Pattern Recognition Letters 29 (2008) 2156–2164

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Subgroup discover in large size data sets preprocessed using stratified instance selection for increasing the presence of minority classes ☆

José-Ramón Cano [a], Salvador García [b,*], Francisco Herrera [b]

[a] Department of Computer Science, University of Jaén, 23700 Linares, Jaén, Spain
[b] Department of Computer Science and Artificial Intelligence, University of Granada, 18071 Granada, Spain

ARTICLE INFO

Article history:
Received 18 September 2007
Received in revised form 25 April 2008
Available online 9 August 2008

Communicated by W. Pedrycz

Keywords:
Large size data sets
Scaling up
Subgroup discovery
Instance selection
Stratification
Minority classes

ABSTRACT

The subgroup discovery is defined as: "given a population of individuals and a property of those individuals, we are interested in finding a population of subgroups as large as possible and in having the most unusual statistical characteristic with respect to the property of interest".

The subgroup discovery algorithms have to face the scaling up problem which appears in the evaluation of large size data sets. In this paper we are interested in the extraction of subgroups from large size data sets. To avoid the scaling up problem, we propose the combination of stratification and instance selection algorithms for scaling down the data set before the subgroup discovery task. In addition, two new stratification models are proposed to increase the presence of minority classes in data sets, which affects to the subgroup discovery process on them. The results show that the subgroup discovery extraction can be executed on large data sets preprocessed independently of the presence of minority classes, which could not be executed in other way.

1. Introduction

The issue of scalability and the effect of increasing the size of data sets are always present in data mining (Domingo et al., 2002). The scaling up problem, due to large size data sets (Khosravi and Kabir, 2007), produces situations where the data mining algorithm cannot be executed. The evaluation necessities to apply the heuristic are expensive computationally and this cost is directly proportional to the size of the data set.

Subgroup discovery as data mining task, is situated at the intersection of predictive and descriptive induction. In subgroup discovery, the rules or subgroups are discovered using heuristics which try to find the best subgroups in terms of rule coverage and distributional unusualness (Lavrač et al., 2004). Subgroup discovery aims at discovering individual rules of interest, which must be represented in explicit symbolic form and which must be relatively simple in order to be recognized as actionable by potential users.

In this paper we are interested in the application of the subgroup discovery task to large data sets analyzing the scaling up problem. Subgroup discovery algorithms are difficult to be evaluated in this kind of data sets due to the fact that their heuristics are expensive computationally, and the costs are directly proportional to the size of the data set.

A possible way to face the scaling up problem that a large size data set produces consists of scaling down the initial data set. The scaling down can be applied by means of a pre-processing stage preceding the subgroup discovery algorithm. The pre-processing suggested in this paper consists in the application of data reduction techniques using instance selection algorithms (Wilson and Martinez, 2000; Liu and Motoda, 2001). The instance selection algorithms choose representative instance subsets following a determined strategy. Those subsets composed by representative instances are used as input to extract models from them (Riquelme et al., 2003). Due to the large size of the data set, it is necessary the stratified execution of the instance selection algorithms (Cano et al., 2005).

The aim of this paper is to propose the combination of the stratified instance selection and subgroup discovery algorithm to face the scaling up problem which affects to subgroup discovery task in large size data sets.

The imbalance data distribution is a problem tackled by several approaches in the specialized literature (Chawla et al., 2004; Zhang and Mani, 2003; Yen and Lee, 2006). This problem consists of the existence of minority classes, those which have few examples with respect to other classes. One of the techniques proposed in the class imbalance problem is to use a data reduction stage (Batista

☆ This work was supported by Projects TIN2005-08386-C05-01 and TIN2005-08386-C05-03.

* Corresponding author. Tel.: +34 958 240598; fax: +34 958 243317.
E-mail addresses: jrcano@ujaen.es (J.-R. Cano), salvagl@decsai.ugr.es (S. García), herrera@decsai.ugr.es (F. Herrera).

0167-8655/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2008.08.001

et al., 2004) (called under-sampling and similar to instance selection) for reducing the bias of the classifiers towards the majority class. In this way, the learning of the model could be oriented to reduce the misclassification of the minority class.

Minority classes also can be presented in subgroup discovery. Their presence in data sets produces that the instance selection stage, in the classic stratified instance selection proposed by Cano et al. (2005), reduces the presence of these minority classes in the subset selected used as input for the subgroup discovery process. In this case, the discovery of subgroups related with minority classes is hard due to the lost of instances which belong to that classes in the instance selection stage. To address this situation, we study two novel stratification proposals which increase the presence of those minority classes. These new proposals make available the subgroup discovery process in large size data sets independently of the presence of minority classes.

In order to do that, the paper is set out as follows. Section 2 is dedicated to the subgroup discovery task, the algorithm considered to develop that task and to present two quality measures of the subgroups discovered. In Section 3, we describe the KDD Cup'99 data set and its characteristics as large size data set used in our study. Section 4 is devoted to analyzing the scaling up problem which appears in the subgroup discovery algorithm when large data sets are used as input. In Section 5, we present the combination of stratified instance selection and the subgroup discovery algorithm to face the scaling up problem, analyzing the classical stratification method and the two novel proposed. Section 6 presents the experimental study developed and deals with the results and their analysis. Finally, in Section 7, we point out our conclusions. An appendix is included containing the description of the instance selection algorithms used in this study and presents the example weighting scheme and the rule quality function which the subgroup discovery algorithm involves.

2. Subgroup discovery

In this section we extend the subgroup discovery description, we present the Apriori-SD algorithm and the quality measures considered to analyze the subgroups discovered.

2.1. Description

Subgroup Discovery was defined by Klöesgen and Wrobel in (Klöesgen, 1996) and (Wrobel, 1997). In subgroup discovery we are interested in the identification of relations between a dependent variable (target variable) and usually more explanatory, independent variables (Lavrač et al., 2004). Subgroup discovery focus its interest on partial relations instead of complete relations; (small) subgroups with interesting characteristics can be sufficient. The discovered subgroups must satisfy two conditions: they should be interpretable for the expert, and they need to be interesting according to the criteria of the user. Interestingness is typically defined by a quality function, which can take certain statistical or other user-defined quality criteria into account.

