# Mining with rarity: A unifying framework

**Article** · January 2004

Source: DBLP

CITATIONS
1,291

READS
3,906

**1 author:**

Gary M. Weiss
Fordham University
**96** PUBLICATIONS **10,474** CITATIONS

SEE PROFILE

# Mining with Rarity: A Unifying Framework

Gary M. Weiss
AT&T Laboratories
30 Knightsbridge Road
Piscataway, NJ 08854
gmweiss@att.com

## ABSTRACT

Rare objects are often of great interest and great value. Until recently, however, rarity has not received much attention in the context of data mining. Now, as increasingly complex real-world problems are addressed, rarity, and the related problem of imbalanced data, are taking center stage. This article discusses the role that rare classes and rare cases play in data mining. The problems that can result from these two forms of rarity are described in detail, as are methods for addressing these problems. These descriptions utilize examples from existing research, so that this article provides a good survey of the literature on rarity in data mining. This article also demonstrates that rare classes and rare cases are very similar phenomena—both forms of rarity are shown to cause similar problems during data mining and benefit from the same remediation methods.

## Keywords

Rare cases, rare classes, class imbalance, small disjuncts, sampling, inductive bias, cost-sensitive learning.

## 1. INTRODUCTION

In life it is often the rare objects that are most interesting. This carries over to data sets, which, after all, represent some aspect of reality. Consequently, in data mining it is often rare objects that are of primary interest. Examples abound and include identifying fraudulent credit card transactions [9], learning word pronunciations [3], predicting pre-term births [22], predicting telecommunication equipment failures [50], and detecting oil spills from satellite images [34]. It is important to study rarity in the context of data mining because rare objects are typically much harder to identify than common objects and most data mining algorithms have a great deal of difficulty dealing with rarity.

Before proceeding, it is useful to discuss what exactly is meant by rarity. Much of the research on rarity relates to *rare classes*, or, more generally, class imbalance. This type of rarity requires labeled examples and is associated with classification problems. The data set used to detect oil spills from satellite images provides a good example of a rare class. Because only 41 of the 937 satellite images contain oil slicks, we can say that oil slicks are rare (i.e., a rare class).[1]

A second type of rarity concerns *rare cases*. Informally, rare cases correspond to a meaningful but relatively small subset of the data, or equivalently, define a small region of the instance space. Rare cases depend only on the distribution of data and therefore are defined for both labeled and unlabeled data, and for supervised and unsupervised data mining tasks. Rare cases are naturally defined by the domain and will share common characteristics. In the case of labeled data, a rare case corresponds to a subconcept, or subclass, that occurs infrequently. For example, if the task is to identify the presence of cancer using blood test results, a rare case may correspond to those test indicators that are associated with a rare form of cancer. Unfortunately, except for artificially generated domains, rare cases are not easily identified. However, unsupervised learning techniques such as clustering may help to identify them, and they may also manifest themselves as *small disjuncts* in classifiers induced from the data. Small disjuncts are those disjuncts in the learned classifier that cover few training examples [23].

Figure 1 shows an artificial domain with two classes, A and B, where A is the rare (minority) class and B is the common (majority) class. Holding with established conventions, the rare class is designated the positive class and the common class is designated the negative class. The true decision boundaries are displayed with solid lines while the learned boundaries are displayed with dashed lines. The labeled examples are represented in the figure using the "+" and "-" symbols.

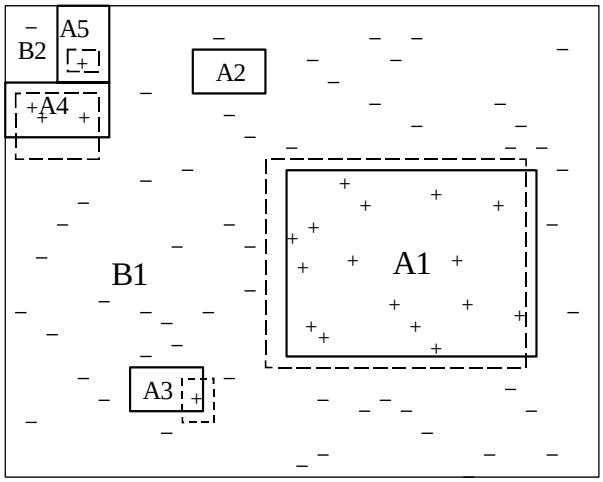


**Figure 1: Graphical representation of a rare class and rare case**

The five subconcepts associated with class A are labeled A1-A5. Subconcepts A2-A5 correspond to rare cases, whereas A1 corresponds to a fairly common case, covering a substantial portion of the instance space. The majority class is comprised of two subconcepts, B1 and B2. Subconcept B1 is a very general case that covers most of the instance space. Subconcept B2, on the other hand, corresponds to a rare case, demonstrating that common classes may contain rare cases. However, we expect rare classes, by their very nature, to contain a greater proportion of rare cases

[1] Rarity must be defined with respect to some distribution. In this case it is defined with respect to the training distribution, although ideally it should be defined with respect to the underlying (but typically unknown) distribution.

(a very rare class cannot contain any common cases). Figure 1 also shows that rare cases A3, A4, and A5 cause small disjuncts to be formed in the learned classifier (rare case A2 is "missed" completely by the classifier). Rare cases, like rare classes, can be considered the result of a form of data imbalance and have in fact been referred to as *within-class* imbalances [24].

An example of a rare class in the context of unsupervised learning can be found in association rule mining and, in particular, in market basket analysis, which looks at how the items purchased by a customer are related. Some groupings of items, such as *peanut butter* and *jelly*, occur frequently and can be considered common cases. Other associations may be extremely rare. For example, *food processor* and *cooking pan* will be an extremely rare association (i.e., case) in a supermarket, not because the items are unlikely to be purchased together, but because neither item is frequently purchased in a supermarket [37].

One final aspect of rarity worth considering is whether rarity is an absolute or relative property. For example, if a case or class covers 1% of a dataset that has one thousand entries, we would certainly say that the class/case is rare—but what if the dataset contains ten million entries? From a practical perspective we are concerned with those properties that make data mining difficult and, as we shall see, both absolute and relative rarity pose problems for conventional data mining systems.

This article discusses the issues associated with mining with rarity and possible methods for effectively mining in the presence of rarity. Unlike most previous work, we consider rare classes and rare cases together, as well as how rarity affects both supervised and unsupervised learning tasks. Consequently we provide a more general analysis of mining with rarity than has been presented previously. This proves to be very useful because rare cases and rare classes cause many of the same problems for data mining and can benefit from some of the same remediation techniques. Thus, this article unifies research topics that previously were considered separate. This article also advances previous work by thoroughly discussing the problems associated with rarity as well as methods for better handling rarity.

## 2. WHY DOES RARITY MAKE DATA MINING DIFFICULT?

There are a number of problems that arise when mining rare classes and rare cases. We divide these problems into categories and describe each one in detail in separate subsections.

### 2.1 Improper Evaluation Metrics

Evaluation metrics play a critical role in data mining. Metrics are used to guide the data mining algorithms and to evaluate the results of data mining. For example, when using a decision tree algorithm to solve a classification task, information gain may be used to guide the construction of the decision tree while accuracy may be used to evaluate the performance of the final tree.

We begin by discussing the evaluation metrics most commonly used to assess the results of data mining. If these metrics do not adequately value rarity, then data mining is not likely to handle rare classes and rare cases very well. Classification accuracy, which computes the fraction of examples that are correctly classified, is the most commonly used evaluation metric for classification tasks. The flaw with this metric with respect to rare classes is well known—rare classes have less impact on accuracy than common classes. For example, for a two-class problem with a class distribution of 90:10, the performance of the classifier on majority-class examples will count nine times as much as the performance on minority-class examples. For similar reasons, accuracy does not value rare cases as much as common cases.

An empirical study by Weiss and Provost [52] supports the conclusion that accuracy leads to poor minority-class performance. These results, averaged over twenty-six data sets, show that the error rate of minority-class classification rules is 2-3 times that of the rules that identify majority-class examples and, just as important, minority-class examples are much less likely to be predicted than majority-class examples. Using terminology from information retrieval, the minority class has much lower *precision* and *recall* than the majority class. Many practitioners have observed that for extremely skewed class distributions the recall of the minority class is often 0—there are no classification rules generated for the minority class.

