Data Set Editing by Ordered Projection

Jesús S. Aguilar[1], José C. Riquelme[1] and Miguel Toro[1]

Abstract. In this paper, an editing algorithm based on the projection of the examples in each dimension is presented. The algorithm, that we have called EOP (Editing by Ordered Projection) has some interesting characteristics: important reduction of the number of examples from the database; lower computational cost in respect of other typical algorithms due to the absence of distance calculations; conservation of the decision boundaries, especially from the point of view of the application of axis-parallel classifiers; reduction of the decision tree size or the number of decision rules. The performance of EOP is showed by comparing the results provided by C4.5 [5] before and after applying it on databases with continuous attributes. These experiments have been realised using some databases from UCI repository [1]. The use of EOP as preprocessing method for the later application of any axis-parallel learning algorithm convert it in a valuable tool in the field of data mining.

1 INTRODUCTION

The data mining researchers, especially those dedicated to the study of algorithms that produce knowledge in some of the usual representations (decision lists, decision trees, association rules, etc.), make usually their tests on standard and accessible databases (most of them with small size). The purpose is to verify and validate independently the results of their algorithms. Nevertheless, these algorithms are modified to solve specific problems, for example real databases that contain much more information (number of examples) than standard used as training. To accomplish the final tests on these real databases with tens of attributes and thousands of examples is a task that takes a lot of time and memory size.

Among all the methodologies used by data mining researchers, those based on axis-parallel classifiers are the most common. These have an important advantage: they are classifiers that provide easy-to-understand decision rules by human and very useful for the expert interested in getting knowledge from the database. The C4.5 tool [Quinlan93] is, probably, the most useful technique of this type. C4.5 and other tools have a similar rule syntax (or equivalent) to the next one (a_i: attribute; m_i, M_i: numeric constants):

If $m_1 \le a_1 \le M_1$ and $m_2 \le a_2 \le M_2$ and ...and $m_n \le a_n \le M_n$ then class

Therefore, it is important to apply to the databases preprocessing techniques to reduce the number of examples or the number of attributes in such a way to decrease the computational cost. These preprocessing techniques are fundamentally oriented to one of the next goals: editing (reduction of the number of examples by eliminating some of them or calculating prototypes) and feature selection (eliminating non-relevant attributes). Our algorithm belongs to the first group.

Editing methods are much related to the nearest neighbours (NN) techniques [2]. Some of them are briefly cited in the following lines. In [3] is proposed to include in the set of prototypes those examples whose classification is wrong using the nearest neighbour technique; [9] proposed to eliminate the examples with incorrect k-NN classification; the works of [6] and [7] follows the same idea. Other variants are based on Voronoi diagrams [4], Gabriel neighbours (two examples are said to be Gabriel neighbours if their diametrical sphere does not contain any other examples) or relative neighbours [8] (two examples p and q are relative neighbours if for all other examples x in the set, is true the next expression $dist(p,q) < max\{dist(p,x), dist(q,x)\}$). All these techniques need to calculate distances between examples which is rather time consuming. If N examples with M attributes are considered, the first methods have $O(MN^2)$; the Voronoi neighbours is $O(MN^3)$; and Gabriel neighbours and relative neighbours are $O(MN^3)$.

The most important characteristics of EOP are the following:

- Considerable reduction of the number of examples.
- Lower computational cost O(MNlogN) than other algorithms.
- Absence of distance calculations.
- Conservation of the decision boundaries, especially interesting for applying classifiers based on axis-parallel decision rules (like C4.5).
- Reduction of the decision tree size or the number of decision rules.

In the last section we have dealt with several databases from UCI repository: small size (Iris), middle size (Breast Cancer and Pima Indian Diabetes) and large size (Letter). To show the goodness of our method we have used C4.5 before and after applying EOP. A five-cross validation for both applications is summarised.

[1] Department of Computer Science. University of Seville. Spain. {aguilar, riquelme, mtoro}@lsi.us.es

2 DESCRIPTION OF THE ALGORITHM

If we choose a region where all examples inside have the same class, maybe we could select some of them, which are not decisive to establish the boundaries of the region. For example, in two dimensions, we need four examples to determine the boundaries of one region, at maximum. In general, in d-dimensions, we will need 2d examples at maximum. Therefore, if a region has more than 2d examples, we could reduce it.

That is the main idea of our algorithm: to eliminate the examples that are not in the boundaries of their regions. The aim is to calculate what set of examples could be covered by a rule and then eliminate those inside the rule that are not establishing the boundaries.

