Available online at www.sciencedirect.com

SCIENCE @ DIRECT®

Applied Soft Computing 6 (2006) 323–332

Applied Soft Computing

www.elsevier.com/locate/asoc

On the combination of evolutionary algorithms and stratified strategies for training set selection in data mining

José Ramón Cano [a], Francisco Herrera [b,*], Manuel Lozano [b]

[a] Department of Computer Science, University of Jaén, 23071 Jaén, Spain
[b] Department of Computer Science and Artificial Intelligence, ETS de Ingeniera Informatica,
University of Granada, 18071 Granada, Spain

Received 31 May 2004; received in revised form 26 January 2005; accepted 14 February 2005

Abstract

In this paper, we present a new approach for training set selection in large size data sets. The algorithm consists on the combination of stratification and evolutionary algorithms. The stratification reduces the size of domain where the selection is applied while the evolutionary method selects the most representative instances. The performance of the proposal is compared with seven non-evolutionary algorithms, in stratified execution. The analysis follows two evaluating approaches: balance between reduction and accuracy of the subsets selected, and balance between interpretability and accuracy of the representation models associated to these subsets. The algorithms have been assessed on large and huge size data sets. The study shows that the stratified evolutionary instance selection consistently outperforms the non-evolutionary ones. The main advantages are: high instance reduction rates, high classification accuracy and models with high interpretability.

Keywords: Evolutionary algorithms; Stratification; Instance selection; Training set selection; Data mining

1. Introduction

In data mining (DM), the construction of classification rules from data is a basic problem [1]. Data sets can be so large that it is impractical to train the classification rule using all available data.

Instance selection is an approach which selects a set of the available data for training [2]. The DM algorithm generates, using the subset selected as input, representation models which predict the outcome class of new unseen instances. In training set selection the objective is to select the highest quality training subsets so the models generated by DM algorithm presents the highest performances [3–5]. There are multiple instance selection approaches focused in data sets quality to classify by means of nearest neighbor classifiers [6–13], but they are not directed toward the extraction of models from the subsets selected.

As training set selection can be viewed as a search problem, it could be solved using evolutionary

* Corresponding author. Tel.: +34 58 244019; fax: +34 58 243317.
E-mail addresses: jrcano@decsai.ugr.es (J.R. Cano), herrera@decsai.ugr.es (F. Herrera), lozano@decsai.ugr.es (M. Lozano).

1568-4946/$ – see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2005.02.006

algorithms (EAs, [14]). EAs offer interesting results in knowledge discovery and DM [15,16]. In [17], the evolutionary instance selection is studied, comparing it performances with non-evolutionary ones, having as conclusion that evolutionary one outperforms the other methods.

When the size of data sets evaluated grows, the issue of scalability is present. The scaling up problem (due to this size) produces excessive storage requirement, increases times complexity and affects to generalization accuracy. In EAs these drawbacks are increased for the size of the chromosome used to represent the solutions in training set selection.

In this paper, we propose the combination of EAs with the stratification strategy for training set selection to face off to the drawbacks introduced by the scaling up problem. In large size data sets it is impractical to evaluate the algorithms over the complete data set so the stratification is a way to carry out the executions. Combining the subset selected per strata we can obtain the subset selected for the whole data set. The stratification reduces the data set size where the selection is carried out, while the EAs select the best local training subset. In that topic there are no proposals in this sense to the best of our knowledge.

The aim of this paper is to study the stratified evolutionary instance selection applied for training set selection in large size data sets. The approach is compared with non-evolutionary ones following the stratified strategy. As representative and efficient EA model has been chosen the heterogeneous recombination and cataclysmic mutation algorithm (CHC algorithm [18]), considering the performances it offers in [17] due to its balanced behavior in diversity and convergency.

In order to do this, this paper is set out as follows. In Section 2, we introduce the main ideas about training set selection, describing the process and the scaling problem which affects it due to large size data sets. In Section 3, we describe the new approach proposed, giving details of how stratified EAs can be applied to the training set selection problem. In Section 4, we explain the methodology followed in the experiments. Section 5 deals with the results and the analysis in large and huge size data sets. Finally, in Section 6, we reach our conclusion.

