Under consideration for publication in Knowledge and Information Systems

Attribute Generation Based on Association Rules

Masahiro Terabe[1], Takashi Washio[2], Hiroshi Motoda[2], Osamu Katai[3] and Testuo Sawaragi[4]

[1]Mitsubishi Research Institute, Inc., Tokyo, Japan
[2]Institute of Scientific and Industrial Research, Osaka University, Osaka, Japan
[3]Graduate School of Informatics, Kyoto University, Kyoto, Japan
[4]Graduate School of Engineering, Kyoto University, Kyoto Japan

Abstract. A decision tree is considered to be appropriate (1) if the tree can classify the unseen data accurately, and (2) if the size of the tree is small. One of the approaches to induce such a good decision tree is to add new attributes and their values to enhance the expressiveness of the training data at the data pre-processing stage. There are many existing methods for attribute extraction and construction, but constructing new attributes is still an art. These methods are very time consuming, and some of them need a priori knowledge of the data domain. They are not suitable for the data mining dealing with large volume of data. We propose a novel approach that the knowledge on attributes relevant to the class is extracted as association rules from the training data. The new attributes and the values are generated from the association rules among the originally given attributes. We elaborate on the method and investigate its feature. The effectiveness of our approach is demonstrated through some experiments.

Keywords: Data Mining; Data Pre-processing; Association Rules; Decision Tree; Attribute Generation

1. Introduction

Data mining becomes a key technique to discover meaningful patterns and rules from large amount of data [2]. It is often used in the fields of business such as marketing and customer support operations. However, one of the major difficulties in the practical approach is scarceness of experts in the application domain and good data mining tools.

In data mining process, a decision tree is often used as knowledge representation of the mined result, because it is easy to understand for human analysts. In general, the appropriateness of the decision tree is evaluated from the next two criteria.

Size: The size of decision tree is evaluated by the number of nodes in the tree. A smaller decision tree is easier to understand. It is also known that a smaller decision tree leads to avoiding overfitting to the data.

Prediction accuracy: The decision tree predicting the correct class of a new instance with higher accuracy is desired.

However, inducing an appropriate decision tree is often difficult. In many cases, the original attributes are not expressive enough. Further, some of them are irrelevant or redundant. Feature selection removes irrelevant and redundant attributes[6]. Feature extraction and construction create new attributes[4, 5], and add them to the original training data. As a result, the description of the original training data is enriched. The decision tree induced from the pre-processed training data can be better than the one induced from the original training data. Constructing new attributes without domain knowledge is computationally very expensive. Preparing appropriate construction operators and applying them in right order are still an art. These processes are very time consuming and some of them need a priori knowledge of the data domain. These features are not suitable for data mining. When the method is applied to data mining, the computational cost of the method should be reasonably small even if the data size is large. Furthermore, domain knowledge is insufficient in the beginning of data mining process, because the human analyst may not be an expert in the data domain.

In this paper, we propose a novel method of attributes generation, which has suitable properties as a data mining method. Our proposed method generates new attributes based on association rules among the original attributes. The rules are extracted from the original training data automatically.

The proposed method has the following properties.

1. It does *not need any a priori knowledge of the attributes and their association.* The knowledge is extracted from the original training data automatically.
2. It adopts Apriori algorithm to generate *attributes association rules* so that the rules are extracted with *reasonable computational cost even if the data size is large.*

These properties are very advantageous compared to the aforementioned traditional attribute generation methods. The first property makes it easy to use the method from the beginning of data mining process when domain knowledge is insufficient. The second enables to apply the method to large scale data.

The paper is organized as follows. In section 2, we briefly explain association rules and Apriori algorithm. In section 3, we propose the novel attribute generation method based on the Apriori algorithm. In section 4, we investigate the performance of our proposed method. In section 5, we discuss the characteristics of the proposed method based on experimental results.

2. Association Rules

Apriori algorithm[1] extracts a co-occurrence pattern of items from the instances in the following form of an association rule.

$$R : \boldsymbol{B} \Rightarrow \boldsymbol{H}, \tag{1}$$

where

$\boldsymbol{B}$: Body which is the condition part of the association rule, and

$\boldsymbol{H}$: Head which is the conclusion part of the association rule.

.

Both "Body" and "Head" are a set of items. This association rule means that "If an instance includes all the items in *Body*, then the instance also includes all the items in *Head* in many cases." In other words, the association rule indicates a co-occurrence pattern of item sets in the data.

Traditional algorithms need much computation time to extract association rules. Apriori algorithm proposed by Agrawal succeeded in reducing the search space efficiently so that the computation time is much smaller than traditional methods even if the data size is large.

In Apriori algorithm, the candidates of association rule are evaluated by two indices, i.e., "support value" and "confidence value". The support value $sup(R)$ of the association rule R is defined as follows.

$$sup(R : \boldsymbol{B} \Rightarrow \boldsymbol{H}) = \frac{n(\boldsymbol{B} \cup \boldsymbol{H})}{N}, \tag{2}$$

where

$n(\boldsymbol{B} \cup \boldsymbol{H})$: number of instances which include all items in both $\boldsymbol{B}$ and $\boldsymbol{H}$, and

N : total number of data.

The support value indicates the ratio of the number of instances including all items appearing in the rule to all instances. Therefore, the rule covers larger portion of the data if the support value of the rule is higher.

The confidence value of the association rule $conf(R)$ is defined as follows.

$$conf(R : \boldsymbol{B} \Rightarrow \boldsymbol{H}) = \frac{n(\boldsymbol{B} \cup \boldsymbol{H})}{n(\boldsymbol{B})}, \tag{3}$$

where

$n(\boldsymbol{B})$: number of instances which include all items in $\boldsymbol{B}$.

If the confidence value is higher, the association is more plausible.

In Apriori algorithm, both the minimum support value and the minimum confidence value are given as threshold parameters. In the association rule extraction process, the rules which do not satisfy these threshold conditions are removed from the set of candidates. The search uses the monotonicity of support values, i.e., if the support of an item set is below the threshold, its super set is pruned from the search. If these minimum thresholds given are decreased, more candidates of association rules are generated and evaluated in the algorithm. As a result, the computation time increases, though a more complete set of good association rules would be extracted. On setting of these minimum thresholds, the trade off between the cost of computation time and the rule covering good association should be taken into account carefully.

3. Proposed Attribute Generation Method

First, the proposed method extracts *Attributes Association Rules* (*AARs*) as the basic knowledge of associations among attributes from the original training

data prepared for decision tree induction. Next, the proposed method generates some new attributes based on the extracted AARs, and adds them to the original training data. Therefore, the data description is extended by this data pre-processing. After this data pre-processing to the original training data, a decision tree is induced by the standard decision tree algorithm.

The AARs represent associations among attributes and the class. Our proposed method consists of the following steps.

Step 1: Description of training data for a decision tree is transformed to the transaction format.

Step 2: Apriori algorithm extracts AARs from the transaction data.

Step 3: Some candidates of new attributes are generated based on AARs.

Step 4: Degree of contribution of the new attribute candidates to identifying the class is evaluated.

Step 5: The new attribute candidates that satisfy a criterion (explained later) are added

The details of each step are explained in the following subsections.

