Decision Tree Construction by Association Rules

F. Säuberlich, W. Gaul

Institut für Entscheidungstheorie und Unternehmensforschung,
Universität Karlsruhe, D-76128 Karlsruhe, Germany

Abstract: Association rules are used to build decision trees with the help of so-called multivariate splits that can take some (or all) attributes for the description of underlying (data) objects into consideration. It is shown that the new DTAR (Decision Tree Construction by Association Rules) method produces small trees with better interpretability than other multivariate decision tree construction techniques. Data sets well known to the data mining and machine learning community are analyzed to demonstrate the abilities of the new method.

1 Introduction

Association rules and decision trees are well-known labels for two families of techniques often used in the field of knowledge discovery in databases and data mining. This statement is confirmed by a recent own survey concerning data mining application reports from the years 1993 to 1998 where decision trees (32 applications), neural networks (27 applications), and association rules (16 applications) got top positions in a ranking according to frequency of application. In a further study concerning data mining software tools we could put together 16 offers from software firms and—here—in 13 cases decision tree algorithms and in 9 cases association rule algorithms were supported by the corresponding software systems (see Gaul, Säuberlich (1999) for an early version of this study).

Information of this kind was used as starting point for the idea to combine two popular data mining areas, namely association rules and decision trees, to suggest a new method for decision tree construction.

1.1 Decision Trees

A decision tree is built from a so-called training data set which consists of interesting (data) objects. Each of these objects is described by a set of realizations of prespecified attributes and a class label. A decision tree contains a *root node*, (zero ore more) *internal nodes*, and (zero, two or more) *leaf nodes*. In the normal case, the root node and all internal nodes have two or more child nodes created by so-called splits where the outcome of a mathematical or logical expression of the realizations of the attributes of those (data) objects that are attached to the father node is used for split selection. This "splitting" process generates subsets of the training data

set attached to the corresponding tree nodes where the degree of purity of the data subsets with respect to class-membership of their (data) objects should improve. Leaf nodes have associated class labels to indicate that a majority (if not all) of the (data) objects of their attached training data subsets belongs to the labeled classes.

One problem in constructing decision trees is the selection of "best" splits in such nodes. Most decision tree construction algorithms use only one of the prespecified attributes at each splitting operation and those which consider more than one attribute, so-called multivariate splits, often produce results that are in general hard to interpret.

Figure 1 shows an example of such a decision tree built on training data depicted in Table 1. Each node contains in the upper half the node number and in the lower half the quantities of the objects within the attached training data subset that are grouped according to the prespecified classes (in this case: two classes). The split criterion is written beside each arc of the tree. Additionally, under each leave node the associated class label is given.

Once that such a decision tree is built it can be used to predict the class labels of further (data) objects—not contained in the set of training data—for which the apriori classification is not known.

1.2 Association Rules

Association rules work with measures for the association between certain quantities. In the starting situation, one has a set of items $I = \{i_1, ..., i_m\}$ and a set of itemsets $D = \{t_1, ..., t_n\}$ of so-called transactions $t_j \in I$, $j = 1, ..., n$. For a pair (Y, Z) of itemsets $Y, Z \subset I$ (e.g., subsets of brands of a product category) an association rule uses bounds for support and confidence measures of Y and Z to check whether the "association of Y and Z" is meaningful. The support $s(Y \cup Z) = \frac{|\{t \in D : Y \cup Z \subseteq t\}|}{|D|}$ gives the share of itemsets in D which contain $Y \cup Z$ and the confidence $c(Y, Z) = \frac{s(Y \cup Z)}{s(Y)}$ describes the

Object No.	Realization of X_1	Realization of X_2	Class Label
1	a	2	A
2	b	1	B
3	a	1	A
4	c	2	A
5	c	1	B

Table 1: Exemplary training data

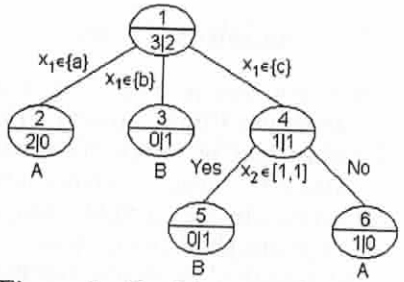

Figure 1: Decision tree built on the training data of Table 1

fact that $c(Y, Z)$ percent of the itemsets in D that contain Y also contain Z. The task of an association rule algorithm is to find all association rules which fulfill prescribed bounds for support and confidence values.

