On Efficient Construction of Decision Trees From Large Databases

Nguyen Hung Son

Institute of Mathematics, Warsaw University
Banacha 2, 02–097, Warsaw, Poland
email: son@mimuw.edu.pl

Abstract. The main task in decision tree construction algorithms is to find the "best partition" of the set of objects. In this paper, we investigate the problem of optimal binary partition of continuous attribute for large data sets stored in *relational databases*. The critical for time complexity of algorithms solving this problem is the number of simple SQL queries necessary to construct such partitions. The straightforward approach to optimal partition selection needs at least $O(N)$ queries, where N is the number of pre-assumed partitions of the searching space. We show some properties of optimization measures related to discernibility between objects, that allow to construct the partition very close to optimal using only $O(\log N)$ simple queries.

Key words: Data Mining, Rough set, Decision tree.

1 Introduction

The philosophy of "rough set" data analysis methods is based on *handling of discernibility between objects* (see [9, 13]). In recent years, one can find a number of applications of rough set theory in Machine Learning, Data Mining or KDD. One of the main tasks in data mining is the classification problem. The most two popular approaches to classification problems are "decision tree" and "decision rule set". Most "rough set" methods are dealing with classification problem by extracting a set of decision rules (see [14, 13, 10]). We have shown in previous papers that the well known discernibility property in rough set theory can be used to build decision tree with high accuracy from data.

The main step in methods of decision tree construction is to find optimal partitions of the set of objects. The problem of searching for optimal partitions of real value attributes, defined by so called cuts, has been studied by many authors (see e.g. [1–3, 11]), where optimization criteria are defined by e.g. height of the obtained decision tree or the number of cuts. In general, all those problems are hard from computational point of view. Hence numerous heuristics have been investigated to develop approximate solutions of these problems. One of major tasks of these heuristics is to define some approximate measures estimating the quality of extracted cuts. In rough set and Boolean reasoning based methods, the quality is defined by the number of pairs of objects discerned by the partition (called *discernibility measure*).

We consider the problem of searching for optimal partition of real value attributes assuming that the large data table is represented in relational data base. The straightforward approach to optimal partition selection needs at least $O(N)$ simple queries, where N is the number of pre-assumed partitions of the searching space. In this case, even the linear complexity is not acceptable because of the time for one step (see [5, 8]). We assume that the answer time for such queries does not depend on the interval length. We show some properties of considered optimization measures allowing to reduce the size of searching space. Moreover, we prove that using only $O(\log N)$ simple queries, one can construct the partition very close to optimal. We have showing that the main part of the formula estimating the quality of the best cut for independent variables from [8] is the same in case of "fully" dependent variables. Comparing [8], we also extend the algorithm of searching for the best cut by adding the global searching strategy.

2 Basic notions

An *information system* [9] is a pair $\mathbb{A} = (U, A)$, where U is a non-empty, finite set called the *universe* and A is a non-empty finite set of *attributes (or features)*, i.e. $a : U \to V_a$ for $a \in A$, where V_a is called *the value set of* a. Elements of U are called *objects or records*. Two objects $x, y \in U$ are said to be discernible by attributes from A if there exists an attribute $a \in A$ such that $a(x) \neq a(y)$. Any information system of the form $\mathbb{A} = (U, A \cup \{dec\})$ is called *decision table* where $dec \notin A$ is called *decision attribute*. Without loss of generality we assume that $V_{dec} = \{1, \ldots, d\}$. Then the set $DEC_k = \{x \in U : dec(x) = k\}$ will be called the k^{th} *decision class of* $\mathbb{A}$ for $1 \leq k \leq d$. Any pair (a, c), where a is an attribute and c is a real value, is called *a cut*. We say that *"the cut* (a, c) *discerns a pair of objects* x, y*"* if either $a(x) < c \leq a(y)$ or $a(y) < c \leq a(x)$.

