2 suggest that tiny subclusters are at the root of the class imbalance problem, it is important to realize that in real-world domains, subclusters are not known and cannot be easily identified. [10,19] attempt to identify very small subclusters and to re-sample them appropriately, but a dangerous shortcoming of these approaches is that they may mistake noisy data for tiny subclusters and, thus, impart too much importance to noise. The traditional methods devised for dealing with the class imbalance problem and discussed in this section – these methods do not attempt to differentiate between the various subclusters of a class, but simply deal with each class as a whole – may, therefore, remain the best approaches, to date, for dealing with the problem.

[18] This factor means that for interval $[a, b]$, data considered close to the boundary are those in $[a, a + 0.25 \times (b - a)]$ and $[a + 0.75 \times (b - a), b]$. If no data were found in these intervals (after 500 random trials were attempted), then the data were sampled from the full interval $[a, b]$ as in the *random oversampling* methodology.

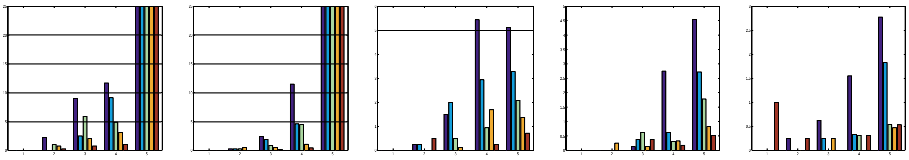

(a) Size=1 (b) Size=2 (c) Size=3 (d) Size=4 (e) Size=5

Fig. 10. Random oversampling: Error rate, corrected.

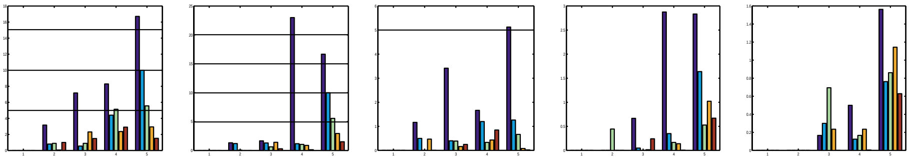

(a) Size=1 (b) Size=2 (c) Size=3 (d) Size=4 (e) Size=5

Fig. 11. Focused oversampling: Error rates, corrected.

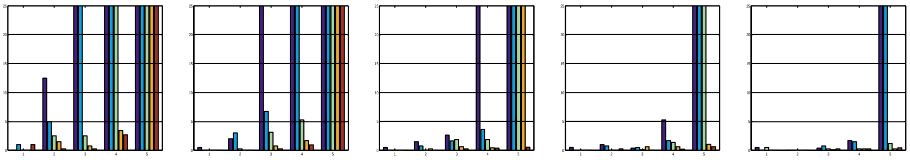

(a) Size=1 (b) Size=2 (c) Size=3 (d) Size=4 (e) Size=5

Fig. 12. Random undersampling: Corrected error rate.

4.2.1. Results on artificial domains

Unlike in Section 3.2.1, we report only on one series of results in the context of each scheme: the corrected error. The uncorrected error, the false positive error and the false negative error were left out due to space and readability constraints. False positive and false negative results will be mentioned in the text if they are of particular interest. In general, the graphs we left out are present in an earlier version of this paper published as a technical report [12].

The format of the results is the same as that used in the last section. The results for random oversampling are displayed in Fig. 10; those for focused oversampling, in Fig. 11; those for random undersampling in Fig. 12; those for focused undersampling in Fig. 13; and those for cost-modifying, in Fig. 14.