We now discuss the use of evaluation metrics to guide the search algorithms employed by data mining systems. In the interest of space we only consider some representative metrics, used to guide the construction of decision trees. Most decision trees are grown in a top-down manner. Test conditions are repeatedly selected and cause new branches (and levels) to be added to the tree. Test selection functions, which identify and evaluate potential test conditions, typically assess the goodness of a test by determining the purity of each branch and then computing an overall purity value, which weights the purity of each branch by the number of examples that follow that branch. These metrics (e.g., information gain) prefer tests that result in a balanced tree where purity is increased for most of the examples to a test that yields high purity for a relatively small subset of the data but low purity for the rest [44]. The problem with this is that a single high purity branch may identify a useful rare case. Another problem occurs when each classification "rule" is assigned a class label. The label is usually assigned based on an error estimate and a threshold (typically 50%). Accuracy may be used to determine the class label, but this will yield an unreliable estimate when the number of examples that are covered is small [39; 54].

Association rule mining systems use the support and confidence metrics to guide the search for association rules. Support measures the number of records that contain the association, while confidence measures the percentage of times that the association is found. In general, association rule mining systems only look for rules that have some minimum support, *minsup*. This allows much of the search space to be pruned. For efficiency reasons *minsup* cannot be set low enough to identify rare associations (this is discussed further in Section 2.3).

In this section we described how some evaluation metrics can make it difficult to mine rare classes and rare cases. These metrics are finally giving way to more sophisticated metrics that better value rarity. Some of these metrics are described in Section 3.1.

### 2.2 Lack of Data: Absolute Rarity

The most fundamental problem with rarity is the associated lack of data. In this section we are concerned with *absolute* rarity, where the number of examples associated with the rare class/case is small in an absolute sense. In this situation the lack of data makes it difficult to detect regularities within the rare classes/cases.

Figure 2 demonstrates the problems that can result from an "absolute" lack of data. The figure only shows the region from Figure 1 surrounding rare case A3. The left side of Figure 2 shows the

original situation, where A3 contains only one positive example, while the right side represents the situation where much more data is available. As can clearly be seen, the learned decision boundaries (shown using dashed lines) much more closely approximate the true decision boundaries when more data is available.

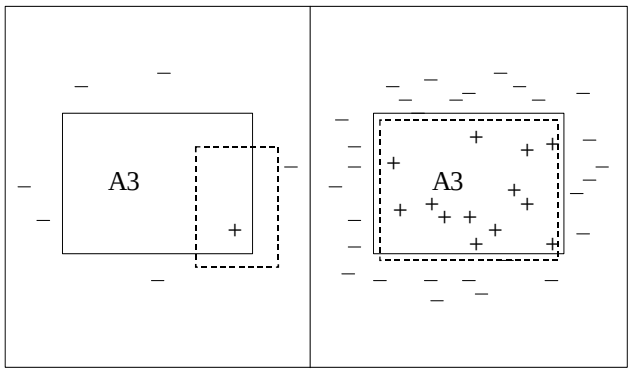


**Figure 2: The impact of an "absolute" lack of data**

So, rare cases may be due to a lack of data. The impact that these rare cases have on classification performance has been analyzed. One study, which employed synthetically generated data sets, showed that rare cases have a much higher misclassification rate than common cases [48]. We refer to this as the *problem with rare cases*. This research further demonstrated something that had previously been assumed—that rare cases cause small disjuncts in the learned classifier. The *problem with small disjuncts*, observed in many empirical studies, is that they (i.e., small disjuncts) generally have a much higher error rate than large disjuncts [2; 3; 23; 47; 48; 51]. We can see that this is again the result of a lack of data. The most thorough empirical study of small disjuncts analyzed thirty real-world data sets and showed that, in the classifiers induced from these data sets, the vast majority of errors are concentrated in the smaller disjuncts [51].

The error-prone nature of small disjuncts is a direct result of rarity. Therefore, an understanding of why small disjuncts are so error prone will help explain why rarity is a problem. One explanation is that some small disjuncts may not represent rare, or exceptional, cases, but rather something else—such as noisy data. Thus, only small disjuncts that are "meaningful" should be kept. Most classifier induction systems have some means of preventing overfitting, to remove subconcepts (i.e., disjuncts) that are not believed to be meaningful. Statistical significance testing is used by some systems to prevent this overfitting. Disjuncts that cover few examples will not pass these significance tests. If a data set has two classes and an equal number of training examples in each, then a disjunct is 99% significant if and only if it covers at least 7 training examples [23]. The basic problem is that the significance of small disjuncts cannot be reliably estimated and consequently significant small disjuncts may be eliminated along with the insignificant ones. Empirical results [23] show that the strategy of eliminating all small disjuncts results in an increase in overall error rate and hence is not a good strategy. Error estimation techniques are also unreliable when there are only a few examples, and hence they suffer from the same basic problem. These approaches work well for large disjuncts because in these cases statistical significance and error rate estimation techniques yield relatively reliable estimates—something they do not do for small disjuncts.

## 2.3 Relative Lack of Data: Relative Rarity

One problem with rarity is that rare "objects" can be hard to find. This holds true even if rarity is relative—that is, objects are not rare in an absolute sense but are rare relative to other objects. A popular phrase that illustrates this is "like a needle in a haystack". This phrase is relevant because it is the large number of strands of hay in the haystack that makes it hard to find the needle.

So why is finding/identifying rare objects (patterns, cases, events, etc.) difficult when data mining? One reason is that the rare objects are not easily located using greedy search heuristics and more global methods are, in general, not tractable. Greedy search heuristics have a problem with rarity for several reasons. First, rare objects may depend on the conjunction of many conditions and therefore examining any single condition in isolation may not provide much information, or guidance. While this may also be true of common objects, with rare objects the impact is greater because the common objects may obscure the true signal (the related issue of data fragmentation is covered in Section 2.4).

As a specific example of this general problem, consider the association rule mining problem described earlier, where we want to be able to detect the association between *food processor* and *cooking pan*. The problem is that both items are rarely purchased in a supermarket, so that even if the two are often purchased together when either one is purchased, this association may not be found. To find this association, the minimum support threshold for the algorithm must be set quite low. However, if this were done, it would cause a combinatorial explosion because frequently occurring items will be associated with one another in an enormous number of ways. This problem has been called the *rare item problem* [37]. The fact that these random co-occurrences will swamp the meaningful associations between rare items is one example of the problem with relative rarity.

## 2.4 Data Fragmentation

Many data mining algorithms employ a divide-and-conquer approach, where the original problem is decomposed into smaller and smaller problems, which results in the instance space being partitioned into smaller and smaller pieces. Decision tree algorithms are a good example of this approach in that they begin with all of the data (all of the instance space) and repeatedly partition it into smaller and smaller pieces. This process may lead to data fragmentation [20]. Data fragmentation is a problem because regularities can then only be found within each individual partition, which will contain less data. While data fragmentation is always a concern, it is more of a concern when mining rare classes/cases, because of the existing "lack of data" problem described in Section 2.2. Thus all iterative divide-and-conquer approaches may have difficulty in the presence of rarity. Data mining algorithms that do not employ a divide-and-conquer approach are therefore more appropriate when mining rare classes/cases. Some of these methods are described in Section 3.

## 2.5 Inappropriate Inductive Bias

Generalizing from specific examples, or induction, requires an extra-evidentiary bias. Without such a bias "inductive leaps" are not possible and learning cannot occur. The bias of a data mining system is therefore critical to its performance. Many learners utilize a general bias in order to foster generalization and avoid overfitting. This bias can adversely impact the ability to learn rare cases and rare classes.

Consider a learner with a maximum-generality bias [23]. Once the learner decides to create a disjunct that covers some set of training examples, it selects the most general set of conditions that satisfy those examples but no others. This can be contrasted with a maximum-specificity bias, which will add all possible conditions that satisfy the training examples. The maximum-generality bias works well for large disjuncts but not for small disjuncts, leading to the observed problem with small disjuncts. Thus we infer that the maximum-generality bias is not appropriate for rare cases, since it may make them overly general. This leads to error-prone classification rules for predicting the minority class, which may subsequently be pruned. Most methods for addressing the problem of small disjuncts (and rare cases) operate by adjusting the bias of the learner.

Inductive bias also plays a role with respect to rare classes. Many induction systems will tend to prefer the more common classes in the presence of uncertainty (i.e., they will be biased in favor of the class priors). As a simple example, imagine a decision tree learner that branches on all possible feature values when splitting a node in the tree. If one of the resulting branches covers no training examples, then there is no evidence on which to base a classification. What label should be associated with the resulting leaf node? Most decision-tree learners will predict the most frequently occurring class, biasing the results against rarer classes.

## 2.6 Noise

Noisy data will affect the way any data mining system behaves, but, what is interesting from the perspective of this article, is that noise has a greater impact on rare cases than on common cases. To see this, consider Figure 3. The left side of Figure 3 reproduces the bottom part of Figure 1, while the right side does the same thing but also introduces some noisy examples. Noise creates a problem when positive-class examples are found in the negative class (class B) and negative-class examples are found in the positive class (class A).