Table 3 shows possible rules generated from the data set of consumer transactions given in Table 2 (articles (items) bought together within single buying situations or consumer transactions that are the basis of a so-called market basket analysis). One possible association rule is "Juice → Coke" with a support value of 2/3 and a confidence value of 4/5. This rule describes the situation that in 80 percent of all transactions of Table 1 in which juice was bought also coke was in the market basket and that the event that these two products were bought together occurred in 4 of the 6 transactions under consideration.

2 Traditional Decision Tree Induction and Some Notation for the New Method

The task of constructing a decision tree from a training data set is also called tree induction in the literature. Almost every tree induction algorithm proceeds in a top-down fashion in the following way: If at a node a specific stop criterion is fulfilled the node is called a leaf node and no further split of the data set attached to this node is made. Otherwise, all possible splits are considered and scored with help of specific selection measures (see Mingers (1989a) for an empirical comparison of selection measures for decision tree induction). A "best" split creates as many child nodes as there are distinct outcomes of the split criterion and the parts of the corresponding partition of the data set attached to the father node are assigned to the child nodes just created.

For our purposes—to suggest a new decision tree induction technique—the following notation is used:

X_ν Attribute with value space

$$A_\nu = \begin{cases} \{a_{\nu_1}, ..., a_{\nu_{W_\nu}}\}, \ a_{\nu_\rho} \in \Re, & \text{if } X_\nu \text{ is categorical} \\ [a_\nu, b_\nu], \ a_\nu, b_\nu \in \Re, & \text{if } X_\nu \text{ is numeric} \end{cases}, \nu = 1, ..., V,$$

x_ν Realization of X_ν, $x_\nu \in A_\nu$,

Transaction	Articles bought (Items)
t_1	Juice, Coke, Beer
t_2	Juice, Coke, Wine
t_3	Juice, Water
t_4	Coke, Beer, Juice
t_5	Juice, Coke, Beer, Wine
t_6	Water

Table 2: Exemplary data set of consumer transactions

Rules with support ≥ 3	Support	Confidence
Juice → Coke	2/3	4/5
Coke → Juice	2/3	1
Coke → Beer	1/2	3/4
Beer → Coke	1/2	1
Juice, Coke → Beer	1/2	3/4

Table 3: Some rules with support $\geq$ 1/2 and confidence $\geq$ 3/4 obtained from the transactions of Table 2

C_j	Class label, $j \in \{1, ..., J\}$,
L	Training data set attached to node p (index p will be omitted),
$(\mathbf{x}, C) \in L$	Element of L, $\mathbf{x} = (x_1, ..., x_V)$ with $x_\nu \in A_\nu$ $(\nu = 1, ..., V)$, $C \in \{C_1, ..., C_J\}$,
$S_k(L)$	Split which divides L into k disjunctive subsets L_i, $i \in \{1, ..., k\}$, *Binary* split if $k = 2$, *Univariate* (*multivariate*) split if only one (more than one) attribute is used for splitting,
N	Number of elements of L, $N = \|L\|$,
N^j	Number of elements $(\mathbf{x}, C) \in L$ with $C = C_j$,
N_i	Number of elements of L_i, $N_i = \|L_i\|$,
N_i^j	Number of elements of L_i with $(\mathbf{x}, C) \in L_i$ and $C = C_j$.

These quantities can be put together in a contingency table as shown in Table 4 from which the outcomes of specific selection measures can be computed.