The results indicate a number of interesting points. First, all the methods proposed to deal with the class imbalance problem present an improvement over C5.0 used without any type of re-sampling or cost-modifying technique (please compare Figs 10–14 with Fig. 2). Nonetheless, not all methods help to the same extent. In particular, of all the methods suggested, undersampling is by far the least effective. This result is actually at odds with previously reported results (e.g. [2]), but we explain this disparity by the fact that in the applications considered by [2], the minority class is the class of interest while the majority class represents everything other than these examples of interest. It follows that in domains such as those in [2] the majority class includes a lot of data irrelevant to the classification task at hand that are worth eliminating by undersampling techniques. In our data sets, on the other hand, the roles of the positive and the negative class are perfectly symmetrical and no examples are irrelevant. Undersampling is, thus, not a very useful scheme in these domains. Focused undersampling does not present any advantages over random undersampling on our data sets either and neither method is recommended in

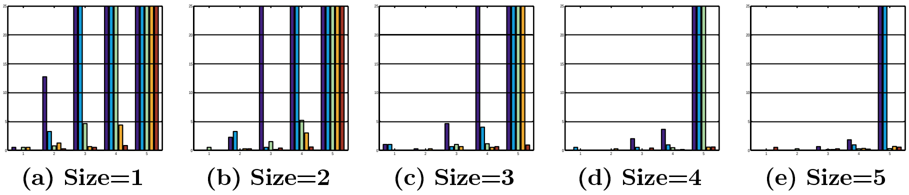

(a) Size=1 (b) Size=2 (c) Size=3 (d) Size=4 (e) Size=5

Fig. 13. Focused undersampling: Error rate, corrected.

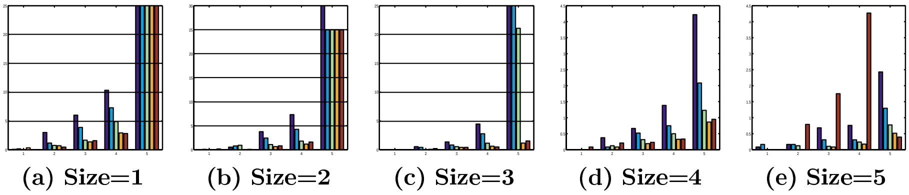

(a) Size=1 (b) Size=2 (c) Size=3 (d) Size=4 (e) Size=5

Fig. 14. Cost modifying: Corrected error rate.

those cases where the two classes play symmetrical roles and do not contain irrelevant data.

The situation in the case of oversampling is quite different. Indeed, oversampling is shown to help quite dramatically at all complexity and training set size levels. Just to illustrate this fact, consider, for example, the situation at size 2 and degree of complexity 4: while in this case, any degree of imbalance (other than the case where no imbalance is present) causes C5.0 difficulties (see Fig. 2(b)), none but the highest degree of imbalance do so when the data is oversampled at random (see figure 10(b)). Contrarily to the case of undersampling, the focused approach does make a difference – albeit, small – in the case of oversampling. Indeed, at sizes 1 and 2, focused oversampling deals with the highest level of complexity better than random oversampling (compare the results at degree of difficulty 5 in Figs 10(a) and (b) on the one hand and in Figs 11(a) and (b), on the other hand). Interestingly, in both oversampling situations, the improvement in overall error does not seem to affect the distribution of the error. Indeed, while the false positive rate does decrease, the false negative one does not significantly increase despite the fact that the size of the positive training set increased dramatically (the graphs illustrating these results can be found in [12]). This is quite an important result since it contradicts the expectation that oversampling would have shifted the error distribution and, thus, not much helped in the case where it is essential to preserve a low false negative error rate while learning the false positive error rate. In summary, oversampling and focused oversampling appear to be quite effective ways of dealing with the problem, at least in situations such as those represented in our training set.

The last method, cost-modifying, is more effective than both random oversampling and focused oversampling in all but a single observed case, that of concept complexity $c = 5$ and size $s = 3$ (compare the results for concept complexity 5 in Figs 10(c) and 11(c) on the one hand to those of Fig. 14(c) on the other). In this case both random and focused oversampling are more accurate than cost-modifying. The generally better results obtained with the cost-modifying method over those obtained by oversampling are in agreement with [13] who suggest that modifying the relative cost of misclassifying each class allows to achieve the same goals as oversampling without increasing the training set size, a step that can harm the performance of a classifier.