The subgroup discovery is a task situated between the predictive and descriptive induction. As difference between classification or association rules and discovered subgroups we can cite Lavrač et al. (2004) where they emphasize that the goal in standard rule learning is to generate one model for each class, consisting of rule sets describing class characteristics in terms of properties occurring in the descriptions of training examples. Subgroup discovery aims at discovery individual rules of interest which must be represented in explicit symbolic form and relatively simple to be recognized as actionable by potential users. Standard classification algorithms use the covering algorithm for rule set construction which hinders the applicability of classification rule induction approaches in subgroup discovery (as Lavrac et al. show in (Lavrač et al., 2004)). In addition, subgroup discovery is different from classification as it addresses different goals: discovery of interesting population subgroups instead of maximizing classification accuracy of the induced rules set. This is manifested in the fact that in subgroup discovery tasks, many more false positives (negative examples incorrectly classified as positives) can be often tolerated than in classification.

There are some proposals on subgroup discovery algorithms, like SD-Map (Atzmueller and Puppe, 2006), SubgroupMiner (Klöesgen and Michael, 2002), CN2-SD (Lavrač et al., 2004) or Apriori-SD (Kavšek and Lavrač, 2006). The first one is discarded from our study due to the fact that it can only be used in two-class data sets. The last three ones are compared in (Kavšek and Lavrač, 2006) and the authors reach as conclusions that Apriori-SD acts very similarly to CN2-SD and is more suitable for predicting the minority classes, while SubgroupMiner found larger and more accurate subgroups when dealing with the majority classes. In large data sets, it is used to find minority classes as it is the case of the KDD Cup'99, where there are minority classes associated to different kinds of attack. Therefore, we have selected Apriori-SD as subgroup discovery algorithm in our study.

2.2. Apriori-SD algorithm

Kavšek et al. analyzes the Apriori-SD algorithm in (Kavšek and Lavrač, 2006), modifying the Apriori-C (which was based originally in the well-known Apriori algorithm (Agrawal et al., 1996) for mining association rules). In this case, the classification rule discovery algorithm Apriori-C (Lavrač et al., 2002) is adapted to subgroup discovery (Kavšek and Lavrač, 2006, 2002).

The association rule learning algorithms (Apriori) can be adapted for classification purposes (Apriori-C) by implementing the following steps: discretize continuous attributes, run an association rule learning algorithm, collect rules whose right-hand side of the rule consists of a single item, representing a value of the target attribute, and use this set of rules to classify the unclassified examples.

The main modifications in the Apriori-C algorithm, making it appropriate for subgroup discovery, involve the implementation of a weighting scheme example in post-processing rule (described in Appendix), a modified rule quality function incorporating example weights into the weighted relative accuracy heuristic (described in Appendix), a probabilistic classification scheme, and the use of the ROC space for improving the evaluation of discovered rules. The pseudo-code of the Apriori-SD algorithm is given in Fig. 1.

The input arguments of the algorithm are: *Examples*, *Classes*, *minSup*, *minConf* and *k*. *Examples* are the set of training examples, *Classes* are the values of the class attribute, parameter *k* determines the threshold for covered example elimination in rule post-processing ensuring the convergence of the algorithm, and parameters *minSup* and *minConf* denote the Apriori minimal support and confidence parameters, constraining rule search.

2.3. Quality measures

As quality measures for each one of the subgroups discovered we will use the following ones:

- *Support*: The support measure computes the frequency of correctly classified covered examples of a rule R_i ($Cond \rightarrow Class$):

$$Sup(R_i) = p(Class, Cond) = \frac{n(Class, Cond)}{N} \quad (1)$$

1. algorithm $APRIORI - SD(Examples, Classes, minSup, minConf, k)$
2. Ruleset= $APRIORI - C(Examples, Classes, minSup, minConf)$
   set all example weights of Examples to 1)
3. Majority= the majority class in Examples
4. Resultset= {}
5. Repeat
   6. BestRule= rule with the highest weighted relative accuracy in Ruleset.
   7. Resultset= Resultset ⋃ BestRule
   8. Ruleset= Ruleset \ decrease the weights of examples covered by BestRule remove from Examples the examples covered more than k-times
9. until Examples={} or Ruleset={}
10. return Resultset= Resultset ⋃ true → Majority

Fig. 1. Pseudocode of Apriori-SD algorithm.

where, $n(Class,Cond)$ is the number of instances of *Class* where the antecedents *Cond* are true and N the number of instances in the data set.

- *Confidence*: To analyze the predictive capabilities of the subgroups discovered, the confidence of each rule is obtained. This measure represents the number of positive instances covered among all the instances covered by the rule:

$$Conf(R_i) = \frac{p(Class|Cond)}{p(Cond)} = \frac{n(Class, Cond)}{n(Cond)} \quad (2)$$

where $n(Cond)$ is the number of instances where the antecedents *Cond* are true.

3. A large size data set: KDD Cup'99

The task for the classifier learning contest organized in conjunction with the KDD Cup'99 conference was to learn a predictive model (i.e. a classifier) capable of distinguishing between legitimate and illegitimate connections in a computer network. A standard set of data to be audited, which includes a wide variety of intrusions simulated in a military network environment, was provided. The 1999 KDD intrusion detection contest used a version of this data set.

Each connection is labeled as either normal, or as an attack, with exactly one specific attack type. Attacks fall into four main categories: dos (denial-of-service, e.g. syn flood), r2l (unauthorized access from a remote machine, e.g. guessing password), u2r (unauthorized access to local superuser) and probing (surveillance and other probing, e.g., port scanning).

The raw training data was about 4 GB of compressed binary dump data from seven weeks of network traffic. This was processed into about five million connection records. It is available in the UCI Repository (Newman et al., 1998). In this study, we work with its ten percent version which is available in the UCI Repository. It is composed by 494,020 instances (connections) with 41 attributes each, and 5 classes (the four previously mentioned and the one normal connection as Table 1 shows). The results for the KDD Cup'99 contest are available in (Elkan, 2000).

In the last years, this data set has been widely studied. In (Gao et al., 2006), Gao et al. proposed a novel LS-SVM intrusion detection model using kernel space approximation through greedy searching. Chen et al. in (Chen et al., 2007) offered an intrusion detection system model based on a general and enhanced flexible neural tree. The paper of Gunes et al. (2007) is dedicated to the application of a hierarchy of self-organizing feature maps to network intrusion detection. Yu et al. proposed in (Yu et al., 2007) an intrusion detection system which automatically tunes the detection model on-the-fly according to the feedback provided by the system operator when false predictions are encountered.

