Reclassification of Linearly Classified Data using Constraint Databases

Peter Revesz and Thomas Triplet

University of Nebraska - Lincoln, Lincoln NE 68588, USA,
revesz@cse.unl.edu, ttriplet@cse.unl.edu

Abstract. In many problems the raw data is already classified according to a variety of features using some linear classification algorithm but needs to be reclassified. We introduce a novel reclassification method that creates new classes by combining in a flexible way the existing classes without requiring access to the raw data. The flexibility is achieved by representing the results of the linear classifications in a linear constraint database and using the full query capabilities of a constraint database system. We implemented this method based on the MLPQ constraint database system. We also tested the method on a data that was already classified using a decision tree algorithm.

1 Introduction

Semantics in data and knowledge bases is tied to classifications of the data. Classifications are usually done by classifiers such as decision trees [7], support vector machines [10], or other machine learning algorithms. After being trained on some sample data, these classifiers can be used to classify even new data.

The reclassification problem is the problem of how to reuse the old classifiers to derive a new classifier. For example, if one has a classifier for disease A and another classifier for disease B, then we may need a classifier for patients who (1) have both diseases, (2) have only disease A, (3) have only disease B, and (4) have neither disease. In general, when combining n classifiers, there are 2^n combinations to consider. Hence many applications would be simplified if we could use a single combined classifier.

There are several natural questions about the semantics of the resultant reclassification. For example, how many of the combination classes are really possible? Even if in theory we can have 2^n combination classes, in practice the number may be much smaller. Another question is to estimate the percent of the patients who fall within each combination class, assuming some statistical distribution on the measured features of the patients.

For example, Figures 1 and 2 present two different ID3 decision tree classifiers for the country of origin and the miles per gallon fuel efficiency of cars. For simplicity, the country of origin is classified as *Europe, Japan*, or *USA*, and the fuel efficiency is classified as *low, medium*, and *high.* Analysis of such decision trees is difficult. For instance, it is hard to tell by just looking at these decision trees whether there are any cars from Europe with a low fuel efficiency.

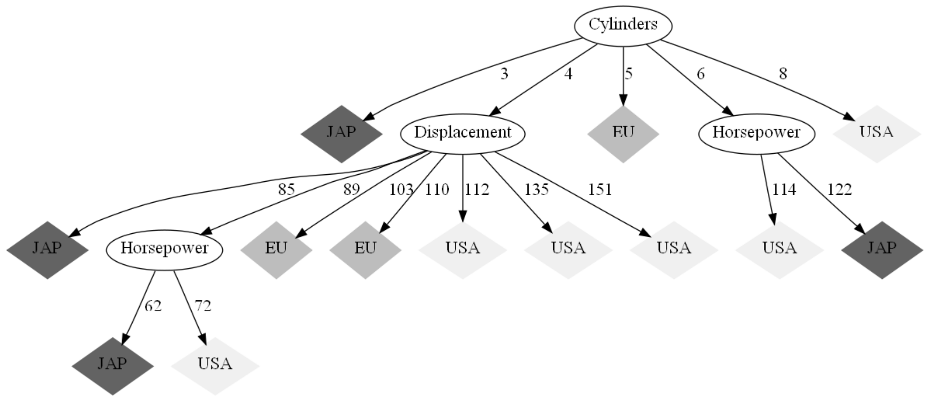

Fig. 1. Decision tree for the country of origin of cars. The tree is drawn using Graphviz [2].

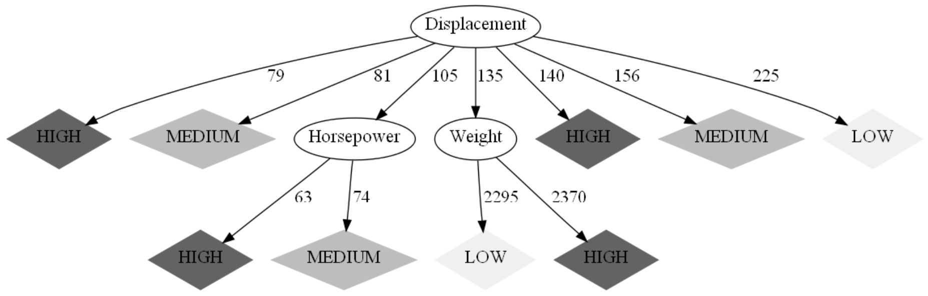

Fig. 2. Decision tree for the miles per gallon fuel efficiency of cars. The tree is drawn using Graphviz [2].

When a decision tree contains all the attributes mentioned in a query, then the decision tree can be used to efficiently answer the query. Here the problem is that the attributes mentioned in the query, that is, country/region of origin and MPG (miles per gallon) fuel efficiency, are not both contained in either decision tree. Reducing this situation to the case of a single decision tree that contains both attributes would provide a convenient solution to the query.