4.2.2. Can our study be applied to real-world domains?

We now ask whether the observations we just made on artificial domains can apply to real world domains. A particular problem with real world domains is that the degree of concept complexity, c,

Table 1
Average Percent Errors obtained on Eight UCI Domains of diverse c, s and i values. Recall that the higher the values of s and c, the higher the size of the domain and its complexity, respectively; but the higher the value of i, the lower the class imbalance

Domain	s	c	i	Error (No Cost)	Error (With Cost)
Sick	4.2	4.9	1.1	0.55%	1.1%
Allbp1	4.2	2.0	−3.3	0.15%	0.3%
Allbp2	4.2	4.4	0.6	1.1%	1.1%
Allbp3	4.2	4.6	0.7	1.25%	2.2%
Hayes-Roth1	−0.2	4.2	4.3	32.16%	26.8%
Hayes-Roth2	−0.2	4.1	4.3	19.65%	19.65%
Hayes-Roth3	−0.2	3.3	3.2	46.4%	23.15%
Monks	1.5	4.8	5.0	0%	0%

cannot be established very easily since, unlike in our experiments on artificial domains, the nature of the data is unknown. For this section, however, we decided to approximate the c value of real-world domains by counting the number of leaves, L, generated by C5.0 and setting c to the following value:

$$c = \log_2 L$$

When tested on the artificial domains for which the real c value is known, we found that this measure approximated the real c value quite closely.

From Section 3.1, we can also derive measures for the training sets size s and the imbalance level i as follows:

$$s = \log_2(32 \times T/5000) \text{ and}$$

$$i = log_2(32 \times X/Y) \text{ and}$$

where T represents the total number of positive and negative examples, X represents the total number of positive examples and Y represents the total number of negative examples.

We now report the results we obtained along with their computed c, s, and i values for different versions of several domains all taken from the UCI Repository for Machine Learning. In particular, the domains are Sick, Allbp, Hayes-Roth and Monks1.[19] Because Allbp and Hayes-Roth both contained three classes and because we are interested in 2-class classification in this paper, we generated 3 data sets each from those two domains, combining, in turn, two old classes into a single new one and keeping the third old class as the second new class. All permutations were considered.[20]

Table 1 lists the results obtained on these 8 resulting data sets along with their s, c and i values. We list the percent error obtained on the non-modified C5.0 learning procedure and the percent error obtained on C5.0 used with a cost matrix aimed at re-establishing the balance of the data. In both cases and in all domains, the results were obtained on a single run of the system where approximately 3/4 of the data was used for training and 1/4 was used for testing.

[19] These domains were chosen because they span a large range of s, c, and i values.

[20] Note that although this paper focuses on 2-class classification problems, the techniques described in Sections 4.1 and 4.2.1 are also applicable to multi-class classification problems and are expected to yield similar results. Note, also that there is no need, as we did here, to express multi-class problems as several 2-class problems. This was done, solely, for comparison purposes.

Although the results obtained on the real-world data sets do not correspond perfectly to the results obtained on the artificial domains,[21] the general trends observed in the artificial domains can also be observed on the real-world ones. In particular, we can note from the results in Table 1 that:

- In large domains (Sick, Allbp1, Allbp2 and Allbp3), no matter what the concept complexity and class imbalance are – even if the class imbalance is very large (see Allbp1) – , C5.0 is capable of classifying the data quite accurately. Using C5.0 with a cost-matrix designed to counter the data imbalance is not useful in these cases, since the data is well-classified without the use of the cost-matrix.
- In very small domains (Hayes-Roth1, Hayes-Roth2 and Hayes-Roth3), both the complexity of the concept and the amount of class imbalance affect classification accuracy. In this particular case, it appears that class imbalance is even more detrimental than concept complexity (see the results for Hayes-Roth3 (which presents a larger imbalance but a smaller concept complexity than Hayes-Roth1 and Hayes-Roth2) as compared to those for Hayes-Roth1 and Hayes-Roth2). Using C5.0 with a cost-matrix is beneficial in two out of the three Hayes-Roth data sets and does not affect the results in the third case.
- In small (but not very small domains), if no class imbalance is present, then classification can be accurate, even for very complex concepts. See, for example, the Monks domain.