Table 1
Class distribution in the original KDD Cup'99 data set

Normal	u2r	Dos	r2l	Probe
97,277	52	391,458	1126	4107

4. Analysis of the scaling up problem for subgroup discovery: the case of Apriori-SD

Apriori-SD extracts a set of rules which are generated using a determined heuristic (see Section 2.2). The evaluation needed to apply the heuristic is expensive computationally, and this cost is directly proportional to the size of the data set.

To execute the Apriori-SD it is necessary to execute previously the Apriori-C, which has as precondition the discretization of the features which compose the instances. The discretization process is other bottleneck in the execution time when the size of the data set increases. As discretizer algorithm we use the Fayyad one (Fayyad and Irani, 1993), as the authors indicate in its previous work (Lavrač et al., 2004).

To analyze the effect of the size of the data set, we will split the data set and create subsets of different sizes (the percentages of instances per class are maintained), executing the discretization method (Fayyad discretization) and the subgroup discovery algorithm (Apriori-SD) over them (the parameters fixed for Apriori-SD appear in Section 6). Table 2 and Fig. 2 present the results of that analysis. The first column of the table is the number of instances in the subset. The second, third and fourth columns, show the execution time of the Fayyad discretization algorithm, the execution time of the Apriori-SD and their combined times, respectively (all in seconds).

The results which appear in Table 2 show that the execution time needed by Apriori-SD when the size of data set increases (due to its own execution time and its discretization method associated) makes it difficult to be used in large size data sets.

Table 2
Fayyad discretization and Apriori-SD execution times in seconds increasing the size of a KDD Cup'99 subset

Number of instances	Fayyad discr. time	Apriori-SD exec.time	Exec. time (Fayyad + Apriori-SD)
200	0.062	0.453	0.515
400	0.156	1.172	1.378
800	0.438	4.625	5.063
1600	2.001	18.656	20.657
3200	6.437	52.203	58.640
6400	26.958	205.750	232.708
12,800	149.438	1548.062	1697.500
25,600	2456.719	Not runnable	Not runnable

5. Stratified instance selection for subgroup discovery

In this paper we face the scaling up problem by means of the application of a pre-processing stage to reduce the initial data set previously to the subgroup discovery process, using instance selection algorithms.

Another process used for reducing the number of instances in training data is the prototype generation, which consists of building new examples by combining or computing several metrics among original data and including them into the subset of training data (Viswanath et al., 2006). Many of the examples of these generated sets may not coincide with examples belonging to the original training set, due to the fact that they are artificially generated. In some applications, this behaviour is not desired, like in Kdd Cup'99 data set, which is composed of information about real connections, and if new instances were generated, it could be possible that they do not correspond to valid real connections. This is the reason why the instance selection has been chosen.

In instance selection we want to isolate the smallest set of instances which enable us to predict the class of a query instance with the same quality as the initial data set (Liu and Motoda, 2001). By reducing the 'useful' data set size we can reduce the space complexity and decrease computational cost of the data mining algorithms that will be applied later, improving their generalization capabilities due to the elimination of noise.

As instance selection algorithms we have selected for this study those which show the best behaviour in (Cano et al., 2003), with low resources consumption and high reduction rates: Cnn (Hart, 1968), Ib3 (Aha et al., 1991) and Evolutionary instance selection based on CHC algorithm (EIS-CHC) (Cano et al., 2003). They are described in Appendix.

The instance selection algorithms are also affected by the size of the input data set (Cano et al., 2005). To avoid the drawbacks associated to large size data sets we apply the instance selection combined with stratified strategy as it was suggested in (Cano et al., 2005) with promising results.

In the following subsections we present the classic stratified strategy and two novel proposals which face the drawback introduced by the classic one in relation to the presence of minority classes.

5.1. Classical stratified instance selection approach

In classic stratified strategy, initial data set D is divided into t disjoint sets D_j, strata of equal size, D_1, D_2, ..., and D_t, keeping the classes distribution within each set in the partitioning process. The test set TS will be the TR complementary one in D:

$$TR = \bigcup_{j \in J} D_j, \quad J \subset \{1, 2, \ldots, t\} \tag{3}$$

$$TS = D \setminus TR \tag{4}$$

Instance selection algorithms are applied to each D_j obtaining a subset selected DS_j (see Fig. 3). The prototype selected set is obtained using DS_j (see Eq. (5)) and it is called Stratified Training Subset Selected (*STSS*),

$$\mathrm{STSS} = \bigcup_{j \in J} DS_j, \quad J \subset \{1, 2, \ldots, t\} \tag{5}$$

This stratification offers an interesting behaviour when the original data set is big enough to keep instances from all the classes in each one of the strata created.

In the original stratification proposal (Cano et al., 2005) the strata are created splitting randomly the initial data set and using the same distribution of classes than the initial one. The situation is that in the KDD Cup'99 data set there are high minority classes so when we split the data set, some of the strata could lose instances from one concrete class (as we can see in Table 3 in the case of class *u2r*). Following the classical stratification strategy there are too few or none instances from the minority classes in

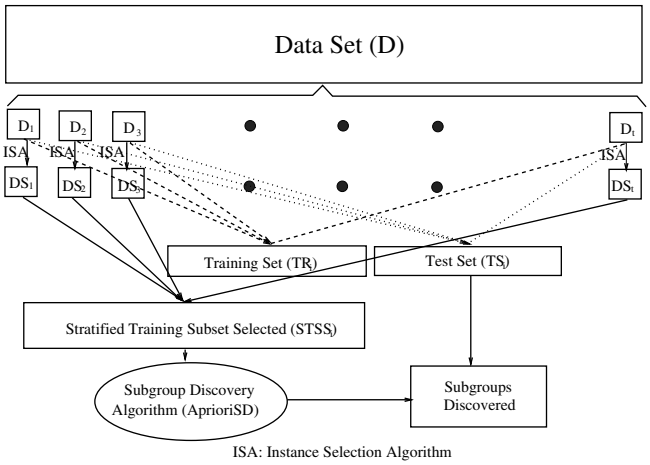

Fig. 3. Stratified instance selection previous the subgroup discovery.

