A Measure for Data Set Editing by Ordered Projections

Jesús S. Aguilar-Ruiz, Juan A. Nepomuceno,
Norberto Díaz-Díaz, and Isabel Nepomuceno

Bioinformatics Group of Seville,
Pablo de Olavide University and University of Seville, Spain
direscinf@upo.es, {janepo, ndiaz, isabel}@lsi.us.es

Abstract. In this paper we study a measure, named *weakness of an example*, which allows us to establish the importance of an example to find representative patterns for the data set editing problem. Our approach consists in reducing the database size without losing information, using algorithm patterns by ordered projections. The idea is to relax the reduction factor with a new parameter, λ, removing all examples of the database whose weakness verify a condition over this λ. We study how to establish this new parameter. Our experiments have been carried out using all databases from UCI-Repository and they show that is possible a size reduction in complex databases without notoriously increase of the error rate.

1 Introduction

Data mining algorithms must work with databases with tends of attributes and thousands of examples when they are used to solve real and specific problems. This kind of databases contain much more information than standard databases, most of them of small size, which are usually used to testing data mining techniques. A lot of time and memory size are necessary to accomplish the final tests on these real databases.

Methodologies based on axis-parallel classifiers are classifiers that provide easy-to-understand decision rules by humans and they are very useful for the expert interest in getting knowledge from databases. These methodologies are the most common among all methodologies used by data mining researchers. If we want to apply one of this type of tools, as C4.5 or k-NN [9], to solve a real problem with a huge amount of data, we should use some method in order to decrease the computational cost of applying these algorithms.

Databases preprocessing techniques are used to reduce the number of examples or attributes as a way of decreasing the size of the database with which we are working. There are two different types of preprocessing techniques: **editing** (reduction of the number of examples by eliminating some of them or finding representatives patterns or calculating prototypes) and **feature selection** (eliminating non-relevant attributes).

Editing methods are related to the nearest neighbours (NN) techniques [4]. For example, in [5], Hart proposed to include in the set of prototypes those examples whose classification is wrong using the nearest neighbour technique; in this way, every member of the main set is closer to a member of the subset of prototypes of the same class than a member of a different class of this subset. In [2] a variant of the previous method is proposed. In [15], Wilson suggests to eliminate the examples which are incorrectly classified with the k-NN algorithm, the works of [13] and [11] follows the same idea. Other variants are based on Voronoi diagrams [7], for example: *Gabriel neighbours* (two examples are Gabriel neighbours if their diametrical sphere does not contain any other examples) or *relative neighbours* [14] (two examples p and q are relative neighbours if for all example x in the set, the following expression is true: $d(p,q) < max\{d(p,x), d(x,q)\}$).

In all previous methods the distances between examples must be calculated, so that, if we are working with n examples with m attributes, the first methods takes a time of $\Theta(mn^2)$, the method proposed in [11] takes $\Theta(mn^2 + n^3)$ and $\Theta(mn^3)$ the methods based on Voronoi diagrams.

We work in this paper in the line proposed by Aguilar-Riquelme-Toro [1], where a first version of editing method by ordered projection technique was introduced. This algorithm works well with continuous attributes. In [10], a second and more elaborated version of this algorithm is proposed and it works simultaneously with continuous and discrete attributes (i.e., nominal attributes) and it conserves the properties of the initial approach. Working with NN-based techniques implies to introduce some initial parameters and defining a distance to calculate the proximity between the different examples of the database. This new method based on ordered projection technique does not need to define any distance and it works with each attribute independently as we will see in the next section. The most important characteristic of this approach to the editing techniques, in addition to the absence of distance calculations, are: the considerable reduction of the number of examples, the lower computational cost $\Theta(mn \log n)$ and the conservation of the decision boundaries (especially interesting for applying classifiers based on axis-parallel decision rules). We are interesting in a measure, the **weakness** of an example, which help us to determine the importance of an example as decision boundary: more weakness implies less relevance. We propose a relaxation to the projection approach eliminating those examples whose weakness is larger than a threshold using a new parameter, λ, in the algorithm.