Though the results, we just presented and analyzed tend to confirm the general trends we observed in the artificial domains, they are not sufficient to establish a perfect correspondence. As a matter of fact, we believe that no perfect correspondence could, in fact, be established because there are factors affecting real-world domains that were not represented in the artificial ones. In particular, the subclusters of our artificial domains all have the same size. This is probably not the case in real-world domains as discussed in [10]. Another variation not present in our artificial data sets is the presence of a large set of features, some relevant some not, that may also affect classification accuracy. In our artificial domains, each domain contained only a single and relevant feature.

Still, the trend we observe in the real-world data sets is close enough to that of our results on artificial domains to suggest that our study on artificial data sets is meaningful for real-world domains as well.

5. Question 3: Are other classifiers also sensitive to class imbalances in the data?

Sections 1 and 2 studied the question of how class imbalances affect classification and how they can be countered all in the context of C5.0, a decision tree induction system. In this section, we are concerned about whether classification systems using other learning paradigms are also affected by the class imbalance problem and to what extent. In particular, we consider two other paradigms which, a priori, may seem less prone to hindrances in the face of class imbalances than decision trees: Multi-Layer Perceptrons (MLPs) and Support Vector Machines (SVMs).

MLPs can be believed to be less prone to the class imbalance problem because of their flexibility. Indeed, they may be thought to be able to take a less global approach to partitioning the space than decision tree learning systems since they get modified by each data point sequentially and repeatedly and thus follow a top-down as well as a bottom-up search of the hypothesis space simultaneously. Even

[21] For example, the “Sick” domain has approximate $s, c,$ and i values of 4, 5, and 1, respectively and a no-cost error of 0.55% is recorded, whereas, for those values on the artificial domains, the no-cost error was of 16.67%.

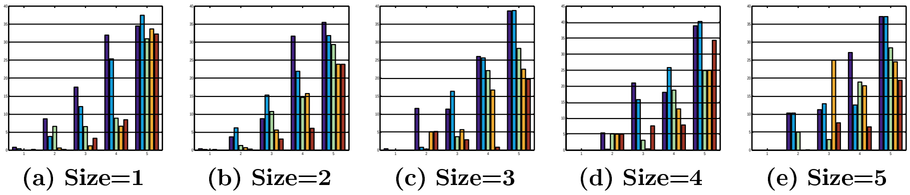

(a) Size=1 (b) Size=2 (c) Size=3 (d) Size=4 (e) Size=5

Fig. 15. MLPs and the class imbalance problem – corrected.

more convincingly than MLPs, SVMs can be believed to be less prone to the class imbalance problem than C5.0 because boundaries between classes are calculated with respect to only a few support vectors, the data points located close to the other class. The size of the data set representing each class may, thus, be believed not to matter given such an approach to classification.

The point of this section is to assess whether indeed MLPs and SVMs are less prone to the class imbalance problem and if so, to what extent. Again, we used domains belonging to the same artificial family as the ones used in the previous section and Section 3.2.1 to make this assessment. Nonetheless, because MLP and SVM training is much less time-efficient than C5.0 training and because SVM training was not even possible for large domains on our machine (because of a lack of memory), we did not conduct as extensive a set of experiments as we did in the previous sections. In particular, because of memory restrictions, we restricted our study of the effects of class imbalances to domains of size 1 for SVMs (for MLPs, we actually conducted our study on all sizes for the imbalance study since we did not have memory problems) and, because of low training efficiency, we only looked at the effect of random oversampling and undersampling for size 1 on both classifiers.[22]

5.1. MLPs and the class imbalance problem

Because of the nature of MLPs, more experiments needed to be run than in the case of C5.0. Indeed, because the performance of MLPs depends upon the number of hidden units it uses, we experimented with 2, 4, 8 and 16 hidden units and reported only the results obtained with the optimal network capacity. Other default values were kept fixed (i.e., all the networks were trained by the Levenberg-Marquardt optimization method, the learning rate was set to 0.01; the networks were all trained for a maximum of 300 epochs or until the performance gradient descended below 10^{-10}; and the threshold for discrimination between the two classes was set to 0.5). This means that the results are reported a-posteriori (after checking all the possible network capacities, the best results are reported).

