Learning decision trees from uncertain data with an evidential EM approach

Nicolas Sutton-Charani*, Sébastien Destercke†, Thierry Denœux‡
*, †,‡ Université Technologie de Compiègne
UMR 7253 Heudiasyc
Rue Roger Couttolenc
60203 COMPIEGNE Cedex FRANCE
†UMR MISTEA and IATE
Campus de la Gaillarde, 2 place Pierre Viala
34060 MONTPELLIER Cedex - FRANCE

***Abstract*—In real world applications, data are often uncertain or imperfect. In most classification approaches, they are transformed into precise data. However, this uncertainty is an information in itself which should be part of the learning process. Data uncertainty can take several forms: probabilities, (fuzzy) sets of possible values, expert assessments, etc. We therefore need a flexible and generic enough model to represent and treat this uncertainty, such as belief functions. Decision trees are well known classifiers which are usually learned from precise datasets. In this paper we propose a methodology to learn decision trees from uncertain data in the belief function framework. In the proposed method, the tree parameters are estimated through the maximization of an evidential likelihood function computed from belief functions, using the recently proposed E^2M algorithm that extends the classical EM. Some promising experiments compare the obtained trees with classical *CART* decision trees.**

***Keywords*—classification; decision trees; belief functions; algorithm EM;**

I. Introduction

Handling data uncertainty is an old and well-known problem in statistics and applied sciences [1], [2]. Most of the time, this uncertainty is removed from samples and datasets, for instance by replacing uncertain data with precise ones [3]. By doing so, a part of the information that should be integrated to the estimation process is lost.

Data uncertainty is of epistemic nature, as data can be assumed to have a unique true value which may be ill-known due to unreliable or imprecise information. Such uncertainty is most often described either by (fuzzy) sets or (partially specified) subjective probabilities. In the past decades, the capacity of probability models to faithfully represent such uncertainty have been questioned by imprecise probabilities tenants [4, chapter 5], [5, chapter 3]. For instance, probabilities cannot distinguish between uniformity due to ignorance (such as an imprecise observation) and uniformity as a physical property. There is therefore a need for a more general framework able to model non-probabilistic uncertainty. The theory of belief functions [6], [7] or Dempster-Shafer theory (DST) satisfies this need by blending probability and set representations. It includes as special cases many uncertainty models such as sets, possibility distributions, probability distributions, ...

The DST was first presented as a statistical approach by Dempster [6] and then extended by Shafer [7] to also deal with non-statistical uncertainty. Most of evidential works then focused on problems such as information fusion or uncertain reasoning [8], [9], [10]. This paper, however, is focused on the issue of learning and predicting from uncertain data.

The EM algorithm, also proposed by Dempster [2], is a means of learning from incomplete data by maximizing a likelihood function computed from set-valued observations. The evidential EM (E^2M) was recently proposed by Denœux [11] as an extension of the EM algorithm that is able to handle uncertain data modeled by belief functions.

In this paper we apply the E^2M to a well-known statistical model: classification decision trees. Indeed, learning decision trees from uncertain data remains a challenge, as there are only few methods tackling this problem in the literature. This paper is organized as follows: after providing the basic background on decision trees and belief function theory in Section II, we present the proposed methodology in Section III. Finally, experiments are given in Section IV to show how the proposed methodology can enhance classical decision trees accuracy.

II. Background

After introducing the needed notations, this section recalls the basis of usual classification trees and the necessary tools of the DST. Finally a brief overview of decision tree methodologies that take into account different types of uncertainty is presented.

A. Formalism

As in any classification problem, the goal is to learn a model to predict the *class label* y of an instance with *attribute* (or feature) values x. During the learning phase, the classifier is built from a learning dataset LD containing samples of (X,Y) joint realisation. Its accuracy is then evaluated on a test dataset TD by comparing the model predictions with the corresponding observed classes.

The attributes $X = (X^1,\dots,X^J)$ take their values on $\Omega_X = \Omega_{X^1} \times \cdots \times \Omega_{X^J}$, the class Y on $\Omega_Y = \{\omega_1,\dots,\omega_K\}$. That is, K different classes can be predicted using J different attributes. Spaces Ω_{X^j} can be categorical or continuous.

A *precise* dataset containing N samples is a set of realisations of the couple (X,Y) and is denoted by

$$D = \begin{pmatrix} x_1, y_1 \\ \vdots \\ x_N, y_N \end{pmatrix} = \begin{pmatrix} x_1^1, \ldots, x_1^J, y_1 \\ \vdots \\ x_N^1, \ldots, x_N^J, y_N \end{pmatrix}.$$

Samples are here assumed to be i.i.d.

B. Decision tree

Decision trees are well-known classifiers [12] that combine good performances with a simple graphical representation allowing for an easy interpretation.

A decision tree is formally a rooted tree structure. Terminal nodes are called *leaves*. To each non-terminal node of the tree is associated an attribute, and to each branch issued from this node is associated a condition on this attribute that determines which data of the sample D go into that branch. A given decision tree therefore determines a partition of the space Ω_X where each element of the partition is associated to a leaf. In the sequel, we will use interchangeably the term *leaf* and its associated partition element.

We will describe a tree

$$\mathscr{P}_H = \{t_1, \ldots, t_H\}$$

by its set of leaves $t_1, \ldots, t_H$. The *leaf index* variable is then defined relatively to $\mathscr{P}_H$ by:

$$Z_H \in \Omega_{Z_H} = \{1, \ldots, H\}$$

In order to lighten the notations, Z_H will be simply denoted Z when there is no ambiguity about the tree to which it refers.