We propose in this paper a novel approach to reclassification that results in a single classifier and enables to answer several semantic questions. Reusability and flexibility are achieved by representing the original classifications in linear constraint databases [6, 8] and using constraint database queries.

Background and Contribution: Earlier papers by Geist [4] and Johnson et al. [5] talked about the representation of decision trees in constraint databases. However, they did not consider support vector machines (SVMs) or the reclassification problem. The reclassification problem is a special problem, and its detailed consideration and explanation of the semantic issues regarding reclassifications are important contributions of this paper. In particular, we point out that the reclassification problem is solved nicely with the use of constraint databases. Furthermore, our experiments compare several different possible approaches to the reclassification problem. The experiments support the idea that the constraint database approach to reclassification is an accurate and flexible method.

The rest of the paper is organized as follows. Section 2 presents a review of linear classifiers. Section 3 presents the reclassification problem. Section 4 describes two novel approaches to the reclassification problem. The first approach is called *Reclassification with an oracle*, and the second approach is called *Reclassification with constraint databases*. Section 5 describes some computer experiments and discusses their significance. Finally, Section 6 gives some concluding remarks and open problems.

2 Review of Linear Classifiers

The problem is the following: we want to classify items, which means we want to predict a characteristic of an item based on several parameters of the item. Each parameter is represented by a variable which can take a finite number of values. The set of those variables is called *feature space*. The actual characteristic of the item we want to predict is called the *label* or *class* of the item. To make the predictions, we use a machine learning technique called *classifier*. A classifier maps a feature space X to a set of labels Y. A linear classifier maps X to Y by a linear function.

Example 1. Suppose that a disease is conditioned by two antibodies A and B. The feature space X is $X = \{Antibody\_A, Antibody\_B\}$ and the set of labels is $Y = \{Disease, no\_Disease\}$. Then, a linear classifier is:

$$y = w_1.Antibody\_A + w_2.Antibody\_B + c$$

where $w_1, w_2 \in \mathcal{R}$ are constant weights and $c \in \mathcal{R}$ is a threshold constant. The $y \in Y$ value can be compared with zero to yield a classifier. That is,

- If $y \leq 0$ then the patient has *no_Disease*.
- If $y > 0$ then the patient has *Disease*.

In general, assuming that each parameter can be assigned a numerical value x_i, a linear classifier is a linear combination of the parameters:

$$y = f(\sum_j w_j x_j) \tag{1}$$

where f is a linear function. $w_i \in \mathcal{R}$ are the weights of the classifiers and entirely define it. $y \in Y$ is the predicted label of the instance.

Decision trees: the ID3 algorithm Decision trees, also called *active classifier* were particularly used in the nineties by artificial intelligence experts. The main reasons are that they can be easily implemented (using ID3 for instance) and that they give an *explanation* of the result.

Algorithmically speaking, a decision tree is a tree:

- An internal node tests an attribute,
- A branch corresponds to the value of the attribute,
- A leaf assigns a classification.

The output of decision trees is a set of logical rules (disjunction of conjunctions). To train the decision tree, we can use the ID3 algorithm, proposed by J.R. Quinlan *et al.* [7] in 1979 in the following three steps:

1. First, the best attribute, A, is chosen for the next node. The best attribute maximizes the information gain.
2. Then, we create a descendant for each possible value of the attribute A.
3. This procedure is eventually applied to non-perfectly classified children.

This best attribute is the one, which maximizes the information gain. The information gain is defined as follows:

$$Gain(S, A) = entropy(S) - \sum_{v \in values(A)} \frac{|S_v|}{|S|}.entropy(S_v)$$

S is a sample of the training examples and A is a partition of the parameters. Like in thermodynamics, the entropy measures the *impurity* of S, purer subsets having a lower entropy:

$$entropy(S) = -\sum_{i=0}^{n} p_i.log_2(p_i)$$

S is a sample of the training examples, p_i is the proportion of *i-valued* examples in S and n is the number of attributes.

ID3 is a greedy algorithm, without backtracking. This means that this algorithm is sensible to local optima. Furthermore, ID3 is inductively biased: the algorithm favors short trees and high information gain attributes near the root. At the end of the procedure, the decision tree perfectly suits the training data including noisy data. This leads to complex trees, which usually lead problems to generalize new data (classify unseen data). According to Occams razor, shortest explanations should be preferred. In order to avoid over-fitting, decision trees are therefore usually pruned after the training stage by minimizing $size(tree) + error\_rate$, with $size(tree)$ the number is leaves in the tree and $error_rate$ the ratio of the number of misclassified instances by the total number of instances (also equal to $1 - accuracy$).

Note: The ID3 decision tree and the support vector machine are linear classifiers because their effects can be represented mathematically in the form of Equation (1).

