A New Algorithm for Optimization of Fuzzy Decision Tree in Data Mining

Abolfazl Kazemi\textsuperscript{a,}\textsuperscript{*}, Elahe Mehrzadegan\textsuperscript{b}

\textsuperscript{a}Assistant Professor, Faculty of Industrial and Mechanical Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran
\textsuperscript{b}MSc. Faculty of Industrial and Mechanical Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran

Received 14 April, 2010; Revised 19 May, 2010; Accepted 7 July, 2010

Abstract

Decision-tree algorithms provide one of the most popular methodologies for symbolic knowledge acquisition. The resulting knowledge, a symbolic decision tree along with a simple inference mechanism, has been praised for comprehensibility. The most comprehensible decision trees have been designed for perfect symbolic data. Classical crisp decision trees (DT) are widely applied to classification tasks. Nevertheless, there are still a lot of problems especially when dealing with numerical (continuous valued) attributes. Some of those problems can be solved using fuzzy decision trees (FDT). Over the years, additional methodologies have been investigated and proposed to deal with continuous or multi-valued data, and with missing or noisy features. Recently, with the growing popularity of fuzzy representation, a few researchers independently have proposed to utilize fuzzy representation in decision trees to deal with similar situations. Fuzzy representation bridges the gap between symbolic and non symbolic data by linking qualitative linguistic terms with quantitative data. In this paper, a new method of fuzzy decision trees is presented. This method proposed a new method for handling continuous valued attributes with user defined membership. The results of crisp and fuzzy decision trees are compared at the end.

Keywords: Data mining, Classification, Decision tree, ID3, Fuzzy

1. Introduction

Data mining is known as the core stage of Knowledge Discovery in Databases (KDD), which is defined by Fayyad et al. [9] as: “the nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data”. In recent years, there is an ongoing demand for systems capable of mining massive and continuous streams of real-world data. Such systems can be used in the fields of temperature monitoring, precision agriculture, urban traffic control, stock market analysis, network security, etc. The complex nature of real-world data has increased the difficulties and the challenges of data mining in terms of data processing, data storage, and model storage requirements [16]. Such a system can handle noise, uncertainty, and asynchrony of the real-world data [7]. Batch classification algorithms like CART [5], ID3 [26], C4.5 [28], and IFN [20] are not suitable for mining continuous data. In almost every real-life field, one is confronted with growing amounts of data coming from measurements, simulations or, simply, from manual data registration and centralization procedures, and, most often, it would be a waste not to take advantage of these data. Recent developments in data storage devices, database management systems, computer technologies, and automatic learning techniques make data analysis tasks easier and more efficient.

In this paper, a new method of fuzzy decision trees is proposed. It is a new method for handling continuous valued attributes with user defined membership.

2. Literature Review

Many decision-tree algorithms have been developed. The most famous algorithm is ID3 that is a simple decision tree learning algorithm developed by Ross Quinlan [25] whose choice of split attribute is based on information entropy. The basic idea of ID3 algorithm is to construct the decision tree by employing a top-down, greedy search through the given sets to test each attribute at every tree node.

C4.5 is an extension of ID3 developed by Prather et al. in 1997 [24]. It improves computing efficiency, deals with continuous values, handles attributes with missing values, avoids over fitting, and performs other functions. To deal with continuous data, CART (classification and regression tree) algorithms have been proposed. CART is a data-exploration and prediction algorithm similar to
In recent years, an alternative representation has grown in popularity. It is a framework which consists of a novel fuzzy inference algorithm to generate fuzzy decision trees from induced crisp decision trees proposed by Zuhair Banda in 2006 [2]. That genetic algorithm is used to optimize and automatically determine the set of fuzzy regions for all branches and simultaneously the degree in which the inference parameters will be applied.

This representation, based on fuzzy sets and used in approximate reasoning, is especially applicable to bridging the conceptual gap between subjective/ambiguous features and quantitative data. Because of the gracefulness of gradual fuzzy sets and approximate reasoning methods used, fuzzy representation is also adequate for dealing with inexact and noisy data. Fuzzy rules, based on fuzzy sets, utilize those qualities of fuzzy representation in a comprehensible structure of rule bases.

In this paper, a new method based on partitioning the continuous-valued attributes is proposed. The proposed method could be used in most algorithms for building Decision Tree without destroying their original properties.