The method is completely heuristic because EOP will work independently with the projection of the example in each dimension, not all dimensions at the same time. This heuristic could seem poor for lack of generality, however the results are quite the opposite.

To show graphically (figure 1) the idea of our algorithm we use a simple two-dimensional database with twelve numbered examples and two labels: I (odd numbers) and P (even numbers).

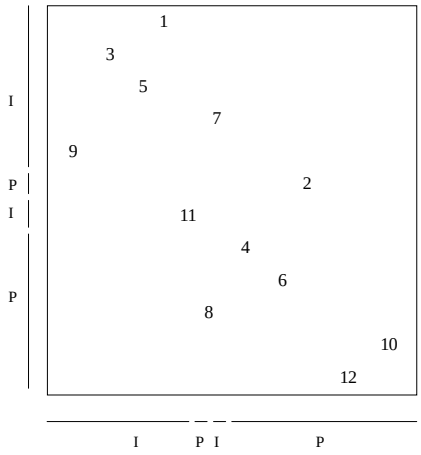

Figure 1. An example of database.

An optimal classifier would obtain the two rules showed in figure 2. However, this classifier must be hierarchical, since it is producing overlapped rules. This is not the case of C4.5 and many others.

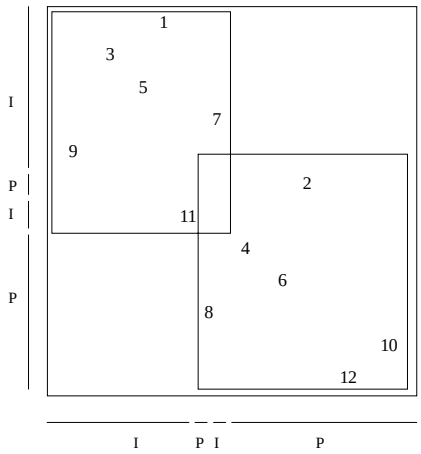

Figure 2. The best solution with overlapped rules.

An axis-parallel classifier might provide one of the following solutions presented in the figures 3, 4, 5 or 6, where rules are not overlapped.

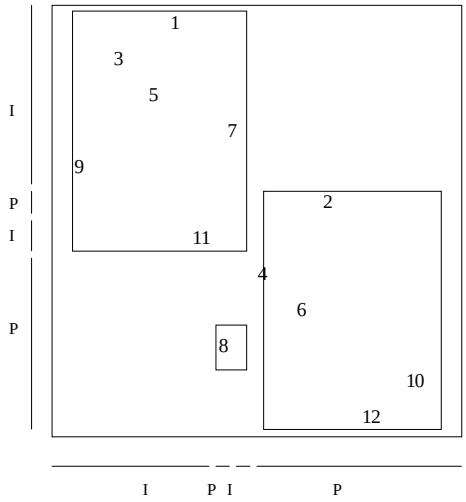

Figure 3. One possible solution.

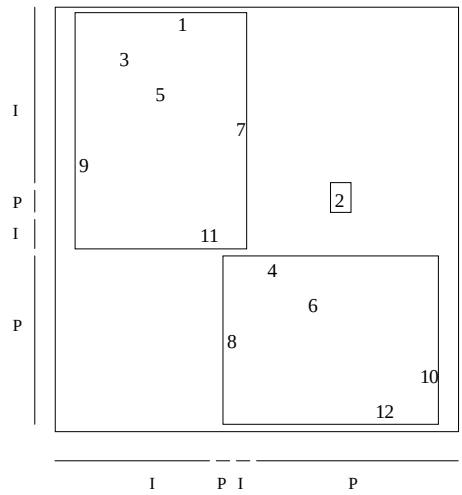

Figure 4. One possible solution.

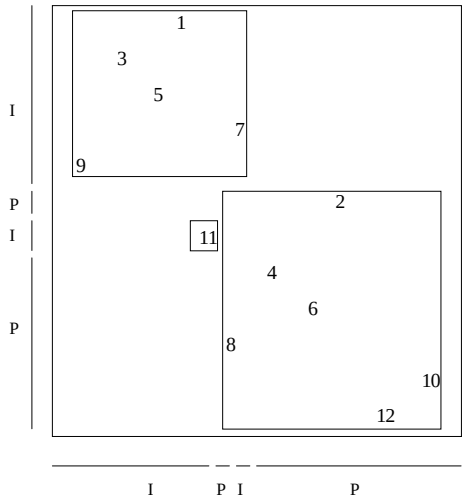

Figure 5. One possible solution.