The decision tree for a given decision table is (in simplest case) a binary directed tree with *test functions* (i.e. boolean functions defined on the information vectors of objects) labelled in internal nodes and decision values labelled in leaves. In this paper, we consider decision trees using cuts as test functions. Every cut (a, c) is associated with test function $f_{(a,c)}$ such that for any object $u \in U$ the value of $f_{(a,c)}(u)$ is equal to 1 (true) if and only if $a(u) > c$. The typical algorithm for decision tree induction can be described as follows:

1. For a given set of objects U, select a cut (a, c_{Best}) of high quality among all possible cuts and all attributes;
2. Induce a partition U_1, U_2 of U by (a, c_{Best}) ;
3. Recursively apply Step 1 to both sets U_1, U_2 of objects until some stopping condition is satisfied.

Developing some decision tree induction methods [3, 11] and some supervised discretization methods [2, 6], we should often solve the following problem: *"for a given real value attribute* a *and set of candidate cuts* $\{c_1, ..., c_N\}$*, find a cut* (a, c_i) *belonging to the set of optimal cuts with high probability."*.

Definition 1 *The d-tuple of integers $\langle x_1, .., x_d\rangle$ is called class distribution of the set of objects $X \subset U$ iff $x_k = card(X \cap DEC_k)$ for $k \in \{1, ..., d\}$. If the set of objects X is defined by $X = \{u \in U : p \leq a(u) < q\}$ for some $p, q \in \mathbb{R}$ then the class distribution of X can be called* **the class distribution in** $[p; q)$.

Any cut $c \in \mathbf{C}_a$ splits the domain $V_a = (l_a, r_a)$ of the attribute a into two intervals: $I_L = (l_a, c)\,; I_R = (c, r_a)$. We will use the following notation:

- U_{L_j}, U_{R_j} – the sets of objects from j^{th} class in I_L and I_R. Let $U_L = \bigcup_j U_{L_j}$ and $U_R = \bigcup_j U_{R_j}$ where $j \in \{1, ..., d\}$;
- $\langle L_1, .., L_d\rangle$ and $\langle R_1, .., R_d\rangle$ – class distributions in U_L and U_R. Let $L = \sum_{j=1}^d L_j$ and $R = \sum_{j=1}^d R_j$
- $C_j = L_j + R_j$ – number of objects in the j^{th} class;
- $n = \sum_{i=1}^d C_j = L + R$ – the total number of objects;

Usually, for a fixed attribute a and the set of all relevant cuts $\mathbf{C}_a = \{c_1, ..., c_N\}$ on a, we use some *measure (or quality functions)* $F : \{c_1, ..., c_N\} \rightarrow \mathbb{R}$ to estimate the quality of cuts. For a given measure F, the *straightforward algorithm* should compute the values of F for all cuts: $F(c_1), .., F(c_N)$. The cut c_{Best} which maximizes or minimizes the value of function F is selected as the result of searching process. The most popular measures for decision tree induction are *"Entropy Function" and "Gini's index"* [4, 1, 11]. In this paper we consider the *discernibility measure* as a quality function. Intuitively, energy of the set of objects $X \subset U$ can be defined by the number of pairs of objects from X to be discerned called $conflict(X)$. Let $\langle N_1, ..., N_d\rangle$ be a class distribution of X, then $conflict(X)$ can be computed by $conflict(X) = \sum_{i<j} N_i N_j$. The cut c which divides the set of objects U into U_1, and U_2 is evaluated by $W(c) = conflict(U) - conflict(U_1) - conflict(U_2)$ i.e. the more is number of pairs of objects discerned by the cut (a, c), the larger is chance that c can be chosen to the optimal set of cut. Hence, in the decision tree induction algorithms based on Rough Set and Boolean reasoning approach, the quality of a given cut c is defined by

$$W(c) = \sum_{i \neq j}^{d} L_i R_j = \sum_{i=1}^{d} L_i \sum_{i=1}^{d} R_i - \sum_{i=1}^{d} L_i R_i \tag{1}$$

This algorithm is called Maximal-Discernibility heuristics or *the MD-heuristics.* The high accuracy of decision trees constructed by using MD-heuristic and their comparison with Entropy-based decision methods has been reported in [7].