3.1. Step 1: Data Description Transformation

First, the proposed method transforms the data description of the original training data to the transaction format.

The training data is described as a set of instance data, *train_data*. An instance in *train_data*, *datum* is represented as follows.

$$datum = \{v_{i,j} | \forall i \in M, j \in \exists N_i\} \cup \{v_{c,j} | j \in \exists N_c\}, \tag{4}$$

where

$$M = \{1, \ldots, m\},$$
$$N_i = \{1, \ldots, n_i\},$$
$$N_c = \{1, \ldots, n_c\}.$$

where $v_{i,j}$ is a value of an attribute a_i, m the number of attributes, n_i the total number of values of a_i, and n_c the total number of values of the class c.

In the proposed method, each pair of attribute "a_i" and its value "$v_{i,j}$" is transformed to an item in form of $item_i = \langle a_i, v_{i,j} \rangle$ and $item_{m+1} = \langle c, v_{c,j} \rangle$. Under this transformation, *datum* in Equation 4 becomes a transaction as follows.

$$trans = \{item_1, \ldots, item_i, \ldots, item_{m+1}\}. \tag{5}$$

The transaction data, *trans_data*, is a set of transactions.

3.2. Step 2: Extraction of Attributes Association Rules (AARs)

Next, the proposed method extracts AARs that represent the association among pairs of attribute and its attribute value.

The attribute value that has strong association with the class value is useful

in predicting the class. Accordingly, the association between the attributes and the class is a measure of the goodness of the attribute. For that reason, we extract AARs, which satisfy the following two conditions.

– The condition part includes only an item set consisting of attributes.
– The conclusion part includes only an item representing the class.

An AAR R that satisfies these conditions is described as follows.

$$R : \textbf{if} \bigcup_{\forall i \in M_s} \{\langle a_i, v_{i,j}\rangle \in ITEM_{a_i} | \exists j \in N_i\} \\ \textbf{then} \{\langle c, v_{c,j}\rangle \in ITEM_c | \exists j \in N_c\}, \tag{6}$$

where

$$M_s \subseteq M,$$

$$ITEM_{a_i} = \bigcup_{\forall \langle a_i, v_{i,j}\rangle \in \forall trans \in trans\_data} \{\langle a_i, v_{i,j}\rangle\},$$

$$ITEM_c = \bigcup_{\forall \langle c, v_{c,j}\rangle \in \forall trans \in trans\_data} \{\langle c, v_{c,j}\rangle\}.$$

This represents a fact that the instance involving these pairs of the attribute and its value in the condition part is concluded with the class $v_{c,j}$ with high confidence. Consequently, a new composed attribute which is the collection of the attributes and their values in these pairs is expected to be useful in predicting the class of an instance.

3.3. Step 3: Generating New Attribute Candidates

Let $\mathcal{R}$ be a set of all AARs extracted in Step 2. Each AAR $R(\in \mathcal{R})$ is a basic unit of this attribute generation algorithm. Let $\boldsymbol{B}$ is the body of R, $\boldsymbol{H}$ is the head of R, $A(\boldsymbol{B}) = \{a_i | \langle a_i, v_{i,j}\rangle \in \boldsymbol{B}\}$, and $C(\boldsymbol{H}) = \{v_{c,j} | \langle c, v_{c,j}\rangle \in \boldsymbol{H}\}$. Define a partition $\mathcal{P}$ such that each element of $\mathcal{P}$ is $P_q = \{R | \forall A(\boldsymbol{B}); A(\boldsymbol{B})\text{s are mutually identical in } \mathcal{R}\}$.

A new attribute candidate is charactrized by the follwing quardaple AN_q for each P_q.

$$AN_q = \langle \mathcal{A}_q, \mathcal{V}_q, \mathcal{S}_q, \mathcal{C}_q \rangle, \tag{7}$$

where

$$\mathcal{A}_q : A(\boldsymbol{B}) \text{ where } \boldsymbol{B} \text{ is the body of } R \text{ in } P_q,$$
$$\mathcal{V}_q = \{V_{q_0}, V_{q_1}, \ldots, V_{q_{|P_q|}}\},$$
$$\mathcal{C}_q = \{sup(R) | R \in P_q\},$$
$$\mathcal{S}_q = \{conf(R) | R \in P_q\},$$

where

$$V_{q_0} = \bigwedge_{\forall R \in P_q} \neg true(\boldsymbol{B})$$
$$V_{q_k} \in \{true(\boldsymbol{B}) | \forall R \in P_q\} \ (k = 1, \ldots, |P_q|).$$

Table 1. New attribute generation algorithm: *NewAttributeCand*.

A set of AARs extracted in Step 2: $\mathcal{R}$
An AAR: $R_p(\in \mathcal{R}) : \boldsymbol{B}_p \Rightarrow \boldsymbol{H}_p$,
where
$\boldsymbol{B}_p = \bigcup_{\forall i \in M_{sp}} \{\langle a_i, v_{i,j}\rangle \in ITEM a_i | \exists j \in N_i\}$; /*condition part*/
$\boldsymbol{H}_p = \{\langle c, v_{c,j}\rangle \in ITEM c | \exists j \in N_c\}$; /*conclusion part*/
$sup(R_p)$; /*support*/ $conf(R_p)$; /*confidence*/
$A(\boldsymbol{B}_p) = \{a_i | \langle a_i, v_{i,j}\rangle \in \boldsymbol{B}_p\}$
A set of new attribute candidate information: $\mathcal{AN}$
A new attribute candidate information: $AN_q = \langle \mathcal{A}_q, \mathcal{V}_q, \mathcal{S}_q, \mathcal{C}_q\rangle \in \mathcal{AN}$,
where
$\mathcal{A}_q$; /* new attribute candidate*/
$\mathcal{V}_q$; /* a set of attribute values*/
$\mathcal{C}_q$; /* a set of support value of each original AAR */
$\mathcal{S}_q$; /* a set of confidence value of each original AAR */
Algorithm:

```
NewAttributeCand(R, AN){
    /* Generate a partition P such that each element of */
    /* P is a partition such that the element */
    /* P_q = {R_p | ∀A(B_p); A(B_p)s are mutually identical in R} */
    q = 1; P_1 = {R_1}; P ← {P_1};
    for(p = 2; p <= |R|; p++){
        /* A(B_p) is identical with a A(B) where B is the body of R ∈ P_q ∈ P */
        if(∃P_q ∈ P | A(B_p) == A(B); B is the body of R ∈ P_q){
            /* Add R_p to P_q*/
            P_q ← P_q ∪ {R_p}
        }
        /* A(B_p) is not identical with any A(B) */
        else {
            q++;
            P_q = {R_p}
            P ← {P_q};
        }
    }

    /* Generate new attribute candidate information AN_q from each P_q*/
    for(q = 1; q <= |P|; q++){
        AN_q = ⟨A_q, V_q, C_q, S_q⟩;
        A_q = A(B);
        V_q = {V_q0, V_q1, ..., V_q|pq|};
            where
            V_q0 = ⋀_{∀R_p ∈ P_q} ¬true(B_k)
            V_qk ∈ {true(B_p) | ∀R_p ∈ P_q} (k = 1, ..., |P_q|)
        C_q = {sup(R_p) | R_p ∈ P_q};
        S_q = {conf(R_p) | R_p ∈ P_q};
        AN ← AN ∪ AN_q
    }
}
```

The attributes in the Body $A(\boldsymbol{B})$ where $\boldsymbol{B}$ is the body of AARs R in P_q are merged into the new attribute of $\mathcal{A}_q$. The value $\mathcal{V}_q$ of a new attribute candidate is defined by the predicate "$true(\boldsymbol{B})$". This predicate becomes true when all of the items in Body $\boldsymbol{B}$ appear in an instance. The set of support and the confidence values of AARs $R \in P_q$ that constitutes the new attribute candidates are recorded as $\mathcal{S}_q$ and $\mathcal{C}_q$ for the evaluation of the candidate in the later step.