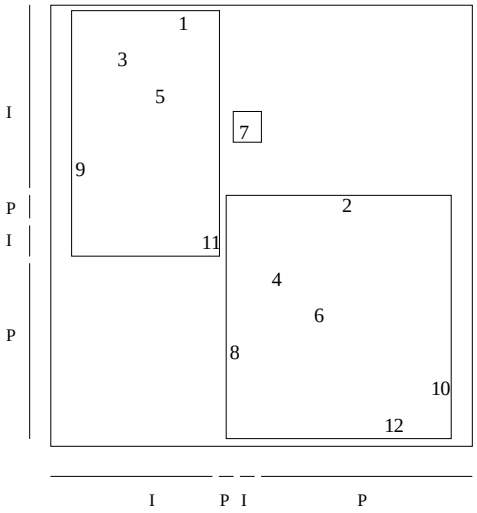

Figure 6. One possible solution.

Before formally exposing the algorithm, we will briefly explain its application. Consider the situation depicted in figure 7: the projection of the examples on the abscissas axis produce four ordered sequences {I, P, I, P} corresponding to the examples {[9, 3, 5, 1, 11], [8], [7], [4, 6, 2, 12, 10]}. Identically, with the projection on the ordinates axis we can obtain the sequences {P, I, P, I} formed by the examples {[12, 10, 8, 6, 4], [11], [2], [9, 7, 5, 3, 1]}. Each sequence represents a rectangular region as possible solution of a classifier and the initial and final examples of the sequence (if it has only one, it is simultaneously the initial and the final one) represent the lower and upper values for each coordinate of this rectangle. For example, in the figure 5 the three rectangles are defined by examples [9, 1, 7], [11] and [8, 2, 12, 10]. Finally, we must do note that the examples 3, 5 and 6 are never the initial or final examples of any sequence. Therefore, they will be candidates to be eliminated because they will never be boundaries. The idea is best understood analysing the non-empty regions obtained by means of projections on every axis, as shows the figure 7 and deleting the examples that are not relevant to establish the boundaries of a rule (figure 8).

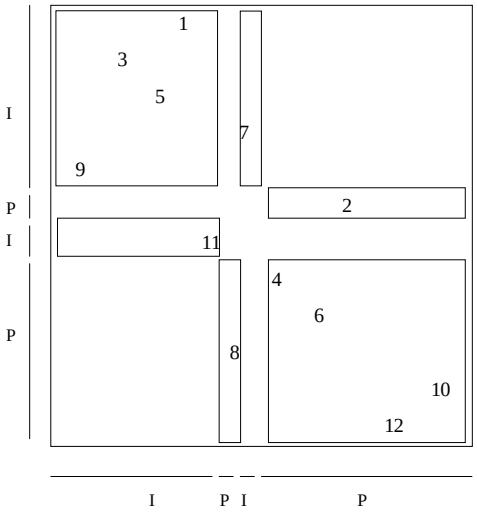

Figure 7. Regions without overlapping on the projection.

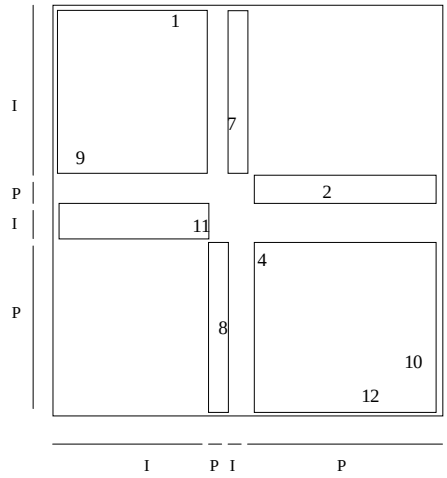

Figure 8. Result of applying EOP.

2.1 Definitions

Def 1: An example E is a tuple formed by the Cartesian product of the set A of attributes A_i and the set of labels C. We define the operations *att* and *lab* to access to the attributes and its label:

$$att : E \times N \rightarrow A_i \text{ and } lab : E \rightarrow C .$$

Def 2: Let the universe U be a sequence of examples. We will say that a database with n examples, each of them with m attributes and a label, forms a particular universe. Then U=<u[1], ..., u[n]>. Since the database is a sequence, the access to an example is achieved by means of its position. Likewise, the access to j-th attribute of the i-th example is made by att(s[i],j), and for knowing its label lab(s[i]).

Def 3: Two disjoint subsequences of examples, randomly generated that contain approximately 70% and 30% of the universe, respectively, have been called training and test.