At the present time some authors think that editing methods are rather old-fashioned because by today's standard technology (even thought today´s data sets are larger) it is not clear whether it is worthwhile to spend the pre-processing time to perform editing. That is why some methods which embedded approaches for (simultaneous) feature selection (or editing) and classification, as SVMs [8], are being used. We are interesting in the study how to relax the projection approach to the editing problem in order to combine this new measure with the parameter of a similar approach to feature selection (eliminating non-relevant attributes), see [12]. A good method (as a new *theory of measure* to preprocessing

techniques) to find out the threshold which reduce the number of examples and the number of attributes without losing information in huge databases, would be a great achievement.

In this paper, we show that in more complicated databases we can relax the reduction factor eliminating those examples whose weakness verify a condition over λ. We have dealt with two different databases of the UCI repository [3] (University of California at Irvine), *heart-statlog* database and *ionosphere* database. k-NN (for $k = 1$) and C4.5 classifiers have been used to classify our database before and after applying our editing method POP_λ (patterns by ordered projections). The condition over the weakness of each example has been relaxed gradually in order to study the importance of this measure and the goodness of our method being applied to algorithms based on axis-parallel classifiers. After having determined the threshold using the λ parameter, we carry out the study of it over the different databases of the UCI repository with continuos attributes [3]. A ten-fold cross-validation has been used for each database.

2 Description of the Algorithm

A database with m attributes and n examples can be seen as a space with m dimensions, where each example takes a value in the rank of each attribute and it has a determined class associated. Each attribute represents an axis of this space, with n points or objects inside and each example has a particular label associated with its corresponding class. For example, if our database has two attributes, we will be in a two-dimensional space (attributes are represented by x and y axis respectively) see Figure 1.

As we said in the previous section, our method does not need to define any distance to compare the different examples, we will work with the projection of each example over each axis of the space.

The main idea of the algorithm is the following: if the dimension of the space is d, in order to locate a region (in the context of the example region means hyper-rectangle although our algorithm will work with any hyperplane) of this space with examples with the same class, we will need only $2d$ examples which will define the borders of this region; for example, in order to define a squared in $\mathbb{R}^2$ we only need four points. So that, if we have more than $2d$ examples in the region with the same class, we can eliminate the rest which are inside. Our objective will be to eliminate examples which are not in the boundaries of the region. The way of finding if an example is inner to a region will be studding if it is inner in each corresponding interval in the projection of the region over the different axis of the space.

An *ordered projected sequence* is the sequence formed by the projection of the space onto one particular axis, i.e., a particular attribute. A *partition* is a subsequence formed from one ordered projection sequence which maintains the projection ordered.

We define the ***weakness*** of an example as the number of times that an example is not a border in a partition (i.e., it is inner to a partition) for every