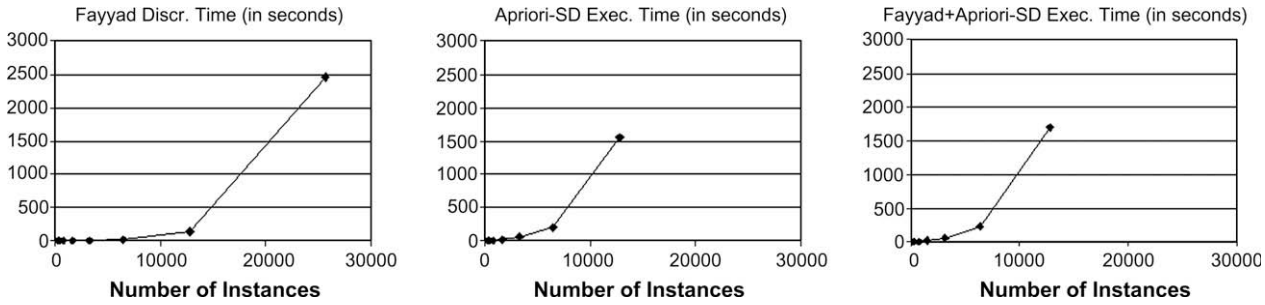

Fig. 2. Graphical representation of the execution time evolution in Fayyad discretization and Apriori-SD when the size of the data set increases.

Table 3
Class distribution per strata in number of instances and ratio class instances/majority class instances, considering 100 strata, in the Classic Stratification Strategy previous to the instance selection process

	Normal		u2r		Dos		r2l		Probe	
	#	Ratio	#	Ratio	#	Ratio	#	Ratio	#	Ratio
Original data set	97,277	24.84%	52	0.013%	391,458	100%	1126	0.287%	4107	1.045%
Strat. classical	972	24.83%	1 or 0	≈0%	3914	100%	11	0.281%	41	1.021%

the strata so for the selection process is difficult or impossible to select them. In that case, in the subset selected after the classic stratified instance selection process ($STSS_i$ subset in Fig. 3) the instances belonging to the minority classes are a few or none so the subgroup discovery task applied over it can't discover subgroup of the minority classes (in the case of KDD Cup'99 data set the minority class u2r is interesting due to it is an attack).

5.2. Two novel approaches for stratified instance selection based on increasing the presence of minority classes

The drawback of the classical stratification model is that it affects notably to minority classes. That situation makes that the Apriori-SD presents more difficulty to extract useful information (subgroups) from these classes.

To avoid the drawback which classic stratified strategy introduces when minority classes appear we offer two novel stratification proposals:

- *Instance selection in all classes (IS-AC):* We just randomly split the majority classes over the strata created. After the majority classes have been divided, the whole minority classes are added to each strata. After the instance selection process developed in each strata, the subsets selected are reunited removing instance duplicities. In the case of KDD Cup'99 the split of the majority classes is the same than the classic stratification, so in each strata will have (in the case of 100 strata) the 1% of the number of instances of the majority classes. The difference appears in the minority classes. Following this strategy, in each strata we keep all the instances of the minority classes (we keep all the instances belonging to r2l and u2r, 1126 and 52, respectively, in each strata). After that, the selection process starts. With this strategy, the most representative instances from all the classes are selected from each strata. The instance selection process selects the most representative instances belonged to all classes, reducing the number of instances. This situation produces that the $STSS_i$ subset, obtained after the reunion of the instances selected from the strata, can present a small number of instances from the minority classes (as Tables 4 and 5 show in the case u2r). The instances which appear in the $STSS_i$ subset are the most representative of all the classes but the minority classes maybe are not enough represented for a proper subgroup discovery task due to the stratified instance selection.That is the reason why we study the IS-MC alternative. The idea is to protect the minority classes from the instance selection process.
- *Instance selection in majority classes (IS-MC):* In the second proposal we randomly split the majority classes over the strata created, but the minority classes are not added nor divided. The selection process is applied without the minority ones, just in the majority classes (see Table 4). The instances which belong to the minority classes are added to the $STSS_i$ subset after the reunion of the subsets selected and before the subgroup discovery task starts. In this case, the instances which appear in $STSS_i$ are the most representative of the majority classes and all the instances belonging to the minority ones. Following this strategy, the instance selection process reduces the initial data set making it available for its use as input in Apriori-SD and does not affect to the presence of instances from the minority classes which makes possible the subgroup discovering process in those classes too.

In Table 4 we show the example of the class distribution in the whole data set, and in one strata considering the original stratification proposal (Strat classical), and the two proposals suggested, splitting the original data set in 100 strata. We present the number of instances and the ratio class instances/majority class instances to detect the minority classes.[1] The strata distribution is offered previous to the instance selection. In the IS-MC case, where the minority classes are not considered for the selection process, the minority classes are tagged with '*' and their ratio in the strata with '-' due to its absence.

With the proposal IS-AC, the idea is that the minority classes are not affected by the splitting process, appearing in small number in each strata. The proposal IS-MC tries to reduce the effect of the instance selection in the minority classes, keeping them till the moment that the subgroups are discovered.

6. Experimental study

The experimental study is defined in two aspects: data sets and algorithm's parameters. They are as follows:

- *Data sets:* The KDD Cup'99 one, divided in 100 strata.
- *Parameters*: The parameters are chosen considering the authors suggestions in the literature. For each one of the algorithms there are:
  - *Apriori-SD*: minimum support = 0.03, minimum confidence = 0.8, $k = 5$. The selection of the cut points for the numeric antecedents of the rules have been done using the Fayyad discretization method (Lavrač et al., 2004).
  - *Cnn*: It has not parameters to be fixed.
  - *Ib3*: acceptance level = 0.9 and drop level = 0.7.
  - *EIS-CHC*: evaluations = 10,000, population = 50 and $\alpha = 0.5$.

The deterministic algorithms have been executed one time for each partition in the 10-fold cross validation and three times the non-deterministic ones.

The experimental section is divided into two sections. In the first one we analyze the effect of stratified instance selection in the execution time of Apriori-SD. In the second one we study the quality of the subgroups discovered using both stratification methods, comparing them with random selections.

6.1. Execution time perspective: analysis of the stratified instance selection with Apriori-SD

As we have shown in Section 4, the Apriori-SD algorithm presents difficulties in its execution with data sets larger than 13,000 instances. So, in Table 5 we analyze the size of the subsets

[1] We consider minority classes those which relation between the number of instances in the class and the number of instances in the biggest class (Ratio in Tables 3 and 4) is smaller than 1% of the biggest one (case of u2r and r2l in KDD Cup'99 data set).