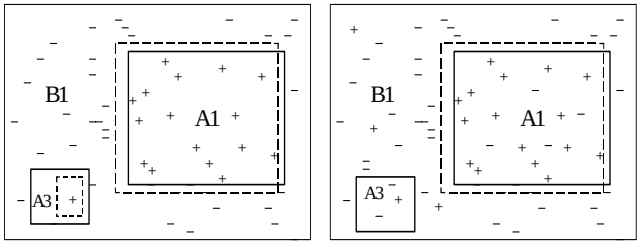


**Figure 3: The effect of noise on rare cases**

Because rare cases have fewer examples to begin with, it will take fewer "noisy" examples to impact the learned subconcept. As we can see in Figure 3, the four noisy data points in A1 have no impact on the learned decision boundary, because of the learner's ability to generalize. However, the two noisy data point in A3 cause the learner to not learn this rare case at all (i.e., there is no decision boundary). In this case the learner cannot distinguish between exceptional (rare) cases and noise, a problem that has been analyzed previously [48]. If the learner were modified to generalize less, so that a portion of A3 were correctly learned, this would most likely have the undesirable effect of having small disjuncts formed to cover the noisy examples in A1 and B1. Thus, noise necessitates the use of overfitting avoidance techniques (e.g., pruning) to eliminate noise-induced small disjuncts and a consequence of this is that some "true" rare cases will not be learned. Should these rare cases be important enough, one should be able to adjust the bias of the learner to include them, even though this will have some undesirable consequences.

# 3. METHODS FOR ADDRESSING RARITY

This section describes methods for dealing with the problems associated with rarity that were listed in the previous section.

## 3.1 More Appropriate Evaluation Metrics

Evaluation metrics that take rarity into account can improve data mining by better guiding the search process and better evaluating the end-result of data mining. Because some metrics are used in both ways, they are discussed together in this section.

Accuracy places more weight on the common classes than on rare classes, which makes it difficult for a classifier to perform well on the rare classes. Because of this, additional metrics are coming into widespread use. Perhaps the most common is ROC analysis and the associated use of the area under the ROC curve (AUC) to assess overall classification performance [4; 40]. AUC does not place more emphasis on one class over the other, so it is not biased against the minority class. ROC curves, like precision-recall curves, can also be used to assess different tradeoffs—the number of positive examples correctly classified can be increased at the expense of introducing additional false positives. ROC analysis has been used by many systems designed to deal with rarity, such as the Shrink data mining system [34].

Precision and recall are metrics from the information retrieval community that are useful for data mining. The precision of a classification rule, or set of rules, is the percentage of times the predictions associated with the rule(s) are correct. If these rules predict class *X* then recall is the percentage of all examples belonging to *X* that are covered by these rule(s). For example, three rules that predict class A may have a precision of 80% and a recall of 10% (i.e., the three rules cover 10% of all A examples).

Many systems have used some variation of precision and recall to guide the data mining process and evaluate the end result [7; 28; 34; 49]. Two examples are the geometric mean (the square root of precision times recall) and the F-measure [45]. The F-measure is parameterized and can be adjusted to specify the relative importance of precision vs. recall (F1-measure counts both equally, F2-measure counts recall twice as much). Rare cases and classes are valued when using these metrics because both precision and recall are defined with respect to the positive (rare) class. Timeweaver [49], a data mining system that employs a genetic algorithm, uses the F-measure after each iteration to evaluate the fitness of the classification rules evolved during that iteration. The parameter to the F-measure that controls the relative importance of precision vs. recall is varied periodically to ensure that a diverse set of classification rules is evolved—so that some rules will have high precision while others will have high recall. The expectation is that this will eventually lead to good solutions that have rules with both high precision and recall. Other research has used the F-measure to compare the performance of different data mining algorithms [17; 29; 30].

The problem with using accuracy to label rules that cover few examples is that it produces very unreliable estimates—and is not even defined if no examples in the training set are covered. Several metrics have therefore been designed to provide better esti-

mates of accuracy for the classification rules associated with rare cases/small disjuncts. One such metric is the Laplace estimate. The standard version of this metric is defined as $(p+1)/(p+n+2)$, where $p$ positive and $n$ negative examples are covered by the classification rule. This estimate moves the accuracy estimate toward ½ but becomes less important as the number of examples increases.

A more sophisticated error-estimation metric for handling rare cases and small disjuncts was proposed by Quinlan [41]. This method improves the accuracy estimates of the small disjuncts by taking the class distribution (class priors) into account. The reasoning is that one would expect disjuncts that predict the majority class to have a lower error rate than those predicting the minority class. However, rather than using the entire training set to estimate the class priors, a more representative (local) set of examples is used. This set only uses training examples that are "close" to the small disjunct—that is, fail to satisfy at most one condition in the disjunct. Quinlan's experimental results demonstrate that when this estimate is used during the data mining process, classification performance improves, most notably for highly skewed class distributions.

Metrics that support cost-sensitive learning are the subject of much research. Because of the importance of cost-sensitive learning, this topic is covered separately in Section 3.8.

## 3.2 Non-Greedy Search Techniques

Section 2.3 described why greedy search techniques can be ineffective at dealing with rarity. In this section non-greedy search methods are described, some of which have specifically been developed to address rarity. The first such method involves genetic algorithms. Genetic algorithms are global search techniques that work with populations of candidate solutions rather than a single solution and employ stochastic operators to guide the search process [21]. These characteristics permit genetic algorithms to cope well with attribute interactions [19; 21] and avoid getting stuck in local maxima, which together make genetic algorithms suitable for dealing with rarity. This may explain why genetic algorithms are being increasingly used for data mining [19]. Several systems have relied to the power of genetic algorithms to handle rarity. Timeweaver [49] uses a genetic algorithm to predict very rare events while Carvalho and Freitas [7; 8] use a genetic algorithm to discover "small disjunct rules."

Decision tree learning algorithms almost always employ a greedy search. Brute [44] was designed to address the limitations of these greedy, hill-climbing search algorithms. Brute performs an exhaustive depth-bounded search for accurate conjunctive rules. The focus is to find accurate rules, even if they cover relatively few training examples. Brute performs quite well when compared to other algorithms, although the length of the generated rules needs to be limited. Brute is capable of locating "nuggets" of information that other algorithms may not be able to find.

Association-rule mining systems generally employ an exhaustive search algorithm [1] and are therefore, in theory, capable of finding rare associations. The problem, previously described in Section 2.3, is that these algorithms become intractable if the minimum level of support is set small enough to find rare associations. Thus, such algorithms are heuristically inadequate for finding rare associations. Section 3.7 describes an extension to these algorithms that makes it possible to find some meaningful rare associations.

## 3.3 Using a More Appropriate Inductive Bias

The bias of most data mining systems favors generality over specialization. While a general bias is good for common cases, it is not appropriate for rare cases and may even cause rare cases to be totally ignored. There have been several attempts to improve the performance of data mining systems with respect to rarity by choosing a more appropriate bias. The simplest approach involves modifying existing systems to eliminate some small disjuncts based on tests of statistical significance or using error estimation techniques. The hope is that these will remove only improperly learned disjuncts. Unfortunately, this approach was shown not only to degrade performance with respect to rarity, but also to degrade overall classification performance [23]. More sophisticated approaches have been developed and are described in this section. The impact of these strategies on rare cases cannot be measured directly, since the rare cases in the "true" concept are generally not known. To judge the efficacy of these strategies we therefore consider whether they improve the performance of the small disjuncts—based on the assumption that rare cases manifest themselves as small disjuncts.

Holte, Acker and Porter [23] changed the bias of an existing learner, CN2, to make the bias more specific. Rather than using CN2's maximum generality bias for all disjuncts, more specific biases were evaluated for the induced small disjuncts. The maximum specificity bias was shown to improve the performance of the small disjuncts but degrade the performance of the large disjuncts, yielding poorer overall performance. This occurred because the "emancipated" examples—those that would previously have been classified by small disjuncts—were then misclassified at an even higher rate by the large disjuncts. Going on the assumption that this change in bias was too extreme, a selective specificity bias was then used. This yielded further improvements, but these improvements were not sufficient to improve *overall* classification accuracy.

Ting [47] subsequently evaluated a very similar approach, which also used a maximum specificity bias. However, in this case steps were taken to ensure that this bias does not affect—and therefore cannot degrade—the performance of the large disjuncts (due to the emancipated examples). The basic approach was to first use C4.5 [42], a decision-tree learner, to determine if an example is covered by a small or large disjunct. If the example was covered by a large disjunct, then C4.5 was used to classify the example; otherwise an instance-based learner was used to classify the example. Instance-based learning was used because it is an extreme example of the maximum specificity bias. While the results of this study are encouraging and show that this hybrid approach can improve the accuracy of the small disjuncts, the results are not conclusive.