The proposed attribute generation process is explained with the following example. The original training data consist of two attributes A_1 (attribute value $V_1 = \{0, 1\}$), A_2 (attribute value $V_2 = \{0, 1\}$), and a class C(class $C = \{0, 1\}$). Furthermore, suppose that the following two AARs are extracted

$$
\begin{aligned}
&R_1 : \textbf{if } \{\langle A_1, 0\rangle, \langle A_2, 0\rangle\} \textbf{ then } \{\langle C, 0\rangle\}, \\
&support : sup(R_1), confidence : conf(R_1), \\
&\text{where} \\
&\boldsymbol{B}_1 = \{\langle A_1, 0\rangle, \langle A_2, 0\rangle\}, \\
&\boldsymbol{H}_1 = \{\langle C, 0\rangle\}, \text{ and} \\
&A(\boldsymbol{B}_1) = \{A_1, A_2\}.
\end{aligned} \tag{8}
$$

$$
\begin{aligned}
&R_2 : \textbf{if } \{\langle A_1, 1\rangle, \langle A_2, 1\rangle\} \textbf{ then } \{\langle C, 0\rangle\}, \\
&support : sup(R_2), confidence : conf(R_2), \\
&\text{where} \\
&\boldsymbol{B}_2 = \{\langle A_1, 1\rangle, \langle A_2, 1\rangle\}, \\
&\boldsymbol{H}_2 = \{\langle C, 0\rangle\}, \text{ and} \\
&A(\boldsymbol{B}_2) = \{A_1, A_2\}.
\end{aligned} \tag{9}
$$

The two AARs R_1 and R_2 have identical $A(\boldsymbol{B})$, i.e., $A(\boldsymbol{B}_1)$ and $A(\boldsymbol{B}_2)$. Therefore, these AARs are in the same partition element $P_1 = \{R_1, R_2\}$. A new attribute candidate AN_1 is generated as follows.

$$
\begin{aligned}
&AN_1 = \langle \mathcal{A}_1, \mathcal{V}_1, \mathcal{S}_1, \mathcal{C}_1 \rangle, \\
&\text{where} \\
&\mathcal{A}_1 = \{A_1, A_2\}, \\
&\mathcal{V}_1 = \{V_{1_0}, V_{1_1}, V_{1_2}\}, \\
&\mathcal{S}_1 = \{sup(R_1), sup(R_2)\}, \text{ and} \\
&\mathcal{C}_1 = \{conf(R_1), sup(R_2)\}.
\end{aligned} \tag{10}
$$

Here, $V_{1_1} = true(\boldsymbol{B}_1)$, $V_{1_2} = true(\boldsymbol{B}_2)$, and $V_{1_0} = \bigwedge_{\forall R \in P_1} \neg true(\boldsymbol{B})$. The attribute generation process explained above is depicted in Figure 1.

3.4. Step 4: Evaluation of Generated Candidates

To evaluate the goodness of generated candidates to induce a decision tree, we adopt the information gain criterion used in the decision tree algorithm ID3[11]. We use an approximated definition of gain, $Gain(\mathcal{A}_q)$, to reduce the computation cost for the large amount of data. In the evaluation of the approximated gain, the information given in the former steps, e.g., support, confidence, and total number of instances, is used, but any other information requiring further heavy computation is not needed. This feature of the approximation enhances the applicability of our proposed method to the large amount of the data through the significant reduction of the computation time.

In detail, the information gain, $Gain(\mathcal{A}_q)$, is calculated by using the support values in $\mathcal{S}_q$ and the confidence values in $\mathcal{C}_q$. Let $R_k \in P_q$ $(k = 1, \ldots, |P_q|)$.

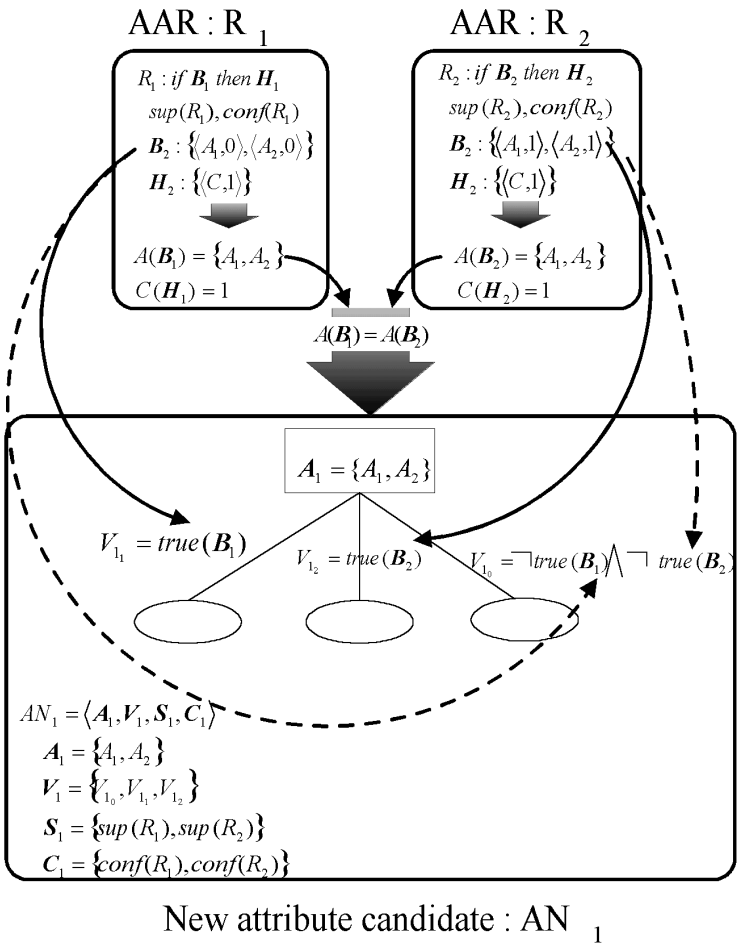


Fig. 1. New attribute caindidate generation from AARs : R_1 andR_2.