For given set of candidate cuts $\mathbf{C}_a = \{c_1, .., c_N\}$ on a, by median of the k^{th} decision class (denoted by $Median(k)$) we mean the cut $c \in \mathbf{C}_a$ minimizing the value $|L_k - R_k|$. Let $c_{min} = \min_i\{Median(i)\}$ and $c_{max} = \max_i\{Median(i)\}$ we have shown (in [8]) the technique for irrelevant cut eliminating called *"Tail cuts can be eliminated"* as follows.

Theorem 1 *The quality function $W : \{c_1, .., c_N\} \rightarrow \mathbb{N}$ defined over the set of cuts is increasing in $\{c_1, ..., c_{min}\}$ and decreasing in $\{c_{max}, ..., c_N\}$.*

This property is interesting because it implies that the best cut c_{Best} can be found in the interval $\{c_{min}, ..., c_{max}\}$ using only $O(d \log N)$ queries to determine the medians of decision classes (by applying Binary Search Algorithm) and to eliminate all tail cuts. Let us also observe that if all decision classes have similar medians then almost all cuts can be eliminated.

Example We consider a data table consisting of 12000 records. Objects are classified into 3 decision classes with the distribution $\langle 5000, 5600, 1400 \rangle$, respectively. One real value attribute has been selected and $N = 500$ cuts on its domain has generated class distributions as shown in Figure 1.

The medians of three decision classes are c_{166}, c_{414} and c_{189}, respectively. The median of every decision class has been determined by *binary search algorithm* using $\log N = 9$ simple queries. Applying Theorem 1 we conclude that it is enough to consider only cuts from $\{c_{166}, ..., c_{414}\}$. In this way 251 cuts have been eliminated by using 27 simple queries only.

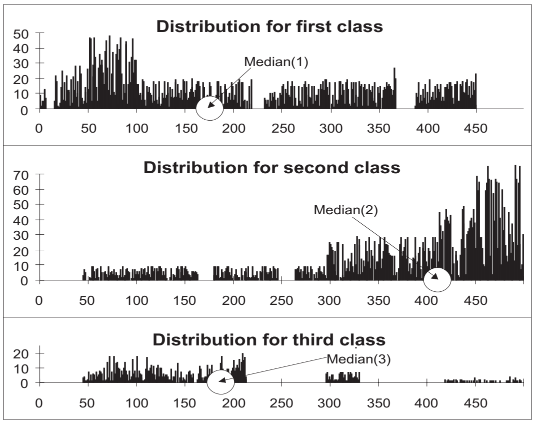

Fig. 1. Distributions for decision classes 1, 2, 3.

3 Divide and Conquer Strategy

The main idea is to apply the *"divide and conquer"* strategy to determine the best cut $c_{Best} \in \{c_1, ..., c_n\}$ with respect to a given quality function.

First we divide the set of possible cuts into k intervals (e.g. $k = 2, 3, ..$). Then we choose the interval to which the best cut may belong with the highest probability. We will use some approximating measures to predict the interval which probably contains the best cut with respect to discernibility measure. This process is repeated until the considered interval consists of one cut. Then the best cut can be chosen between all visited cuts.

The problem arises how to define the measure evaluating the quality of the interval $[c_L; c_R]$ having class distributions: $\langle L_1, ..., L_d \rangle$ in $(-\infty; c_L)$; $\langle M_1, ..., M_d \rangle$ in $[c_L; c_R)$; and $\langle R_1, ..., R_d \rangle$ in $[c_R; \infty)$. This measure should estimate the quality of the best cut among those belonging to the interval $[c_L; c_R]$. In next Section we present some theoretical considering about the quality of the best cut in $[c_L; c_R]$. These results will be used to construct the relevant measure to estimate the quality of the whole interval. We consider two specific probabilistic models for distribution of objects in the interval $[c_L; c_R]$.