3 The Reclassification Problem

The need for reclassification arises in many situations. Consider the following.

Example 2. One study found a classifier for the origin of cars using

$$X_1 = \{acceleration, cylinders, displacement, horsepower\}$$

and

$$Y_1 = \{Europe, Japan, USA\}$$

where acceleration from 0 to 60 mph is measured in seconds (between 8 and 24.8 seconds), cylinders the number of cylinders of the engine (between 3 and 8 cylinders), displacement in cubic inches (between 68 and 455 cubic inches) and horsepower the standard measure of the power of the engine (between 46 and 230 horsepower).

A sample training data is shown in the table below.

Origin

Acceleration	Cylinders	Displacement	Horsepower	Country
12	4	304	150	USA
9	3	454	220	Europe

Another study found another classifier for the fuel efficiency of cars using

$$X_2 = \{acceleration, displacement, horsepower, weight\}$$

and

$$Y_2 = \{low, medium, high\}$$

where the weight of the car is measured in pounds (between 732 and 5140 lbs.).

A sample training data for the second study is shown in the table below.

Efficiency

Acceleration	Displacement	Horsepower	Weight	MPG
20	120	87	2634	medium
15	130	97	2234	high

Suppose we need to find a classifier for

$$X = X_1 \cup X_2 = \{acceleration, cylinders, displacement, horsepower, weight\}$$

and

$$\begin{aligned} Y = Y_1 \times Y_2 = \{&Europe - low, Europe - medium, Europe - high,\\ &Japan - low, Japan - medium, Japan - high,\\ &USA - low, USA - medium, USA - high\} \end{aligned}$$

Building a new classfier for (X, Y) seems easy, but the problem is that there is no database for (X, Y). Finding such a database would require a new study with more data collection, which would take a considerable time. That motivates the need for reclassification. As Section 4.1 shows, a classifier for (X, Y) can be built by an efficient reclassification algorithm that uses only the already existing classifiers for (X_1, Y_1) and (X_2, Y_2).

4 Novel Reclassification Methods

We introduce now several new reclassification methods. Section 4.1 describes two variants of the *Reclassification with an oracle* method. While oracle-based methods do not exist in practice, these methods give a limit to the best possible practical methods. Section 4.2 describes the practical *Reclassification with constraint databases* method. A comparison of these two methods is given later in Section 5.

4.1 Reclassification with an Oracle

In theoretical computer science, researchers study the computational complexity of algorithms in the presence of an oracle that tells some extra information that can be used by the algorithm. The computational complexity results derived this way can be useful in establishing theoretical limits to the computational complexity of the studied algorithms.

Similarly, in this section we study the reclassification problem with a special type of oracle. The oracle we allow can tell the value of a missing attribute of each record. This allows us to derive essentially a theoretical upper bound on the best reclassification that can be achieved. The reclassification with oracle method extends each of the original relations with the attributes that occur only in the other relation. Then one can take a union of the extended relations and apply any of the classification algorithms one chooses. We illustrate the idea behind the *Reclassification with an oracle* using an extension of Example 2 and ID3.

Example 3. First, we add a weight and an MPG attribute to each record in the *Origin* relation using an oracle. Suppose we get the following:

Origin

Acceleration	Cylinders	Displacement	Horsepower	Weight	Country	MPG
12	4	304	150	4354	USA	low
9	3	454	220	3086	Japan	medium

Second, we add a cylinders and a country attribute to each record in the *Efficiency* relation using an oracle. Suppose we get the following:

Efficiency

Acceleration	Cylinders	Displacement	Horsepower	Weight	Country	MPG
20	6	120	87	2634	USA	medium
15	4	130	97	2234	Europe	high

After the union of these two relations, we can train an ID3 decision tree to yield a reclassification as required in Example 2.

A slight variation of the *Reclassification with an oracle* method is the *Reclassification with an X-oracle.* That means that we only use the oracle to extend the original relations with the missing X attributes. For example, in the car example, we use the oracle to extend the *Origin* relation by only *weight*, and the *Efficiency* relation by only *cylinders.*

When we do that, then the original classification for *MPG* (derived from the second study) can be applied to the records in the extended *Origin* relation. Note that this avoids using an oracle to fill in the *MPG* values, which is a Y or target value. Similarly, the original classification for *country* (derived from the first study) can be applied to the records in the extended *Efficiency* relation.

The *Reclassification with an X-oracle* also is not a practical method except if the two original studies have exactly the same set of X attributes because oracles do not exist and therefore can not be used in practice.

4.2 Reclassification with Constraint Databases

The *Reclassification with Constraint Databases* method has two main steps:

Translation to Constraint Relations: We translate the original linear classifiers to a constraint database representation. Our method does not depend on any particular linear classification method. It can be an ID3 decision tree method [7] or a support vector machine classification [10] or some other linear classification method.