3. Decision Trees Algorithm

In decision-tree algorithms, examples, described by features of some descriptive language and with known decision assignments, guide the tree-building process. Each branch of the tree is labeled with a condition. To reach a leaf, all conditions on its path must be satisfied. A decision-making inference procedure (class assignments in this case) matches features of new data with those conditions, and classifies the data based on the classification of the training data found in the satisfied leaf. Tree-building is based on recursive partitioning, and for computational efficiency it usually assumes independence of all attributes. ID3 and CART are the two most popular such algorithms. While ID3 aims at knowledge comprehensibility and is based on symbolic domains, CART is naturally designed to deal with continuous domains but lacks the same level of comprehensibility.

The recursive partitioning routine selects one attribute at a time, usually the one which maximizes some information measure for the training examples satisfying the conditions leading to the node. This attribute is used to split the node, using domain values of the attribute to form additional conditions leading to sub trees. Then, the same procedure is recursively repeated for each child node, with each node using the subset of the training examples satisfying the additional condition. A node is further split unless all attributes are exhausted, when all examples at the node have the same classification, or when some other criteria are met.

The recursive tree-building can be described as follows:

1. \[ I_N = -\sum_k P_k \log P_k \] Where \( C \) is the set of decisions and \( p_k \) is the probability (estimated from data) that an example found present in the node has classification \( K \).

2. For each remaining attribute \( a_i \) (previously unused on the path to \( N \)), compute the information gain based on this attribute splitting node \( N \). The gain \( G_j = I_N - \sum_j W_j I_{N_j} \), where \( D_i \) denotes the set of features associated with \( a_i \), \( L_N_j \) is the information content at the \( j \)th child of \( N \), and \( W_j \) is the proportion of \( N_j \)’s examples that satisfy the condition leading to that node.

3. Expand the node using the attribute which maximizes the gain.

The above tree-building procedure in fact creates a partition of the description space, with guiding principles such as having "large blocks" and unique classifications of training data in each of the blocks. It is quite natural to make classification decisions based on those partitions in such a way that a new data element is classified the same way as the training data from the same partition block. Of course, problems arise if a portion block contains training data without unique classifications. This may result from a number of factors, such as an insufficient set of features,
noise or errors. Another potential problem arises if a block has no training data. This may result from an insufficient data set. Following the above intuitive decision procedure, in the inference stage a new sample’s features are compared against the conditions present of the tree. This, of course, corresponds to deciding on the partition block that the new example falls into. The classification of the examples of that leaf whose conditions are satisfied by the data is returned as the algorithm’s decision. For example, assuming that the shaded node contains samples with a unique decision, any other sample described in particular by the same two features "young-age" and "blond-hair" would be assigned the same unique classification.

ID3 assumes symbolic features, and any attempt to avoid this assumption trades its comprehensibility. Quinlan has extensively investigated ID3 extensions to deal with missing features, inconsistency (when a leaf contains examples of different classes), and incompleteness (when a branch for a given feature is missing out of a node).

ID3 algorithm [5] applies to a set of data and generates a decision tree for classifying the data. Fuzzy ID3 algorithm is extended to apply to a fuzzy set of data (several data with membership grades) and generates a fuzzy decision tree using fuzzy sets defined by a user for all attributes. A fuzzy decision tree consists of nodes for testing attributes, edges for branching by test values of fuzzy sets defined by a user and leaves for deciding class names with certainties. An example of fuzzy decision trees is shown in Fig. 1 below.

![Fuzzy decision tree](image)

There are roughly a dozen publications on fuzzy decision trees (FDT). While Janikow investigates another problem of traditional DT (FDT and missing attribute values [15]), our aim is to exploit the fuzzy ID3 algorithm [27] for the addressed purposes.

4. The Proposed Algorithm

Our algorithm is very similar to ID3. However, whereas ID3 selects the test attribute based on the information gain which is computed by the probability of ordinary data, our algorithm does it by the probability of membership values for data.

The recursive tree-building can be described as defined in section 3. In the FDT described in this section, the membership function for attribute values is user defined. We use the algorithm, which is summarized in the following:

1. Generates the root node
2. Tests for leaf node (see section 2 for three conditions)
3. Finds a test attribute
   a. Divides the data according to this attribute
   b. Generates new nodes for fuzzy subsets
4. Makes recursion of the process for the new nodes from point 2.

In our method, only point 3a is modified as follows:

At first, we must define the cut points. In order to choose the cut points, first, the attribute values are arranged in an ascending order. Then, we have some possible cut points between data with different classes. The flowchart of the proposed algorithm is shown in Fig. 2.

We use the following abbreviations: the information of data (D), the class information entropy E (attribute, D) after discretization with attribute, and the information gain G (attribute, D).