$S_k(L)$	C_1	C_2	...	C_J	Sum
L_1	N_1^1	N_1^2	...	N_1^J	N_1
L_2	N_2^1	N_2^2	...	N_2^J	N_2
⋮	⋮			⋮	⋮
L_k	N_k^1	N_k^2	...	N_k^J	N_k
Sum	N^1	N^2	...	N^J	N

Table 4: Contingency table for split $S_k(L)$

Most decision tree induction algorithms only use univariate splits. For numeric attributes binary splits can be given as questions of the form "Is $x_\nu \in [\tau, \tau']$?" $(\tau \leq \tau' \in A_\nu)$. For categorical attributes each subset $B \subset A_\nu$ of the attribute space can be considered for such binary splitting operations.

Well-known existing approaches for multivariate decision tree induction consider binary splits and use a linear combination of the attributes as split criterion as OC1 by Murthy et al. (1994) or CART-LC by Breiman et al. (1984). A recent approach from Fukuda et al. (1996) applies so-called numeric association rules and uses two numeric attributes for splitting. All these methods are only capable for numeric attributes and—as the splitting criterion has the form of a linear combination of the attributes or of a region in a two-dimensional value space—the results are in general not easy to interpret.

After having built a decision tree in the top-down fashion mentioned earlier so-called pruning strategies often are applied to select subtrees of the decision tree just constructed that avoid overfitting effects with respect to the training data (see Esposito et al. (1997) or Mingers (1989b) for surveys concerning pruning techniques).

Against this background we consider binary multivariate splits for numeric and categorical attributes composed of subsets (subintervals in the numeric case) of the single value spaces of the alternatives. Since the number of possible splits of this form is in general very large we use a modification of an association rule algorithm to handle the search space. As we focus on tree construction issues, we do not consider pruning strategies. A main reason is that we want to perform a fair comparison of algorithms concerning decision tree induction aspects. Afterwards each of the algorithms compared can be subjected to the same pruning activities.

3 Combination of Decision Trees with Association Rules: The DTAR Method

Our approach for DTAR (Decision Tree Construction by Association Rules) is based on the following steps:

- Generate association rules $W \rightarrow C$ with
  T-tupels $W = (w_{\nu_1}, ..., w_{\nu_T})$, $\nu_\mu < \nu_{\mu+1}$ for $\mu = 1, ..., T-1$ $(T \leq V)$ where
  $w_\nu = \begin{cases} B_\nu & , B_\nu \subset A_\nu, \quad \text{if } X_\nu \text{ is categorical} \\ [\tau_\nu, \tau'_\nu] & , \tau_\nu \leq \tau'_\nu,\ \tau_\nu, \tau'_\nu \in A_\nu, \quad \text{if } X_\nu \text{ is numeric} \end{cases}, \nu \in \{1, ..., V\}$,
  and class labels $C \in \{C_1, ..., C_J\}$.
- Compute support values
  $s(W) = |\{(\mathbf{x}, C) \in L : x_{\nu_1} \in w_{\nu_1} \wedge ... \wedge x_{\nu_T} \in w_{\nu_T}\}|/N$,
  $s(W \cup C_j) = |\{(\mathbf{x}, C) \in L : x_{\nu_1} \in w_{\nu_1} \wedge ... \wedge x_{\nu_T} \in w_{\nu_T} \wedge C = C_j\}|/N$.
- Determine the outcome of the underlying selection measure with the help of the support and confidence values of the corresponding rules.
- Chose an association rule with a "best" outcome of the underlying selection measure and use the question "Is $x_{\nu_1} \in w_{\nu_1} \wedge ... \wedge x_{\nu_T} \in w_{\nu_T}$?" as split criterion.

For computational purposes we modified the Apriori Algorithm from Agrawal and Srikant (1994). The main task is the generation of so-called frequent itemsets (with a support value greater or equal than a minimum support threshold) from which rules with required confidence values can be generated in an easy way. Considering the fact that all subsets of a frequent itemset must also be frequent and starting with the frequent itemsets with just one element, in the k-th recursion only the frequent itemsets with $k-1$ elements are used to generate candidates for frequent itemsets with k elements. In this manner, the search space for frequent itemsets is reduced considerably.