Table 4
Class distribution per strata in number of instances and ratio class instances/majority class instances, considering 100 strata, in the three kinds of stratifications previous to the instance selection process

	Normal		u2r		Dos		r2l		Probe	
	#	Ratio	#	Ratio	#	Ratio	#	Ratio	#	Ratio
Original data set	97,277	24.84%	52	0.013%	391,458	100%	1126	0.287%	4107	1.045%
Strat. classical	972	24.83%	1 or 0	≈0%	3914	100%	11	0.281%	41	1.021%
IS-AC	972	24.83%	52	1.328%	3914	100%	1126	28.76%	41	1.021%
IS-MC	972	24.83%	52*	–	3914	100%	1126*	–	41	1.021%

Table 5
Size of the subsets selected by the stratified instance selection algorithms and its class distribution

Algorithm	KDD Cup'99 subset size selected (instances)	Normal	Dos	Probe	u2r	r2l
Cnn IS-AC	113,535	5794	106,545	914	38	244
Ib3 IS-AC	7401	1499	4328	892	51	631
EIS-CHC IS-AC	7355	1644	4718	253	35	705
Cnn IS-MC	118,972	3062	113,844	888	52	1126
Ib3 IS-MC	5754	1739	1986	851	52	1126
EIS-CHC IS-MC	5342	1008	2875	281	52	1126

that the stratified instance selection algorithms offer for the KDD Cup'99 data set. The first column in the table presents the name of the instance selection algorithms and the model of stratification considered. The second column contains the number of instances which compose the subset selected after the stratified instance selection execution. The remaining columns indicate the distribution of the instances in the classes.

Analyzing Table 5 we can point out the following conclusions:

- Cnn presents low reduction rates, independently of the stratification model considered, so the subsets selected are still too large to be used as input for the Apriori-SD algorithm. The others instance selection methods (Ib3 and EIS-CHC) reduce significantly the initial data set defeating the scaling up problem.
- The first Stratification method (IS-AC) maintains the original majority classes in the same proportion in the subsets selected, while the second stratification model (IS-MC) offers more balanced proportions among classes.

With those subsets selected, we proceed with the subgroup discovery process that could not be executed using the whole initial data set. Due to the large size of the subsets selected by the Cnn, in this stage we use the one obtained for Ib3 and EIS-CHC with both stratification methods.

Table 6 presents the execution times of the subgroup discovery process. In the first column we offer the name of the instance selection algorithm, the stratification used, the following discretization method and the subgroup discovery algorithm applied. The second, third and fourth columns present the execution time consumed for the discretization method, the subgroup discovery execution time and the combined times for both, respectively.

Analyzing Table 6 we can point out that the pre-processing produces an important reduction in execution time as in discretization as in the subgroup discovery process. The combination of stratification and instance selection algorithm let us carry out a subgroup discovery task in large size data sets which could not be evaluated in other way. The stratification IS-MC with instance selection associated offers the smallest subsets and that situation reduces the execution time of Apriori-SD.

6.2. Subgroup discovery quality perspective: analysis of the stratified instance selection with Apriori-SD

In this section we analyze the subgroups discovered considering both stratification methods combined with the instance selection algorithms which present the biggest reduction rates (EIS-CHC and Ib3). In addition, we will compare the quality (confidence and support) of the subgroups extracted using the subsets selected by the stratified instance selection with the quality of the subgroups extracted using random selections over the original data set of the same size. The objective is to evaluate if the selection that the instance selection algorithms offers improves the random one, leaving us with the extraction of higher quality subgroups for all classes.

Table 7 shows the classes distribution using each one of the selection methods combined with stratification, including random selection for analyzing the instance selection algorithms quality:

In Table 8 we show the number of subgroups extracted per class in previous cases (combinations of EIS-CHC, Ib3 and random and IS-AC and IS-MC with Apriori-SD).

In Tables 9 and 10 we present some of the subgroups discovered. We offer the subgroup with highest confidence for each class if it is present, and their quality measures (confidence and support in test), using both stratification strategies and the random selections of the same size:

Table 6
Discretization and subgroup discovery execution time (in seconds) results for the subsets previously selected

Algorithm	Discr. time (Fayyad)	Apriori-SD exec. time	Exec. time (Fayyad + Apriori-SD)
Ib3 IS-AC + Fayyad + Apriori-SD	1090.342	9907.085	10997.427
EIS-CHC IS-AC + Fayyad + Apriori-SD	1180.383	9625.901	10806.284
Ib3 IS-MC + Fayyad + Apriori-SD	851.077	6441.992	8144.146
EIS-CHC IS-MC + Fayyad + Apriori-SD	722.969	5466.016	6188.985

Table 7
Classes distribution using instance and random selection keeping both stratification methodologies

Algorithm	KDD Cup'99 subset size selected (instances)	Normal	Dos	Probe	u2r	r2l
EIS-CHC IS-AC	7355	1644	4718	253	35	705
Random 1	7355	1454	5825	58	1	17
EIS-CHC IS-MC	5342	1008	2875	281	52	1126
Random 2	5342	801	3327	36	52	1126
Ib3 IS-AC	7401	4428	1399	892	51	631
Random 3	7401	1463	5876	45	0	17
Ib3 IS-MC	5754	2089	1636	851	52	1126
Random 4	5754	929	3599	48	52	1126

Table 8
Number of subgroups per class in each selection using Apriori-SD

Class	EIS-CHC IS-AC + Apriori-SD	EIS-CHC IS-MC + Apriori-SD	Ib3 IS-AC	Ib3 IS-MC
Normal	8	7	14	12
dos	12	14	11	17
Probe	0	4	6	9
u2r	0	0	0	0
r2l	4	6	0	9
	Random 1 + Apriori-SD	Random 2 + Apriori-SD	Random 3 + Apriori-SD	Random 4 + Apriori-SD
Normal	11	11	9	9
dos	8	8	9	7
Probe	0	0	0	0
u2r	0	0	0	0
r2l	0	0	0	6

Table 9
Subgroup discovered using EIS-CHC with IS-AC and IS-MC, and random selections combined with Apriori-SD