2. Training set selection

This section describes the training set selection, showing the drawbacks introduced by the size of data sets evaluated. In Section 2.1 we situate the training set selection process in the data reduction domain. Section 2.2 is dedicated to describe the key points of training set selection. Finally, Section 2.3 presents the scaling up problem induced by the large size of data sets.

2.1. Data reduction

As we know, DM techniques learn models from the data sets [19,20]. When the size of these data sets is large, DM techniques present problems to generate models which generalize properly [21]. To address this situation, data reduction is a feasible way. The objective of data reduction methods is to select the most representative information from data set. This information, introduced as input in DM algorithm, can increase the models capabilities and generalization properties.

To describe data reduction, we consider data as flat file and composed by terms called attributes or features. Each line in the file consists of attribute-values and forms an instance in a multi-dimensional space defined by the attributes. Data reduction can be achieved in many ways:

- *by selecting features* [22], we reduce the number of columns in the data set;
- *by discretizing feature-values* [23], we reduce the number of possible values of discretized features;
- and *by selecting instances* [2,24], we reduce the number of rows in the data set.

We focus our attention in data reduction by means of instance selection (IS). In IS 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 initial data set [25].

By reducing the 'useful' data set size we can:

- reduce space complexity;
- decrease computational cost;
- diminish the size of formulas obtained by a subsequent induction algorithm on the reduced

and less noise data sets. This may facilitate interpretation tasks.

IS raises the problem of defining relevance for a prototype subset. From the statistical viewpoint, relevance can be partly understood as the contribution to the overall accuracy, that would be e.g. obtained by a subsequent induction. We emphasize that removing instances does not necessarily lead to a degradation of the results: we have observed experimentally that a little number of instances can have performances c-omparable to those of the whole sample, and sometimes higher. Two reasons come to mind to explain such an observation. First, some noises or repetitions in data could be deleted by removing instances. Second, each instance can be viewed as a supplementary degree of freedom. If we reduce the number of instances, we can sometimes avoid over-fitting situations.

The training set selection is developed by means of IS algorithms. In the following section this process is described.

2.2. Model construction via training set selection

There may be situations in which there are too much data and these data in most cases are not equally useful in the training phase of the learning algorithm. IS mechanisms have been proposed to choose the most suitable points in the data set to become instances for the training data set used by the learning algorithm.

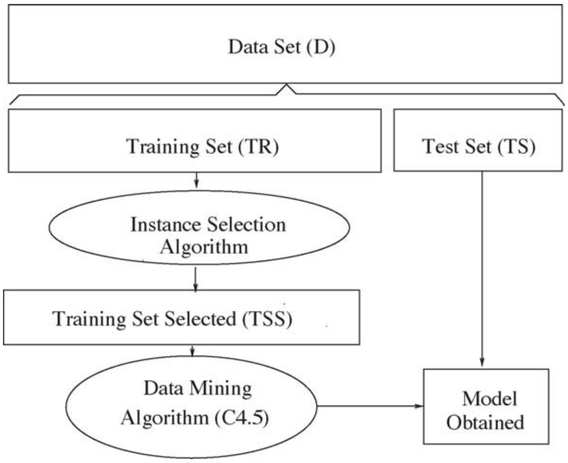
Data Set (D)
Training Set (TR)
Test Set (TS)
Instance Selection Algorithm
Training Set Selected (TSS)
Data Mining Algorithm (C4.5)
Model Obtained

Fig. 1. The training set selection process.

Fig. 1 shows the general framework for the application of the IS algorithm in training set selection. Starting from the data set, TR, the IS algorithm finds a suitable subset, TSS, then a learning or DM algorithm is applied to evaluate each subset selected (C4.5 in our case [26]) to obtain the model from the data set. This model is assessed using the test data set, TS.

2.3. The scaling up problem

The majority IS algorithms cannot deal with large size data sets, In this section we study the effect of the data set size in both groups of algorithms, evolutionary and non-evolutionary.