The results are presented in Fig. 15 which reports on the performance of MLPs on imbalanced domains of concept complexities $c = 1 \dots 5$, training set sizes $s = 1 \dots 5$, and imbalance levels $i = 1 \dots 5$ and in Fig. 16 which reports on the performance of MLPs on (a) imbalanced domains, (b) imbalanced but oversampled domains, and (c) imbalanced but undersampled domains of concept complexity $c = 1 \dots 5$, training set size $s = 1$, and imbalance levels $i = 1 \dots 5$. The format used to report these results is the same as the one used in the previous two sections.

There are several important differences between the results obtained with C5.0 and those obtained with MLPs. In particular, in all the MLP graphs a large amount of variance can be noticed in the results

[22] Unlike in Question 2 for C5.0, our intent here is not to compare all possible techniques for dealing with the class imbalance problem with MLPs and SVMs. Instead, we are only hoping to shed some light on whether these two systems do suffer from class imbalances and get an idea of whether some simple remedial methods can be considered for dealing with the problem.

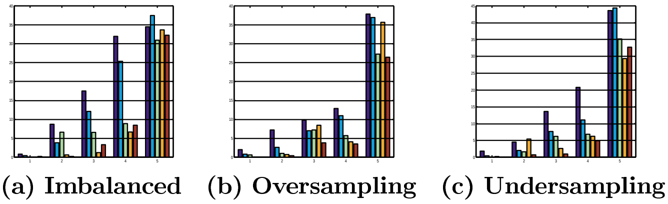


Fig. 16. Lessening the class imbalance problem in MLP networks – corrected.

despite the fact that all results were averaged over five different trials. The conclusions derived from these graphs should thus be thought of reflecting general trends rather than specific results. Furthermore, a careful analysis of the graphs reveals that MLPs do not seem to suffer from the class imbalance problem in the same way as C5.0. Looking, for example, at the graphs for size 1 for C5.0 and MLP (see Fig. 2(a) on the one hand and Fig. 15(a) on the other hand), we see that C5.0 displays extreme behaviors: it either does a perfect (or close to perfect) job or it misclassifies 25% of the testing set (see Fig. 2(a)). For MLP, this is not the case and misclassification rates span an entire range. As a result, MLP seems less affected by the class imbalance problem than C5.0. For example, for size $= 1$, and concept complexity 4, C5.0 run with imbalance levels 4, 3, 2, and 1 (see Fig. 2(a)) misclassify 25% of the testing set whereas MLP (see Fig. 15(a)) misclassifies the full 25% of the testing set for only imbalance levels 2 and 1 – the highest degrees of imbalance (some misclassification also occurs at imbalance levels 3 and 4, but not as drastic as for levels 2 and 1). Note that the difficulty displayed by MLPs at concept complexity 5 for all sizes is probably caused by the fact that, once again for efficiency reasons, we did not try networks of capacity greater than 16 hidden units. We, thus, ignore these results in our discussion.