Join: The linear constraint relations are joined together using a constraint join operator [6, 8].

Example 4. Figure 1, which we saw earlier, shows an ID3 decision tree for the country of origin of the cars obtained after training by 50 random samples from a cars database [1]. A straightforward translation from the original decision tree to a linear constraint database does not yield a good result for problems where the attributes can have real number values instead of only discrete values. Real number values are often used when we measure some attribute like weight in pounds or volume in centiliters.

Hence we improve the naive translation by introducing comparison constraints $>, <, \geq, \leq$ to allow continuous values for some attributes.

That is, we translate each node of the decision tree by analyzing all of its children. First, the children of each node are sorted based on the possible values of the attribute. Then, we define an interval around each discrete value based on the values of the previous and the following children. The lower bound of the interval is defined as the median value between the value of the current child and the value of the previous child. Similarly, the upper bound of the interval is defined as the median value of the current and the following children. For instance, assume we have the values $\{10, 14, 20\}$ for an attribute for the children. This will lead to the intervals $\{(-\infty, 12], (12, 17], (17, +\infty)\}$.

In the following, let a be the acceleration, c the number of cylinders, d the displacement of the engine, h the horsepower, w the weight of the car, *country* the origin of the car, and *mpg* the miles per gallon of the car. We use the depth-first algorithm with the above heuristic on the cars data from [1] to generate the following MLPQ [9] constraint database:

```
Origin(a,c,d,h,country) :-  c = 3, country ='JAPAN'.

Origin(a,c,d,h,country) :-  c = 4, d > 111, country ='USA'.

Origin(a,c,d,h,country) :-  c = 4, d > 96, d <= 111, country ='EUROPE'.

Origin(a,c,d,h,country) :-  c = 4, d > 87, d <= 96, h > 67, country ='USA'.

                                   ⋮
```

Similarly, we used another decision tree to classify the efficiency of the cars. Translating the second decision tree yielded the following constraint relation:

```
Efficiency(a,d,h,w,mpg) :-  d <= 103, mpg = 'low'.

Efficiency(a,d,h,w,mpg) :-  d > 103 , d <= 112 , h < 68, mpg = 'high'.

Efficiency(a,d,h,w,mpg) :-  d > 420, mpg = 'low'.

                                   ⋮
```

Now the reclassification problem can be solved by a constraint database join of the *Origin* and *Efficiency* relations. The join is expressed by the following Datalog query:

```
Car(a,c,d,h,w,country,mpg) :- Origin(a,c,d,h,country),
                              Efficiency(a,d,h,w,mpg).
```

The reclassification can be used to predict for any particular car its country of origin and fuel efficiency. For example, if we have a car with $a = 19.5, c = 4, d = 120, h = 87$, and $w = 2979$, then we can use the following Datalog query:

```
Predict(country,mpg) :- Car(a,c,d,h,w,country,mpg),
                        a = 19.5, c = 4, d = 120, h = 87, w = 2979.
```

The prediction for this car is that it is from Europe and has a low fuel efficiency. Note that instead of Datalog queries, in the MLPQ constraint database system one also can use logically equivalent SQL queries to express the above problems.

Semantic Analysis of the Reclassification: Returning to the semantics questions raised in the introduction, one can test each constraint row of the *Cars* relation whether it is satisfiable or not. That allows the testing of which combination classes (out of the nine target labels) is possible. Moreover, the size of each region can be calculated. Assuming some simple distributions of the cars within the feature space, one can estimate the number of cars in each region. Hence one also can estimate the percent of cars that belong to each combination class.

5 Experiments and Discussion

The goal of our experiments is to compare the *Reclassification with constraint databases* and the *Reclassification with an oracle* methods. It is important to make the experiments such that those abstract away from the side issue of which exact classification algorithm (ID3, SVM etc.) is used for the original classification.

In our experiments we use the ID3 method as described in Example 4. Therefore, we compared the *Reclassification with constraint databases* assuming that ID3 was the original classification method with the Reclassification with an oracle assuming that the same ID3 method was used within it. We also compared *Reclassification with constraint databases* with the original linear classification (decision tree) for each class. We chose the ID3 decision tree method for the experiments because it had a non-copyrighted software. Using ID3 already gave interesting results that helped compare the relative accuracy of the methods. Likely a more complex decision tree method would make the accuracy of all the practical algorithms, including the reclassification methods, proportionally better without changing their relative order.

