A Classification Method using Decision Tree for Uncertain Data

Annie Mary Bhavitha S¹, Sudha Madhuri²

¹Pursuing M.Tech(CSE), Nalanda Institute of Engineering & Technology, Siddharth Nagar, Sattenapalli, Guntur, Affiliated to JNTUK, Kakinada, A.P., India.

²Asst. Professor, Department of Computer Science Engineering, Nalanda Institute of Engineering & Technology, Siddharth Nagar, Sattenapalli, Guntur, Affiliated to JNTUK, Kakinada, A.P., India.

Abstract - The Decision Tree is one of the most popular classification algorithms in current use in Data Mining and Machine Learning. They are also used in many different disciplines including medical diagnosis, cognitive science, artificial intelligence, game theory, engineering. This paper presents an algorithm for building decision trees in an uncertain environment. Our algorithm will use the theory of belief functions in order to represent the uncertainty about the parameters of the classification problem. Our method will be concerned with both the decision tree building task and the classification task. The theory of belief functions provides a non-Bayesian way of using mathematical probability to quantify subjective judgments. Whereas a Bayesian assesses probabilities directly for the answer to a question of interest, a belief-function user assesses probabilities for related questions and then considers the implications of these probabilities for the question of interest.

Keywords: Decision tree, uncertain data, Classification.

I. INTRODUCTION

Decision trees are one of the most widely used classification techniques especially in artificial intelligence. Their popularity is basically due to their ability to express knowledge in a formalism that is often easier to interpret by experts and even by ordinary users. Despite their accuracy when precise and certain data are available, the classical versions of decision tree algorithms are not able to handle the uncertainty in classification problems. Hence, their results are categorical and do not convey the uncertainty that may occur in the attribute values or in the case class.

In this paper, we present a classification method based on the decision tree approach having the objective to cope with the uncertainty that may occur in a classification problem and which is basically related to human thinking, reasoning and cognition.

Our algorithm will use the belief function theory as understood in the transferable belief model (TBM) [1, 2] and which seems offering a convenient framework thanks to its ability to represent epistemological uncertainty. Moreover, the TBM allows experts to express partial beliefs in a much more flexible way than probability functions do. It also allows to handle partial or even total ignorance concerning classification parameters. In addition to these advantages, it offers appropriate tools to combine several pieces of evidence.

This paper is composed as follows: we start by introducing decision trees, then we give an overview of the basic concepts of the belief function theory. In the main part of the paper, we present our decision tree algorithm based on the evidence theory. The two major phases will be detailed: the building of a decision tree and the classification task. Our algorithm will be illustrated by an example in order to understand its real unfolding.
II. DECISION TREES

Decision trees present a system using a top-down strategy based on the divide and conquer approach where the major aim is to partition the tree in many subsets mutually exclusive. Each subset of the partition represents a classification sub problem. A decision tree is a representation of a decision procedure allowing to determine the class of a case. It is composed of three basic elements [3]:
- Decision nodes specifying the test attributes.
- Edges corresponding to the possible attribute outcomes.
- Leaves named also answer nodes and labeled by a class.
The decision tree classifier is used in two different contexts:
1. Building decision trees where the main objective is to find at each decision node of the tree, the best test attribute that diminishes, as much as possible, the mixture of classes with each subset created by the test.
2. Classification where we start by the root of the decision tree, then we test the attribute specified by this node. The result of this test allows to move down the tree branch relative to the attribute value of the given example. This process will be repeated until a leaf is encountered. So, the case is classified by tracing out a path from the root of the decision tree to one of its leaves [4].