Independency model: Let us consider an arbitrary cut c lying between c_L and c_R and let us assume that $\langle x_1, x_2, ..., x_d \rangle$ is a class distribution of the interval $[c_L; c]$. In this model we assume that $x_1, x_2, ..., x_d$ are independent random variables with uniform distribution over sets $\{0, ..., M_1\}$, ..., $\{0, ..., M_d\}$, respectively. One can observe that for all $i \in \{1, .., d\}$ $E(x_i) = \frac{M_i}{2}$ and $D^2(x_i) = \frac{M_i(M_i+2)}{12}$. We have shown in [8] the following theorem:

Theorem 2 *The mean $E(W(c))$ of quality $W(c)$ for any cut $c \in [c_L; c_R]$ satisfies*

$$E(W(c)) = \frac{W(c_L) + W(c_R) + conflict([c_L; c_R])}{2} \tag{2}$$

and the standard deviation of $W(c)$ is equal to

$$D^2(W(c)) = \sum_{i=1}^{n} \left[\frac{M_i(M_i+2)}{12} \left(\sum_{j \neq i} (R_j - L_j) \right)^2 \right] \tag{3}$$

where $conflict([c_L; c_R]) = \sum_{i<j} M_i M_j$.

Full dependent model: In this model, we assume that the values $x_1, ..., x_d$ are proportional to $M_1, ..., M_d$, i.e.

$$\frac{x_1}{M_1} \simeq \frac{x_2}{M_2} \simeq ... \simeq \frac{x_d}{M_d}$$

In this model we have the following theorem:

Theorem 3 *In full independent model, quality of the best cut in interval $[c_R; c_L]$ is equal to*

$$W(c_{Best}) = \frac{W(c_L) + W(c_R) + conflict([c_L; c_R])}{2} + \frac{[W(c_R) - W(c_L)]^2}{8 \cdot conflict([c_L; c_R])} \tag{4}$$

if $|W(c_R) - W(c_L)| < 2 \cdot conflict([c_L; c_R])$. Otherwise it is evaluated by

$$\max\{W(c_L), W(c_R)\}.$$

3.1 Evaluation measures

These are two extreme cases of independent and "fully" dependent random variables of object distribution between decision classes. For real–life data one can expect that the variables are "partially" dependent. Hence we base our heuristic on hypothesis that the derived formula for the quality of the best cut in $[c_L; c_R]$

$$Eval\,([c_L; c_R]) = \frac{W(c_L) + W(c_R) + conflict([c_L; c_R])}{2} + \Delta \tag{5}$$

where the value of Δ is defined by:

$$\Delta = \frac{[W(c_R) - W(c_L)]^2}{8 \cdot conflict([c_L; c_R])} \quad \text{(in the dependent model)}$$

$$\Delta = \alpha \cdot \sqrt{D^2(W(c))} \qquad \text{for some } \alpha \in [0;1]; \quad \text{(in the independent model)}$$

The choice of Δ and the value of parameter α from $[0;1]$ can be tuned in learning process or are given by expert.