Another research study used a similar hybrid method. C4.5 is again used to identify examples as belonging to a small or large disjunct, but this time the training examples that fall into each small disjunct are fed into a genetic-algorithm based learner [7, 8]. This learner then generates classification rules to specifically cover the examples that fall into each individual small disjunct. Examples that fall into a large disjunct are classified using C4.5 whereas those that fall into a small disjunct are classified using the rules induced specifically for that disjunct. When classifier performance is compared the hybrid approach outperforms C4.5 in 9 cases, C4.5 outperforms the hybrid approach in 2 cases and the differences are not statistically significant in 11 cases.

One final study [3] advocates the use of instance-based learning for domains with many rare cases/small disjuncts, because of the highly specific bias associated with this learning method. The authors of this study were mainly interested in learning word pronunciations, which have "pockets of exceptions" (i.e., rare cases) that cause many small disjuncts to be formed during learning. Results are not provided to demonstrate that instance-based learning outperforms others learning methods in this situation. Instead the authors argue that instance-based learning methods should be used because they store all examples in memory, while other approaches ignore examples when they fall below some utility threshold (e.g., due to pruning). Returning to Figure 1, most learners will not learn any part of A3 because there is only one positive example, and hence any examples that are near that point will be misclassified. However, an instance-based learning algorithm will match some examples in region A3 to the positive example and thus will often assign the correct classification.

In summary, there have been several attempts to select an inductive bias that will perform better in the presence of small disjuncts. These methods have shown only mixed success, although this may be the result of researchers using overall classification accuracy to evaluate the approach, rather than focusing on the benefit to the small disjuncts. This approach to addressing rarity appears promising and worthy of further study.

## 3.4 Knowledge/Human Interaction

Knowledge can always improve the data mining process. However, it is especially useful for very difficult problems, such as when rare classes/cases are present. Knowledge can take many forms. Knowledge can provide better descriptions of the examples, by providing more sophisticated features. Feature engineering may be guided by an expert's domain knowledge and by knowledge of what interactions between variables may be most useful. Most experts will naturally tend to include features that are useful for predicting rare, but important, cases.

Data Mining is inherently an interactive process. This is especially true for unsupervised learning tasks, such as mining association rules. For example, for association rule mining a human may indicate which results are interesting and warrant further mining and which are uninteresting (e.g., the association between name and social security number) and do not warrant further exploration. This interaction is especially important when mining rare classes/cases, since the user may have domain knowledge that can aid in the search process. This viewpoint is supported by the following quote "Only in rare cases will users wish to see patterns with miniscule support. In those cases it is more likely that users will start the mining on a small filtered sample (which may be the result of a previous drill-down operation)." [32]

## 3.5 Learn only the Rare Class

As discussed throughout Section 2, if we try to learn a set of classification rules for all classes, the rare classes may be largely ignored. One solution to this problem is to only learn classification rules that predict the rare class. One data mining system that utilizes this recognition-based approach is Hippo [27]. Hippo uses a neural network and learns only from the positive (rare) examples, thus *recognizing* patterns amongst the positive examples, rather than differentiating between positive and negative examples. Support vector machines have also utilized this same approach to learn rare classes, with some success [43].

Systems that learn only the rare class may still train using examples belonging to all classes. Brute [44], Shrink [33] and Ripper [13] are three such data mining systems. Brute has been used to look for flaws in the Boeing manufacturing process [44]. Because the goal is to find failures (rare cases), Brute focuses only on the performance of the rules that predict failures. The advantage of Brute's approach is summarized nicely in [34], "by measuring performance only of the positive predicting rules Brute is not influenced by the invariably high accuracy of the negative examples that are not covered by the positive predicting rules." Shrink uses a similar approach to detect rare oil spills from satellite radar images [33]. Based on the assumption that there will be many more negative examples than positive examples, Shrink labels mixed regions (i.e., regions with positive and negative examples) with the positive class. The task then is to search for the "best" positive regions, which have the highest ratio of positive to negative examples. Ripper [13] is a rule induction system that utilizes a separate-and-conquer approach to iteratively build rules to cover previously uncovered training examples. Each rule is grown by adding conditions until no negative examples are covered. It normally generates rules for each class from the most rare class to the most common class. Given this architecture, it is quite straightforward to learn rules only for the minority class—a capability that Ripper provides.

Not all of the classification rules generated by these systems need be used. Most data mining systems produce an estimate of the quality of each rule, typically the estimated precision of the rule based on the training data. Classification rules can then be ordered based on this quality metric and the user can then choose only the best $m$ of $n$ rules, $m \leq n$. Varying the value of $m$ will generate a precision/recall curve and this curve can then be used to select a specific solution, based on requirements for the problem. When using this general approach, data mining systems need not determine, a priori, at what point to stop generating positive prediction rules.

## 3.6 Segmenting the Data

One way to deal with rarity is to reduce the degree of rarity by carefully segmenting the data. Segmenting the data effectively partitions the original data-mining problem into separate subproblems. Imagine that some target event is rare, only occurring .001% of the time. It may be that by segmenting the data into two regions, R1 and R2, this target event may occur 20% of the time in R1 and .0001% of the time in R2. One can then mine R1 without the problems associated with extremely rare classes. While it will be even more difficult to mine R2 for the target events, this may be acceptable, since R1 may contain the majority of these events.

As a real-world example, imagine that a telephone company is interested in identifying all dedicated dial-up lines used to connect a home computer to an Internet Service Provider. The telephone company will have the calling history for each phone line, which includes the time and duration of each call, the day of week of each call, the calling numbers dialed, etc. In addition, the telephone company will have some modest list of phone lines for which the purpose of the line is known (i.e., a training set). Since most lines are not dedicated PC data lines, this data-mining task involves rarity. One strategy is to partition the phone lines into three segments based on weekly minutes of usage: low usage, medium usage, and high usage. Since most dedicated PC lines will have a relatively high number of weekly minutes of usage, one would expect these lines to be much less rare in the high usage segment than in the other segments. Thus, one can probably

do a fairly good job of identifying the dedicated lines in the high usage segment—which will probably account for most dedicated data lines. Thus, the problem has been simplified. Segmentation can be viewed as a specific instance of how knowledge (Section 3.4) can be used to address rarity.

## 3.7 Accounting for Rare Items

Association rule mining can suffer from the rare item problem (described in Section 2.3), in which significant associations between rarely occurring items may be missed, because the minimum support value, *minsup*, must not be set low, in order to avoid a combinatorial explosion of associations. This problem can be solved by specifying multiple minimum levels of support to reflect the frequencies of the associated items in the distribution [37]. Specifically, the user can specify a different minimum support for each item. The minimum support for an association rule is then the lowest *minsup* value amongst the items in the rule. Association rule mining systems are tractable mainly because of the downward closure property of support: if a set of items satisfies *minsup* then so do all of its subsets. While this downward closure property does not hold with multiple minimum levels of support, the standard Apriori algorithm for association rule mining can be modified to satisfy the sorted closure property for multiple minimum levels of support [37]. The use of multiple minimum levels of support then becomes tractable. Empirical results indicate that the new algorithm is able to find meaningful associations involving rare items without producing a huge number of meaningless rules involving common items.

## 3.8 Cost-Sensitive Learning

In many data mining tasks (e.g., medical diagnosis), it is the rare classes/cases that are of primary interest. Metrics that do not take this into account generally do not perform well in these situations. One solution is to use cost-sensitive learning methods [38]. These methods can exploit the fact that the value of correctly identifying the positive (rare) class outweighs the value of correctly identifying the common class. For two-class problems this is done by associating a greater cost with false negatives than with false positives. This strategy is appropriate for most medical diagnosis tasks because a false positive typically leads to more comprehensive (i.e., expensive) testing procedures that will ultimately discover the error, whereas a false negative may cause a life-threatening condition to go undiagnosed, which could lead to death.

Assigning a greater cost to false negatives than to false positives will improve performance with respect to the positive (rare) class. If this misclassification cost ratio is 3:1, then a region that has ten negative examples and four positive examples will nonetheless be labeled with the positive class. Thus non-uniform costs can bias the classifier to perform well on the positive class—where in this case the bias is desirable. One problem with this approach is that specific cost information is rarely available. This is partially due to the fact that these costs often depend on multiple considerations that are not easily compared [10]. For example, in the medical diagnosis task the considerations involve the probability that an undiagnosed condition will lead to death, the "cost" of a false-positive on a patient's well being, etc. Thus, without specific cost information, it may be more practical to only predict the rare class (Section 3.5) and generate an ordered list of the best positive-predicting rules. Then one can decide where to place the threshold *after* data mining is complete.