To test the effect of increasing the data set size, we have evaluated large and huge size data sets. The main difficulties the algorithms have to face are the following:

- Efficiency: The efficiency of non-evolutionary IS algorithms evaluated is at least of $\Theta(n^2)$, being n the number of instances in the data set. There are another set of IS algorithms (like *Rnn* in [27], *Snn* in [28], *Shrink* in [29], etc.) but most of them present an efficiency order much greater than $\Theta(n^2)$. Logically, when the size grows, the time needed by each algorithm also increases.
- Resources: Most of the algorithms assessed need to have the complete data set stored in memory to carry out their execution. If the size of the data set was too big, the computer would need to use the disk as swap memory. This loss of resources has an adverse effect on efficiency due to the increased access to the disk.
- Generalization: Algorithms are affected in their generalization capabilities due to the noise and over fitting effect introduced by larger size data sets.
- Representation: EAs are also affected by representation, due to the size of their chromosomes. When the size of these chromosomes is too big, the algorithms experience convergence difficulties, as well as costly computational time.

These drawbacks produce considerable degradation in the behavior of IS algorithms. There is a group of them that can't be applied due to its efficiency (the case of *Snn* in [28] with $\Theta(n^3)$).

Algorithms evaluated directly over the whole large size data sets are unefficacy and unefficient.

3. The evolutionary stratified instance selection algorithm

Trying to avoid the drawbacks previously mentioned, our proposal is directed towards the hybridation between EAs and stratified strategy. In Section 3.1 we offer the application of EAs in training set selection. Section 3.2 describes the stratified strategy and its integration with IS algorithms.

3.1. Evolutionary algorithms in training set selection

The application of EAs in training set selection is accomplished by tackling two important issues: the specification of the representation of the solutions and the definition of the fitness function.

3.1.1. Representation

Let us assume a data set denoted TR with n instances. The search space associated with the instance selection of TR is constituted by all the subsets of TR. Then, the chromosomes should represent subsets of TR. This is accomplished by using a binary representation. A chromosome consists of n genes (one for each instance in TR) with two possible states: 0 and 1. If the gene is 1, then its associated instance is included in the subset of TR represented by the chromosome. If it is 0, then this does not occur.

3.1.2. Fitness function

Let TSS be a subset of instances of TR to evaluate and be coded by a chromosome. We define a fitness function that combines two values: the classification performance (clas_per) associated with TSS and the percentage of reduction (perc_red) of instances of TSS with regards to TR:

$$\text{fitness}\,(\text{TSS}) = \alpha \cdot \text{clas\_rat} + (1-\alpha)\cdot \text{perc\_red}. \quad (1)$$

The 1-NN classifier is used for measuring the classification rate, clas_rat, associated with TSS. It denotes the percentage of correctly classified objects from TR using only TSS to find the nearest neighbour. For each object y in TSS, the nearest neighbour is searched for amongst those in the set TSS\{y}. Whereas, perc_red is defined as:

$$\text{perc\_red} = 100(|\text{TR}| - |\text{TSS}|)/|\text{TR}|. \quad (2)$$

The objective of the EAs is to maximize the fitness function defined, i.e., maximize the classification performance and minimize the number of instances obtained. In the experiments presented in this paper, we have considered the value $\alpha = 0.5$ in the fitness function, as per a previous experiment in which we found the best trade-off between precision and reduction with this value.

The evolutionary model applied is the CHC algorithm, described in Section 4.1.2.

3.2. Stratification strategy

This strategy divides the initial data set into strata. The strata are disjoints sets with equal class distribution. The number of strata will determine the size of them (see Fig. 2).

Using the proper number of strata, we can reduce significantly the set size where the selection will be applied. This situation allows us to avoid the drawbacks commented in Section 2.3.

Following the 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. We maintain class distribution within each set in the partitioning process.