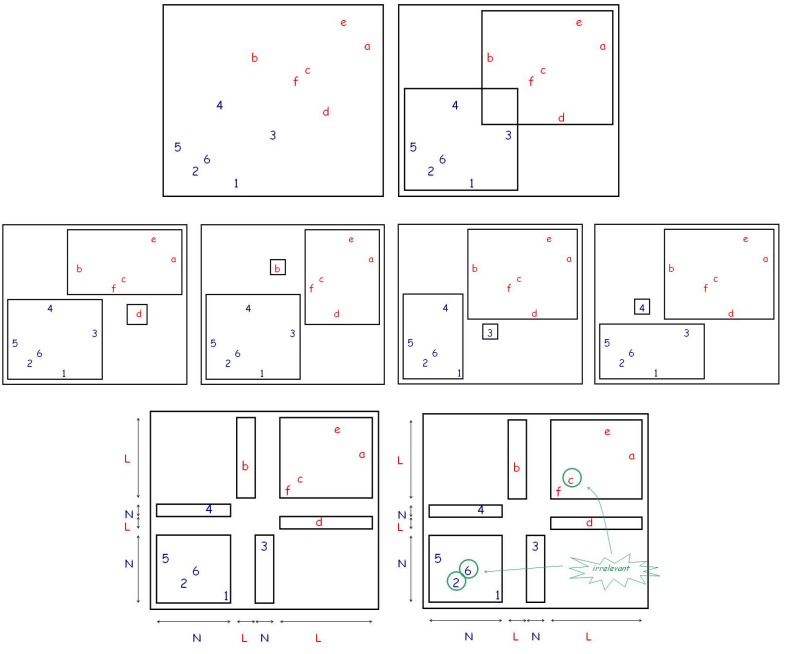

Fig. 1. A two-dimensional database with twelve examples

partition obtained from ordered projected sequences of each attribute. We call ***irrelevant examples*** those examples whose weakness is equal to the number of attributes.

In order to illustrate the method, we have designed a simple two–dimensional labelled database. This database is depicted in Figure 1, picture 1, and it contains twelve examples from two different class: N (numbers) and L (letters). An optimal classifier would obtain two rules, examples with *numbers* label and *letters* label, see picture 2 in the Figure 1, with overlapped rules. The classifier must be hierarchical because it produces overlapped rules. This is no the case of an axis parallel classifier which does not produce overlapped rules. For example, C4.5 and many others similar classifiers would produce situations like we can see in picture 2, 3, 4 and 5 in Figure 1.

The aim of our algorithm is to built regions containing all the examples of the same class and to eliminate those examples which are not necessaries to define the regions, that is, those examples which are not in the borders of the regions. If we consider the situation depicted in picture number 7 in Figure 1, each region only contains examples of the same class in a maximal way. The projection of the examples on the abscissa axis, for the first attribute, it will produce four ordered sequences $\{N, L, N, L\}$ corresponding to {[5, 2, 6, 4, 1], [b], [3], [f, c, d, e, a]}. Respectively on the ordinates axis, will produce the sequences $\{N, L, N, L\}$ formed by the examples {[1, 2, 6, 5, 3], [d], [4], [f, c, b, a, e]}. Each sequence represents a rectangular region as possible solution of a classifier and 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. In this situation, 5 and 1 are border for the first attribute.

According to this figure, the weakness of each examples would be 0 to examples '1', '3' and 'f'; 1 to '4', 'd', 'e', '5', 'b' and 'a'; and 2 to example '2', '6' and 'c'. Last examples have weakness equal to the dimension, therefore they are not necessaries to define the subregions, they are irrelevant examples. So, they are removed from the database, see picture 8 in the Figure 1.

2.1 Algorithm

Given the database E, let be n and n' the initial and the final number of examples ($n \geq n'$), let be m the number of attributes and let be $\lambda \in \mathbb{R}$ the initial parameter to relax the measure of the weakness. The POP_λ-algorithm (algorithm for patterns by projections ordered) is the following:

.....

Procedure POP_λ (in: ($E_{n\times m}$, λ), out: $E_{n'\times m}$)
for each example $e_i \in E$,$i \in \{1,\ldots,n\}$
$weakness(e_i) := 0$
end for
for each attribute a_j, $j \in \{1,\ldots,m\}$
$E_j := QuickSort(E_j, a_j)$ in increasing order
$E_j := ReSort(E_j)$
for each example $e_i \in E_j$, $i \in \{1,\ldots,n\}$
if e_i is nor border
$weakness(e_i) := weakness(e_i) + 1$
end if
end for
end for
for each example $e_i \in E$, $i \in \{1,\ldots,n\}$
if $weakness(e_i) \geq m \cdot \lambda$
remove e_i from E
end if
end for
end POP_λ

.....

The computational cost of POP is $\Theta(mn \log n)$. This cost is much lower than other algorithms proposed in the bibliography, normally $\Theta(mn^2)$.