5.1 Experiment with the dataset "Primary Biliary Cirrhosis"

The *Primary Biliary Cirrhosis (PBC)* data set, collected between 1974 and 1984 by the Mayo Clinic about 314 patients [3], contains the following features:

1. case number,
2. days between registration and the earliest of death, transplantion, or study analysis time,
3. age in days,
4. sex (0=male, 1=female),
5. asictes present (0=no or 1=yes),
6. hepatomegaly present (0=no or 1=yes),
7. spiders present (0=no or 1=yes),
8. edema (0 = no edema, 0.5 = edema resolved with/without diuretics, 1 = edema despite diuretics),
9. serum bilirubin in mg/dl,
10. serum cholesterol in mg/dl,
11. albumin in mg/dl,
12. urine copper in μg/day,
13. alkaline phosphatase in Units/liter,
14. SGOT in Units/ml,
15. triglicerides in mg/dl,
16. platelets per cubic ml/1000,
17. prothrombin time in seconds,
18. status (0=alive, 1=transplanted, or 2=dead),
19. drug (1=D-penicillamine or 2=placebo), and
20. histologic stage of disease (1, 2, 3, 4).

We generated the following three subsets from the original data set:
DISEASE with features (3, 4, 5, 7, 8, 9, 10, 13, 14, 16, 17, 20),
DRUG with features (3, 4, 6, 7, 8, 9, 10, 11, 13, 16, 17, 19), and
STATUS with features (3, 4, 5, 6, 7, 8, 9, 10, 11, 16, 17, 18).

In each subset, we used the first eleven features to predict the the twetfth, that is, the last feature.

The results of the experiment are shown in Figures 4, 5, 6 and 7.

It can seen from Figures 4, 5 and 6 that the accuracy of the *Reclassification with constraint databases* has significantly improved compared to the original linear classification (ID3) of a single class.

Figures 7 shows that the *Reclassification with constraint databases* and the *Reclassification with an oracle* perform very similarly. Hence the practical *Reclassification with constraint databases* method achieves what can be considered as the theoretical limit represented by the *Reclassification with an oracle* method. Note that by theoretical limit we mean only a maximum achievable with the use of the ID3 linear classification algorithm. Presumably, if we use for example support vector machines, then both methods will improve proportionally.

5.2 Experiment with the dataset "cars"

In this experiment we used the car data set (pieces of which we used in the examples of this paper) from [1] and the MLPQ constraint database system [9]. The results of the experiment cv are shown in Figures 8, 9 and 10.

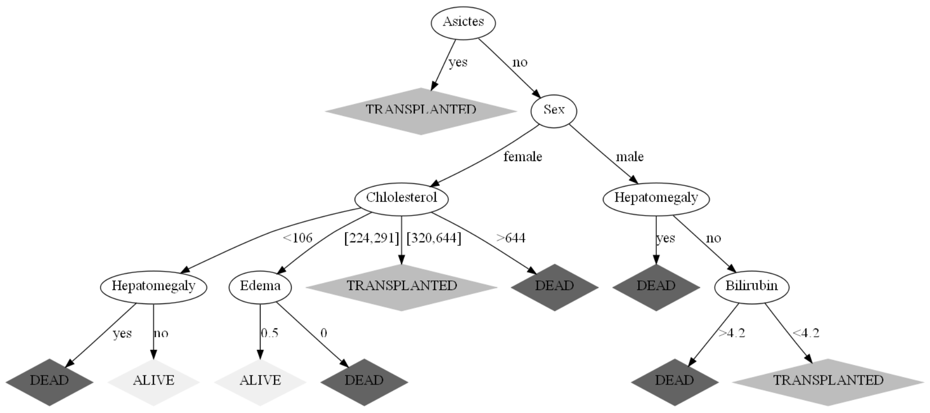

Fig. 3. Tree representation of the constraints using for the prediction of the status of a patient using PBC data. Note that this tree is different from the decision tree generated using the standard ID3 algorithm (here the values of the attributes are defined using constraints). The tree is drawn using Graphviz [2].

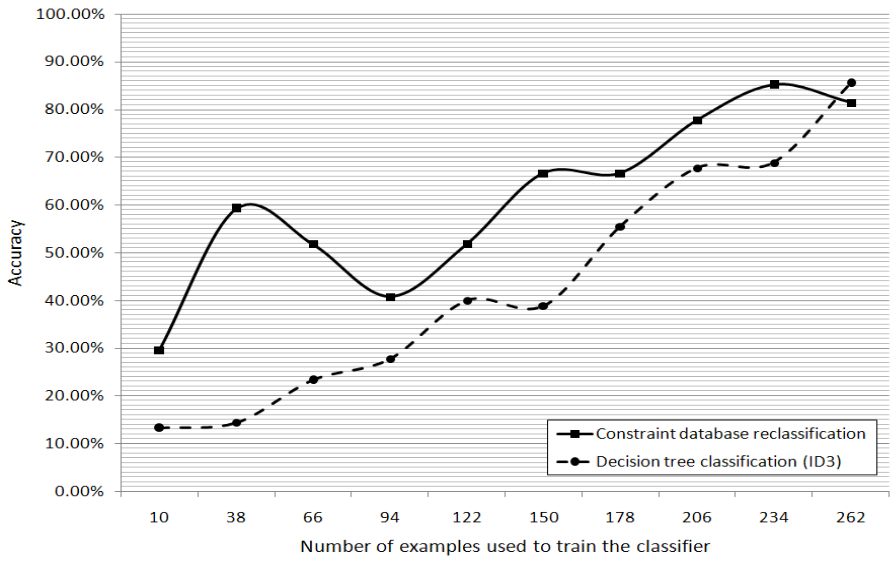

Fig. 4. Comparison of the *Reclassification with constraint databases* (solid line) and the original *Classification with a decision tree (ID3)* for the prediction of the class DISEASE-STAGE of the patients (dashed line) methods using PBC data.