$$
\begin{aligned}
&Gain(\mathcal{A}_q) \\
&= -\sum_{j=1}^{|n_c|} \frac{n(v_{c,j})}{N} \log_2 \frac{n(v_{c,j})}{N} \\
&- \sum_{k=1}^{|P_q|} \frac{sup(R_k)}{conf(R_k)} \left\{ -conf(R_k) \log_2 conf(R_k) - p_k \log_2 \frac{p_k}{n_c - 1} \right\} \\
&- \left\{ 1.0 - \sum_{k=1}^{|P_q|} \frac{sup(R_k)}{conf(R_k)} \right\} \{ -r_j \log_2 r_j \}, \qquad (11)
\end{aligned}
$$

where

$$
\begin{aligned}
N &: \text{number of training data,}\\
sup(R_k) \in \mathcal{S}_q &: \text{support value of } R_k \text{ which corresponds to } V_{q_k},\\
conf(R_k) \in \mathcal{C}_q &: \text{confidence value of } R_k \text{ which corresponds to } V_{q_k},\\
p_k &= 1.0 - conf(R_k),\\
r_j &= \frac{N_{q_0}(v_{c,j})}{N - N\sum_{k=1}^{|P_q|}\frac{sup(R_k)}{conf(R_k)}},\\
n(v_{c,j}) &: \text{number of instances whose class is } v_{c,j},\\
N_{q_0}(v_{c,j}) &= n(v_{c,j}) - \sum_{k=1}^{|P_q|} N_{q_k}(v_{c,j}),\\
&: \text{number of instances whose attribute value and class}\\
&\quad \text{value are } V_{q_0} \text{ and } v_{c,j} \text{ respectively,}\\
N_{q_k}(v_{c,j}) &= \begin{cases} N \cdot sup(R_k) & (v_{c,j} \equiv C(\boldsymbol{H}_k))\\ \dfrac{N \cdot \frac{sup(R_k)}{conf(R_k)}(1.0 - conf(R_k))}{n_c - 1} & (\text{otherwise}) \end{cases}.
\end{aligned}
$$

In this equation, the first term of the right hand side represents the amount of information which is necessary to predict the class of the instances without using any information of the attributes. The second term represents the sum of the information amount needed to predict the classes of the instances having the attribute value $V_{q_k} \in \mathcal{V}_q$, $k = 1, \ldots, |P_q|$. The third term stands for the amount of information to classify the instances having the attribute value V_{q_0}. For more detail, refer to the appendix.

If this information gain of the candidate is larger than 0, the candidate is considered to be informative for classification.

3.5. Step 5: Adding New Attributes

The new attribute candidates $\mathcal{A}$ whose information gains $Gain(\mathcal{A}_q)$ are larger than 0, are added to the attributes of the original data. By adding these new attributes to the data, the descriptive power of the data is expected to be improved, and a better decision tree may be induced.

4. Experiment

4.1. Conditions of Experiment

To confirm the effectiveness of our proposed method for the improvement of decision tree induction, experiments have been conducted for several sample data sets. In these experiments, we use C4.5 for decision tree induction[10]. All of the functional options of C4.5 are set to defaults. The pruned decision trees are used for evaluation.

The sample data sets are selected from the UCI Machine Learning Repository [9]. They are selected from the data sets used in the Zheng's paper [14] to compare the results. The specifications of the test data sets are summarized in Table 2.

Table 2. The specifications of data sets for experiment.

Data Set	# of Training Data	# of Test Data	# of Attribute	# of Class
Monk1	124	432	6	2
Monk2	169	432	6	2
Monk3	122	432	6	2
Promoters	8124	CV10	22	2
Phenome	12960	CV10	8	52
Stress	12960	CV10	8	5
Letter	12960	CV10	8	163
Tic-Tac-Toe	958	CV10	9	2

CV10: 10 fold cross validation.

Because the proposed method can only deal with nominal attributes, the data sets that include only nominal attributes are used. The Monk's data sets (Monk1, Monk2, Monk3) are artificial data designed for evaluation of machine learning algorithms. The other five data sets are real-world domain data. They are from a molecular biology domain (Promoters), three linguistic domains (Phoneme, Stress, Letter), and a game domain (Tic-Tac-Toe).

We mainly evaluate the following two aspects for data mining.

Improvement of decision tree: Two indices are used to evaluate the effect of the proposed method for improvement of the decision tree. One is the size of the decision tree, and the other is the prediction accuracy.

Applicability to large data sets: The applicability to the large data is evaluated. In the experiment, we investigate the computation time for various data size.

The features of decision tree are evaluated using 10 fold cross validation except the Monk's data sets.

4.2. Effect to Improve Decision Tree

The experimental results on Monk's data sets of our proposed method (AARs method) is summarized in Table 3. For comparison, we also give the result of some other attribute generation methods: AQ17-DCI, AQ17-HCI[4], CI[13], ID2-of-3[8], XofN[14] and decision trees generated with original attribute (C4.5). In this experiment, we set the minimin support and minimum confidence 0.05 and 0.95 respectively.

In the Monk1 data set, both the size and the prediction accuracy are improved well by AARs method. On the other hands, AARs method does not demonstrate good performance in the Monk2 and Monk3 data sets.

Next, we evaluate the experimental results with real-world domain data. The experimental results are summarized in Table 5 and Table 6. For AARs method, we also investigate performance with several settings of minimum confidence and minimum support. The result of AARs method is chosen for the case where the induced decision tree's prediciton accuracy is the best among the several settings. The best setting of minimum support and minimum confidence is summarized in Table 4.

In Promoter and Tic-Tac-Toe data sets, AARs method improves both prediction accuracy and size over the C4.5. On the other hands, in the three linguistic

Table 3. The accuracy(%) and size on the Monk's data sets.

Algorithm	Monk1		Monk2		Monk3	
	Accuracy	Size	Accuracy	Size	Accuracy	Size
C4.5	75.7	18	65.0	31	97.2	12
AQ17-DCI	100.0	N/A	100.0	N/A	94.2	N/A
AQ17-HCI	100.0	N/A	93.1	N/A	100.0	N/A
CI	100.0	14	67.1	22	95.8	14
ID2-of-3	100.0	18	98.1	24	97.2	21
XofN	100.0	17	100.0	13	100.0	9
AARs	100.0	8	75.5	35	92.8	11

Table 4. The setting of minimum support and minimum confidence when the best accuracy is demonstrated.

Data Set	AARs	
	minimum support	minimum confidence
Promoters	0.25	0.95
Phoneme	0.001	0.80
Stress	0.01	0.85
Letter	0.003	0.90
Tic-Tac-Toe	0.05	0.45

data sets (Phenome, Letter, Stress), AARs method can not improve the size though the prediction accuracy is improved from the C4.5.

4.3. Applicability to Large Scale Data

Next, the applicability of the proposed method to large scale data is evaluated. One of the main factors that affect the computation time of the proposed method is the number of training data. Various sizes of the test data sets have been generated as follows. An instance is picked up from Monk1 data set, and the class is changed to an erroneous value with probability 5%. This process is repeated until the number of data needed is attained. The training data sets having the size of 10,000, 50,000, 100,000, and 500,000 are prepared. A personal computer having the specification of OS: Linux OS, CPU: Pentium 166 Hz, and main memory: 128 M bytes is used in this experiment. The experimental results are shown in Table 7 from which the following is concluded.

- The computation time for data pre-processing also increases when the data size becomes large. The increase is almost proportional to the size of the data.
- The effect to improve decision tree is maintained even if the data size is large.

Table 5. The accuracy(%) on the real-world domains.