Def 4: An ordered projected sequence is a sequence formed by the projection of the universe onto the i-th attribute. This sequence is sorted and it contains the numbers of the examples. For example, in figure 1, for the first attribute we have {9, 3, 5, 1, 11, 8, 7, 4, 6, 2, 12, 10} and for the second attribute {12, 10, 8, 6, 4, 11, 2, 9, 7, 5, 3, 1}.

Def 5: A partition in subsequences of the ordered sequence is the set of subsequences formed from the ordered sequence of an attribute in such a way to maintain the projection order, group the examples with the same label and every two consecutive partitions are disjoint. In the figure 1, we have for the first attribute {[9, 3, 5, 1, 11], [8], [7], [4, 6, 2, 12, 10]} and for the second attribute {[12, 10, 8, 6, 4], [11], [2], [9, 7, 5, 3, 1]} (figure 7).

Def 6: If an example is in the left or right extreme of a partition, the example is called border. If the partition only have one example, it is border. The remainder are not border, are inner. For example, in the partition obtained in the previous definition the examples 9, 11, 8, 4 and 10 are borders for the first attribute.

Def 7: The weakness of an example is defined as the number of times that this example is not border in a partition (i. e., it is inner to a partition) for every partition obtained from ordered projected sequences of each attribute.

In the previous example, let {[9, 3, 5, 1, 11], [8], [7], [4, 6, 2, 12, 10]} and {[12, 10, 8, 6, 4], [11], [2], [9, 7, 5, 3, 1]} be the partitions, then the weakness of each example is given by:

weakness=0 ⇒ examples {9, 11, 4}
weakness=1 ⇒ examples {1, 8, 7, 2, 12, 10}
weakness=2 ⇒ examples {3, 5, 6}

Def 8: Those examples whose weakness is equal to the number of attributes of the database are called irrelevant. In our example, there are three irrelevant examples: {3, 5, 6}, that they do not appear in the solution (figure 8).

2.2 Algorithm

Input: E: training file (N examples, M attributes)
Output: E edited training file (N* examples)

For each example e_j of E with j in {1,...,N}
    weakness(e_j) <-- 0
For each attribute i in {1,..., M}
    E_i <-- Sort E in increasing order by attribute i
    For each example e_j of E_i with j in {1,...,N}
        If e_j is not border
            weakness(e_j) <-- weakness(e_j)+1
For each example e_j of E with j in {1,...,N}
    If weakness(e_j)=M
        remove e_j from E

The computational cost of the algorithm is O(MNlogN), much lower than other algorithms proposed in the bibliography.

3 EXPERIMENTS

Tests have been achieved over four databases of varying complexity from UCI repository: Iris (150, 4, 3), Cancer (683, 9, 2), Pima (768, 8, 2) and Letter (20.000, 16, 26).

The experiments follow the next steps for each database:

- Random generation of five tests. Each test is composed by a training and test files, approximately with 70% and 30%, respectively.
- Application of C4.5 to each training/test files.
- Application of EOP to each training file E (not to test), obtaining E* editing training file.
- Application of C4.5 to each editing training file, testing with the original test files.
- Analysis of training file reduction in relation to decision tree size and error rate.

It is very important to point out that we could have obtained the tree size and the error rate by applying C4.5 before and after EOP, without using test files. The results would have been better in that case. However, the experiments designed are using the same test file in each execution. The next figure 9 shows the procedure.

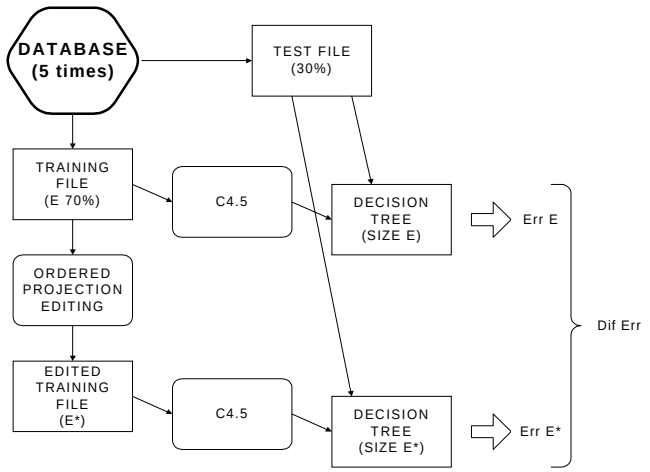

Figure 9. Applying EOP and C4.5.