EIS-CHC IS-AC + Apriori-SD	EIS-CHC IS-MC + Apriori-SD
IF protocol_type = icmp THEN Class dos Confidence: 0.991; Support: 0.569	IF land = 0 AND logged_in = 0 AND dst_host_count = 255 THEN Class dos Confidence: 0.951; Support: 0.786
IF service = http THEN Class normal Confidence: 0.965; Support: 0.145	IF dst_host_srv_diff_host_rate $\geqslant$ 0.015 AND dst_host_srv_diff_host_rate < 0.075 THEN Class normal Confidence: 0.996; Support: 0.066
IF service = ftp_data AND logged_in = 1 THEN Class r2l Confidence: 0.196; Support: 0.001	IF service = ftp AND logged_in = 1 THEN Class r2l Confidence: 0.464; Support: 0.001 IF diff_srv_rate $\geqslant$ 0.570 AND dst_host_srv_count = 1 THEN Class probe Confidence: 0.704; Support: 0.004
Random 1 + Apriori-SD	Random 2 + Apriori-SD
IF dst_host_same_src_port_rate $\geqslant$ 0.305 THEN Class dos Confidence: 0.944; Support: 0.569	IF dst_host_same_src_port_rate < 0.305 THEN Class dos Confidence: 0.944; Support: 0.569
IF protocol_type = tcp AND src_bytes = SF THEN Class normal Confidence: 0.952; Support: 0.182	IF dst_host_srv_diff_host_rate $\geqslant$ 0.015 AND dst_host_srv_diff_host_rate < 0.275 THEN Class normal Confidence: 0.979; Support: 0.078

Table 10
Subgroup discovered using Ib3 with IS-AC and IS-MC, and random selections combined with Apriori-SD

Ib3 IS-AC + Apriori-SD	Ib3 IS-MC + Apriori-SD
IF serror_rate $\geqslant$ 442.015 THEN Class dos Confidence: 1.000; Support: 0.564	IF src_bytes = SF AND count $\geqslant$ 510.489 THEN Class dos Confidence: 1.000; Support: 0.459
IF hot = 0 AND logged_in = 1 THEN Class normal Confidence: 0.990; Support: 0.140	IF service = http AND flag < 4.5 THEN Class normal Confidence: 0.993; Support: 0.104
IF diff_srv_rate $\geqslant$ 0.530 AND dst_host_srv_count = 1 THEN Class probe Confidence: 0.705; Support: 0.004	IF num_outbound_cmds = 0 AND diff_srv_rate $\geqslant$ 0.754 AND dst_host_diff_srv_rate $\geqslant$ 0.505 THEN Class probe Confidence: 1.000; Support: 0.004 IF service = ftp AND logged_in = 1 THEN Class r2l Confidence: 0.464; Support: 0.001
Random 3 + Apriori-SD	Random 4 + Apriori-SD
IF dst_host_srv_serror_rate $\geqslant$ 0.0150 THEN Class dos Confidence: 0.983; Support: 0.175	IF srv_rerror_rate $\geqslant$ 0.015 AND srv_rerror_rate < 0.355 THEN Class dos Confidence: 0.996; Support: 0.216
IF protocol_type = tcp AND dst_host_srv_count = 255 THEN Class normal Confidence: 0.973; Support: 0.111	IF dst_host_srv_diff_host_rate $\geqslant$ 0.015 AND dst_host_srv_diff_host_rate < 0.075 THEN Class normal Confidence: 0.996; Support: 0.066 IF is_guest_login = 1 THEN Class r2l Confidence: 0.471; Support: 0.001

Paying attention to Tables 7–10, we can point out the following:

- The random 1 process selects too few instances which belong to the minority classes so it is difficult to extract subgroups from them. The random 2, which retains all the instances for the minority classes u2r and r2l is not able to generate subgroups for them when Apriori-SD is applied. Random 3 and 4 show the same behaviour as the two previous ones.
- The instance selection algorithms combined with IS-MC, and using the same number of instances as the random selections, extract subgroups from most of the classes. The proper selection of the representative instances of each class by means of instance selection algorithms improve the number and the quality of subgroups discovered as Tables from 7 and 10 indicate.
- For the stratification IS-AC, there are subgroups discovered from the majority classes but none of them from the minority classes in the subsets selected. The small number of instances in those minority classes is not enough to discover rules with Apriori-SD with measure levels higher than the indicated by the authors (support greater than 0.03 and confidence greater than 0.8).
- As we have just indicated, the use of stratification IS-MC lets us extract subgroups for the majority of classes. There are not subgroups for the u2r class, which has the minimal representation (52 instances in a data set composed by 494,020). That class presents too few instances for the discovering process with Apriori-SD.
- Considering the quality of the subgroups, we can indicate that the ones discovered using the previous process of stratified instance selection present better behaviour in confidence with similar support. The use of the most representative instances improves the quality of the subgroups extracted.
- Both stratification methods combined with instance selection algorithms improve the number of subgroups discovered, compared with the random processes.

As conclusion, we point out that the combination of the instance selection algorithms with stratification lets us run the Apriori-SD subgroup discovery algorithm in large size data sets that could not be used as input using the original data set. The instance selection in majority classes proposed allows us the extraction of subgroups for most of the classes, including most of the minority ones, with high levels of confidence and support measures.

7. Concluding remarks

This paper addresses the scaling up problem involved when large size data sets are used as input in subgroup discovery algorithms. To avoid the drawbacks introduced by the data set size, we propose the combination of stratification and instance selection previous the subgroup discovery task.

An experimental study has been carried out to analyze the effect of the data set size in subgroup discovery algorithms, and how the proposal can face this effect. The main conclusion reached is that the combination of stratification and instance selection algorithms as pre-processing makes available the subgroup discovery task in large size data sets, which could not be executed in other way. In addition, the proper election of the instances previous the subgroup discovery process improves the quality of the subgroups extracted. As stratification method we stress the second one proposed, instance selection in majority classes (IS-MC), because it suits better large data sets with different class distributions than the other stratification strategies.