An itemset W in our approach is a tupel of subsets and subintervals selected from T out of the V attribute spaces, respectively. Adequate subsets or

subintervals of the underlying attribute spaces constitute the items in this context and are put together in the set I. All elements $g \in I$ for which a class label $C \in \{C_1, ..., C_J\}$ exists so that $s(g \cup C) \geq s_{min}$ are called frequent and put together in the set G_1. In the k-th recursion we have to secure that only those itemsets with $k-1$ elements are joined which have the first $k-2$ elements in common and for which the remaining two elements are subsets from different attribute spaces. Thus, we avoid to create itemsets in which several elements belong to the same attribute space. Figure 2 shows the DTAR method in pseudo code where the item(sub)set notation is used for the construction of the T-tupels $W = (w_{\nu_1}, ..., w_{\nu_T})$.

$$I = \{\{w_\nu\} | w_\nu = \begin{cases} B_\nu & , B_\nu \subset A_\nu, \\ [\tau_\nu, \tau'_\nu] & , \tau_\nu \leq \tau'_\nu,\ \tau_\nu, \tau'_\nu \in A_\nu, \end{cases} \begin{matrix} \text{if } X_\nu \text{ is categorical} \\ \text{if } X_\nu \text{ is numeric} \end{matrix}, \quad \nu \in \{1, ..., V\}\},$$

$$L = \{(\mathbf{x}, C) | x_\nu \in A_\nu,\ C \in \{C_1, ..., C_J\}\},\ s_{min} \in [0, 1],$$

$$G_1 = \{g \in I | \exists C \in \{C_1, ..., C_J\} : s(g \cup C) \geq s_{min}\},$$

$$k = 2$$

while $G_{k-1} \neq \emptyset$ **do**

$$\overline{H}_k = \{\{g_1, ..., g_{k-1}, g'_{k-1}\} | g = \{g_1, ..., g_{k-1}\}, g' = \{g'_1, ..., g'_{k-1}\} \in G_{k-1},$$
$$g_\mu = g'_\mu, \mu = 1, ..., k-2, g_{k-1} \subset A_\nu, g'_{k-1} \subset A_{\nu'} \text{ with } \nu < \nu'\},$$

$$H_k = \{h \in \overline{H}_k | \forall g'' = \{g''_1, ..., g''_{k-1}\} \subset h : g'' \in G_{k-1}\},$$

$$G_k = \{g \in H_k | \exists C \in \{C_1, ..., C_J\} : s(g \cup C) \geq s_{min}\},$$

$$k = k + 1$$

end while

answer $\bigcup_{\kappa=1}^{k-1} G_\kappa$

Figure 2: Pseudo code for DTAR

The outcome of the chosen selection measure depends on the binary split created with the help of W and denoted as $S_2(L, W)$. It can be computed in a direct way using the corresponding support and confidence values. The contingency table shown in Table 5 describes this situation.

$S_2(L,W)$	C_1	C_2	...	C_J	Sum
$L_1(W)$	$N \cdot s(W \cup C_1)$	$N \cdot s(W \cup C_2)$	...	$N \cdot s(W \cup C_J)$	$N \cdot s(W)$
$L_2(W)$	$N^1 - N \cdot s(W \cup C_1)$	$N^2 - N \cdot s(W \cup C_2)$	...	$N^J - N \cdot s(W \cup C_J)$	$N \cdot (1 - s(W))$
Sum	N^1	N^2	...	N^J	N

Table 5: Contingency table for rule $W \rightarrow C$

As only binary splits are considered the contingency table consists of two rows. The support of W denotes the share of objects of L fulfilling the corresponding splitting criterion. The number of objects fulfilling the criterion and belonging to class C_j is equivalent to the value $N \cdot s(W \cup C_j)$. All other

values of the contingency table can be computed directly with help of these support values and the marginal values already known.