The algorithm constructs the ordered projected sequence over each axis of the space (attribute) and it calculates the weakness for each example. The value of the projections need to be sorted when we are working with each attribute. We use QuickSort algorithm, [6], and a second sorting, we call it Resort, is made in order to create regions containing examples of the same class in a maximal way. The examples sharing the same value for an attribute are not necessary

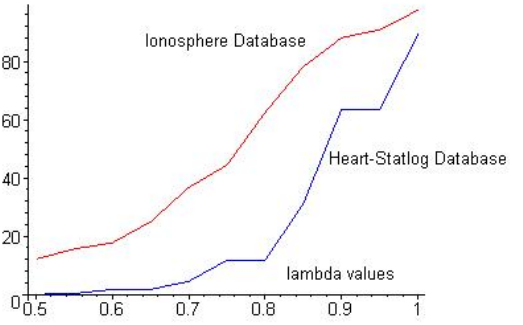

Fig. 2. Applying POP_λ over two different databases. Ordinate axis shows the *percentage of retention*. In abscissa axis different values of the λ parameter.

nearer to those examples that have the same class and have another value. The solution to that problem consists of resorting the interval containing repeated values. The heuristic is applied to obtain the least number of changes of class. Therefore, the algorithm sort by value and, in case of equality, by class (*Resort* sorting). In the case of working with a database with nominal attributes, an other more elaborated version of this kind of algorithm could be considered: discrete attributes does not need to be sorted and the weakness of all the examples except one which has the least weakness obtained for the continuous attributes is increased. We are not interesting in this first approach in database with nominal attributes. Finally, examples verifying the condition over λ parameter are eliminated of the database. This parameter permit us to control the level of reduction of our editing method.

3 Experiments

Our tests have been achieved over two different databases: *heart-statlog* database and *ionosphere* database, both obtained from the UCI repository [3]. The main objective is to compare the performance of our editing method when the λ parameter is modified. We have a measure for each example, the weakness of the example, which determines its importance as a decision boundary, we relate this measure with the parameter of our editing algorithm. Our objective is to study how to establish a threshold to eliminate examples of the database, we want to determinate a parameter.

A ten-fold cross-validation is made dividing the database in ten parts and taking blocks of nine parts which are our training set and the other one is the test set. We apply our reducing method to the training set and then, after having applied the corresponding classifier algorithm, we use the test set to validate the process. This operation is realized ten times each one with the different ten subset we have built.

Table 1. Computational Cost in Seconds and Error Rate (*average - standard deviation*) for C4.5 and k-NN (with $k = 1$) algorithms over the different databases obtained with POP_λ