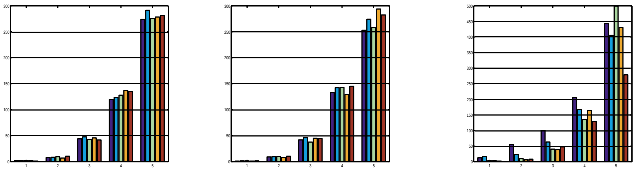
Another important difference that can be seen by looking at the graphs for size 5 of both C5.0 (Fig. 2(e)) and MLP (Fig. 15(e)) is that while the overall size of the training set makes a big difference in the case of C5.0, it doesn't make any difference for MLP: except for the highest imbalance levels combined with the highest degrees of complexity, C5.0 does not display any noticeable error at training set size 5 – the highest. MLPS's on the other hand do. This may be explained by the fact that the number of epochs necessary for MLP networks to process large quantities of data is greater than it is for smaller quantities, and extra time was not provided in our experiments.

Because MLP generally suffers from the class imbalance problem, we asked whether, like for C5.0, this problem can be lessened by simple techniques. For the reason of efficiency noted earlier and for reasons of conciseness of report, we restricted our experiments to the cases of random oversampling and random undersampling and to the smallest size (size 1) case. The results of these experiments are shown in Fig. 16 which display the results obtained with no re-sampling at all (a repeat of Fig. 15(a)), random oversampling and random undersampling.

The results in these figures show that both oversampling and undersampling have a noticeable effect for MLPs, though once again, oversampling seems more effective. The difference in effectiveness between undersampling and oversampling, however, is less pronounced in the case of MLPs than it was in the case of C5.0. As a matter of fact, undersampling is much less effective than oversampling for MLP in the most imbalanced cases, but it has comparable effectiveness in all the other ones. This suggests that like for C5.0, simple methods for counteracting the effect of class imbalances should be considered when using MLPs.

5.2. SVMs and the class imbalance problem

Like for MLPs, more experiments needed to be run with SVMs than in the case of C5.0. Actually, even more experiments were run with SVMs than with MLPs. We ran SVMs with a Gaussian Kernel



(a) **Imbalanced** (b) **Oversampling** (c) **Undersampling**

Fig. 17. The class imbalance problem in SVMs – corrected.

but since the optimal variance of this kernel is unknown, we tried 10 different possible variance values for each experiment. We experimented with variances 0.1, 0.2, etc. up to 1. We did not experiment with modifications to the soft margin threshold (note that such experiments would be equivalent to the cost-modification experiments of C5.0). Like for MLPs, the results are reported a-posteriori (after checking the results with all the possible variances, we report only the best results obtained).

As mentioned before, because of problems of memory capacity, the results are reported for a training set size of 1 and it was not possible to report results similar to those reported for MLP in Fig. 15. Instead, we report results similar to those of Fig. 16 for MLPs in Fig. 17 for SVMs. In particular, these figures show results obtained by SVMs with no resampling at all, random oversampling and random undersampling for size 1.

The results displayed in Fig. 17(a) show that there is a big difference between C5.0 and MLPs on the one hand and SVMs on the other. Indeed, while for both C5.0 and MLPs, the leftmost column in a cluster of columns – those columns representing the highest degree of imbalances – were higher than the others (Figs 2(a) and 15 or 16 (a)), in the case of SVMs (Fig. 17(a)) the 5 columns in the clusters display a flat value or the leftmost columns have lower values than the rightmost ones (see the case of concept complexity 4 in particular).

This is quite interesting since it suggests that SVMs are absolutely not sensitive to the class imbalance problem. As a matter of fact, Figs 17(b) and (c) show that oversampling the data at random does not help in any way and undersampling it at random even hurts the SVM's performance.

All in all, this suggests that when confronted to a class imbalance situation, it might be wise to consider using SVMs since they are robust to such problems. Of course, this should be done only if SVMs fare well on the particular problem at hand as compared to other classifiers. In our domains, for example, up to concept complexity 3 (included), SVMs (Fig. 17(a)) are competitive with MLPs (Fig. 16(a)) and only slightly less competitive with oversampled C5.0 (Fig. 10(a)). At complexity 4, oversampled MLPs (Fig. 16(b)) and C5.0 (Fig. 10(b)) are more accurate than SVMs (Fig. 17(a)).