Appendix

The description of the instance selection algorithms considered in the study is the following:

- *Cnn* (Hart, 1968) – tries to find a consistent subset, which correctly classifies all of the remaining points in the sample set. The Cnn algorithm finds a subset *S* of the training set *TR* such that every member of *TR* is closer to a member of *S* of the same class than to a member of *S* of a different class. The subset *S* can be used to classify all the instances in *TR* correctly. A description of the algorithm is given in Fig. 4.
- *Ib3* (Aha et al., 1991) – instance *x* from the training set *TR* is added to the new set *S* if the nearest *acceptable* instance in *S* (if there are not *acceptable* instances a random one is used) has different class than *x*. The *acceptable* concept is defined as the confidence interval:

$$\frac{p+\frac{z^2}{2n}\pm z\sqrt{\frac{p(p-1)}{n}+\frac{z^2}{2n^2}}}{1+\frac{z^2}{n}} \tag{6}$$

  where *z* is the confidence factor (0.9 is used to accept, 0.7 to reject). *p* is the classification accuracy of a *x* instance (while *x* is added to *S*). *n* is the number of classification-trials for the given instance (while added to *S*). The algorithm proceeds as shown in Fig. 5.
- Evolutionary instance selection based on *CHC* algorithm (*EIS-CHC*) (Cano et al., 2003; Eshelman, 1991) – Evolutionary algorithms (Back et al., 1997) are general-purpose search algorithms that use principles inspired by natural genetic populations to evolve solutions to problems, and they have been used to solve the instance selection problem, with promising results (Kuncheva, 1995; Kim, 2006). The election of *CHC* as instance selection algorithm is based on its behaviour showed in (Cano et al., 2003; Cano et al., 2005). During each generation the *EIS-CHC* develops the following steps:
  1. It uses a parent population of size *pop* to generate an intermediate population of *pop* individuals, which are randomly paired and used to generate *pop* potential offspring.
  2. Then, a survival competition is held where the best *pop* chromosomes from the parent and offspring populations are selected to form the next generation.

  Other important characteristics of this algorithm are:
  - *CHC* also implements a form of heterogeneous recombination using *HUX*, a special recombination operator. *HUX* exchanges half of the bits that differ between parents, where the bit position to be exchanged is randomly determined. *CHC* also employs a method of incest prevention. Before applying *HUX* to two parents, the Hamming distance between them is measured. Only those parents who differ from each other by some number of bits (mating threshold) are mated. The initial threshold is set at *L*/4, where *L* is the length of the chromosomes. If no offspring are inserted into the new population then the threshold is reduced by 1.
  - No mutation is applied during the recombination phase. Instead, when the population converges or the search stops making progress (i.e., the difference threshold has dropped to zero and no new offspring are being generated which are better than any members of the parent population) the population is reinitialized to introduce new diversity to the search. The chromosome representing the best solution found over the course of the search is used as a template to re-seed the population. Re-seeding of the population is accomplished by randomly changing 35% of the bits in the template chromosome to form each of the other $n-1$ new chromosomes in the population. The search is then resumed.

1. $TR = Examples\_Set$, $S = \emptyset$, $fail = true$
2. $S = S \cup \{x_{C_1}, x_{C_2}, ..., x_{C_M}\}$, where x_{C_i} is any example that belongs to class i
3. While $fail = true$
   4. $fail = false$
   5. For each example x in TR
      6. If x is misclassified by using S
         7. $S = S \cup \{x\}$
         8. $fail = true$
9. Return S

Fig. 4. Pseudocode of Cnn algorithm.

1. For each instance x in TR
2. Let a be the nearest acceptable instance in S to x (if there are no acceptable instances in S, let a be a random instance in S)
3. If class(a)≠class(x) then add x to S.
4. For each instance s in S
   5. If s is at least as close to x as a is
      6. Then update the classification record of s and remove s from S if its classification record is significantly poor
7. Remove all non-acceptable instance from S
8. Return S

Fig. 5. Pseudocode of Ib3 algorithm.

The description of the example weighting scheme which Apriori-SD involves is the following:

In this weighting scheme each covered positive examples are not deleted when the currently *best* rule is selected in the post-processing step of the algorithm. Instead, each time a rule is selected, the algorithm stores with each example a count i of how many times (with how many rules) the example has been covered so far. Weights of positive examples covered by the selected rule decrease according to the formula $w(e_j, i) = \frac{1}{i+1}$. In the first iteration all target class examples are assigned the same weight $w(e_j,0) = 1$, while in the following iterations the contributions of examples are inverse proportional to their coverage by previously selected rules. In this way the examples already covered by one or more selected rules decrease their weights while rules covering

many yet uncovered target class examples whose weights have not been decreased will have a greater chance to be covered in the following iterations. Covered examples are completely eliminated when their weights fall below a given threshold (e.g., when an example has been covered more than k times).

The description of the rule quality function which includes example weights considered in Apriori-SD is the following:

Weighted relative accuracy (*WRAcc*) is used in subgroup discovery to evaluate the quality of induced rules. Weighted relative accuracy (Lavrač et al., 1999; Todorovski et al., 2000) is defined as follows (considering the rule $X \rightarrow Y$, and $p(\,)$ the corresponding probability):

$$WRAcc(X \rightarrow Y) = p(X) \cdot (p(Y|X) - p(Y)) \tag{7}$$

Weighted relative accuracy consists of two components: generality $p(x)$, and relative accuracy $P(Y—X) - P(Y)$. Relative accuracy, is the accuracy gain of rule $X \rightarrow Y$ relative to the fixed rule $true \rightarrow Y$, which predicts all instances to be of class Y; rule $X \rightarrow Y$ is only interesting if it improves upon this *default* accuracy.

The rule quality measure *WRAcc* used in Apriori-SD has been further modified to enable handling example weights, which provide the means to consider different parts of the example space when selecting the best rules. The modified *WRAcc* measure is defined as follows (*wWRAcc*):

$$wWRAcc(X \rightarrow Y) = \frac{n'(X)}{N'} \cdot \left(\frac{n'(X,Y)}{n'(X)} - \frac{n(Y)}{N} \right) \tag{8}$$

where N' is the sum of the weights of all examples, $n'(X)$ is the sum of the weights of all covered examples, $n'(X,Y)$ is the sum of the weights of all correctly covered examples, $n(Y)$ is the number of examples of class Y, and N is the number of all examples.

References

Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A.I., 1996. Fast discovery of association rules. In: Advances in Knowledge Discovery and Data Mining. AAAI Press, pp. 307–328.

Aha, D.W., Kibler, D., Albert, M.K., 1991. Instance-based learning algorithms. Mach. Learn. 6, 37–66.

Atzmueller, M., Puppe, F., 2006. SD-Map – A fast algorithm for exhaustive subgroup discovery. In: Proc. 10th European Conf. on Principles and Practice of Knowledge Discovery in Databases (PKDD 2006), LNAI, vol. 4213, Springer-Verlag, pp. 6–17.