	C4.5				**k-NN**			
	Heart-Statlog		**Ionosphere**		**Heart-Statlog**		**Ionosphere**	
	CCS	ER $\pm\sigma$	CCS	ER $\pm\sigma$	CCS	ER $\pm\sigma$	CCS	ER $\pm\sigma$
Original	0.08	21.7 $\pm$6.6	0.16	14.3 $\pm$7.9	0.09	24.6 $\pm$8.1	0.06	15.4 $\pm$8.2
POP$_{\lambda=1}$	0.08	24.0 $\pm$6.0	0.18	14.3 $\pm$7.9	0.05	24.8 $\pm$8.6	0.05	16.3 $\pm$8.8
POP$_{\lambda=0.95}$	0.13	20.8$\pm$10.8	0.15	15.4 $\pm$8.7	0.05	28.4 $\pm$5.9	0.05	16.8 $\pm$9.3
POP$_{\lambda=0.90}$	0.08	20.8$\pm$10.8	0.14	14.8 $\pm$7.7	0.05	28.4 $\pm$5.9	0.05	17.1 $\pm$9.4
POP$_{\lambda=0.85}$	0.05	22.1 $\pm$6.7	0.14	12.0 $\pm$7.2	0.03	24.9$\pm$11.4	0.06	17.1 $\pm$9.4
POP$_{\lambda=0.80}$	0.03	27.4$\pm$11.4	0.12	14.3 $\pm$6.8	0.02	39.4 $\pm$8.8	0.05	16.3 $\pm$8.9
POP$_{\lambda=0.75}$	0.04	37.4$\pm$11.4	0.09	17.7 $\pm$8.1	0.01	39.4 $\pm$8.8	0.05	15.4 $\pm$8.2
POP$_{\lambda=0.70}$	0.03	40.3$\pm$14.8	0.08	25.1$\pm$14.5	0.01	45.4 $\pm$9.5	0.04	15.7 $\pm$8.5
POP$_{\lambda=0.65}$	0.02	42.0$\pm$14.2	0.06	30.2$\pm$11.7	0.00	41.3 $\pm$9.5	0.04	19.7$\pm$11.5
POP$_{\lambda=0.60}$	0.02	42.0$\pm$14.2	0.04	47.2$\pm$13.7	0.00	41.3 $\pm$9.5	0.04	24.8$\pm$11.2
POP$_{\lambda=0.55}$	0.01	43.0$\pm$15.8	0.04	58.3$\pm$16.6	0.00	46.6 $\pm$6.5	0.03	44.1 $\pm$9.5
POP$_{\lambda=0.50}$	0.01	9.3$^*\pm$18.5	0.04	53.4$\pm$17.0	0.00	42.9 $\pm$7.1	0.03	51.2 $\pm$7.9

We apply POP_λ algorithm with $\lambda \in \{1, 0.95, 0.9, 0.85, 0.8, 0.75, 0.7, 0.65, 0.6, 0.55, 0.5\}$. Figure 2 shows the percentage of retention of our method varying with the different values of λ (100% means that the database has not any reduction)[1]. We notice that both functions are increasing, that is because we have imposed the condition " $weakness(e_i) \geq m \cdot \lambda$" in the algorithm and for each new valor of the parameter, we remove the new examples that verifies the condition and the old ones. If we have put $=$ instead $\geq$, the graphic would not be increasing and the process would not be cumulative. If a group of examples with the same weakness is removed for a value of λ, they will be removed when the parameter decreases. We are interested in relaxing the condition over λ in order to remove examples from the database gradually. A possible threshold could be establish seeing the graphics but we must verify that we do not lose any knowledge in order to make the classification phase.

The results of the classifications using C4.5 and k-NN techniques are shown in Table 1. We presents the CCS, computational cost in seconds, and the ER, error rate, for the original database and the different reduced databases. ER is the classification error produced when the test set validates the model that we have constructed applying the different techniques to the different databases. Values are the average and the standard deviation of complete 10-fold cross-validation (the sum of the 10 experiments for each value obtained). We can observe that the computational cost is decreasing while the lambda value is decreasing too. So, if we found a lambda value less than 1 without losing information, we would manage reduce the database size and the computational cost would decreased too.

[1] Percentages of retention for each value of λ are the average of the ten different experiments realized in the 10-fold CV.

The purpose is to study the relevance of POP_λ as a reduction method for the different values of λ. The best situation could be getting the λ which produces the greatest reduction in our database and the least error rate when the classification methods is applied. We observe between $\lambda = 0.85$ and $\lambda = 0.80$ a possible threshold: for $\lambda = 0.80$ the number of examples are removed dramatically from the database and the error rate seems to increase. We have a good candidate to be our threshold. We must verify it and we must study how to establish the valour of the parameter.