Most modern data mining systems can handle cost-sensitivity directly, in which case cost information can be passed to the data-mining algorithm. In the past such systems often did not have this capability. In this case cost-sensitivity was obtained by altering the ratio of positive to negative examples in the training data, or, equivalently, by adjusting the probability thresholds used to assign class labels [16]. While these "indirect" methods for handling cost sensitivity do work, they have some undesirable consequences. This has caused some techniques for handling rarity to be misused. This will be discussed in detail in Section 4.

## 3.9 Sampling

One of the most common techniques for dealing with rarity is sampling. The basic idea is to eliminate or minimize rarity by altering the distribution of training examples. Typically the class distribution is altered to reduce the problems associated with rare classes, but the distribution of cases can also be altered to deal with rare cases [24; 48]. In most of this section we focus on the use of sampling to reduce class imbalance.

### *3.9.1 Basic Sampling Methods*

The basic sampling methods include under-sampling and over-sampling. Under-sampling eliminates majority-class examples while over-sampling, in its simplest form, duplicates minority-class examples. Both of these sampling techniques decrease the overall level of class imbalance, thereby making the rare class less rare.

These sampling methods have several drawbacks. Under-sampling discards potentially useful majority-class examples and thus can degrade classifier performance. Because over-sampling introduces additional training cases, it can increase the time necessary to build a classifier. Worse yet, because over-sampling often involves making exact copies of examples, it may lead to overfitting [11; 15]. As an extreme case, classification rules may be introduced to cover a single, replicated, example. More importantly, over-sampling introduces *no new data*—so it does not address the fundamental "lack of data" issue described in Section 2.2. This explains why some studies have shown simple over-sampling to be ineffective at improving recognition of the minority class [15, 36] and why under-sampling may be a better choice [15]. However, a study that used artificial domains came to the opposite conclusion [28]. The next section describes advanced sampling methods that can overcome some of the weaknesses described in this section.

### *3.9.2 Advanced Sampling Methods*

Advanced sampling methods may use intelligence when removing/adding examples or combine under-sampling and over-sampling techniques. One under-sampling strategy [35] only removes majority-class examples that are redundant with other examples or border regions with minority-class examples, figuring that they may be the result of noise. SMOTE [11], on the other hand, over-samples by introducing new, non-replicated minority-class examples. Minority-class examples are generated by adding examples from the line segments that join the $k$ minority-class nearest neighbors (SMOTE uses $k$=5). This causes additional generalization, as opposed to the specialization that may arise from exactly replicating examples. A mixture-of-experts approach [17] has been used to combine the results of many classifiers, each induced after over-sampling or under-sampling the data with different over/under-sampling rates. This approach recognizes the

fact that it is still unclear which sampling method performs best, what sampling rate should be used—and that the proper choice is probably domain specific. Results indicate that the mixture-of-experts approach performs well, generally outperforming another method (AdaBoost) with respect to precision and recall on text classification problems, and doing especially well at covering the rare, positive, examples.

A different approach involves identifying a good class distribution for data mining—one that will lead to good results—and then generating samples with that distribution. Chan and Stolfo [9] run a set of preliminary experiments to identify a good class distribution and then sample in such a way as to generate multiple training sets with the desired class distribution. Each training set typically includes all minority-class examples and a subset of the majority-class examples; however, each majority-class example is guaranteed to occur in at least one training set, so no data is wasted. The learning algorithm is applied to each training set and meta-learning is used to form a composite learner from the resulting classifiers. This approach can be used with any learning method and Chan and Stolfo evaluate it using four different learning algorithms. The same basic approach for partitioning the data and learning multiple classifiers has been used with support vector machines. The resulting SVM ensemble [53] was shown to outperform both under-sampling and over-sampling. While these ensemble approaches are effective for dealing with rare classes, they assume that a good class distribution is known. This can be estimated using some preliminary runs, but this increases the time required to learn.

Another method that uses this general approach employs a progressive-sampling algorithm to build larger and larger training sets, where the ratio of positive to negative examples added in each iteration is chosen based on the performance of the various class distributions evaluated in the previous iteration [52]. Experimental results indicate that the class distribution will generally converge to a good, nearly optimal, value for learning. This approach assumes that not all possible training examples are immediately available, but rather that there is a cost associated with procuring each example. This method will perform well when such costs exist since it takes procurement costs into account. This can be contrasted with most other sampling schemes, where it is assumed that there is some large collection of examples, ready, without cost, for learning.

The methods described thus far are designed to reduce class imbalance. While existing research [52] indicates that reducing class imbalance will tend to also reduce within-class imbalances (i.e., make rare cases less rare), it is natural to ask whether sampling can be used in a more direct manner to reduce within-class imbalances—and if this is beneficial. This question has been studied using artificial domains [24]. The results indicate that it is not sufficient to eliminate between-class imbalances (rare classes) in order to learn complex concepts that contain rare cases. Only when the within-class imbalances are also eliminated can the concept be learned well. This suggests that sampling should be used to improve the performance associated with rare cases. Unfortunately, there are a few problems with implementing the strategy for real-world domains. First, one does not know, a priori, anything about the rare case(s). However, assuming that rare cases cause small disjuncts, one could sample based on disjunct size, with the goal of equalizing the sizes of the disjuncts in the induced classifier. An alternative method would be to use clustering to identify possible rare cases and then sample to equalize these cluster sizes. A second issue is that there generally will not be additional examples available for the minority class—most sampling strategies use all minority-class examples. However, this could be addressed by generating new examples using a technique such as SMOTE [11].

## 3.10 Other Methods

This section describes a variety of other methods for dealing with rarity. In some cases one could argue that these methods fit into one of the previously established categories (3.1-3.9), but we believe that these methods warrant their own categories.

### *3.10.1 Boosting*

Boosting algorithms, such as AdaBoost [46], are iterative algorithms that place different weights on the training distribution each iteration. After each iteration boosting increases the weights associated with the incorrectly classified examples and decreases the weights associated with the correctly classified examples. This forces the learner to focus more on the incorrectly classified examples in the next iteration. Because rare classes/cases are more error-prone than common classes/cases [51; 52], it is reasonable to believe that boosting may improve their classification performance because, overall, it will increase the weights of the examples associated with these rare cases/classes. Note that because boosting effectively alters the distribution of the training data, one could consider it a type of advanced sampling technique.

AdaBoost's weight-update rule has been made cost-sensitive, so that examples belonging to rare class(es) that are misclassified are assigned higher weights than those belonging to common class(es). The resulting system, Adacost [18], has been empirically shown to produce lower cumulative misclassification costs than AdaBoost and thus, like other cost-sensitive learning methods, can be used to address the problem with rare classes.

Boosting algorithms have been developed to directly address the problem with rare classes. In each iteration of boosting, RareBoost [31] scales false-positive examples in proportion to how well they are distinguished from true-positive examples and scales false-positive examples in proportion to how well they are distinguished from true-negative examples. Because AdaCost, unlike RareBoost, does not stratify these measures separately, it is believed that AdaCost may sometimes over-emphasize recall, thus leading to poorer precision. A second algorithm that uses boosting to address the problems with rare classes is SMOTEBoost [12]. This algorithm recognizes that boosting may suffer from the same problems as over-sampling (e.g., overfitting), since boosting will tend to weight examples belonging to the rare classes more than those belonging to the common classes—effectively duplicating some of the examples belonging to the rare classes. Instead of changing the distribution of training data by updating the weights associated with each example, SMOTEBoost alters the distribution by adding new minority-class examples using the SMOTE algorithm [11]. Empirical results indicate that this approach allows SMOTEBoost to achieve higher F-values than Adacost [12].

Boosting has been analyzed from a theoretical perspective to determine whether it is guaranteed to improve the classification performance of any base learner for the rare class [30]. This analysis shows that no such guarantee exists. Rather, the performance improvement from boosting is shown to be strongly tied to the choice of the base learning algorithm and boosting will perform poorly if the base learner always achieves poor precision or precision. This analysis shows, however, that if the base learner can

effectively trade-off precision and recall, then boosting can significantly improve the performance of the base learner. A practical result is that one can often determine which base learner to use for boosting based on the performance of the learner without boosting.