6. Conclusion

The purpose of this paper was to explain the nature of the class imbalance problem, compare various simple strategies previously proposed to deal with the problem and assess the effect of class imbalances on different types of classifiers.

Our experiments allowed us to conclude that the class imbalance problem is a relative problem that depends on 1) the degree of class imbalance; 2) the complexity of the concept represented by the data; 3) the overall size of the training set; and 4) the classifier involved.

More specifically, we found that the higher the degree of class imbalance the higher the complexity of the concept and the smaller the overall size of the training set, the greater the effect of class imbalances in classifiers sensitive to the problem. This observation was explained by the fact that high complexity and imbalance levels as well as small training set sizes give rise to some very small subclusters that cannot be classified accurately. Conversely, however, the class imbalance problem causes no harm when all subclusters have a reasonable size, thus dispelling the belief that classification errors will necessarily occur if one class is represented by a large data set and the other, by a small one.

For the case where class imbalances yield very small subclusters, we tested three different types of classifiers on different instances of the problem. These classifiers were not sensitive to the class imbalance problem in the same way: C5.0 was the most sensitive of the three, MLPs came next and displayed a different pattern of sensitivity (a grayer-scale type compared to C5.0's which was more categorical); and SVMs came last since they were shown not to be at all sensitive to the problem.

Finally, for classifiers sensitive to the class imbalance problem, it was shown that simple re-sampling methods could help a great deal whereas they do not help, and in certain cases, even hurt the classifier insensitive to class imbalances. An extensive and careful study of the classifier most affected by class imbalances, C5.0, reveals that while random re-sampling is an effective way to deal with the problem, random oversampling is a lot more useful than random undersampling. More "intelligent" oversampling helps even further, but more "intelligent" undersampling does not. The cost-modifying method seems more appropriate than the over-sampling and even focused over-sampling method except in one case of very high complexity and medium-range training set size.

7. Future work

The work in this paper presents a systematic study of class imbalance problems on a large family of domains. Nonetheless, this family does not cover all the known characteristics that a domain may take. For example, we did not study the effect of irrelevant data in the majority class. We assume that such a characteristic should be important since it it may make undersampling more effective than oversampling or even cost-modifying on domains presenting a large variance in the distribution of the large class. Other characteristics, such as the number of features present and their relevance, should also be studied since they may reveal other strengths and weaknesses of the remedial methods surveyed in this study.

In addition several other methods for dealing with class imbalance problems should be surveyed. Two approaches in particular are 1) over-sampling by creation of new synthetic data points not present in the original data set but presenting similarities to the existing data points and 2) learning from a single class rather than from two classes, trying to *recognize* examples of the class of interest rather than *discriminate* between examples of both classes.

Finally, it would be interesting to combine, in an "intelligent" manner, the various methods previously proposed to deal with the class imbalance problem. Preliminary work on this subject was previously done by [1,4,5], but much more remains to be done in this area.

References

[1] N. Chawla, K. Bowyer, L. Hall and P. Kegelmeyer, SMOTE: Synthetic Minority Over-sampling TEchnique, *International Conference on Knowledge Based Computer Systems*, 2000.

[2] P. Domingos, Metacost: A General Method for Making Classifiers Cost-sensitive, *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1999, pp. 155–164.

[3] C. Elkan, The Foundations of Cost-Sensitive Learning, *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, 2001.

[4] A. Estabrooks, *A Combination Scheme for Inductive Learning from Imbalanced Data Sets*, MCS Thesis, Faculty of Computer Science, Dalhousie University, 2000.

[5] A. Estabrooks and N. Japkowicz, A Mixture-of-Experts Framework for Concept-Learning from Imbalanced Data Sets, *Proceedings of the 2001 Intelligent Data Analysis Conference*, 2001, pp. 34–43.

[6] T.E. Fawcett and F. Provost, Adaptive Fraud Detection, *Data Mining and Knowledge Discovery* **3**(1) (1997), 291–316.