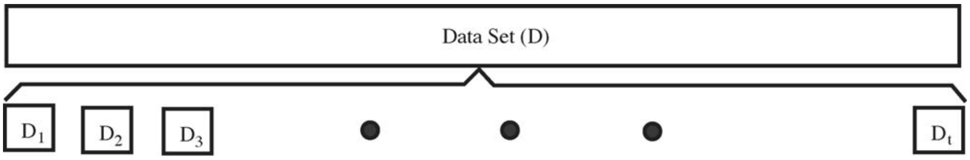

Fig. 2. Stratified strategy.

In the stratified strategy, the IS algorithm is applied in each D_j to obtain its DS_j associated. The training set selected is obtained using the DS_j (see Eq. (3)) and it is called stratified training subset selected (STSS).

$$\text{STSS} = \bigcup_{j \in J} DS_j, J \subset \{1, 2, \ldots, t\} \tag{3}$$

The test set TS will be the TR complementary one in D.

$$\text{TR} = \bigcup_{j \in J} D_j, J \subset \{1, 2, \ldots, t\} \tag{4}$$

$$\text{TS} = D \backslash \text{TR} \tag{5}$$

After evaluate all strata by means of the IS algorithm, we have to reunite the results in a last phase, where the final subset is created (STSS).

The evolutionary stratified instance selection algorithm follows this model, using CHC as IS algorithm to select each DS_j.

4. Experimental methodology

We have carried out our study in training set selection using two size problems: large and huge. We evaluate the behavior of the algorithms increasing the size of the data set where they are assessed.

Section 4.1 is dedicated to describe the algorithms which appear in the experiments. In Section 4.2 we present the data sets evaluated. Section 4.3 shows the stratification and partition of the data sets that were considered, and finally, in Section 4.4, we describe the table contents that report the results.

4.1. Instance selection algorithms for experiments

The algorithms selected in this study are the most efficacy ones shown in [17] due to its balanced behavior in diversity and convergency. The nature of the algorithms, evolutionary or not let us classify them in two groups:

4.1.1. Non-evolutionary algorithms

In this section we present the summary of non-evolutionary IS algorithms included in this study. The algorithms used will be:

- *Cnn* [30]: It tries to find a consistent subset, which correctly classifies all of the remaining points in the sample set. However, this algorithm will not find a minimal consistent subset.
- Drop1 [31]: Essentially, this rule tests to see if removing an instance would degrade leave-one-out cross-validation generalization accuracy, which is an estimate of the true generalization ability of the resulting classifier.
- Drop2 [31]: Drop2 changes the order of removal of instances. It initially sorts the instances in TR by the distance to their nearest enemy (nearest instance belonging to another class). Instances are then checked for removal beginning at the instance furthest from its nearest enemy. This tends to remove instances furthest from the decision boundary first, which in turn increases the chance of retaining border points.
- Drop3 [31]: Drop3 uses a noise filtering pass before sorting the instances in TR. This is done using the rule: Any instance not classified by its k-nearest neighbours is removed.
- *Ib*2 [29]: It is similar to *Cnn* but using a different selection strategy.
- *Ib*3 [29]: It outperforms *Ib*2 introducing the acceptable instance concept to carry out the selection. The parameters associated to *Ib*3 appear in Table 1.

Table 1 presents the parameters associated with the algorithms.

4.1.2. Evolutionary algorithms

EAs are general-purpose search algorithms that use principles inspired by natural genetic populations to evolve solutions to problems [32]. We have evaluated the CHC algorithm as representative and efficient EA model [18]. It has been chosen due to in [17], the CHC is the algorithm which shows the best balance between convergency and diversity among the EAs, selecting the smallest and most representative subsets of instances among all the algorithms studied.

Table 1
Algorithm's parameters

Algorithm	Parameters
*Ib*3	Aceptance level = 0.9; drop level = 0.7
CHC	Population = 50; evaluations = 10000

During each generation the CHC develops the following steps:

(1) It uses a parent population of size n to generate an intermediate population of n individuals, which are randomly paired and used to generate n potential offspring.
(2) Then, a survival competition is held where the best n chromosomes from the parent and offspring populations are selected to form the next generation.

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.

The parameters associated to CHC algorithm appear in Table 1.