III. BELIEF FUNCTION THEORY

The theory of belief functions is based on two ideas: the idea of obtaining degrees of belief for one question from subjective probabilities for a related question, and Dempster’s rule for combining such degrees of belief when they are based on independent items of evidence. We can derive degrees of belief for statements made by witnesses from subjective probabilities for the reliability of these witnesses. Degrees of belief obtained in this way differ from probabilities in that they may fail to add up to 100%. Suppose, for example, that Betty tells me a tree limb fell on my car. My subjective probability that Betty is reliable is 90%; my subjective probability that she is unreliable is 10%. Since they are probabilities, these numbers add up to 100%. But Betty’s statement, which must be true if she is reliable, is not necessarily false if she is unreliable. From her testimony alone, I can justify a 90% degree of belief that a limb fell on my car, but only a 0% (not 10%) degree of belief that no limb fell on my car. (This 0% does not mean that I am sure that no limb fell on my car, as a 0% probability would; it merely means that Betty's testimony gives me no reason to believe that no limb fell on my car.) The 90% and the 0%, which do not add to 100%, together constitute a “belief function.”

In this example, we are dealing with a question that has only two answers (Did a limb fall on my car? Yes or no.). Belief functions can also be derived for questions for which there are more than two answers. In this case, we will have a degree of belief for each answer and for each set of answers. If the number of answers (or the size of the “frame”) is large, the belief function may be very complex.

Let Θ be the frame of discernment representing a finite set of elementary hypotheses related to a problem domain. We denote by \(2^\Theta\) the set of all the subsets of \(\Theta\). To represent degrees of belief, Shafer [5] introduces the so-called basic belief assignment (called initially basic 'probability' assignments, an expression that has created serious confusion). They quantify the part of belief that supports a subset of hypotheses without supporting any strict subset of that set by lack of appropriate information [2]. A basic belief assignment (bba) is a function denoted \(m\) that assigns a value in [0, 1] to every subset \(A\) of \(\Theta\). This function \(m\) is defined here by:

\[
m : 2^\Theta \to [0,1] \text{ such that } m(\emptyset) = 0 \text{ and } \sum_{A \subseteq \Theta} m(A) = 1
\]

The subsets \(A\) of the frame of discernment \(\Theta\) which \(m(A)\) are strictly positive, are called focal elements of the bba. The credibility \(\text{Bel}\) and the plausibility \(\text{Pl}\) are defined by:

\[
\text{Bel}(A) = \sum_{B \subseteq A} m(B)
\]
\[
\text{Pl}(A) = \sum_{A \subseteq B \neq \emptyset} m(B)
\]

The quantity \(\text{Bel}(A)\) expresses the total belief fully committed to the subset \(A\) of \(\Theta\). \(\text{Pl}(A)\) represents the maximum amount of belief that might support the subset \(A\). Within the belief function model, it is easy to express the state of total ignorance. This is done by the so-called vacuous belief function which only focal element is the frame of discernment \(\Theta\). It is defined by [5]:

\[
m = \frac{1}{|\Theta|} \cdot \text{Pl}
\]
m(Θ) = 1 and m(A) = 0 for A ≠ Θ.

**IV. DECISION TREE USING THE BELIEF FUNCTION THEORY**

In this section, we detail our decision tree algorithm based on the belief function theory. First, we present the decision tree building phase, then the classification phase. The two phases will be illustrated by examples in order to understand their unfolding.

**4.1 Decision tree building phase**

In this part, we define the main parameters of a decision tree within the belief function framework, then we present our algorithm for building such decision trees.

We propose the following steps to build the tree:

1. Compute the average pignistic probability function BetPT taken over the training set T. Then compute the entropy of the class distribution in T. This value $\text{Info}(T)$ is equal to:

$$
\text{Info}(T) = -\sum_{i=1}^{n} \text{BetP}_{T_i}(C_i) \log_2 \text{BetP}_{T_i}(C_i)
$$

Compute the information gain provided by each attribute A as:

$$\text{Gain}(T, A) = \text{Info}(T) - \text{InfoA}(T).$$

2. Our task is at first to define $\text{InfoA}(T)$ for each attribute. The idea is to apply the same procedure as in the computation of $\text{Info}(T)$, but restricting ourselves to the set of objects that share the same value for the attribute A and averaging these conditional information measures. For each attribute value $a_m$, we build the subset $T_m$ made of the cases in T whose value for the attribute is $a_m$. We compute the average belief function $\text{BelPT}_m$, then apply the pignistic transformation to it in order to compute the pignistic probability $\text{BetPT}_m$. From it, we compute $\text{Info}(T_m)$ where $T_m$ represents the training subset when the value of the attribute A is equal to $a_m$.