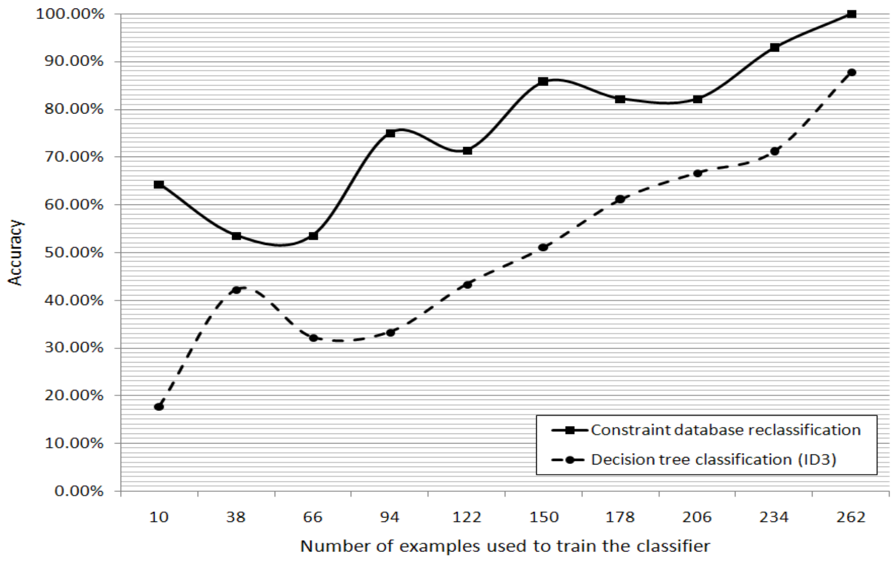

Fig. 5. Comparison of the *Reclassification with constraint databases* (solid line) and the original *Classification with a decision tree (ID3)* for the prediction of the class STATUS of the patients (dashed line) methods using PBC data.

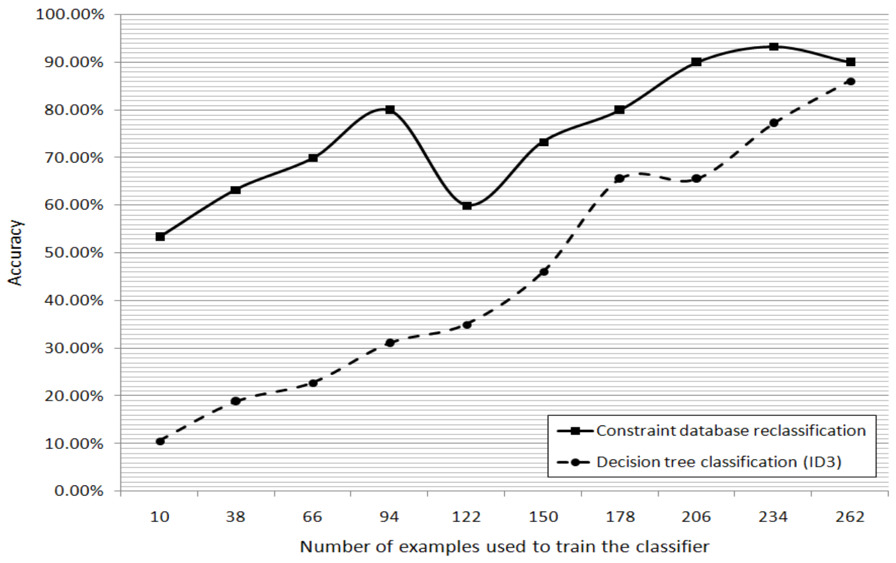

Fig. 6. Comparison of the *Reclassification with constraint databases* (solid line) and the original *Classification with a decision tree (ID3)* for the prediction of the class DRUG of the cars (dashed line) methods using PBC data.