4.2. Data sets for experiments

To evaluate the behavior of the algorithms applied in different size data sets, we have carried out the experiments increasing complexity and size of data sets. We have selected large and huge size data sets as we can see in Table 2. These data sets can be found in the University of California, Irvine, Repository (UCI Repository [33]).

Table 2
Data sets

Data set	Instances	Features	Classes
Adult	30132	14	2
Kdd Cup'99	494022	41	23

4.3. Partitions and stratification: An specific model

We have evaluated each algorithm in a 10-fold cross validation process. In the validation process TR_i, $i = 1, \ldots, 10$ is a 90% of D and TS_i its complementary 10% of D.

In our experiments, we have executed the IS algorithms following two perspectives for the 10-fold cross validation process:

In the first one, we have executed the IS algorithms as we can see in Fig. 3. We call it 10-fold cross validation classic *(*Tfcv classic*)*. The idea is use this result as baseline versus the stratification ones.

In Tfcv classic the subsets TR_i and TS_i, $i = 1, \ldots, 10$ are obtained as Eqs. (6) and (7) indicate:

$$\mathrm{TR}_i = \bigcup_{j \in J} D_j, J = \{j/1 \leq j \leq b \cdot (j-1) \text{ and } (i \cdot b) + 1 \leq j \leq t\} \tag{6}$$

$$\mathrm{TS}_i = D \backslash \mathrm{TR}_i \tag{7}$$

where t is the number of strata, and b is the number of strata grouped ($b = t/10$, to carry out the 10-fold cross validation).

Each set TSS_i is obtained by the IS algorithm applied to TR_i subset.

The second way is to execute the IS algorithms in a stratified process as Fig. 4 shows. We call it 10-fold cross validation strat *(*Tfcv strat*)*.

In Tfcv strat each TR_i is defined as we can see in Eq. (6), by means of the union of D_j subsets (see Fig. 4).

In Tfcv strat (see Fig. 4) STSS_i is generated using the DS_j instead of D_j (see Eq. (8)).

$$\mathrm{STSS}_i = \bigcup_{j \in J} DS_j, J = \{j/1 \leq j \leq b \cdot (i-1) \text{ and } (i \cdot b) + 1 \leq j \leq t\} \tag{8}$$

STSS_i contains the instances selected by IS algorithms in TR_i following the stratified strategy.

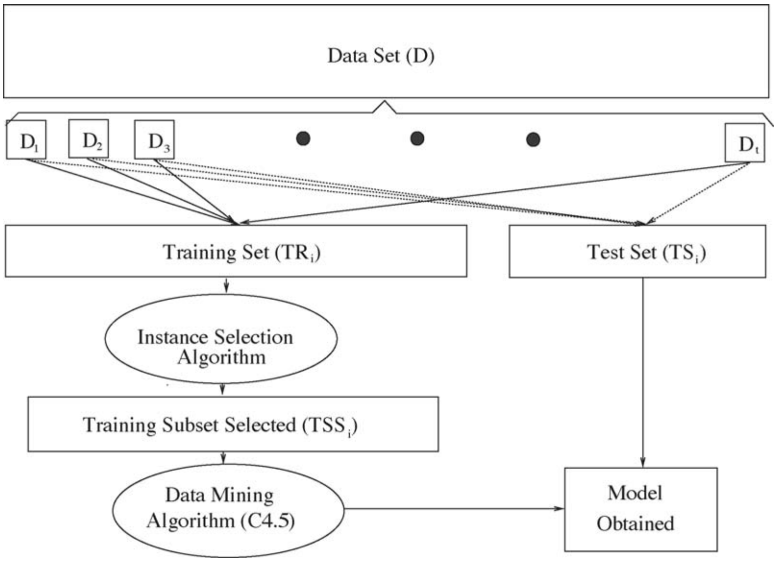
Data Set (D)
D1
D2
D3
Dt
Training Set (TRi)
Test Set (TSi)
Instance Selection Algorithm
Training Subset Selected (TSSi)
Data Mining Algorithm (C4.5)
Model Obtained

Fig. 3. Training set selection in 10-fold cross validation.