Data Set	C4.5	CI	ID2-of-3	XofN	AARs
Promoters	76.3	81.0	87.6	88.5	90.4
Phoneme	81.1	82.3	83.1	83.9	83.6
Stress	82.7	83.8	86.2	87.6	86.1
Letter	73.7	65.6	75.1	76.9	76.6
Tic-Tac-Toe	84.7	94.2	94.9	98.4	99.7

Table 6. The size on the real-world domains.

Data Set	C4.5	CI	ID2-of-3	XofN	AARs
Promoters	22.6	15.0	11.2	13.9	7.0
Phoneme	2339.2	1634.5	1188.4	1506.0	3221.6
Stress	2077.3	1074.1	961.5	739.6	1751.6
Letter	3394.9	1024.9	1654.8	2242.4	2825.8
Tic-Tac-Toe	128.5	82.0	95.8	42.8	23.7

Table 7. The number of training data and computation time for pre-processing.

# of Training Data	C4.5 (Original Data)		AARs (Pre-processed Data)		Pre-processing Computation Time (sec)
	Accuracy (%)	Size	Accuracy (%)	Size	
10,000	78.7	90	95.0	9	4
50,000	82.9	79	95.1	12	13
100,000	85.4	79	95.0	12	22
500,000	80.3	90	95.0	13	114

5. Discussion

5.1. Basic Features of Proposed Method

The basic features of the proposed method are well demonstrated in case of the decision tree induced from Monk's data sets. Monk's data sets are artificial data prepared for classification problem[12]. As indicated in the experimental result in section 4.2, AARs method improves the prediction accuracy and the size of the induced decision tree over the other attribute generation methods. However, AARs method does not demonstrate good performance in Monk2 and Monk3 data sets. We explain the reason of the performance difference.

Each data set has 6 attributes $a_1, \ldots, a_6$ and 2 class values, $Class = \{0, 1\}$. Each data set contains its target concept. The logical descriptions of the target concept are as follows.

Monk1: if $(a_1 = a_2)$ or $(a_5 = 1)$ then $Class = 1$.

Monk2: if $(a_n = firstvalue)$ for *exactly two* choices of n in $\{1, 2, \ldots, 6\}$ then $Class = 1$.

Monk3: if either $(a_5 = 3$ and $a_4 = 1)$ or $(a_5 \neq 3$ and $a_1 \neq 3)$ then $Class = 1$. Monk3 data has 5% additional noise (missclasifications) in the training data.

Here, "$a_x = a_y$" means that these attributes take the same attribute value. The discussion on the performance of the proposed method is summarized for each data as follows.

Monk1: By applying AARs method to the original training data, sixteen new attributes $a_7, \ldots, a_{22}$ are generated. The decision trees induced from the original data and the pre-processed data by AARs method are depicted in Figure 2 respectively. The decision tree induced from the training data pre-processed by AARs method includes new attributes a_7 in the root node, and a_{10} as a node in the next level. The new attribute a_7 implies a target concept "**if** $(a_1 = a_2)$ **then** $Class = 1$" appropriately with conjunction of two original

attributes. These included new attributes imply the target concept so that AARs method improves the prediction accuracy and size of decision trees.

Monk2: Thirteen new attributes $a_7, \ldots, a_{19}$ are generated. In this example, we notice the limitation of AARs method and its new attribute representation. As new attributes, AARs method can generate only conjunction (not disjunction) of original attributes. Even under this representation constraint on new attributes, the proposed method can extract the correct target concept of the data in form of AARs such as,

$$\textbf{if}\,\{\langle a_1,1\rangle,\langle a_2,1\rangle,\langle a_3,0\rangle,\langle a_4,0\rangle,\langle a_5,0\rangle,\langle a_6,0\rangle\}\,\textbf{then}\,\{Class=1\}$$

$$\ldots$$

$$\textbf{if}\,\{\langle a_1,0\rangle,\langle a_2,0\rangle,\langle a_3,0\rangle,\langle a_4,0\rangle,\langle a_5,1\rangle,\langle a_6,1\rangle\}\,\textbf{then}\{\,Class=1\}$$

However, such conjunctive associations of many attributes rarely appear, i.e., their support is very small in this data set. For this reason, extracting such associations and generating new attributes are very difficult for this data set. Simmilarly to AARs method, CI[13] also represents new attributes as conjunction of original attributes. Therefore, CI does not improve the decision tree in Monk2.
On the other hands, the XofN and its new attribute representation X-of-N[14] can represent the target concept as follows.

$$\textbf{if}\,X-of-\{\langle a_1,1\rangle,\langle a_2,1\rangle,\langle a_3,0\rangle,\langle a_4,1\rangle,\langle a_5,1\rangle,\langle a_6,1\rangle\,\}=2$$

$$\textbf{then}\,\{Class=1\}$$

The target concept of Monk2 is more suitable for X-of-N representation than AARs' representation. This is one of the major reasons why the improvement of decision tree is different between XofN and AARs method.

Monk3: The twenty one new attributes $a_7, \ldots, a_{27}$ are generated. The decision tree from the data pre-processed by AARs method includes new attributes a_{13}. By the data pre-processing, the size of decision tree becomes smaller. However, the prediction error is slightly increased. The proposed method extracts one of the target concepts "**if** $(a_5 = 3$ and $a_4 = 1)$ **then** $Class = 1$". However, the other target concept "**if** $(a_5 \neq 3$ and $a_1 \neq 3)$ **then** $Class = 1$" is not extracted by AARs because of the representation constraint mentined above. By this limitation of extracting the latter target concept, the good set of new attributes which expresses the target concept appropriately can not be generated. This fact gives a bad effect in prediction accuracy of decision tree induced from the data pre-processed by AARs method.

5.2. Effectiveness of the Proposed Method in Data Mining

In most of the data sets, AARs method improves the prediction accuracy. AARs method improves both prediction accuracy and size particulary in Monk1, Promoter, and Tic-Tac-Toe. The target concepts in these data sets are suitable to represent with AARs representation. On the other hands, AARs method does not work well in Monk2 and Monk3 data sets where target concepts are not suitable for AARs method's new attribute representation. These target concepts are more suitable for representation using X-of-N and negation.

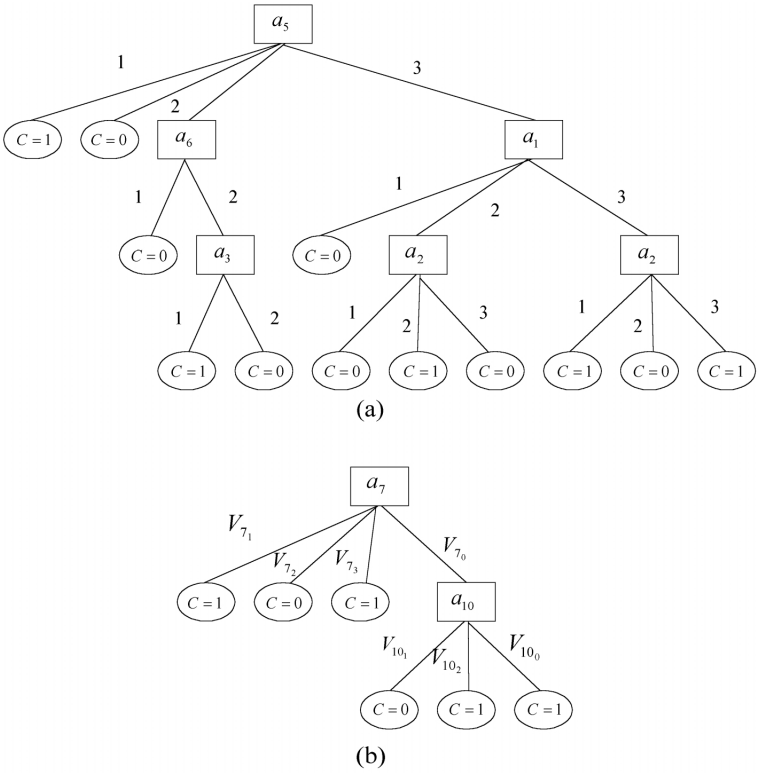


Fig. 2. Two decision trees induced from Monk1 data set: (a) original, (b) pre-processed by AARs method.