### *3.10.2 Two Phase Rule Induction*

Induction techniques that deal with rare classes must try to maximize precision and recall. Most induction techniques try to optimize these two competing measures simultaneously. According to one view this is simply too difficult to accomplish for complex problems. PNrule [29] uses two-phase rule induction to focus on each measure separately. In the first phase, if high precision rules cannot be found then lower precision rules are accepted, as long as they have relatively high recall. So, the first phase focuses on recall. In the second phase precision is optimized. This is accomplished by learning to identify false positives within the rules from phase 1. Returning to the needle and haystack analogy, this approach identifies regions likely to contain needles in the first phase and then learns to discard the strands of hay within these regions in the second phase. Two phase rule induction deals with rare cases by handling two of the problems described in Section 2. The presence of the second phase permits the first phase to be sensitive to the problem of small disjuncts (Section 2.2) while the second phase allows the false positives to be grouped together, addressing the problem of data fragmentation (Section 2.4).[2]
Experimental results [29] indicate that PNrule performs competitively with other disjunctive learners on easy problems, but that as more complex concepts are introduced (with many rare cases), the performance of the other learners degrades, while PNrule maintains high performance. These results also show that PNrule outperforms AdaBoost, especially for rare classes. The reason provided for this is that boosting lacks the ability of PNrule to collect the relevant false positives and to explicitly learn rules to exclude them.

### *3.10.3 Place Rare Cases into Separate Classes*

Rare cases make data mining difficult, especially if there are a large number of rare cases. The underlying issue is that different rare cases may have little in common between them, making it difficult for one learner to assign the same class value to all of them. One possible solution is to reformulate the original problem so that the rare cases are viewed as separate classes. Japkowicz [25] implemented this approach by 1) separating each class into subclasses using clustering, 2) relabeling the training examples based on the subclasses (i.e., clusters) and then 3) re-learning on the revised training set. Because multiple clustering experiments were used in step 1, step 2 involves learning multiple models, which are subsequently combined using voting. The performance results from this study are promising, but not conclusive, and additional research is needed.

## 4. DISCUSSION

This section discusses various issues associated with addressing rarity. Section 4.1 provides a mapping from each problem associated with rarity, listed in Section 2, to method(s) from Section 3 for addressing that problem. In Section 4.2 we compare and contrast the methods for dealing with rarity and point out drawbacks with some of these methods. Finally, in Section 4.3 we discuss the relationship between rare classes and rare cases. We show that these two forms of rarity are related and have similar problems and "solutions". Throughout this section we refer to the problems in Section 2 and methods for addressing these problems in Section 3 using the corresponding section number (e.g., 2.2 or 3.5).

## 4.1 Mapping of Problems to Solutions

Table 1 summarizes the mapping of problems with rarity to the methods for addressing these problems. Note that for each problem multiple solutions are available. In these cases we list the best (most direct, most useful) solutions first and italicize those solutions that only indirectly address the underlying problem.

| **Data Mining Problem** | **Method to Address the Problem** |
|---|---|
| 2.1: Improper evaluation metrics<br>- for evaluating final result<br>- to guide data mining | 3.1: More appropriate eval. metrics<br>3.8: Cost-sensitive learning |
| 2.2: Absolute Rarity | 3.9: Over-Sampling<br>Remainder identical to cell below |
| 2.3: Relative Rarity<br>"needles in a haystack" | 3.5: Learn only the rare class<br>3.6: Segmenting the data<br>3.9: Sampling (over- and under-)<br>3.2: Non-greedy search techniques<br>3.10.2: Two-phase rule induction<br>3.7: Accounting for rare items<br>3.8: Cost-sensitive learning<br>3.4: Knowledge/human interaction<br>3.10.3: Rare cases into separate classes<br>3.1: More appropriate eval. metrics<br>3.3: More appropriate inductive bias<br>3.10.1: *Boosting* |
| 2.4: Data Fragmentation<br>"rare classes/cases split apart" | 3.2: Non-greedy search techniques<br>3.10.2 Two-phase rule induction<br>3.5: Learn only the rare class<br>3.10.3: Rare cases into separate classes<br>3.9: *Sampling* |
| 2.5: Inappropriate Bias | 3.3: More appropriate inductive bias<br>3.1: *Appropriate evaluation metrics*<br>3.8: *Cost-sensitive learning* |
| 2.6: Noise | 3.9.2 Advanced Sampling<br>3.3: More appropriate inductive bias |

**Table 1: A mapping of data mining problems associated with rarity to methods for addressing these problems.**

Most of Table 1 requires no explanation, although the precise ordering of methods within each cell is certainly open to debate. We choose to focus our discussion on Problem 2.2 and Problem 2.3. First consider Problem 2.3, the problem of relative rarity. Table 1 lists twelve methods for dealing with this problem—which includes all of the methods listed in Section 3. For this problem most methods are quite good at dealing with the underlying problem and hence the ordering of the methods is somewhat arbitrary. Those methods that are not specifically geared toward solving this problem tend to be listed later (e.g., cost-sensitive learning). Because Problem 2.2, the problem with absolute rarity, shares many characteristics with the problem of relative rarity, they share the same solutions. In particular, note that when there is a problem with absolute rarity there will also be a problem with relative rarity (assuming the data set is not very small). For Problem 2.2 we highlight the over-sampling method by listing it first,

[2] Whereas the data fragmentation problem usually relates to keeping positive examples together, in this case the issue is keeping the false positives together, to avoid the problem with splintered false positives [6].

since this is the only method that directly addresses the problem with absolute rarity (i.e., it introduces additional rare examples). Specifically, over-sampling can try to address the underlying problem by duplicating existing rare examples (not necessarily a good idea), synthetically generating new rare examples, or, ideally, procuring new rare examples (e.g., buying more data, surveying additional people likely to belong to a rare class, etc.). For the "relative" data problem, sampling is instead used to rebalance the distribution, to reduce between-class or within-class imbalances. Thus, under-sampling can prevent the common cases/classes from hiding the rare cases/classes. As we shall see in the next section, rebalancing the data has its drawbacks.

## 4.2 Comparative Evaluation of the Solutions

The previous section summarizes the mapping of problems associated with rarity to methods for addressing these problems. The critical question then becomes, which methods are best? Unfortunately, there have been no comprehensive empirical studies that evaluate all, or even most, of the methods listed in Table 1. Each research study typically compares its method for handling rarity to a base learner that has no special modifications for handling rarity, or to one or two very similar methods. Sampling techniques have generated the most research in this area and there are a few studies [15; 28] that compare sampling methods. Unfortunately, even in this case the conclusions are not consistent. The best we can do is to compare the ways some of the methods operate, describe drawbacks with these methods and discuss some misconceptions that result in some of these methods being misapplied. Because sampling is the most-used methods for dealing with rarity, much of this discussion revolves around sampling.

### *4.2.1 Equivalence of Sampling and Other Methods*

There exists the notion that various methods for dealing with rarity and class imbalance are *equivalent*. In particular, Breiman [5] establishes the connection between the distribution of training-set examples, the prior probability of each class, the costs of mistakes on each class and the placement of the decision threshold. Thus, for example, one can make false negatives twice as costly as false positives by using cost-sensitive learning *or* by increasing the ratio of positive to negative examples in the training set by a factor of two. Each of the quantities listed by Breiman can be altered to help deal with rarity—and many of the techniques (e.g., sampling) in Section 3 rely on this. As it turns out, in practice the above-mentioned equivalence does *not* hold, even though it may hold theoretically. One practical consideration is that the "precise relationship among these things is complex and task- and method-specific" [11]. A second issue is that one does not have complete freedom to vary these quantities. This especially causes problems with sampling and has caused many practitioners and researchers to misuse sampling, or, at the very least, not properly identify how sampling addresses the problems associated with class imbalance.

As a concrete example, suppose a training set has 10,000 examples and a class distribution of 100:1, so that there are only 100 positive examples. One way to improve the identification of the rare class is to impose a greater cost for false negatives than for false positives. Suppose we choose a ratio of 100:1, even though we may not have actual cost information. This is theoretically equivalent to using a training distribution with a balanced (1:1) class distribution. However, can one generate such a distribution in practice? For the equivalence to truly hold, one must sample from the original distribution in order to balance the classes. However, this is generally not possible, because, except for artificial domains, we do not know the true distribution and additional, unused, training examples are typically not available. In practice what would be done is to discard majority-class examples (under-sample), duplicate minority-class examples (over-sample), or use some combination of both in order to achieve the desired 1:1 class ratio. As discussed in Section 3.9, these sampling methods introduce problems—they either discard potentially useful data or duplicate examples, which may lead to overfitting. Even generating new, synthetic, minority-class examples violates the equivalence, since these examples will, at best, only be a better *approximation* of the true distribution.