The subset TS_i is defined by means of Eq. (7). Both, TR_i and TS_i are generated in the same way in Tfcv classic and Tfcv strat.

4.4. Table of results

In the following section we will offer the structure of table where we present the results.

Our table shows the results obtained by the evolutionary and non-evolutionary IS algorithms, respectively. In order to observe the level of robustness achieved by all algorithms, the table presents the average in the 10-fold cross validation process of the results offered by each algorithm in the data sets evaluated. Each column shows:

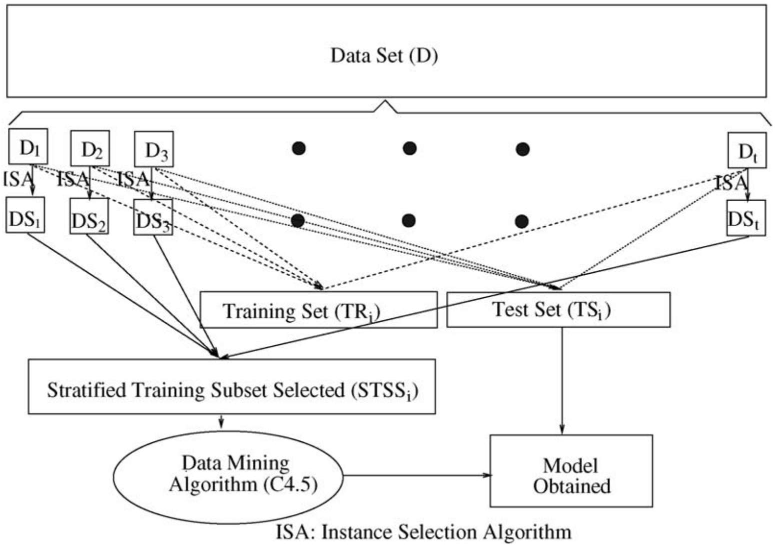
Data Set (D)
D1
D2
D3
Dt
ISA
ISA
ISA
ISA
DS1
DS2
DS3
DSt
Training Set (TRi)
Test Set (TSi)
Stratified Training Subset Selected (STSSi)
Data Mining Algorithm (C4.5)
Model Obtained
ISA: Instance Selection Algorithm

Fig. 4. Training set selection in stratified 10-fold cross validation.

Table 3
Results for adult data set

	Tim.	Num. rules	% reduc.	%Ac. trn (TR)	%Ac. test (TS)
C4.5 *classic*	2	327		88.72	85.40
Cnn *strat*	1	21	97.34	52.17	36.45
Drop1 *strat*	44	1	95.09	24.92	26.31
Drop2 *strat*	48	179	70.33	85.61	83.09
Drop3 *strat*	41	75	95.57	82.96	77.29
*Ib*2 *strat*	1	12	99.57	49.42	36.37
*Ib*3 *strat*	3	162	76.69	85.17	82.73
Icf *strat*	33	138	85.62	79.99	82.21
CHC *strat*	20172	4	99.38	83.78	82.76

- The first column shows the name of the algorithm. In this column the name is followed by the sort of validation process: *strat* or *classic* meaning the validation process followed.
- The second column contains the average execution time (in seconds) associated to each algorithm. The algorithms have been run in a Pentium 4, 2.4 Ghz, 256 RAM, 40 GB HD.
- The third column offers the mean number of rules associated to the decision tree generated by C4.5.
- The fourth column shows the average reduction percentage from the initial training sets.
- The fifth column contains the accuracy when C4.5 is applied using the training set selected. It shows the training accuracy associated to the model generated using the subset selected (STSS) in TR.
- Last column offers the test accuracy of the algorithms in the model obtained by C4.5 (TS assessed using STSS).

5. Experimental results and analysis

This section shows the results and its associated analysis in the evaluation of large and huge size data sets.

5.1. Experimental results

Tables 3 and 4 contain the results obtained in the evaluation of Adult and Kdd Cup'99 data sets, respectively. The number of strata used in Adult data set is $t = 10$. In Kdd Cup'99 the number of strata is $t = 100$.