Therefore, the idea consists of trying to maintain the knowledge that there was in the original training file into the reduced training file. This purpose is proved by testing both training files.

The results are shown in the next table. The meaning of the columns is as follows:

- E: size of training files (70% original database size).
- E*: size of edited training file (DB+EOP).
- %Red: reduction of E in comparison to E*.
- Size E: decision tree size, generated by C4.5 from E.
- Size E*: decision tree size, generated by C4.5 from E*.
- %Dec Size: reduction percentage of decision tree size.
- %Err E: error rate provided by the decision tree of E from the original test file.
- %Err E*: error rate provided by the decision tree of E* from the original test file.
- Dif Err: difference between both error rates: |E*-E|.

The analysis of the results indicates that:

- Iris (150, 4, 3) is reduced more than 27%, although sizes and error rates are the same.
- Breast Cancer (699, 9, 2) is reduced more than 72%, the tree size is 30% smaller, although the error rate is 1.2 greater.
- Pima (768, 8, 2) is a known database because its outliers. Only 12.4% of reduction is reached and 2.8% of decrease in the decision tree size. Surprisingly the error rate is improved.
- Letter (20000, 16, 26) is reduced more than 59% and the decision tree size 32%. The error rate is worse, because it is 5.7 greater. However, in this case it is important to note that the edited training file (±5700) is smaller than the original test file (±6000).

Table 1. Results.

DataBase	*E*	*E**	*%Red*	*Size E*	*Size E**	*%Dec Size*	*%Err E*	*%Err E**	*Dif Err*
Iris	107	77.2	27.8	9.4	9.4	+0.0	7.0	7.0	+0.0
Breast Cancer	475.6	132.2	72.2	30.2	21	+30.5	4.3	5.5	+1.2
Pima Diabetes	535.8	469.2	12.4	143.4	139.4	+2.8	30.3	29.6	-0.7
Letter	14041	5745.8	59.1	2252.6	1518.6	+32.6	14.1	19.8	+5.7

4 CONCLUSIONS

We show an editing algorithm (EOP: Editing by Ordered Projection). Its main application is as a preprocessing method for axis-parallel classifiers (like C4.5). EOP has an important characteristic: it does not need distance calculations and, therefore, it is not necessary to define it. NN-based techniques need to initially set some parameters, EOP does not. The computational cost is lower than other methods O(M N log N). The test set has been realised with four different databases from UCI repository and the results are very interesting which show that our algorithm is a robust method to reduce databases for studying other learning algorithms without loosing decision boundaries. EOP is deterministic; it is neither dependent on random values nor the order of example processing.

At the same time, we are presenting a measure, named weakness for an example, which can help to determine the importance of an example as decision boundary. More weakness implies less relevance. Thus, in more complicated databases we could relax the reduction factor for eliminating which weakness is greater or equal than M-i, instead of M, as it is used in the algorithm above.

We are studying now the possibility of applying it to discrete attributes.

5 ACKNOWLEDGEMENTS

This work has been supported by the Spanish research agency CICYT under grant TIC99-0351.

6 REFERENCES

[1] Blake, C. and Merz, E. K. UCI Repository of machine learning databases. (1998).

[2] Cover, T. y Hart, P. Nearest Neighbor Pattern Classification. IEEE Transactions on Information Theory, 13. (1967).

[3] Hart, P.E. The Condensed Nearest Neighbor Rule. IEEE Transactions on Information Theory, IT-14. (1968).

[4] V. Klee. On the complexity of d-dimensional Voronoi diagrams. Arch. Math., vol. 34, pp 75-80. (1980).

[5] Quinlan, J. C4.5: Programs for Machine Learning. Morgan Kaufmann, Publishers, San Mateo, California. (1993).

[6] Ritter, G. L., Woodruff, H.B., Lowry, S.R. y Isenhour, T.L. An algorithm for a Selective Nearest Neighbor Decision Rule. IEEE Transactions on Information Theory, 21. (1975).

[7] Tomek, I. An Experiment with the Edited Nearest-Neighbor Rule. IEEE Transactions on Systems, Man, an Cybernetics SMC-6. (1976).

[8] G. T. Toussaint. The relative neighbourhood graph of a finite planar set. Pattern Recognition, vol. 12, nº 4, 1980, pp. 261-268. (1980).

[9] Wilson, D. Asymptotic Properties of Nearest Neighbor Rules using Edited Data. IEEE Transactions on Systems, Man and Cybernetics 2. (1972).