3.2 Local and Global Search

We present two strategies of searching for the best cut using formula 5 called *local* and *global search*. In local search algorithm, first we discover the best cuts on every attribute separately. Next, we compare all locally best cuts to find out the globally best one. The details of local algorithm can be described as follows:

```
ALGORITHM: Searching for semi-optimal cut
PARAMETERS: $k \in \mathbb{N}$ and $\alpha \in [0;1]$.
INPUT: attribute $a$; the set of candidate cuts $\mathbf{C}_a = \{c_1, .., c_N\}$ on $a$;
OUTPUT: The optimal cut $c \in \mathbf{C}_a$

begin
 $Left \leftarrow \min$; $Right \leftarrow \max$;      {see Theorem 1}
 while ($Left < Right$)
   1.Divide $[Left; Right]$ into $k$ intervals with equal length by $(k + 1)$
     boundary points i.e.
                 $p_i = Left + i * \frac{Right - Left}{k}$;
     for $i = 0, .., k$.
   2.For $i = 1, .., k$ compute $Eval([c_{p_{i-1}}; c_{p_i}], \alpha)$ using Formula (5). Let
     $[p_{j-1}; p_j]$ be the interval with maximal value of $Eval(.)$;
   3.$Left \leftarrow p_{j-1}$; $Right \leftarrow p_j$;
 endwhile;
 Return the cut $c_{Left}$;
end
```

One can see that to determine the value $Eval([c_L; c_R])$ we need only $O(d)$ simple SQL queries of the form:`SELECT COUNT FROM ... WHERE attribute BETWEEN` c_L `AND` c_R. Hence the number of queries necessary for running our algorithm is of order $O(dk \log_k N)$. In practice we set $k = 3$ because the function $f(k) = dk \log_k N$ over positive integers is taking minimum for $k = 3$. For $k > 2$, instead choosing the best interval $[p_{i-1}; p_i]$, one can select the best union $[p_{i-m}; p_i]$ of m consecutive intervals in every step for a predefined parameter $m < k$. The modified algorithm needs more – but still $O(\log N)$ – simple questions only.

The global strategy is searching for the best cut over all attributes. At the beginning, the best cut can belong to every attribute, hence for each attribute we keep the interval in which the best cut can be found (see Theorem 1), i.e. we have a collection of all potential intervals

$$\textbf{Interval\_Lists} = \{(a_1, l_1, r_1), (a_2, l_2, r_2), ..., (a_k, l_k, r_k)\}$$

Next we iteratively run the following procedure

- remove the interval $I = (a, c_L, c_R)$ having highest probability of containing the best cut (using Formula 5);
- divide interval I into smaller ones $I = I_1 \cup I_2 ... \cup I_k$;
- insert $I_1, I_2, ..., I_k$ to **Interval_Lists**.

This iterative step can be continued until we have one–element interval or the time limit of searching algorithm is exhausted. This strategy can be simply implemented using priority queue to store the set of all intervals, where priority of intervals is defined by Formula 5.

3.3 Example

In Figure 2 we show the graph of $W(c_i)$ for $i \in \{166, ..., 414\}$ and we illustrated the outcome of application of our algorithm to the reduce set of cuts for $k = 2$ and $\Delta = 0$.
First the cut c_{290} is chosen and it is necessary to determine to which of the intervals $[c_{166}, c_{290}]$ and $[c_{290}, c_{414}]$ the best cut belongs. The values of function $Eval$ on these intervals is computed:

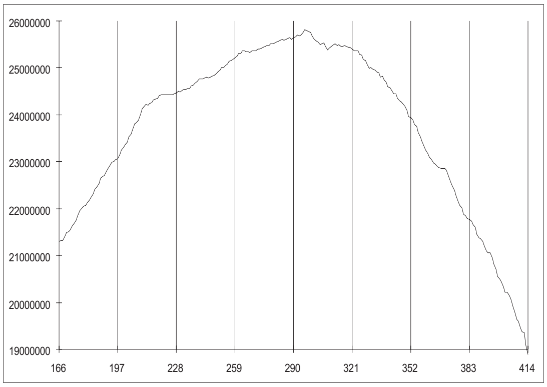

Fig. 2. Graph of $W(c_i)$ for $i \in \{166, .., 414\}$.

$$Eval([c_{166}, c_{290}]) = 23927102, \qquad Eval([c_{290}, c_{414}]) = 24374685.$$

Hence, the best cut is predicted to belong to $[c_{290}, c_{414}]$ and the search process is reduced to the interval $[c_{290}, c_{414}]$. The above procedure is repeated recursively until the selected interval consists of single cut only. For our example, the best cut c_{296} has been successfully selected by our algorithm. In general, the cut selected by the algorithm is not necessarily the best. However, numerous experiments on different large data sets shown that the cut c^* returned by the algorithm is close to the best cut c_{Best} (i.e. $\frac{W(c^*)}{W(c_{Best})} \cdot 100\%$ is about 99.9%).