The adopted new attribute representation is different from other attribute generation methods. From the practical point of view, attribute generation methods which adopt different representations should be applied to the original data, and the generated attributes should be selected by an attribute selection method.

When the training data size becomes large, the computation time required by the proposed method increases. However the increase is almost proportional to the size of the data. This result guarantees the applicability of the proposed method to large data sets.

The levels of minimum support and minimum confidence are main parameters of the proposed method for new attribute generation. When the minimum support is set small, the associations among attributes which appear not frequently in the data is extracted as AARs. On the other hand, the association

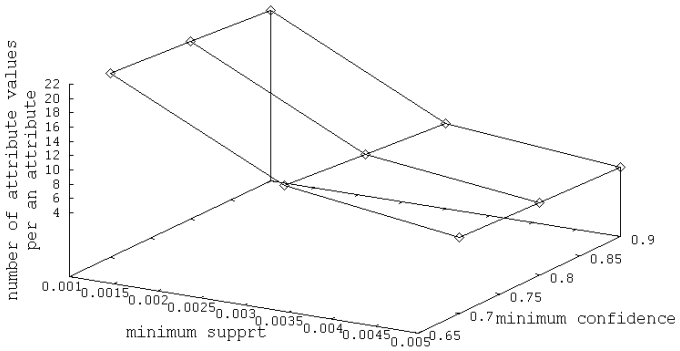


Fig. 3. The relation between minimum thresholds and the number of attribute values per new attributes (Phenome).

rules which is not plausible are extracted as AARs when the minimum confidence is set small. Thus, smaller the values of the minimum support and the minimum confidence are, more numbers of AARs are extracted. Accordingly, each new attribute takes many attribute values. The relations between these parameters' thresholds and the number of attribute values per new attribute in Phoneme data set is depicted in Figure 3. By including such new attributes taking many attribute values, the size of decision tree becomes larger.

The association pattern appears less frequently when either the number of attribute values or the number of classes increases. Therefore, the minimum support should be set smaller when the number of attribute values per attribute or number of classes is larger. Figure 4 depicts the relationship between (a) the product of the number of attribute values per attribute and the number of classes and (b) the minumum support value when the prediction accuracy of induced decision tree is the best. The results are for the real world data used in section 4.2. The negative correlation is confirmed between these two values. The regression equation is,

$$y = 0.34x^{-0.67} \ (R^2 = 0.820), \tag{12}$$

where

x : product of the number of attribute values per attribute and the number of classes,

y : minimum confidence when the prediction accuracy of induced decision tree is the best, and

R^2 : coefficient of detemination.

The coefficient of determination R^2 is calculated as follows. This represents the fraction of the deviation component explained by this regression in the data.

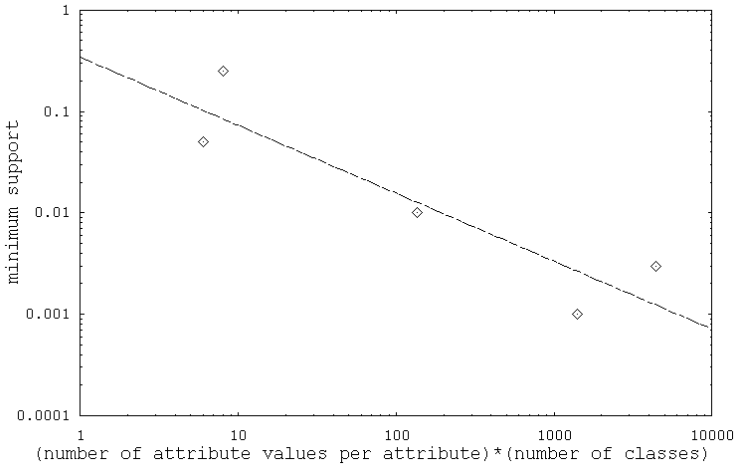


Fig. 4. The relationship between x: product of the number of attribute values per attribute and classes and y: the minimum support value when the prediction accuracy of induced decision tree is the best.

$$R^2 = \frac{\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}, \tag{13}$$

where

$\{x_i, y_i\}$: (observed) data$(i = 1, \ldots, n)$,

$\hat{y}_i$: estimates on y_i predicted with the regression equation, and

$\bar{y}$: mean of y.

The regression equation indicates that the minimum support should be set smaller when the product of the number of attribute values and classes is larger.

6. Related Work

Many investigated attribute generation methods. ID2-of-3[8] adopts M-of-N new attribute representation, and generates binary attributes. XofN[14], an extension of ID2-of-3 represents new nominal attribute by X-of-N. As discussed in section 5, these methods improve the prediction accuracy and size when these new attribute representation is appropriate to describe the target concept. As for representation of new attributes, CI[13] is one of the closest to AARs method, because it also uses conjunction to represent the target concept of data. However, they did not discuss the applicapability to data mining dealing with large volume data.

Bloedorn et al.[4] investigated the method of data-driven constructive induction (DCI). In the method, new attributes are generated by applying the various attribute construction operators to the original ones. Though the evaluation of the new attributes are conducted by using training data, which operators to apply need to be selected from the prepared repository by human user.

Lavrăc[5] investigated the method of attribute generation based on a priori knowledge of attributes in the framework of inductive logic programming setting. Though the algorithm works very in many cases, the applicability to the large scale data has not been assessed.

Liu et al. proposed a classifier induction algorithm which induces the classifiers from association rules[7]. The algorithm shows better performance on classification problem than C4.5. However, their algorithm does not induce a decision tree but only rule a classifier.

7. Conclusion

In this paper, we proposed a novel data pre-processing method to improve the performance of decision tree induction. The effectiveness of our method has been demonstrated through experiments using a subset of U.C. Irvine data sets. Without using a priori knowledge of attributes and associations among them, our method extracts the knowledge of association among attributes in form of AARs from the data automatically, and uses them to generate new attributes. The computation time required by the method remains small even if the size of the training data is large. These features are highly advantageous for the purpose of data mining.

The following issues remain for our future work.

- By applying the proposed method to the decision tree algorithm such as C4.5, we expect to induce appropriate decision trees. We plan to evaluate the effect to improve decision trees by our method together with the attribute grouping function implemented to C4.5[10].
- Currently, our method is not directly applicable to continuous valued attributes. Adopting an function dealing with continuous valued attributes to our method is a future work.
- In processing new attribute evaluation by Equation 11, we mainly consider processing time. The process does not need much calculation cost, and this feature is appropriate for data mining method. However, attribute selection mechanisms should be investigated because increase of the number of attributes requires the expansion of the database and more calculation to induce the decision tree.