### *4.2.2 Unintended Consequences of Sampling*

Another significant concern with sampling is that the impact that sampling has on rare classes is often not fully understood, or considered. This has definite consequences for data mining. Increasing the proportion of examples belonging to the rare class will have two effects. First, it will help address Problems 2.2 and 2.3, the problems with absolute and relative rarity. However, if no other action is taken, it will also have a second effect—it will impose non-uniform error costs, causing the learner to be biased in favor of predicting the rare class. In some situations (described in the next section) this second effect is not desired. The intent in these situations is to improve performance with respect to the rare class by having *more data* available for that class, not by biasing the data mining algorithm toward that class. Thus, this bias should be removed. This can be done using the equivalences noted in Section 4.2.1 to account for the differences between the training distribution and the underlying distribution [16; 52]. For example, the bias can be removed by adjusting the decision thresholds.

One study [52] on class distribution shows the advantages of adjusting for this bias. This study shows that by altering the class distribution of the training data so that it deviates from the natural, underlying, distribution, improved classifier performance is possible. However, classifier performance was improved more when the bias just described was removed by adjusting the decision thresholds within the classifier. Other research studies that investigate the use of sampling to handle rare cases and class imbalance do not remove this bias—and worse yet, do not even discuss the implications of this decision. This issue must be considered much more carefully in future studies.

### *4.2.3 Some Recommendations and Guidelines*

Many of the methods for addressing rarity are still in the research stage or are not widely implemented (e.g., two-phase rule-induction) or are widely available but the advantages for dealing with rarity are not conclusively proven (e.g., boosting). Still others are domain specific and cannot always be utilized (e.g., segmentation). In this section we focus our attention on two of the most widely used techniques for dealing with rarity—sampling and cost-sensitive learning.

Given the previous discussion, what recommendations can we make concerning the use of sampling and cost-sensitive learning? First, one should use all available training data, regardless of the resulting distribution of classes (or cases), if feasible. Thus, no data is discarded and no information is lost. If cost information is known, then a cost-sensitive learning algorithm should be used. If specific cost information is not known, but the performance of the rare class is unsatisfactory, then several options should be investigated. One option is to use cost-sensitive learning and vary the

cost ratios, in order to improve the performance of the rare classes at the expense of the common classes. By using this method instead of sampling, "all of the data can be used (to produce the tree) thus throwing away no information, and learning speed is not degraded due to duplicate instances" [14, page 239]. This recommendation is also supported by research that shows that such methods outperform over-sampling and under-sampling [28]. Other alternatives from Table 1 are also available, most notably learning only rules that predict the rare class.

If the amount of training data must be limited because of tractability issues or because training data is costly to acquire, then sampling must be used. If one class is very rare, the sampling method should include all examples belonging to that class. Cost information, if available, can be used to help select the relative sampling rate of the classes. If sampling is being used to reduce the size of the training set—and not to impose non-uniform misclassification costs—then any bias imposed by the sampling scheme should be eliminated, as described in Section 4.2.2.

Ideally, the relative sampling rate between classes should be chosen so that the generated distribution provides the best results. Unfortunately, as shown by Weiss and Provost [52], there is no general answer as to which class distribution will perform best, and the answer is surely method and domain dependent. Nonetheless, the empirical results from this study do provide some recommendations about which class distributions often perform best. A better approach is to try to determine, for a given domain and learning algorithm, which class distribution will perform best. A progressive sampling scheme has been shown to be effective at finding a near-optimal class distribution for learning [52], although this does require some additional computation. Additional research is needed into ways to efficiently identify a good class distribution for learning.

It is worth pointing out that state-of-the-art commercial data mining systems (e.g., SAS Enterprise Miner) currently provide many of the features necessary to deal with rarity. For example, they allow one to sample so that the training data deviates from the underlying distribution and can be configured to automatically remove the bias imposed by this sampling scheme. Such systems now can also handle complex misclassification functions, as required to perform cost-sensitive learning. Some of these systems can also dynamically alter the decision thresholds of induced classifiers, in order to evaluate how these changes affect overall performance. Thus one can see how different misclassification costs would affect learning without re-learning. Finally, modern data mining systems also permit one to evaluate the end result of data mining using a wide variety of metrics (e.g., ROC curves, lift curves, etc.), which makes it easier to assess the performance of rare classes.

## 4.3 Rarity: Rare Cases versus Rare Classes

Both rare cases and rare classes are problematic for data mining. This section begins by describing the connection between rare cases and rare classes and then shows that both forms of rarity cause similar problems for data mining—and that these problems can often be addressed using the same methods.

Earlier in this article it was stated that rare classes tend to have a higher proportion of rare cases than common cases. This is supported by an empirical study that shows that, over twenty-six two-class learning problems, minority-class rules consistently are made up of smaller disjuncts than the majority-class rules [52]. Specifically, these results show that for the eighteen data sets with a class distribution of at least 2:1, only in two cases does the majority class have a smaller average disjunct size than the minority class. Thus, between-class and within-class imbalances are linked. Therefore, we expect that when class imbalance (i.e., between class imbalance) is reduced, then within-class imbalance will also be reduced.

We now argue that rare classes and rare cases are not just linked, but are *very similar* phenomena. That is, both forms of rarity affect data mining in a similar way and hence the problems associated with both forms of rarity can be addressed using the same methods. Consider the problems associated with rarity. Of the six problems listed in Section 2 and summarized in Table 1, all apply equally to rare cases and rare classes. For example, data fragmentation can be a problem for rare classes because the examples belonging to the rare classes can become separated; similarly, it can be a problem for rare cases because the examples belonging to each rare case can become separated. Thus, both rare classes and rare cases share the same fundamental problems. This equivalence should not be surprising since a rare case can be viewed as a rare class (Method 3.10.3, which maps rare cases into separate classes, relies on this).

Next we cover the methods for addressing rarity. Methods 3.1-3.4 apply equally to both rare classes and rare cases. Method 3.5, which involves learning only the rare class, by definition only applies to rare classes. However, if we could identify rare cases, then one could generalize it to include rare cases (i.e., only predict one rare case at a time). This is in fact what Method 3.10.3 does (it uses clustering to split each rare case into a rare class and then learns each class separately). Thus we consider Methods 3.5 and 3.10.3 to be specific instances of the same general method—a method that handles both rare classes and rare cases.

Method 3.6, which involves segmenting the data, can also apply to rare classes and rare cases, although, as in the last case, it *may* be harder to segment rare cases than rare classes. Method 3.7 deals with associations between rare items, and therefore applies only to rare cases (associations are considered to be cases, although one could argue they are a distinct form of rarity). Method 3.8, cost sensitive learning, applies most directly to rare classes, since misclassification costs are normally assigned based on the actual and predicted class value. Costs can be assigned based on characteristics of each example, but since rare cases are not easily identified a priori, this method does not seem applicable, in practice, to rare cases. Since Method 3.9, sampling, has been used to reduce within-class imbalances as well as between-class imbalances [24], it can be applied to both rare cases and rare classes. Both boosting (3.10.1) and two-phase induction (3.10.2) can be used to address the problems with rare classes and rare cases. As was shown earlier, Method 3.10.3 is a counterpart of Method 3.5, and the general technique shared by both can be applied to both forms of rarity.

The discussion in this section therefore shows that rare classes and rare cases suffer from similar problems and that most methods for addressing rarity apply to both rare cases and rare classes. A few methods apply mainly to rare classes, but could be applied to rare cases if the rare cases could be more easily identified. While this is generally not possible for real-world domains, as described earlier unsupervised learning methods such as clustering can help (alternatively one could rely on small disjuncts to help identify rare cases). This section therefore shows that rare cases and rare classes can be viewed using a common framework and that methods developed to address a problem with one of these will usually

benefit the other. This common framework is especially promising given that most recent research on rarity has focused on between-class imbalances, while within-class imbalances may actually be more problematic [24]. Thus, we may be able to leverage methods for dealing with between-class imbalances to deal with within-class imbalances.

## 5. CONCLUSION

This article investigates the issues related to mining in the presence of rare classes and rare cases. The problems that can arise from these two forms of rarity are categorized and described in detail. Methods for addressing these problems are also described and a mapping from "problems" to "solutions" is provided. In most instances the descriptions of the problems and solution methods include descriptions of relevant research and data mining programs; thus, this article also provides a good survey of the literature on data mining with rare classes and rare cases. This article also demonstrates that rare classes and rare cases both suffer from the same set of problems and generally can benefit from the same solutions. Thus both forms of rarity can be viewed as being instances of the same basic phenomenon. This realization should certainly impact future research.