In order to proof the goodness of our parameter, in Table 2, we carry out the study of the situation for this two values over all the databases from UCI repository with continuous attributes [3]. We show the percentage of reduction, PR, in order to indicate the number of examples which are removed from the database, values are the average of complete 10-fold cross-validation process. We must consider how the error changes when the database is reduced considerably. Our aim is to ascertain the value of λ which reduce more the database without losing information. For example, for Heart–Statlong, the Error Rate from the original database using C4.5 is 21.7 ± 6.6, but if we apply $POP_{\lambda=0.85}$ (Table 2) the error would be only 22.1 ± 6.7. That is, we have managed to reduce the database in a 68.8% (100-PR) without losing knowledge. If we take the same database and configuration but using k-NN, similar behavior is observed. In general, for $\lambda = 0.85$, databases would be reduced to 61.8% of the original size and error rate would be incremented from 18.6 ± 4.8 to 24.8 ± 5.8 using C4.5, and from 15.4 ± 4.6 to 23.1 ± 5.0 using 1-NN. We have drawn in bold in both tables the data which are relevant according to t-Student statistical distribution.

Paying attention to results obtained with $\lambda = 0.8$ we have to say that although databases was reduced dramatically, the error rate is incremented notably. These Experiments show us how to establish an appropriate value of λ parameter in order to apply the POP_λ algorithm reducing the database up to the limit and conserving the knowledge of the original database.

In summary, we can state that with lambda values minor than 1 it is possible a higher database size reduction without losing information. But this reduction is limited to a 0.85 lambda value. We have established a method to find out a threshold to relax the reduction factor over the algorithm for finding representative patterns for dat aset editing. We have proven the goodness of our level with all the databases of the UCI repository with only continuous attributes.

4 Conclusions

We have defined a new parameter, λ, which helps us to remove all examples in a database which verify a condition over it, its "$weakness \geq$(number attributes)$\cdot\lambda$". Therefore we have established a threshold via a measure over each example in order to reduce the number of examples of a database.

After analyzing our approach using some databases from the UCI–Repository, we conclude that it is possible to reduce the database size up to a 40% without losing any knowledge. Furthermore, the computational cost is decreased by allowing to remove examples with weakness less than the number of attributes.

Table 2. Error Rate for C4.5 and k-NN (with $k = 1$) algorithms over databases from UCI repository with continuous attributes. Every database is considered before and after applying POP_λ for $\lambda = 0.85$, for the first table, $\lambda = 0.80$ for the second one. PR is the percentage of reduction of the reduction algorithm.

	C4.5			**k-NN**		
	Original	$POP_{\lambda=0.85}$		Original	$POP_{\lambda=0.85}$	
Data Base	ER $\pm\sigma$	ER $\pm\sigma$	PR	ER $\pm\sigma$	ER $\pm\sigma$	PR
Heart-Statlog	21.7 $\pm$6.6	22.1 $\pm$6.7	31.2	24.6 $\pm$8.1	**24.9 $\pm$ 11.4**	31.2
Ionosphere	14.3 $\pm$7.9	12.0 $\pm$7.2	78.2	15.4 $\pm$8.2	17.1 $\pm$9.4	78.2
Balance-Scale	25.4 $\pm$7.2	**66.0 $\pm$ 17.7**	10.8	20.2 $\pm$5.6	**54.0 $\pm$ 13.1**	10.8
Breast-W	5.2 $\pm$2.7	**17.0 $\pm$ 9.9**	5.2	5.2 $\pm$4.7	**23.2 $\pm$ 7.8**	5.2
Bupa	15.4 $\pm$0.0	**42.0 $\pm$ 0.0**	42.0	0.0 $\pm$0.0	**22.6 $\pm$ 0.0**	42.0
Diabetes	26.4 $\pm$7.3	28.1 $\pm$7.7	55.3	29.6 $\pm$4.0	**35.0 $\pm$ 5.8**	55.3
Glass	51.3 $\pm$19	51.4 $\pm$19	97.6	39.8 $\pm$12	39.8 $\pm$12	97.6
Iris	2.0 $\pm$0.0	**12.7 $\pm$ 0.0**	26.0	0.0 $\pm$0.0	**3.3 $\pm$ 0.0**	26.0
Lung-Cancer	15.6 $\pm$0.0	**9.4 $\pm$ 0.0**	96.0	0.0 $\pm$0.0	**3.1 $\pm$ 0.0**	96.0
Page-Blocks	1.5 $\pm$0.0	**9.3 $\pm$ 0.0**	13.0	0.3 $\pm$5.6	**23.5 $\pm$ 3.6**	13.0
Pima-Indians-Diabetes	15.9 $\pm$0.0	**23.6 $\pm$ 0.0**	58.0	0.0 $\pm$0.0	**11.9 $\pm$ 1.8**	58.0
Segment	3.3 $\pm$1.3	3.8 $\pm$1.5	92.0	3.2 $\pm$1.6	3.2 $\pm$1.6	92.0
Sonar	45.5$\pm$19.3	45.5$\pm$19.3	100.0	49.4$\pm$16.1	49.4$\pm$16.1	100.0
Vehicle	27.8 $\pm$3.4	28.2$\pm$4.6	88.1	31.9 $\pm$4.6	31.8 $\pm$4.4	88.1
Waveform-5000	25.1 $\pm$1.5	25 $\pm$1.8	97.1	26.8 $\pm$1.5	26.8 $\pm$1.6	97.1
Wine-5000	1.1 $\pm$0.0	1.1 $\pm$0.0	98.0	0.0 $\pm$0.0	0.0 $\pm$0.0	98.0
Average	18.6 $\pm$4.8	24.8 $\pm$5.8	61.8	15.4 $\pm$4.6	23.1 $\pm$5.0	61.8