References

[1] Agrawal, R. and Srikant, R.: Fast Algorithms for mining association rules, *Proceedings of the 20th VLDB Conference*, pp.487–499 (1994).

[2] Berry, M.J.A. and Linoff G.S.: *Data Mining Techniques For Marketing, Sales, and Customer Support*, Wiley (1997).

[3] Bloedorn, E., Wnek, J. and Michalski, R.S. : Multistrategy Constructive Induction AQ17-MCI, *Proceedings of the 2nd International Workshop on Multistrategy Learning*, pp.188–203 (1993).

[4] Bloedorn, E. and Michalski, R.S.: Data-Driven Constructive Induction: A Method and Its Application, *IEEE Intelligent Systems & their applications*, Vol.13, No.2, pp.30–37 (1998).

[5] Lavrăc, N., Gamberger, D., and Turney, P. : A Relevancy Filter for Constructive Induction, *IEEE Intelligent Systems & their applications*, Vol.13, No.2, pp.50–56 (1998).

[6] Liu, H. and Motoda, H. (Eds.): *Feature Selection For Knowledge Discovery and Data Mining*, Klewer Academic Publishers (1998).

[7] Liu, B., Hsu, W., and Ma, Y.: Integrating Classification and Association Rule Mining, *Proceedings of the 4th Conference on Knowledge Discovery and Data Mining*, pp.80–86 (1998).

[8] Murphy, P.M. : ID2-of-3: Constructive induction of M-of-N concepts for discriminators in decision trees, *Proceedings of the 8th International Workshop on Machine Learning*, pp.183-187 (1991).

[9] Blake, C., Keogh, E. and Merz, C.J.: UCI Repository of Machine Learning Databases, http://www.ics.uci.edu/ mlearn/MLRepository.html, Irvine, CA: University of California, Department of Information and Computer Science (1998).

[10] Quinlan, R. : *C4.5: Programs for Machine Learning*, Morgan Kaufmann (1993).

[11] Russell, S. and Norvig, P.: *Artificial Intelligence, A Modern Approach*, Prentice-Hall (1995).

[12] Thrun, S., Bala, J., and Bloedorn, E. et al. : The MONK's Problems: A Performance Comparison of Different Learning Algorithms, *Technical Report of Carnegie Mellon University, CMU-CS-91-197* (1991).

[13] Zheng, Z. : Constructing Conjunctive Tests for Decision Trees, *Proceedings of the 5th Australian Joint Conference on Artificial Intelligence*, pp.355-360 (1992).

[14] Zheng, Z. : Constructing Nominal X-of-N Attributes, *Proceedings of the 14th International Joint Conference of Artificial Intelligence*, pp.1064–1070 (1995).

A. On Evaluation Equation (11)

We give additional explanation on how to derive the evaluation equation (11) for a new attribute candidate $\mathcal{A}_q$. Here, its attribute values $\mathcal{V}_q$ are assumed as $\mathcal{V}_q$=$\{V_{q1},\ldots,V_{q_{|P_q|}},V_{q0}\}$).

The new attribute candidate is evaluated using the approximated information gain $Gain(\mathcal{A}_j)$ when it is adopted at the root node of the decision tree.

First, the amount of information G_{root}, which is necessary to predict the class correctly in the root node is calculated with the number of instances $n(v_{c,j})$ which belong to each class $v_{c,j}$.

$$G_{root} = -\sum_{j=1}^{n_c} \frac{n(v_{c,j})}{N} \log_2 \frac{n(v_{c,j})}{N}, \tag{14}$$

where

N : number of training data,

$n(v_{c,j})$: number of instances whose class is $v_{c,j}$.

Next, the amount of information which is neccesary to predict the class in the child node $child_k$ which has the following the attribute value $\{V_{q_k}|k = 1,\ldots,|P_q|\} \in \mathcal{V}_q$ defined by the AAR R_k, is calculated as follows.

From the definition of support value $sup(R_k)$ and confidence value $conf(R_k)$ of R_k, the number $N^1_{q_k}$ of instances falling into $child_k$ is equal to the number of instances which match the condition part of AAR R_k. Therefore, $N^1_{q_k}$ is calculated with support value $sup(R_k)$ and confidence value $conf(R_k)$ as follows.

$$N^1_{q_k} = N \cdot \frac{sup(R_k)}{conf(R_k)}. \tag{15}$$

Furthermore, the number of instance $N^2_{q_k}$ which belongs to the class $C(\boldsymbol{H}_k)$ is calculated by the support value $sup(R_k)$ of R_k.

$$N^2_{q_k} = N \cdot sup(R_k). \tag{16}$$

On the other hands, the number of instances which belong to other classes can not be calculated using the index of AARs alone. Thus, we take the conservative approximation by considering the case that the amount of information to predict the class correctly is the maximum. The case occurs when the numbers of instances that belong to each class are even. The number of instance $N^3_{q_k}$ is calculated as follows.

$$\begin{aligned} N^3_{q_k} &= \frac{N^1_{q_k} - N^2_{q_k}}{n_c - 1} \\ &= \frac{N \cdot \frac{sup(R_k)}{conf(R_k)}(1.0 - conf(R_k))}{n_c - 1}. \end{aligned} \tag{17}$$

Hence, the approximated amount of information G_{child_k} which is necessary to predict the class correctly in the child node $child_k$ can be estimated as follows.

$$
\begin{aligned}
&G_{child_k} \\
&= \frac{N^1_{q_k}}{N}\left\{-\left(\frac{N^2_{q_k}}{N^1_{q_k}}\right)\log_2\left(\frac{N^2_{q_k}}{N^1_{q_k}}\right) - (n_c-1)\cdot\left(\frac{N^3_{q_k}}{N^1_{q_k}}\right)\log_2\left(\frac{N^3_{q_k}}{N^1_{q_k}}\right)\right\} \\
&= \frac{sup(R_k)}{conf(R_k)}\left\{-conf(R_k)\log_2 conf(R_k) - p_k\log_2\frac{p_k}{n_c-1}\right\}, \qquad (18)
\end{aligned}
$$

where

$$p_k = 1.0 - conf(R_k).$$

$N^1_{q_0}$, the number of instances which belong to the child node $child_0$ corresponding to the attribute value V_{q_0} is calculated by substracting the number of instance belonging to the other child node $child_k$ from the number of all instance N.

$$
\begin{aligned}
N^1_{q_0} &= N - \sum_{k=1}^{|P_q|} N^1_{q_k} \\
&= N\left\{1.0 - \sum_{k=1}^{|P_q|}\frac{sup(R_k)}{conf(R_k)}\right\}. \qquad (19)
\end{aligned}
$$

$N_{q_0}(v_{c,j})$, the number of instances which belong to the class $v_{c,j}$ is estimated by the number of instances of the same class in other children.