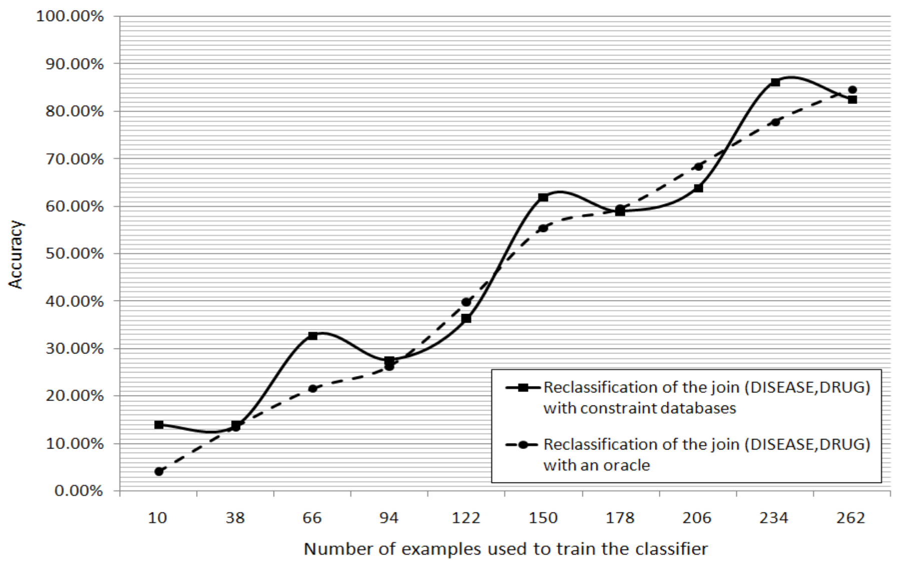

Fig. 7. Comparison of the *Reclassification with constraint databases* (solid line) and the *Reclassification with an oracle* (dashed line) methods using PBC data.

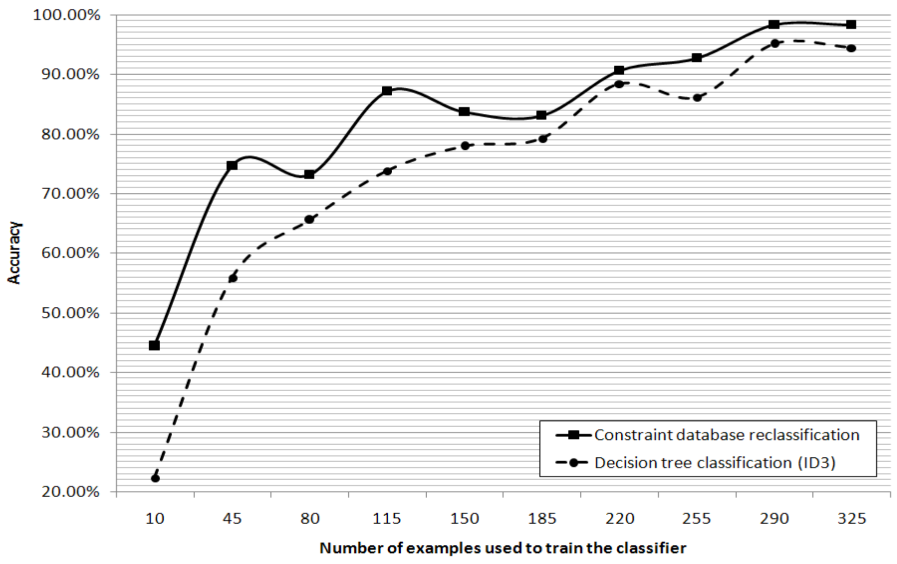

Fig. 8. Comparison of the *Reclassification with constraint databases* (solid line) and the original *Classification with a decision tree (ID3)* for the prediction of the class ORIGIN of the cars (dashed line) using cars data.

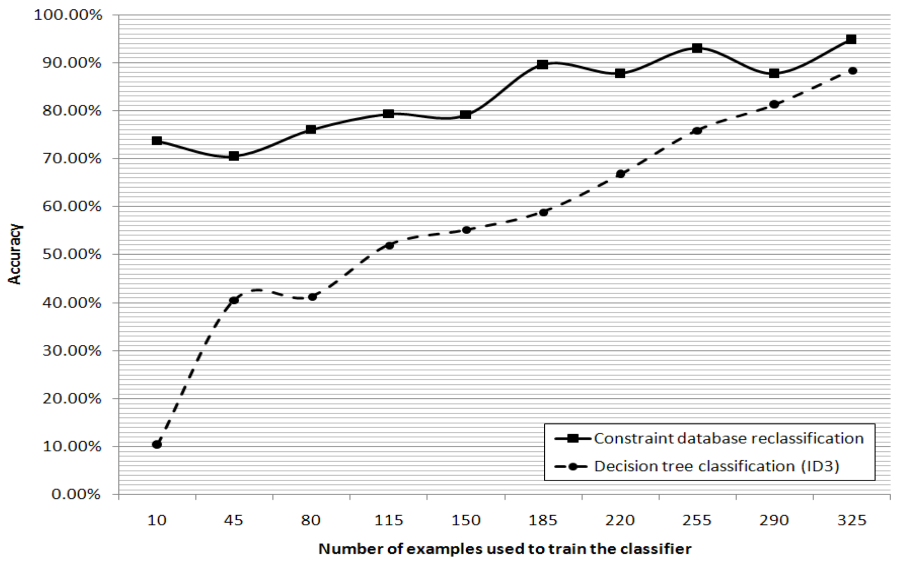

Fig. 9. Comparison of the *Reclassification with constraint databases* (solid line) and the original *Classification with a decision tree (ID3)* for the prediction of the class MPG efficiency of the cars (dashed line) using cars data.