For attributes A \((i = 1, 2,...)\), find the best cut point, calculate the information gains G (Ai, D), and select the test attribute Amax that maximize them.

The new method will be illustrated with data from Table 1 [29]. The data D with µ are given in Table 1 too.

First the attribute values height and weight are arranged in an ascending order (in Table 2 and 3).

From Table 3 we select the discrimination with maximum information gain among all the candidate cut points. We get the attribute weight as the testing attribute in the root node.

Two out of three new subsets completely belong to the same class (D2 = \{8\} and D3 = \{2, 3, 5\})(table 4). So, two leaf nodes are produced.
Fig 2. The flowchart of the proposed algorithm

### Table 1: Data (D)

<table>
<thead>
<tr>
<th>number</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>height</td>
<td>160</td>
<td>180</td>
<td>170</td>
<td>175</td>
<td>160</td>
<td>175</td>
<td>165</td>
<td>180</td>
</tr>
<tr>
<td>weight</td>
<td>60</td>
<td>80</td>
<td>75</td>
<td>60</td>
<td>75</td>
<td>60</td>
<td>60</td>
<td>70</td>
</tr>
<tr>
<td>hair color</td>
<td>blond</td>
<td>black</td>
<td>black</td>
<td>red</td>
<td>black</td>
<td>red</td>
<td>blond</td>
<td>blond</td>
</tr>
<tr>
<td>class</td>
<td>c₁</td>
<td>c₂</td>
<td>c₁</td>
<td>c₂</td>
<td>c₁</td>
<td>c₂</td>
<td>c₁</td>
<td>c₁</td>
</tr>
<tr>
<td>μ</td>
<td>1</td>
<td>0.8</td>
<td>0.2</td>
<td>0.7</td>
<td>1</td>
<td>0.3</td>
<td>1</td>
<td>0.5</td>
</tr>
</tbody>
</table>

### Table 2: Data sorted by height

<table>
<thead>
<tr>
<th>number</th>
<th>1</th>
<th>5</th>
<th>7</th>
<th>3</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>height</td>
<td>160</td>
<td>160</td>
<td>165</td>
<td>170</td>
<td>175</td>
<td>175</td>
<td>180</td>
<td>180</td>
</tr>
<tr>
<td>class</td>
<td>c₁</td>
<td>c₂</td>
<td>c₂</td>
<td>c₂</td>
<td>c₁</td>
<td>c₂</td>
<td>c₁</td>
<td>c₂</td>
</tr>
<tr>
<td>μ</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.2</td>
<td>0.7</td>
<td>0.3</td>
<td>0.5</td>
<td>0.8</td>
</tr>
</tbody>
</table>
Now we obtain for the hair color attribute:

\[ I(D) = 0.971 \]

\[ E(hair\ color, D) = \frac{2.5}{5.5} \times \left( -\frac{1.5}{2.5} \log \frac{1.5}{2.5} - \frac{1.0}{2.5} \log \frac{1.0}{2.5} \right) + \]
\[ \frac{2.0}{5.5} \times \left( -\frac{0.0}{2.0} \log \frac{0.0}{2.0} - \frac{2.0}{2.0} \log \frac{2.0}{2.0} \right) + \]
\[ \frac{1.0}{5.5} \times \left( -\frac{0.7}{1.0} \log \frac{0.7}{1.0} - \frac{0.3}{1.0} \log \frac{0.3}{1.0} \right) \]
\[ = -0.602 \]

\[ G(hair\ color, D) = I(D) - E(hair\ color, D) = 0.369 \]

\[ E(height, D, cut\ point\_1) = 0.939 \]
\[ E(height, D, cut\ point\_2) = 0.971 \]
\[ E(height, D, cut\ point\_1\_2) = 0.909 \]
\[ E(weight, D, cut\ point\_1) = 0.867 \]
\[ E(weight, D, cut\ point\_2) = 0.606 \]
\[ E(weight, D, cut\ point\_1\_2) = 0.538 \]
\[ G(height, D, cut\ point\_1) = 0.032 \]
\[ G(height, D, cut\ point\_2) = 0.000 \]
\[ G(height, D, cut\ point\_1\_2) = 0.062 \]
\[ G(weight, D, cut\ point\_1) = 0.104 \]
\[ G(weight, D, cut\ point\_2) = 0.365 \]
\[ G(weight, D, cut\ point\_1\_2) = 0.433 \]

For the third subset \( D_1 \) = \{4, 6, 7\} in Table 5 we have to repeat the induction process.