$$N_{q_0}(v_{c,j}) = n(v_{c,j}) - N_q(v_{c,j}), \qquad (20)$$

where

$N_q(v_{c,j})$: number of instances whose attribute values are $\{V_{q_k}|k=1,\ldots,|P_q|\}$, and class is $v_{c,j}$,

$$N_q(v_{c,j}) = \sum_{k=1}^{|P_q|} N_{q_k}(v_{c,j}),$$

$N_{q_k}(v_{c,j})$: number of instances whose attribute values are V_{q_k} and class is $v_{c,j}$,

$$N_{q_k}(v_{c,j}) = \begin{cases} N\cdot sup(R_k) & (v_{c,j} \equiv C(\boldsymbol{H}_k)) \\ \dfrac{N\cdot\frac{sup(R_k)}{conf(R_k)}(1.0-conf(R_k))}{n_c-1} & (\text{otherwise}) \end{cases}.$$

As mentioned above, the information quantity G_{child_0} which is necessary to predict the class correctly in the child node $child_0$ is calculated as follows.

$$
\begin{aligned}
&G_{child_0} \\
&= \frac{N^1_{q_0}}{N}\sum_{j=1}^{n_c}\left\{-\left(\frac{N_{q_0}(v_{c,j})}{N^1_{q_0}}\right)\log_2\left(\frac{N_{q_0}(v_{c,j})}{N^1_{q_0}}\right)\right\}
\end{aligned}
$$

$$= \left\{ 1.0 - \sum_{k=1}^{|P_q|} \frac{sup(R_k)}{conf(R_k)} \right\} \sum_{j=1}^{n_c} \{-r_j \log_2 r_j\},$$

where

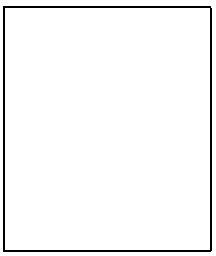
$$r_j = \frac{N_{q_0}(v_{c,j})}{N - N \sum_{k=1}^{|P_q|} \frac{sup(R_k)}{conf(R_k)}}.$$

The approximated information gain $Gain(\mathcal{A}_q)$ is, therefore, calculated by,

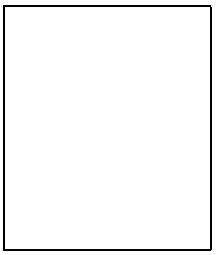
$$Gain(\mathcal{A}_q) = G_{root} - \sum_{k=1}^{|P_q|} G_{child_k} - G_{child_0}. \tag{21}$$

This leads to Equation (11).

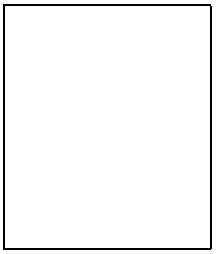
Author Biographies



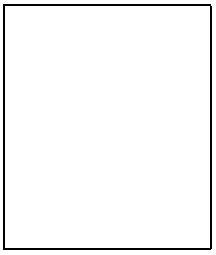
Masahiro Terabe is a staff researcher in the department of Safety Science and Policy of Mitsubishi Research Institute, Inc., Japan. His current research interests include machine learning, knowledge acquisition, knowledge discovery, data mining. He received his Bs (1993) and Ms (1995) degrees in precision engineering from Kyoto University. He is a member of AAAI, Japanese Society for Artificial Intelligence, and Society of Instrument and Control Engineers.



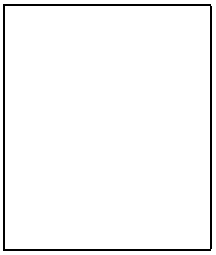
Hiroshi Motoda is a professor in the division of Intelligent Systems Science at the Institute of Scientific and Industrial Research of Osaka University. Before joining the university, he had been with Hitachi since 1967 and reached the position of a senior chief research scientist at the Advanced Research Laboratory where he headed an AI group and conducted research on machine learning, knowledge acquisition, diagrammatic reasoning and information filtering. His current research interests includes, in addition to these, scientific knowledge discovery and data mining. He received his Bs (1965), Ms (1967) and PhD (1972) degrees in nuclear engineering from university of Tokyo. He is now on the editorial board of Artificial Intelligence in Engineering, International Journal of Human-Computer Studies and Knowledge and Information Systems: An International Journal. He received the outstanding achievement award from JSAI (1999). He is a member of AAAI, IEEE Computer Society, JSAI, JSSST, IPSJ and JCSS.



Takashi Washio graduated at Dept. of Nuclear Eng., Tohoku Univ. in 1983, and took his Ms.E. and Ph.D. in the same department in 1985 and 1988 respectively. He was a visiting researcher in Nuclear Reactor Lab. of Massachusetts Institute of Technology (MIT) from 1988 to 1990, and was a senior researcher in Mitsubishi Research Institute, Inc. in Tokyo, Japan from 1990 to 1996. He has researched on the techniques of qualitative reasoning, diagnosis theory and risk analysis for large-scale plants. He became an associate professor of Institute for the Scientific and Industrial Research (ISIR), Osaka Univ. in 1996. His current research interests are automated scientific law discovery and industrial data mining techniques. He is the member of AAAI, Japanese Society for Artificial Intelligence, Society of Instrument and Control Engineers, Information Processing Society of Japan and Japan Society for Fuzzy Theory and Systems.



Osamu Katai graduated Kyoto University at the Dept. of Mechanical Engineering in 1969, then proceeded to the Graduate School of Engineering and received Ms. E. and Dr. of Engineering in 1971 and 1979, respectively. Since 1971, he has been with Kyoto University, first as an Instructor at the Dept. of Precision Engineering and now is a Professor at the Dept. of Systems Science in the Graduate School of Informatics. From 1980 to 1981, he was a visting researcher at INRIA (National Research Institute on Informatics and Automation, France). His current research interests are on the methodologies and theoretical frameworks on "symbiotic systems" where artificial systems and natural systems can coexist harmoniously. Particular interests are on bioinformatic systems, co-evolving systems, autonomous robots, agents, environmental design, ecological design, community design, etc. He is a member of Japanese Society for Artificial Intelligence, Society of Instrument and Control Engineers, etc.



Testuo Sawaragiis an associate professor at the Dept. of Precision Engineering, Graduate School of Engineering, Kyoto University, Japan. He received his B.S., M.S. and Ph.D. degrees in Systems Engineering from Kyoto University in 1981, 1983 and 1988, respectively. From 1986 to 1994, he was an instructor at the Department of Precision Mechanics, Faculty of Engineering, Kyoto University, and in 1994 he was with the current department as an associate professor. From 1991 to 1992, he was a visiting scholar at Dept. of Engineering-Economic Systems, Stanford University, USA. He has been engaged in the researches on Systems Engineering, Cognitive Science and Artificial Intelligence, particularly in the development of human-machine collaborative systems, modeling the transfer of human cognitive skills into machines. He is a member of the Society of Instrument and Control Engineers, the Institute of Systems, Control and Information Engineers, Japanese Society for Artificial Intelligence, Japan Society for Fuzzy Theory and Systems, Human Interface Society, JASME, and IEEE.

correspondence and offprint requests to: Masahiro Terabe, Safety Science and Policy Department, Mitsubishi Research Institutes, Inc., 2-6-1 Ohtemachi, Chiyoda, Tokyo 100-8141, Japan. Email:terabe@mri.co.jp