For each frequent itemset W the support values $s(W \cup C_j)$ for all possible class labels C_j ($j = 1, ..., J$) have to be determined in course of the algorithm. Then, the outcome of the chosen selection measure can be computed directly by using these support and confidence values, e.g., the outcome of the Gini-Index is given by

$$\text{Gini}_{S_2(L,W)}(L) = \sum_{i=1}^{2} \frac{N_i}{N} \sum_{j=1}^{J} \left(\frac{N_i^j}{N_i}\right)^2 - \sum_{j=1}^{J} \left(\frac{N^j}{N}\right)^2 = s(W) \cdot \underbrace{\sum_{j=1}^{J} \left(\frac{s(W \cup C_j)}{s(W)}\right)^2}_{c(W,C_j)^2}$$

$$+(1 - s(W)) \cdot \sum_{j=1}^{J} \left(\frac{N^j/N - s(W \cup C_j)}{1 - s(W)}\right)^2 - \sum_{j=1}^{J} \left(\frac{N^j}{N}\right)^2 .$$

To illustrate the DTAR method the training data of Table 1 are used. Figure 3 shows the frequent itemsets, the rules with corresponding values for support (Sup), confidence (Conf), and the outcome of the gini-index (Gini), as well as the decision tree built by DTAR. Compared with Figure 1 only one split is required to get a perfect division of the data set.

$s_{\min} = 2/5$

$I = \{\{a\},\{b\},\{c\},\{a,b\},\{a,c\},\{b,c\},\{[1,1]\},\{[2,2]\}\}$

$G_1 = \{\{a\},\{a,b\},\{a,c\},\{b,c\},\{[1,1]\},\{[2,2]\}\}$

$H_2 = \{\{\{a\},[1,1]\},\{\{a\},[2,2]\},\{\{a,b\},[1,1]\},$
$\{\{a,b\},[2,2]\},\{\{a,c\},[1,1]\},\{\{a,c\},[2,2]\},$
$\{\{b,c\},[1,1]\},\{\{b,c\},[2,2]\}\}$

$G_2 = \{\{\{a,c\},[2,2]\},\{\{b,c\},[1,1]\}\}$

DTAR finds the following rules

Rule	Sup	Conf	Gini
$\{a\} \rightarrow A$	2/5	1	0,2133
$\{a,b\} \rightarrow A$	2/5	2/3	0,0133
$\{a,c\} \rightarrow A$	3/5	3/4	0,18
$\{b,c\} \rightarrow B$	2/5	2/3	0,2133
$\{[1,1]\} \rightarrow B$	2/5	2/3	0,2133
$\{[2,2]\} \rightarrow A$	2/5	1	0,2133
$\{\{a,c\},[2,2]\} \rightarrow A$	2/5	1	0,2133
$\{\{b,c\},[1,1]\} \rightarrow B$	2/5	1	0,48

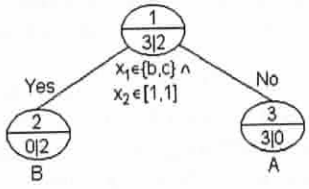

Figure 3: DTAR results: Frequent itemsets, rules, and the decision tree built from the training data of Table 1

4 Applications and Comparisons

We used five data sets well known from the STATLOG Project (see Michie et al. (1994)) to compare our new DTAR method with two multivariate decision tree induction algorithms (OC1 and CART-LC) and with one univariate technique (CART). Table 6 shows some characteristics of the underlying data sets.

Name	Attributes	Numeric Attributes	Objects	Classes
Australian Credit	14	6	690	2
Diabetes of Pima Indians	8	8	768	2
German Credit	20	7	1000	2
Heart Disease	13	7	270	2
Vehicle Silhouette	18	18	846	4

Table 6: Characteristics of the data sets used for comparisons

Only numeric attributes were analyzed since OC1 and CART-LC cannot handle categorical attributes. As mentioned before no pruning strategies were considered. For all data sets we conducted 10-fold cross validations. As comparison criteria the *accuracy* of a tree (the percentage of (data) objects for which the class label is correctly predicted by the tree), the number of *leave nodes* (a measure for the complexity of the tree structure), and the *maximal depth* of a tree (the length of a longest path in the tree) are used. Table 7 shows the results of this comparisons.