4 Conclusions

The problem of optimal binary partition of continuous attribute domain for large data sets stored in *relational data bases* has been investigated. We reduced the number of simple queries from $O(N)$ to $O(\log N)$ to construct the partition very close to the optimal one. We plan to extend these results for other measures.
Acknowledgement: This paper was supported by KBN grant 8T11C02519 and Wallenberg Foundation – WITAS project.

References

1. Chmielewski, M. R., Grzymala-Busse, J. W.: Global discretization of attributes as preprocessing for machine learning. In. T.Y. Lin, A.M. Wildberger (eds.). Soft Computing. Rough Sets, Fuzzy Logic Neural Networks, Uncertainty Management, Knowledge Discovery, Simulation Councils, Inc., San Diego, CA 294–297
2. Dougherty J., Kohavi R., Sahami M.: Supervised and unsupervised discretization of continuous features. In. Proceedings of the Twelfth International Conference on Machine Learning, Morgan Kaufmann, San Francisco, CA
3. Fayyad, U. M., Irani, K.B.: On the handling of continuous-valued attributes in decision tree generation. Machine Learning **8**, 87–102
4. Fayyad, U. M., Irani, K.B.: The attribute selection problem in decision tree generation. In. Proc. of AAAI-92, San Jose, CA. MIT Press
5. J. E. Gehrke, R. Ramakrishnan, and V. Ganti. RAINFOREST - A Framework for Fast Decision Tree Construction of Large Datasets. In Proc. of the 24^{th} International Conference on Very Large Data Bases, New York, New York, 1998.
6. Nguyen, H. Son: Discretization Methods in Data Mining. In L. Polkowski, A. Skowron (Eds.): *Rough Sets in Knowledge Discovery* **1**, Springer Physica-Verlag, Heidelberg, 451–482.
7. H.S. Nguyen and S.H. Nguyen. From Optimal Hyperplanes to Optimal Deciison Trees, *Fundamenta Informaticae* **34**No 1–2, (1998) 145–174.
8. Nguyen, H. Son: Efficient SQL-Querying Method for Data Mining in Large Data Bases. Proc. of Sixteenth International Joint Conference on Artificial Intelligence, IJCAI-99, Morgan Kaufmann Publishers, Stockholm, Sweden, pp. 806-811.
9. Pawlak Z.: *Rough sets: Theoretical aspects of reasoning about data*, Kluwer Dordrecht.
10. Polkowski, L., Skowron, A. (Eds.): *Rough Sets in Knowledge Discovery* **Vol. 1,2**, Springer Physica-Verlag, Heidelberg.
11. Quinlan, J. R. *C4.5. Programs for machine learning.* Morgan Kaufmann, San Mateo CA.
12. Skowron, A., Rauszer, C.: The discernibility matrices and functions in information systems. In. R. Słowiński (ed.). Intelligent Decision Support – Handbook of Applications and Advances of the Rough Sets Theory, Kluwer Academic Publishers, Dordrecht 311–362
13. J. Komorowski, Z. Pawlak, L. Polkowski and A. Skowron,(1998). Rough sets: A tutorial. In: S.K. Pal and A. Skowron (eds.), Rough - fuzzy hybridization: A new trend in decision making, Springer-Verlag, Singapore, pp. 3-98.
14. Ziarko, W.: Rough set as a methodology in Data Mining. In Polkowski, L., Skowron, A. (Eds.): *Rough Sets in Knowledge Discovery* **Vol. 1,2**, Springer Physica-Verlag, Heidelberg, pp. 554–576.