3. $\text{InfoA}(T)$ will be equal to the weighed sum of the different $\text{Info}(T_m)$ relative to the considered attribute. These $\text{Info}(T_m)$ will be weighted by the proportion of each attribute value in the training set.

$$
\text{InfoA}(T) = \sum_{m=1}^{k} \left| T_m \right| \text{Info}(T_m)
$$

$$
= -\sum_{m=1}^{k} \left| T_m \right| \sum_{i=1}^{n} \text{BetP}_{T_m}(C_i) \log_2 \text{BetP}_{T_m}(C_i)
$$

4. Once the different attribute information gains are computed, we choose the attribute with the highest value of the information gain.

**4.2. Decision tree building algorithm:**

Let $T$ be a training set composed by objects characterized by $l$ symbolic attributes $(A1, A2, ..., Al)$ and that may belong to the set of classes $\Theta = \{C1, C2, ..., Cn\}$. For each object $I_j$ ($j = 1..p$) of the training set will correspond a basic belief assignment expressing the quantity of beliefs exactly committed to the subsets of classes.

Our algorithm which uses a Top-Down Induction of Decision Trees (TDIDT) approach, will have the same skeleton as an ID3 algorithm [6]. Their steps are described as follows:

1. Generate the root node of the decision tree including all the objects of the training set.

2. Verify if this node satisfies or not the stopping criterion:

   • If yes, declare it as a leaf node and compute its corresponding bba as we mentioned in the last section.
   • If not, look for the attribute having the highest information gain. This attribute will be designed as the root of the decision tree related to the whole training set.

3. Apply the partitioning strategy by developing an edge for each attribute value chosen as a root. This partition leads to several training subsets.

4. Repeat the same process for each training subset from the step 2 while verifying the stopping criterion. If this latter is satisfied, declare the node as a leaf and compute its assigned bba, else repeat the same process.

5. Stop when all the nodes of the latter level of the tree are leaves.

We have to mention that we get the same results as ID3 if all the bba are 'certain'. That is when the class
assigned for each training example is unique and known with certainty.

**EXAMPLE 1:**
Now, we present a simple example illustrating our decision tree building algorithm within a belief function framework. Let T be a small training set (see table 1). It is composed of five objects characterized by three symbolic attributes defined as following:

- Eyes = {Brown, Blue};
- Hair = {Dark, Blond};
- Height = {Short, Tall}

As we work in a supervised learning context, the possible classes are already known. We denote them by C1, C2 and C3. For each object Ij (j = 1..5) belonging to the training set T, we assign a bba mj expressing our beliefs on its actual class. These functions are defined on the same frame of discernment Θ = {C1, C2, C3}.