Back, T., Fogel, D., Michalewicz, Z., 1997. Handbook of Evolutionary Computation. Oxford University Press.

Batista, G.E.A.P.A., Prati, R.C., Monard, M.C., 2004. A study of the behavior of several methods for balancing machine learning training data. SIGKDD Explorations 6 (1), 20–29.

Cano, J.-R., Herrera, F., Lozano, M., 2003. Using evolutionary computation as instance selection for data reduction in KDD: An experimental study. IEEE Trans. Evol. Comput. 7 (6), 561–575.

Cano, J.-R., Herrera, F., Lozano, M., 2005. Stratification for scaling up evolutionary prototype selection. Pattern Recognition Lett. 26, 953–963.

Chen, Y., Abraham, A., Yang, B., 2007. Hybrid flexible neural-tree-based intrusion detection systems. Internat. J. Intell. Systems 22, 337–352.

Chawla, N.V., Japkowicz, N., Kolcz, A., 2004. Editorial: Special issue on learning from imbalanced data sets. SIGKDD Explorations 6 (1), 1–6.

Domingo, C., Gavaldá, R., Watanabe, O., 2002. Adaptive sampling methods for scaling up knowledge discovery algorithms. Data Mining Knowledge Discov. 6 (2), 131–152.

Elkan, C., 2000. Kdd'99 knowledge discovery contest. ACM SIGKDD Explorations Newslett. 1 (2), 78.

Eshelman, L.J., 1991. The CHC adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination. Found. Genetic Algorithms 1, 265–283.

Fayyad, U.M., Irani, K.B., 1993. Multi-interval discretisation of continuous valued attributes for classification learning. In: Proceedings of the 13th International Joint Conference on Artificial Intelligence. Morgan Kaufmann, pp. 1022–1027.

Gao, H., Wang, X., Yang, H., 2006. LS-SVM based intrusion detection using kernel space approximation and kernel-target alignment. In: Proc. 6th World Congress on Intelligent Control and Automation, Dalian, China.

Gunes, H., Nur, A., Heywood, M.-I., 2007. A hierarchical SOM-based intrusion detection system. Eng. Appl. Artif. Intell. 20, 439–451.

Hart, P.E., 1968. The condensed nearest neighbour rule. IEEE Trans. Inform. Theory 18 (3), 431–433.

Kavšek, B., Lavrač, N., 2006. APRIORI-SD: Adapting association rule learning to subgroup discovery. Appl. Artif. Intell. 20 (7), 543–583.

Kavšek, B., Lavrač, N., Bullas, J.C., 2002. Rule induction for subgroup discovery: A case study in mining UK traffic accident data. In: Proc. Internat. Multi-Conf. on Information Society (IS'02), pp. 127–130.

Khosravi, H., Kabir, E., 2007. Introducing a very large dataset of handwritten Farsi digits and a study on their varieties. Pattern Recognition Lett. 28 (10), 1133–1141.

Kim, K., 2006. Artificial neural networks with evolutionary instance selection for financial forecasting. Expert Systems Appl. 30 (3), 519–526.

Klöesgen, W., 1996. Explora: A multipattern and multiestrategy discovery assistant. In: Advances in Knowledge Discovery and Data Mining. MIT Press, pp. 249–271.

Klöesgen, W., Michael, M., 2002. Census data mining – An application. In: Proc. 5th European Conf. on Principles of Data Mining, Knowledge Discovery (PKDD'02), pp. 65–79.

Kuncheva, L., 1995. Editing for the k-nearest neighbours rule by a genetic algorithm. Pattern Recognition Lett. 16, 809–814.

Lavrač, N., Flach, P., Zupan, B., 1999. Rule evaluation measures: A unifying view. In: Proc. 9th Internat. Workshop on Inductive Logic Programming, pp. 174–185.

Lavrač, N., Flach, P., Kavšek, B., Todorovski, L., 2002. Adapting classification rule induction to subgroup discovery. In: Proc. IEEE Internat. Conf. on Data Mining, pp. 266–273.

Lavrač, N., Cestnik, B., Gamberger, D., Flach, P., 2004. Decision support through subgroup discovery: Three case studies and the lessons learned. Machine Learn. 57, 115–143.

Lavrač, N., Kavšek, B., Flach, P., Todorovski, L., 2004. Subgroup discovery with CN2-SD. J. Machine Learn. Res. 5, 153–188.

Liu, H., Motoda, H. (Eds.), 2001. Instance Selection and Construction for Data Mining. Kluwer Academic Publishers.

Newman, D.J., Hettich, S., Blake, C.L., Merz, C.J., 1998. UCI Repository of machine learning databases. University of California, Irvine, Dept. of Information and Computer Sciences. <http://www.ics.uci.edu/~mlearn/MLRepository.html>.

Riquelme, J.C., Aguilar, J.S., Toro, M., 2003. Finding representative patterns with ordered projections. Pattern Recognition 36, 1009–1018.

Todorovski, L., Flach, P., Lavrač, N., 2000. Predictive performance of weighted relative accuracy. In: Proc. 4th European Conf. on Principles of Data Mining and Knowledge Discovery, pp. 255–264.

Viswanath, P., Narasimba, M., Bhatnagar, S., 2006. Partition based pattern synthesis technique with efficient algorithms for nearest neighbor classification. Pattern Recognition Lett. 27 (14), 1714–1724.

Wilson, D.R., Martinez, T.R., 2000. Reduction techniques for instance-based learning algorithms. Machine Learn. 38, 257–268.

Wrobel, S., 1997. An algorithm for multi-relational discovery of subgroups. In: Proc. 1st European Conf. on Principles of Data Mining and Knowledge Discovery. Springer, pp. 78–87.

Yen, S.J., Lee, Y.S., 2006. Cluster-based sampling approaches to imbalanced data distributions. In: Proc. Data Warehousing and Knowledge Discovery (DaWak 2006), Lecture Notes in Computer Science, vol. 4081, Springer-Verlag, pp. 427–436.

Yu, Z., Tsai, J.J.P., Weigert, T., 2007. An automatically tuning intrusion detection system. IEEE Trans. Systems Man Cybernet. 37 (2), 373–384.

Zhang, J., Mani, I., 2003. kNN approach to unbalanced data distributions: A case study involving information extraction. In: Proc. ICML'2003 Workshop on Learning from Imbalanced Datasets, 2003.