	Criterion	OC1	CART-LC	CART	DTAR
Australian Credit	Accuracy	68.70%	73.62%	72.32%	74.20%
	Leave Nodes	65	82.3	127.2	59.2
	Maximal Depth	12	13.2	16.3	12.2
Diabetes of Pima Indians	Accuracy	67.58%	68.10%	71.09%	71.48%
	Leave Nodes	81.3	95.2	122.5	54.4
	Maximal Depth	11.4	13.3	14.1	12.1
German Credit	Accuracy	62.90%	62.90%	64.70%	63.50%
	Leave Nodes	172.3	193.7	248.1	124.3
	Maximal Depth	21.1	20.2	20.7	17
Heart Disease	Accuracy	64.44%	66.67%	65.56%	72.22%
	Leave Nodes	35.9	42	53	20.2
	Maximal Depth	9.3	10.1	10.6	6.8
Vehicle Silhouette	Accuracy	71.04%	70.45%	71.99%	72.22%
	Leave Nodes	111.7	122.1	126.3	68.4
	Maximal Depth	16.1	17.3	17.6	9.5

Table 7: Comparison of the trees built on the data sets of Table 6

In four of the five data sets analyzed the DTAR method came out best with respect to accuracy. Only for the German Credit data set the CART algorithm constructed trees with an accuracy slightly better than DTAR. With regard to the number of leave nodes in all cases the trees constructed by the DTAR method showed the best results. With respect to maximal depth only for two of the data sets under consideration OC1 was better than DTAR. All in all, DTAR generated trees that are comparable or even better than those constructed by OC1 and CART-LC although the additional ability of DTAR to handle categorical attributes was not taken into consideration.

5 Conclusions and Outlook

Association rules and decision trees are popular areas of data mining. We use a combination of techniques from these areas to cope with the problem of multivariate split selection for the construction of decision trees. Our new DTAR method competes favourably with other multivariate decision tree induction algorithms with respect to accuracy and tree structure (number of leave nodes, maximal depth). Additionally, DTAR can handle categorical attributes and the results are in general better to interpret than those of comparable algorithms. Further analyses and simulations are on the way to get additional and more extensive comparisons and to proof the usability of the new method on larger data sets.

References

AGRAWAL, R. and SRIKANT, R. (1994): Fast Algorithms for Mining Association Rules. Proceedings of the 20th VLDB Conference, Santiago, Chile, 1994.

BREIMAN, L., FRIEDMAN, J.H., OLSHEN, R.A., and STONE, C.J. (1984): Classification and Regression Trees. Wadsworth International Group, Belmont, California.

ESPOSITO, F., MALERBA, D., and SEMERARO, G. (1997): A Comparative Analysis of Methods for Pruning Decision Trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 19, No. 5, 476-493.*

FUKUDA, T., MORIMOTO, Y., and MORISHITA, S. (1996): Constructing Efficient Decision Trees by Using Optimized Numeric Association Rules. Proceedings of the 22nd VLDB Conference, Mumbai (Bombay), India, 1996.

GAUL, W. and SÄUBERLICH, F. (1999): Classification and Positioning of Data Mining Tools. In: Gaul, W., Locarek-Junge, H. (eds.) (1999): Classification in the Information Age. Springer, Berlin, Heidelberg, 143-152.

MICHIE, D., SPIEGELHALTER, D.J., and TAYLOR, C.C. (eds.) (1994): Machine Learning, Neural and Statistical Classification. Ellis Horwood, 1994.

MINGERS, J. (1989a): An Empirical Comparison of Selection Measures for Decision Tree Induction. *Machine Learning, Vol. 3, No. 4, 319-342.*

MINGERS, J. (1989b): An Empirical Comparison of Pruning Methods for Decision Tree Induction. *Machine Learning, Vol. 4, No. 2, 227-243.*

MURTHY, S.K., KASIF, S., and SALZBERG, S. (1994): A System for Induction of Oblique Decision Trees. *Journal of Artificial Intelligence Research, Vol. 2, 1-32.*