5.2. Analysis of the results

The analysis is developed paying attention to the balance among reduction, accuracy and interpretability offered by the algorithms assessed. The study of third, fourth and sixth columns in Tables 3 and 4

Table 4
Results for Kdd Cup'99 data set

	Tim.	Num. rules	% reduc.	%Ac. trn (TR)	%Ac. test (TS)
C4.5 *classic*	365	252		99.97	99.94
Cnn *strat*	8	83	81.61	98.48	96.43
Drop1 *strat*	7	3	99.97	38.63	34.97
Drop2 *strat*	105	82	76.66	81.40	76.58
Drop3 *strat*	131	59	56.74	77.02	75.38
*Ib*2 *strat*	7	58	82.01	95.81	95.05
*Ib*3 *strat*	3	74	78.92	99.13	96.77
Icf *strat*	242	68	23.62	99.98	99.53
CHC *strat*	1960	9	99.68	97.68	97.53

allow us to make the following comments according to the balance among the three objectives:

- Most of the non-evolutionary algorithms which present higher reduction rates, decrease their prediction capabilities. Stratified CHC is the one which provides the highest reduction rates in the problems evaluated, independently of their size. In both cases, it reduces more than 99% of the initial data set.
- In the case of accuracy rate (sixth column), C4.5 assessed without any kind of reduction is the best one. Our proposal is only slightly improved by non-evolutionary algorithms which offer lower reduction rates.
- Most of non-evolutionary algorithms present much bigger decision trees than evolutionary one. C4.5 applied without any kind of reduction generates the biggest decision trees. Our proposal presents the smaller decision trees among all algorithm assessed.

The main conclusion that can be drawn is that Stratified CHC is the algorithm among the studied which offers the most interpretable training set selected, with the best accuracy and reduction rates associated in large and huge size data sets.

Briefly summarizing this comments, we conclude that when stratified CHC is applied to training set selection in large size data sets it produces the most interpretable models with highest accuracy and reduction rates. This small model size increases the decision tree's speed in classification, reducing its storage necessities and increasing its human interpretability.

6. Conclusions

In this paper the stratified evolutionary instance selection algorithm for training set selection in large size data sets is presented. It selects the minor number of most representative instances which maintain classification capabilities and increase the interpretability of the models generated by DM algorithms. While the stratification face off the scaling up problem, CHC selects the most representative instances.

The main conclusions reached in the experimental study are the following:

- *Stratified* CHC outperforms the classical algorithms, simultaneously offering the best balance among data reduction, accuracy and interpretability. Our proposal significantly reduces the size of the decision tree associated to the model obtained. This characteristic produces decision trees that are easier to interpret.
- In large and huge size data sets, non-evolutionary algorithms do not present balanced behavior. If the algorithm reduces the size then its accuracy rate is poor. When accuracy increases there is no reduction. The stratified version of CHC offers the best results when the data set size increases.

Therefore, as a final concluding remark, we consider Stratified strategy combined with CHC to be the best mechanism for training set selection in DM. Our proposal offers as principal characteristics its balance among reduction, precision and interpretability.

It has become a powerful tool to obtain smaller and high quality training sets and therefore scaling down data. CHC can select the most representative instances, satisfying the objectives of high accuracy and reduction rates, and the most interpretable models. Stratified strategy permits the reduction of the search space so we can carry out the evaluation of the algorithms in acceptable execution time.

Acknowledgement

This work was supported by TIC2002-04036-C05-01.

References

[1] M. Garofalakis, D. Hyun, R. Rastogi, K. Shim, Building decision trees with constraints, Data Mining Knowledge Discovery 7 (2003) 187–214.

[2] H. Liu, H. Motoda, On issues of instance selection, Data Mining Knowledge Discovery 6 (2002) 115–130.

[3] C.R. Reeves, S.J. Taylor, Selection of training data for neural networks by a genetic algorithm, Par. Prob. Solv. Nat. (1998) 633–642.

[4] M. Sebban, T. Nock, J.H. Chauchat, R. Rakotomalala, Impact of learning set quality and size on decision tree perfomances, Int. J. Comp. Syst. Sig. 1 (1) (2000) 85–105.