	C4.5			**k-NN**		
	Original	$POP_{\lambda=0.80}$		Original	$POP_{\lambda=0.80}$	
Data Base	ER $\pm\sigma$	ER $\pm\sigma$	PR	ER $\pm\sigma$	ER $\pm\sigma$	PR
Heart-Statlog	21.7 $\pm$6.6	**27.4 $\pm$ 11.4**	11.7	24.6 $\pm$8.1	**39.4 $\pm$ 8.8**	11.7
Ionosphere	14.3 $\pm$7.9	14.3 $\pm$6.8	62.5	15.4 $\pm$8.2	16.3 $\pm$8.9	62.5
Balance-Scale	25.4 $\pm$7.2	**6.0 $\pm$ 17.8**	10.8	20.2 $\pm$5.6	**54.0 $\pm$ 13.2**	10.8
Breast-W	5.2 $\pm$2.7	**50.4 $\pm$ 29**	1.3	5.2 $\pm$4.7	**34.9 $\pm$ 9.0**	5.2
Bupa	15.4 $\pm$0.0	**42.0 $\pm$ 0.0**	42.0	0.0 $\pm$0.0	**22.6 $\pm$ 0.0**	42.0
Diabetes	26.4 $\pm$7.3	**49.5 $\pm$ 8.2**	26.4	29.6$\pm$4.0	**47.0 $\pm$ 4.1**	26.4
Glass	51.3 $\pm$19	52.6$\pm$17.5	94.2	39.8$\pm$12.0	39.8 $\pm$12	94.2
Iris	2.0 $\pm$0.0	**12.7 $\pm$ 0.0**	26.0	0.0 $\pm$0.0	**3.3 $\pm$ 0.0**	26.0
Lung-Cancer	15.6 $\pm$0.0	**12.5 $\pm$ 0.0**	90.0	0.0 $\pm$0.0	**9.4 $\pm$ 0.0**	90.0
Page-Blocks	1.5 $\pm$0.0	**9.3 $\pm$ 0.0**	13.0	0.3 $\pm$5.6	**23.6 $\pm$ 0.0**	13.0
Pima-Indians-Diabetes	15.9 $\pm$0.0	**48.8 $\pm$ 0.0**	28.0	0.0 $\pm$0.0	**32.2 $\pm$ 0.0**	13.0
Segment	3.3$\pm$1.4	4.1 $\pm$1.5	87.0	3.2 $\pm$1.6	3.7 $\pm$1.5	87.0
Sonar	45.5$\pm$19.3	45.5$\pm$19.3	100.0	49.4$\pm$16.1	49.4$\pm$16.1	100.0
Vehicle	27.8 $\pm$3.4	29.0 $\pm$6.4	79.6	31.9 $\pm$4.6	32.8 $\pm$5.3	79.6
Waveform-5000	25.1 $\pm$1.5	25.1 $\pm$1.6	92.6	26.8 $\pm$1.5	26.9 $\pm$1.6	92.6
Wine-5000	1.1 $\pm$0.0	1.1 $\pm$0.0	96.6	0.0 $\pm$0.0	0.0 $\pm$0.0	96.6
Average	18.6 $\pm$4.8	30.6 $\pm$7.5	53.9	15.4 $\pm$4.6	27.2 $\pm$5.0	53.9