<table>
<thead>
<tr>
<th>Ij</th>
<th>Eyes</th>
<th>Hair</th>
<th>Height</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>I1</td>
<td>Brown</td>
<td>Dark</td>
<td>Short</td>
<td>m1</td>
</tr>
<tr>
<td>I2</td>
<td>Brown</td>
<td>Dark</td>
<td>Tall</td>
<td>m2</td>
</tr>
<tr>
<td>I3</td>
<td>Blue</td>
<td>Blond</td>
<td>Tall</td>
<td>m3</td>
</tr>
<tr>
<td>I4</td>
<td>Blue</td>
<td>Dark</td>
<td>Tall</td>
<td>m4</td>
</tr>
<tr>
<td>I5</td>
<td>Blue</td>
<td>Dark</td>
<td>Short</td>
<td>m5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>C1</th>
<th>C2</th>
<th>C3</th>
<th>C1 ∪C2</th>
<th>C1 ∪C3</th>
<th>C2 ∪C3</th>
<th>Θ</th>
</tr>
</thead>
<tbody>
<tr>
<td>BelTbr</td>
<td>0.15</td>
<td>0.25</td>
<td>0</td>
<td>0.7</td>
<td>0.15</td>
<td>0.25</td>
<td>1</td>
</tr>
<tr>
<td>mTbr</td>
<td>0.15</td>
<td>0.25</td>
<td>0</td>
<td>0.3</td>
<td>0.3</td>
<td>0</td>
<td>0.3</td>
</tr>
<tr>
<td>BelTbl</td>
<td>0.27</td>
<td>0.27</td>
<td>0.1</td>
<td>0.54</td>
<td>0.37</td>
<td>0.43</td>
<td>1</td>
</tr>
<tr>
<td>mTbl</td>
<td>0.27</td>
<td>0.27</td>
<td>0.1</td>
<td>0</td>
<td>0.06</td>
<td>0.3</td>
<td></td>
</tr>
</tbody>
</table>

where
- \( m_1(C_1) = 0.3; \) \( m_1(C_1 \cup C_2) = 0.4; \) \( m_1(\Theta) = 0.3; \)
- \( m_2(C_2) = 0.5; \) \( m_2(C_1 \cup C_2) = 0.2; \) \( m_2(\Theta) = 0.3; \)
- \( m_3(C_1) = 0.8; \) \( m_3(\Theta) = 0.2; \)
- \( m_4(C_2) = 0.1; \) \( m_4(C_1) = 0.3; \) \( m_4(C_2 \cup C_3) = 0.2; \)
- \( m_5(\Theta) = 0.4; \)
- \( m(C_2) = 0.7; \) \( m(\Theta) = 0.3; \)

In order to find the root relative to the decision tree, we have to compute the average belief function BelT related to the whole training set T. BelT and its corresponding bba mT are presented in the following table (see table 2):

<table>
<thead>
<tr>
<th></th>
<th>C1</th>
<th>C2</th>
<th>C3</th>
<th>C1 ∪C2</th>
<th>C1 ∪C3</th>
<th>C2 ∪C3</th>
<th>Θ</th>
</tr>
</thead>
<tbody>
<tr>
<td>BelTbr</td>
<td>0.22</td>
<td>0.26</td>
<td>0.06</td>
<td>0.6</td>
<td>0.28</td>
<td>0.36</td>
<td>1</td>
</tr>
<tr>
<td>mTbr</td>
<td>0.22</td>
<td>0.26</td>
<td>0.06</td>
<td>0.12</td>
<td>0</td>
<td>0.04</td>
<td>0.5</td>
</tr>
</tbody>
</table>

The pignistic transformation of mT gives as results:

\[
\text{BetPT}(C_1) = 0.38; \quad \text{BetPT}(C_2) = 0.44; \\
\text{BetPT}(C_3) = 0.18; \\
\text{Hence}
\]
\[
\text{Info}(T) = - \sum_{i=1}^{3} \text{BetPT}_i(C_i) \log \text{BetPT}_i(C_i) = 1.496
\]

Once the entropy related to the whole set T is calculated, the second step is to find the information gain of each attribute in order to choose the root of the decision tree.

Let's illustrate the computation for the eye attribute. Let BelTbr be the average belief function relative to the objects belonging to T and having brown eyes whereas, BelTbl for the ones having blue eyes. mTbr, mTbl, BetPTbr and BetPTbl are respectively the bba and the pignistic probability relative to the values brown and blue of the eyes (see table 3 and table 4).

\[
\text{Gain}(T, \text{Eyes}) = \text{Info}(T) - \text{Infoeys}(T) = 0.0228
\]

By similar analysis for the hair and height attributes, we get:

\[
\text{Gain}(T, \text{Hair}) = 0.1876; \quad \text{Gain}(T, \text{Height}) = 0.0316;
\]