[7] R.C. Holte, L.E. Acker and B.W. Porter, Concept Learning and the Problem of Small Disjuncts, *Proceedings of the Eleventh Joint International Conference on Artificial Intelligence*, 1989, pp. 813–818.

[8] N. Japkowicz, C. Myers and M. Gluck, A Novelty Detection Approach to Classification, *Proceedings of the Fourteenth Joint Conference on Artificial Intelligence*, 1995, pp. 518–523.

[9] N. Japkowicz, Learning from Imbalanced Data Sets: A Comparison of Various Solutions, in: *Proceedings of the AAAI'2000 Workshop on Learning from Imbalanced Data Sets*, N. Japkowicz, ed., AAAI Press, 2000.

[10] N. Japkowicz, Concept-Learning in the Presence of Between-Class and Within-Class Imbalances, *Advances in Artificial Intelligence: Proceedings of the 14th Conference of the Canadian Society for Computational Studies of Intelligence*, 2001, pp. 67–77.

[11] N. Japkowicz, Supervised versus Unsupervised Binary-Learning by Feedforward Neural Networks, *Machine Learning* **42**(1/2) (2001), 97–122.

[12] N. Japkowicz and S. Stephen, The Class Imbalance Problem: A Systematic Study, *Technical Report TR-2001-07*, University of Ottawa, 2001.

[13] S. Lawrence, I. Burns, A.D. Back, A.C. Tsoi and C.L. Giles, Neural Network Classification and Unequal Prior Class Probabilities, in: *Tricks of the Trade, Lecture Notes in Computer Science State-of-the-Art Surveys*, G. Orr, R.-R. Muller and R. Caruana, eds, Springer Verlag, 1998, pp. 299–314.

[14] M. Kubat and S. Matwin, Addressing the Curse of Imbalanced Data Sets: One-Sided Sampling, *Proceedings of the Fourteenth International Conference on Machine Learning*, 1997, pp. 179–186.

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

[16] P.M. Murphy and D.W. Aha, *UCI Repository of Machine Learning Databases*, University of California at Irvine, Department of Information and Computer Science, 1994.

[17] D. Lewis and J. Catlett, Heterogeneous Uncertainty Sampling for Supervised Learning, *Proceedings of the Eleventh International Conference of Machine Learning*, 1994, pp. 148–156.

[18] C.X. Ling and C. Li, Data Mining for Direct Marketing: Problems and Solutions, *International Conference on Knowledge Discovery & Data Mining*, 1998.

[19] A. Nickerson, N. Japkowicz and E. Milios, Using Unsupervised Learning to Guide Re-Sampling in Imbalanced Data Sets, *Proceedings of the Eighth International Workshop on AI and Statitsics*, 2001, pp. 261–265.

[20] M. Pazzani, C. Merz, P. Murphy, K. Ali, T. Hume and C. Brunk, Reducing Misclassification Costs, *Proceedings of the Eleventh International Conference on Machine Learning*, 1994, pp. 217–225.

[21] R. Quinlan, C5.0: An Informal Tutorial, http://www.rulequest.com/see5-unix.html, 2001.

[22] D.E. Rumelhart, G.E. Hinton and R.J. Williams, Learning Internal Representations by Error Propagation, in: *Parallel Distributed Processing*, D.E. Rumelhart and J.L. McClelland, eds, MIT Press, Cambridge, MA, 1986, pp. 318–364.

[23] C. Schaffer, Overfitting Avoidance as Bias, *Machine Learning* **10** (1993), 153–178.

[24] J. Swets, R. Dawes and J. Monahan, Better Decisions through Science, *Scientific American* (2000), 82–97.

[25] G. Weiss, Learning with Rare Cases and Small Disjuncts, *Proceedings of the Twelfth International Conference on Machine Learning*, 1995, pp. 558–565.

[26] G.M. Weiss and F. Provost, The Effect of Class Distribution on Classifier Learning: An Empirical Study, *Technical Report ML-TR-44*, Department of Computer Science, Rutgers University, 2001.