Table 4

<table>
<thead>
<tr>
<th>number</th>
<th>1</th>
<th>4</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>5</th>
<th>3</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>weight</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>70</td>
<td>75</td>
<td>75</td>
<td>80</td>
</tr>
<tr>
<td>class</td>
<td>( c_1 )</td>
<td>( c_1 )</td>
<td>( c_2 )</td>
<td>( c_2 )</td>
<td>( c_1 )</td>
<td>( c_1 )</td>
<td>( C_2 )</td>
<td>( C_2 )</td>
</tr>
<tr>
<td>( \mu )</td>
<td>1</td>
<td>0.7</td>
<td>0.3</td>
<td>1</td>
<td>0.5</td>
<td>1</td>
<td>0.2</td>
<td>0.8</td>
</tr>
</tbody>
</table>

For the third subset \( D_1 \) = \{4, 6, 7\} in Table 5 we have to repeat the induction process.

Table 5

<table>
<thead>
<tr>
<th>number</th>
<th>1</th>
<th>4</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>height</td>
<td>160</td>
<td>165</td>
<td>175</td>
<td>165</td>
</tr>
<tr>
<td>weight</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>hair color</td>
<td>blond</td>
<td>red</td>
<td>red</td>
<td>blond</td>
</tr>
<tr>
<td>class</td>
<td>( c_1 )</td>
<td>( c_1 )</td>
<td>( c_2 )</td>
<td>( c_2 )</td>
</tr>
<tr>
<td>( \mu )</td>
<td>1</td>
<td>0.7</td>
<td>0.3</td>
<td>1</td>
</tr>
</tbody>
</table>

Possible discretizations of the height and hair color attributes are shown in Table 6:

Table 6

<table>
<thead>
<tr>
<th>number</th>
<th>1</th>
<th>7</th>
<th>4</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>height</td>
<td>160</td>
<td>165</td>
<td>175</td>
<td>175</td>
</tr>
<tr>
<td>class</td>
<td>( c_1 )</td>
<td>( c_1 )</td>
<td>( c_1 )</td>
<td>( c_2 )</td>
</tr>
<tr>
<td>( \mu )</td>
<td>1</td>
<td>1</td>
<td>0.7</td>
<td>0.3</td>
</tr>
</tbody>
</table>

Now we obtain for the hair color attribute:

\[ I(D) = 0.971 \]

\[ G(hair\ color, D) = 0.027 \]

The maximum information gain favors the height attribute (with two cut points) to be used for discretization in this node (in table 7).

Table 7

<table>
<thead>
<tr>
<th>number</th>
<th>1</th>
<th>7</th>
<th>4</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>height</td>
<td>160</td>
<td>165</td>
<td>175</td>
<td>175</td>
</tr>
<tr>
<td>class</td>
<td>( c_1 )</td>
<td>( c_1 )</td>
<td>( c_1 )</td>
<td>( c_2 )</td>
</tr>
<tr>
<td>( \mu )</td>
<td>1</td>
<td>1</td>
<td>0.7</td>
<td>0.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 8 Results of the proposed algorithm

<table>
<thead>
<tr>
<th>Data Set</th>
<th>#Attribute</th>
<th>#Classes</th>
<th>Samples</th>
<th>ID3</th>
<th>New-Alg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Out Look</td>
<td>4</td>
<td>3</td>
<td>16</td>
<td>8</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1.3</td>
<td>1</td>
</tr>
<tr>
<td>*Postoperative Patient</td>
<td>8</td>
<td>3</td>
<td>90</td>
<td>30</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.1</td>
<td>3</td>
</tr>
</tbody>
</table>

- If there are no more attributes for classification (in ID3 there is a null class for this leaf node).
- More than one class name may be assigned to one leaf node (the real advantage of FDT).
- The fuzzy sets of all attributes are user defined. Each attribute is considered as a linguistic variable. (In our opinion, this is not necessary. The membership function can be calculated from the boundary points of the interval using the algorithm in section 4.)

5. Conclusion and Future Work

This paper proposes a method that is based on partitioning the continuous-valued attributes. The suggested method could be used in most algorithms for building DT (e.g. Fuzzy ID3) without destroying their original properties. Future work should be done in fuzzy classification of larger data sets and in investigations of more fuzzy operators for "AND" and "OR", respectively. We plan to extend our experiments with a novel cut-point-strategy. In this case, the resulting cut points depend on the density of attribute values. This algorithm can also be developed with changing the final condition for accepting the assumptions that are not given training pair.

6. References