According to the gain criterion, the hair attribute will be chosen as the root of the decision tree and branches are created below the root for each of its possible value
(Dark, Blond). So, we get the following decision tree (see figure 1):

![Decision Tree](image)

Figure 1: First generated decision tree

We notice that the training subset Tblo contains only one example, thus the stopping criterion is satisfied. The node relative to Tblo is therefore declared as a leaf defined by the bba m3 of the example I3. For the subset Tda, we apply the same process as we did for T until the stopping criterion holds. The final decision tree induced by our algorithm is given by (see figure 2):

![Decision Tree](image)

Figure 2: The final decision tree

4.3 Case’s Classification

Once the decision tree is constructed, the following phase will be the classification of unseen examples referring to as new objects. On one hand, our algorithm is able to ensure the standard classification where the unseen example attribute values are assumed to be certain. As in an ordinary tree, it consists on starting from the root node and repeating to test the attribute at each node by taking into account the attribute value until reaching a leaf. Contrary to the classical decision tree where a unique class is attached to the leaf, in our decision tree, the unseen example classes will be defined by a basic belief assignment related to the reached leaf. In order to make a decision and to get the probability of each singular class, we propose to apply the pignistic transformation to the basic belief assignment related to the reached leaf, and to use this probability distribution to compute the expected utilities required for optimal decision making.

On the other hand, as we deal with an uncertain context, our classification method allows also classifying unseen examples characterized by uncertainty in the values of their attributes. In our method, we assume that new examples to classify are not only described by certain attribute values but may also be characterized by means of disjunction values for some attributes. They may even have attributes with unknown values.

EXAMPLE 2:

Let’s continue example 1 and assume that an unseen example is characterized by:

Hair = Dark; Eyes = Blue or Brown; Height = Tall.

Using the decision tree (see figure 2) relative to the training set T, gives us two possible leaves for this case:

- The first leaf characterized by m2 as a b.p.a. This leaf is induced by the path corresponding to dark hair, brown eyes and tall as height.
- The second is the one corresponding to the Path defined by dark hair, blue eyes and tall as height.

This leaf is labeled by the b.p.a m4. By applying the disjunctive rule of combination, we get m24 = m2 v m4 defined by:

\[ m24(C2) = 0.05; m24(C1 U C2) = 0.02; m24(C2 U C3) = 0.25; m24(\Theta) = 0.68; \]

Thus, the unseen example classes are described by m24. Applying the pignistic transformation on m24 gives us:

\[ \text{BetP}24(C1) = 0.24; \text{BetP}24(C2) = 0.41; \text{BetP}24(C3) = 0.35; \]

It seems that the most probable class for this example to belong is C2 with the probability of 0.41.

V. CONCLUSION

In this paper, we propose an algorithm to generate a decision tree under uncertainty within the belief function framework. The interest of the TBM appears essentially in its ability to cope with partial ignorance, and at the level of the leaves conjunctive and disjunctive rules can be used in a coherent way as they provide conjunctive and disjunctive aggregation rules.
First, we have interested to the decision tree building phase by taking into consideration the uncertainty characterized the classes of the training examples. Next, we have ensured the classification task of new examples where some of their attribute values are assumed to be uncertain. Either in the decision tree building task or in the classification task, the uncertainty is handled within the theory of belief functions which presents a convenient framework for coping with lack of information.

REFERENCES


AUTHORS PROFILE

S. Annie mary Bhavitha Pursuing M.Tech(CSE) from Nalanda Institute of Engineering & Technology, Siddharth Nagar, Sattenapalli, Guntur Affiliated to JNTUK, Kakinada, A.P., India. My research Interests are Data mining.

M. Sudha Madhuri, working as Asst. Professor, Department of Computer Science Engineering at Nalanda Institute of Engineering & Technology, Siddharth Nagar, Sattenapalli, Guntur Affiliated to JNTUK, Kakinada, A.P., India. My research Interests are Data Mining and Computer Networks.