In spite of having introduced a new parameter and treating a problem of editing (some authors consider editing is also rather old-fashioned because it is worthwhile to spend the pre-processing time with today´s standard technologies), this paper begins a way to consider the preprocessing problem, such editing as features selection, as a problem of election of two parameters. As a future work, the combination of POP_λ-algorithm with SOAP-algorithm [12] is proposed. Thus we will obtain an algorithm to preprocess a database working with two parameters in order to remove such examples as attributes.

References

1. Aguilar, J.S.; Riquelme, J.C.; Toro, M.: Data set editing by ordered projection, in: Proceedings of the 14th European Conference on Arti2cial Intelligence (ECAI'00), Berlin, Germany, (2000), pp. 251-255.
2. Aha, D.W.; Kibler, D.; Albert, M.K.: Instance-based learning algorithms, Mach. Learning 6 (1991), pp. 37-66.
3. Blake, C.; Merz, E.K.: UCI repository of machine learning databases, (1998).
4. Cover, T. and Hart, P.: Nearest Neighbor Pattern Classification. IEEE Transactions on Information Theory, IT-13 (1) (1967), pp. 21-27.
5. Hart, P.: The condensed nearest neighbor rule, IEEE Trans. Inf. Theory 14 (3) (1968), pp. 515-516.
6. Hoare, C.A.R.: Quicksort, Comput.J. 5 (1) (1962), pp. 10-15.
7. Klee, V.: On the complexity of d-dimensional voronoi diagrams, Arch. Math. 34 (1980), pp. 75-80.
8. Neumann, Julia; Schnörr, Christoph; Steidl, Gabriele: SVM-based Feature Selection by Direct Objective Minimisation Pattern Recognition, Proc. of 26th DAGM Symposium, LNCS, Springer, August (2004).
9. Quinlan, J.R.: C4.5: Programs for Machine Learning, Morgan Kauffmann, San Mateo, CA, (1993).
10. Riquelme, José C.; Aguilar-Ruiz, Jesús S.; Toro, Miguel: Finding representative patterns with ordered projections Pattern Recognition 36 (2003), pp. 1009-1018.
11. Ritter, G.; WoodruI, H.; Lowry, S.; Isenhour, T.: An algorithm for a selective nearest neighbor decision rule, IEEE Trans. Inf. Theory 21 (6) (1975), pp. 665-669.
12. Ruiz, R.; Riquelme, Jose C.; Aguilar-Ruiz, Jesus S.: NLC: A Measure Based on Projections 14th International Conference on Database and Expert Systems Applications, DEXA 2003Lecture Notes in Computer Science, Springer-VerlagPrague, Czech Republic, 1-5 September, (2003).
13. Tomek, I.: An experiment with the edited nearest-neighbor rule, IEEE Trans. Syst. Man Cybern. 6 (6) (1976), pp. 448-452.
14. Toussaint, G.T.: The relative neighborhood graph of a finite planar set, Pattern Recognition 12 (4) (1980), pp. 261-268.
15. Wilson, D.R.; Martinez, T.R.: Improved heterogeneous distance functions, J. Artif. Intell. Res. 6 (1) (1997), pp. 1-34.