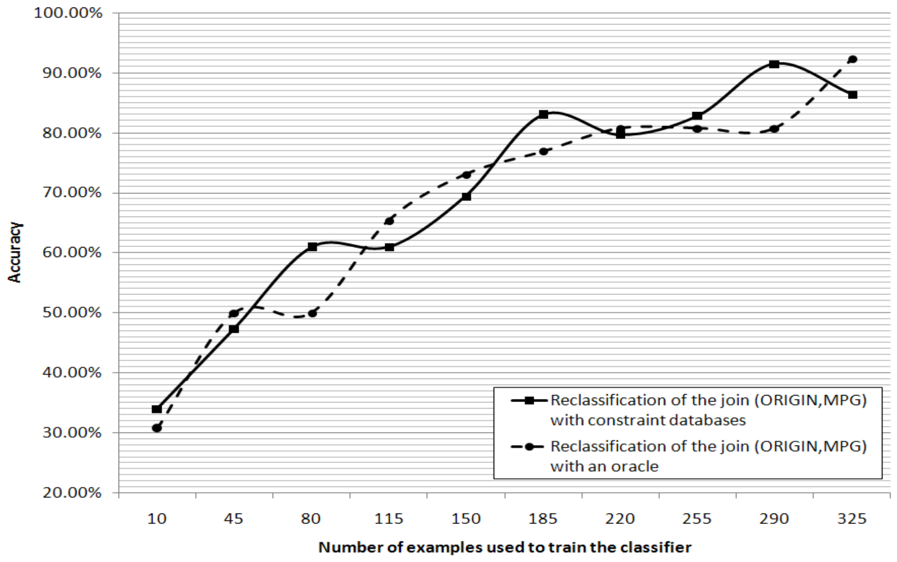

Fig. 10. Comparison of the *Reclassification with constraint databases* (solid line) and the *Reclassification with an oracle* (dashed line) using cars data.

This second experiment agrees with the first data set in that constraint database-based reclassification performs better than the original linear decision tree-based classification.

6 Conclusions

The most important conclusion that can be drawn from the study and the experiments is that the Reclassification with constraint databases method improves the accuracy of linear classifier such as decision trees. The proposed method is also close to the theoretical optimal when joining two classes and is safe to use in practice.

There are several open problems. We plan to experiment with other data sets and use the linear Support Vector Machine algorithm in addition to the ID3 algorithm in the future. Also, when an appropriate data can be found, we also would like to test the *Reclassification method with X-oracles.*

Acknowledgement: The work of the first author of this paper was supported in part by a Fulbright Senior U.S. Scholarship from the Fulbright Foundation-Greece.

References

1. D. Donoho and E. Ramos. The CRCARS dataset. Exposition of Statistical Graphics Technology, Toronto, 1983.
2. J. Ellson, E. Gansner, E. Koutsofios, S. North, and G. Woodhull. Graphviz and dynagraph – static and dynamic graph drawing tools. In M. Junger and P. Mutzel, editors, *Graph Drawing Software*, pages 127–148. Springer-Verlag, 2003.
3. T. R. Fleming and D. P. Harrington. *Counting Processes and Survival Analysis.* Wiley, New York, 1991.
4. I. Geist. A framework for data mining and KDD. In *Proc. ACM Symposium on Applied Computing*, pages 508–13. ACM Press, 2002.
5. T. Johnson, L. V. Lakshmanan, and R. T. Ng. The 3W model and algebra for unified data mining. In *Proc. IEEE International Conference on Very Large Databases*, pages 21–32, 2000.
6. G. M. Kuper, L. Libkin, and J. Paredaens, editors. *Constraint Databases.* Springer-Verlag, 2000.
7. J. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.
8. P. Revesz. *Introduction to Constraint Databases.* Springer-Verlag, 2002.
9. P. Revesz, R. Chen, P. Kanjamala, Y. Li, Y. Liu, and Y. Wang. The MLPQ/GIS constraint database system. In *Proc. ACM SIGMOD International Conference on Management of Data*, 2000.
10. V. Vapnik. *The Nature of Statistical Learning Theory.* Springer-Verlag, 1995.