[5] B. Bolat, T. Yildirim, A data selection method for probabilistic neural networks, J. Electr. Electron. Eng. 4 (2) (2004) 1137–1140.

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

[7] J.C. Riquelme, J.S. Aguilar, M. Toro, Finding representative patterns with ordered projections, Pattern Rec. 36 (4) (2003) 93–102.

[8] S.W. Kim, B.J. Oommen, A brief taxonomy and ranking of creative prototype reduction schemes, Pattern Anal. Appl. 6 (2003) 232–244.

[9] J.S. Sanchez, R. Barandela, A.I. Marquez, R. Alejo, J. Badenas, Analysis of new techniques to obtain quality training sets, Pattern Rec. Lett. 24 (7) (2003) 1015–1022.

[10] J.S. Sanchez, High training set size reduction by space partitioning and prototype abstraction, Pattern Rec. 37 (2004) 1561–1564.

[11] J.H. Chen, H.M. Chen, S.Y. Ho, Design of nearest neighbor classifiers using an intelligent multi-objective evolutionary algorithm, Lec. Notes Comp. Sci. 3157 (2004) 262–271.

[12] J.R. Cano, F. Herrera, M. Lozano, Stratification for scaling up evolutionary prototype selection, Pattern Rec. Lett., submitted to publication.

[13] Y. Jiang, Z.H. Zhou, Editing training data for kNN classifiers with neural network ensemble, Lec. Notes Comp. Sci. 3173 (2004) 356–361.

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

[15] A.A. Freitas, Data Mining and Knowledge Discovery with Evolutionary Algorithms, Springer-Verlag, 2002.

[16] W. Romao, A.A. Freitas, I.M.S. Gimenes, Discovering interesting knowledge from a science and technology database with a genetic algorithm, Appl. Soft Comp. J. 4 (2) (2004) 121–137.

[17] J.R. Cano, F. Herrera, M. Lozano, Using evolutionary algorithms as instance selection for data reduction in kdd: an experimental study, IEEE Trans. Comp. 7 (6) (2003) 561–575.

[18] L.J. Eshelman, The chc adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination, Found. Gen. Algorithms 1 (1991) 265–283.

[19] P. Adriaans, D. Zantinge, Data Mining, Addison-Wesley, 1996

[20] I.H. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques with JAVA Implementations, Morgan Kaufmann, 2000.

[21] T. Oates, D. Jensen, The effects of training set size on decision tree complexity, in: Proceedings of the 14th International Conference on Machine Learning, 1997, pp. 254–262.

[22] H. Liu, H. Motoda, Feature Selection for Knowledge Discovery and Data Mining, Kluwer Academic Publishers, 2001.

[23] H. Liu, F. Hussain, C.L. Tan, M. Dash, Discretization: an enabling technique, Data Mining Knowledge Discovery 6 (2002) 393–423.

[24] H. Brighton, C. Mellish, Advances in instance selection for instance-based learning algorithms, Data Mining Knowledge Discovery 6 (2002) 153–172.

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

[26] J.R. Quinlan, C4. 5: Programs for Machine Learning, Morgan Kaufmann, 1993.

[27] G.W. Gates, The reduced nearest neighbour rule, IEEE Trans. Inform. Theory 18 (5) (1972) 431–433.

[28] G.L. Ritter, H.B. Woodruff, S.R. Lowry, T.L. Isenhour, An algorithm for a selective nearest neighbour decision rule, IEEE Trans. Inform. Theory 21 (6) (1975) 665–669.

[29] D. Kibbler, D.W. Aha, Learning representative exemplars of concepts: an initial case of study, in: Proc. 4th Int. Workshop on Machine Learning, 1987, 24–30.

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

[31] D.R. Wilson, T.R. Martinez, Instance pruning techniques, in: Proc. 14th Int. Conf., 1997, 403–411.

[32] D.E. Goldberg, The Design of Competent Genetic Algorithms: Steps Toward a Computational Theory of Innovation, Kluwer Academic Publishers, 2002.

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