## 6. ACKNOWLEDGEMENTS

The content of this article was influenced by previous research on the impact of small disjuncts and class distribution on data mining. The author would therefore like to thank Haym Hirsh and Foster Provost for their collaboration on this earlier research. The author would also like to acknowledge Nathalie Japkowicz for work that demonstrates that rare cases, as well as rare classes, should be covered in any study on rarity in data mining.

## 7. REFERENCES

[1] R. Agarwal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207-217, 1993.

[2] K. Ali, and M. Pazzani HYDRA-MM: learning multiple descriptions to improve classification accuracy. *International Journal of Artificial Intelligence Tools*, 4, 1995.

[3] A. van den Bosch, T. Weijters, H. J. van den Herik, and W. Daelemans. When small disjuncts abound, try lazy learning: A case study. In *Proceedings of the Seventh Belgian-Dutch Conference on Machine Learning*, pages 109-118, 1997.

[4] A. Bradley. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7): 1145-1159, 1997.

[5] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and regression trees.* Chapman and Hall/CRC Press, Boca Raton, Fl, 1984.

[6] C. Cardie. Improving minority class prediction using case-specific feature weights. In *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 57-65, Morgan Kaufmann, 1997.

[7] D. R. Carvalho, and A. A. Freitas. A genetic algorithm for discovering small-disjunct rules in data mining. *Applied Soft Computing* 2(2):75-88, 2002.

[8] D. R. Carvalho, and A A. Freitas. New results for a hybrid decision tree/genetic algorithm for data mining. In *Proceedings of the Fourth International Conference on Recent Advances in Soft Computing*, pages 260-265, 2002.

[9] P. K. Chan, and S. J. Stolfo. Toward scalable learning with non-uniform class and cost distributions: a case study in credit card fraud detection. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pages 164-168, 2001.

[10] N. V. Chawla. C4.5 and imbalanced data sets: investigating the effect of sampling method, probabilistic estimate, and decision tree structure. In *Workshop on Learning from Imbalanced Datasets II*, International Conference on Machine Learning, 2003.

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

[12] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. Bowyer. SMOTEBoost: Improving prediction of the minority class in boosting. In *Proceedings of Principles of Knowledge Discovery in Databases*, 2003.

[13] W. W. Cohen. Fast effective rule induction. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 115-123, 1995.

[14] C. Drummond, and R. C. Holte. Exploiting the cost (in)sensitivity of decision tree splitting criteria. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 239-246, 2000.

[15] C. Drummond and R. C, Holte. C4.5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In *Workshop on Learning from Imbalanced Data Sets II*, International Conference on Machine Learning, 2003.

[16] C. Elkan. The foundations of cost-sensitive learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 239-246, 2001.

[17] A. Estabrooks, and N. Japkowicz. A mixture-of-experts framework for learning from unbalanced data sets. In *Proceedings of the 2001 Intelligent Data Analysis Conference*, pages 34-43, 2001.

[18] W. Fan, S. J. Stolfo, J. Zhang, and P. K. Chan. AdaCost: misclassification cost-sensitive boosting. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 99-105, 1999.

[19] A. A. Freitas. Evolutionary computation. *Handbook of Data Mining and Knowledge Discovery*, Oxford University Press, pages 698-706, 2002.

[20] J. H. Friedman, R. Kohavi, and Y. Yun. Lazy decision trees. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 717-724, 1996.

[21] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989.

[22] J. W. Grzymala-Busse, Z. Zheng, L. K. Goodwin, and W. J. Grzymala-Busse. An approach to imbalanced data sets based on changing rule strength. In *Learning from Imbalanced Data Sets: Papers from the AAAI Workshop*, pages 69-74, AAAI Press Technical Report WS-00-05, 2000.

[23] R. C. Holte, L. E. Acker, and B. W. Porter. Concept learning and the problem of small disjuncts. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 813-818, 1989.

[24] N. Japkowicz. Concept learning in the presence of between-class and within-class imbalances. In *Proceedings of the Fourteenth Conference of the Canadian Society for Computational Studies of Intelligence*, pages 67-77, Springer-Verlag, 2001.

[25] N. Japkowicz. Supervised learning with unsupervised output separation. In *International Conference on Artificial Intelligence and Soft Computing*, pages 321-325, 2002.

[26] N. Japkowicz. Class imbalances: are we focusing on the right issue? In *International Conference on Machine Learning Workshop on Learning from Imbalanced Data Sets II*, 2003.

[27] N. Japkowicz, C. Myers, and M. A. Gluck. A novelty detection approach to classification. In *Proceedings of the Fourteenth Joint Conference on Artificial Intelligence*, pages 518-523, 1995.

[28] N. Japkowicz, and S. Stephen. The class imbalance problem: a systematic study. *Intelligent Data Analysis*, 6(5):429-450, 2002.

[29] M. V. Joshi, R. C. Agarwal, and V. Kumar. Mining needles in a haystack: classifying rare classes via two-phase rule induction In *SIGMOD '01 Conference on Management of Data*, pages 91-102, 2001.

[30] M. V. Joshi, R. C. Agarwal, and V. Kumar. Predicting rare classes: can boosting make any weak learner strong? In *Proceedings of the Eighth International Conference on Knowledge Discovery and Data Mining*, pages 297-306, 2002.

[31] M. V. Joshi, V. Kumar, and R. C. Agarwal. Evaluating boosting algorithms to classify rare cases: comparison and improvements. In *First IEEE International Conference on Data Mining*, pages 257-264, November 2001.

[32] R. Kohavi. Data mining with MineSet: what worked, what did not, and what might. In *Workshop on Commercial Success of Data Mining*, Fourth International Conference on Knowledge Discovery and Data Mining, 1998.

[33] M. Kubat, R. Holte, and S. Matwin. Learning when negative examples abound. *Machine Learning: ECML-97, Lecture Notes in Artificial Intelligence 1224*, pages 146-153, Springer, 1997.

[34] M. Kubat, R. C. Holte, and S. Matwin. Machine learning for the detection of oil spills in satellite radar images. *Machine Learning*, 30(2):195-215, 1998.

[35] M. Kubat, and S. Matwin. Addressing the curse of imbalanced training sets: one-sided selection. In *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 179-186, Morgan Kaufmann, 1997.

[36] C. Ling, and C. Li. Data mining for direct marketing problems and solutions. *In Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pages 73-79, 1998.

[37] B. Liu, W. Hsu, and Y. Ma. Mining association rules with multiple minimum supports. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 337-341, 1999.

[38] M. Pazzani, C. Merz, P. Murphy, K. Ali, T. Hume, and C. Brunk. Reducing misclassification costs. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 217-225, 1994.

[39] F. Provost, and P. Domingos. Tree Induction for probability-based rankings. *Machine Learning*, 52(3).

[40] F. Provost, and T. Fawcett. Robust classification for imprecise environments. *Machine Learning*, 42: 203-231, 2001.

[41] J. R. Quinlan. Improved estimates for the accuracy of small disjuncts. *Machine Learning* 6:93-98, 1991.

[42] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1993.

[43] B. Raskutti, and A. Kowalczyk. Extreme re-balancing for SVMs: a case study. In *Workshop on Learning from Imbalanced Data Sets II*, International Conference on Machine Learning, 2003.

[44] P. Riddle, R. Segal and O. Etzioni. Representation design and brute-force induction in a Boeing manufacturing design. *Applied Artificial Intelligence*, 8:125-147, 1994.

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

[46] R. E. Schapire. A brief introduction to boosting. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 1401-1406, 1999.

[47] K. M. Ting. The problem of small disjuncts: its remedy in decision trees. In *Proceeding of the Tenth Canadian Conference on Artificial Intelligence*, pages 91-97, 1994.

[48] G. M. Weiss. Learning with rare cases and small disjuncts. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 558-565, Morgan Kaufmann, 1995.

[49] G. M. Weiss. Timeweaver: a genetic algorithm for identifying predictive patterns in sequences of events. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 718-725, Morgan Kaufmann, 1999.

[50] G. M. Weiss, and H. Hirsh. Learning to predict rare events in event sequences. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pages 359-363, 1998.

[51] G. M. Weiss, and H. Hirsh. A quantitative study of small disjuncts. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence*, pages 665-670, AAAI Press, 2000.

[52] G. M. Weiss, and F. Provost. Learning when training data are costly: the effect of class distribution on tree induction. *Journal of Artificial Intelligence Research*, 19:315-354, 2003.

[53] R. Yan, Y. Liu, R. Jin, and A. Hauptmann. On predicting rare classes with SVM ensembles in scene classification. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2003.

[54] B. Zadrozny, and C. Elkan. Learning and making decisions when costs and probabilities are both unknown. In *Proceedings of the Seventh International Conference on Knowledge Discovery and Data Mining*, pages 204-213, 2001.