We can associate to each leaf t_h the following quantities:

- by an abuse of language we will denote by t_h the partition element $t_h = A_h^1 \times \ldots \times A_h^J$ where $A_h^j \subseteq \Omega_{X^j}$, $j = 1, \ldots, J$ is associated to branches leading to t_h. By definition, for any two distinct leaves t_h and $t_{h'}$ we have $t_h \cap t_{h'} = \emptyset$.
- the probability $\pi_h = P(Z = h)$ of leaf t_h
- a multinomial distribution $\alpha_h = (\alpha_h^1, \ldots, \alpha_h^K)$ where $\alpha_h^k = P(Y = \omega_k \,|\, Z = h)$ is the probability of the k^{th} class inside leaf t_h.

Usually, if a data item falls into leaf t_h, the predicted class is $\arg\max_{\omega_k \in \Omega_Y} \alpha_h^k$, i.e. the class with highest probability.

The parameters of a tree $\mathscr{P}_H$ can be summarized as a multidimensional parameter θ_H:

$$\theta_H = \begin{pmatrix} \alpha_1^1 & \cdots & \alpha_1^K & \pi_1 \\ \vdots & \ddots & \vdots & \vdots \\ \alpha_H^1 & \cdots & \alpha_H^K & \pi_H \end{pmatrix} \in \Theta_H$$

where Θ_H is the space of all possible parameters given a tree containing H leaves. In the case of precise data, the parameters π_h, α_h^k, $h = 1, \ldots, H, k = 1, \ldots, K$ are simply estimated by proportions:

$$\widehat{\pi_h} = |\{x_i : x_i \in t_h, i = 1, \ldots, N\}| / N \quad (1)$$

$$\widehat{\alpha_h^k} = \frac{|\{(x_i, y_i) : x_i \in t_h, y_i = \omega_k, i = 1, \ldots, N\}|}{|\{x_i : x_i \in t_h, i = 1, \ldots, N\}|} \quad (2)$$

These notations are unusual to describe decision trees, but they will be instrumental when introducing the learning problem with uncertain data. We can already notice that the distributions $\pi_h, \alpha_h, i = 1, \ldots, H$ can be viewed as a mixture of multinomial distributions. The next small example illustrates those notations.

Example 1 Let us consider a data set with $N = 12$ data items, $J = 2$ attributes and $K = 3$ classes. Spaces are described as follows:

$$\Omega_{X^1} = [1, 10], \quad \Omega_{X^2} = [13, 150], \quad \Omega_Y = \{a, b, c\}.$$

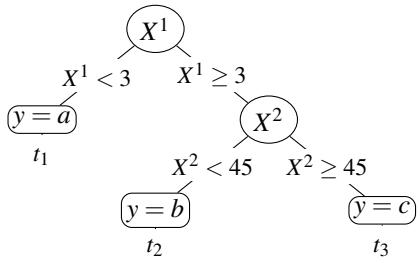

Fig. 1. Decision tree illustration

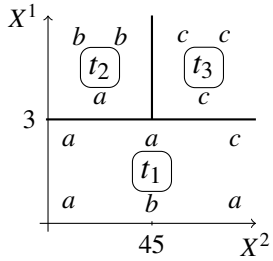

Fig. 2. Partitioned attribute space

Figures 1 and 2 respectively represent a possible decision tree and its corresponding partition of the attribute space with the learning dataset represented in it. To leaf t_2 are associated the values $A_2^1 \times A_2^2 = [3, 10] \times [13, 45[$, $\widehat{\pi_2} = 3/12$, $\widehat{\alpha_2} = (1/3, 2/3, 0)$, and the prediction for any test data item in t_2 (for example, $x^1 = 6$ and $x^2 = 30$) will be $y = b$.

As illustrated in Fig. 1, each non-terminal node of a tree is associated to a split of its parent. There are two main kinds of decision trees in the literature: CART trees [12], where each split is binary (as in Example 1) and C4.5 [13] where a non-terminal node is split in as much branches as the number of modalities of the variable associated to the node (preliminary discretization can be required). In this paper, the approach will be applied on CART trees, but it extends easily to C4.5.

The construction of a CART tree goes as follows: given a tree $\mathscr{P}_H = \{t_1, \ldots, t_H\}$, for every leaf t_h, find the best split s among all possible splits that divides t_h into two new leaves t_L, t_R and the resulting tree is $\mathscr{P}_{H+1} = \{t_1, \ldots, t_{h-1}, t_L, t_R, t_{h+1} \ldots, t_H\}$. The construction starts with the root node $\mathscr{P}_1$ and proceeds recursively. By a small abuse of notation we denote by $\mathscr{P}_H(s) = \mathscr{P}_{H+1}$ the tree obtained from $\mathscr{P}_H$ with the split s, and by $\theta_H(s) = \theta_{H+1}$ its parameter.

The quality of a split s dividing a node t_h is usually measured by how well it separates its classes. This is measured by a so-called impurity measure i (a *pure node* is a node that contains samples of the same class).

The tree learning requires to specify some stopping criteria such as minimum purity gain threshold, maximum number of leaves, maximum tree depth, ... Those criteria are called *pre-pruning* whereas *post-pruning* procedures aim to reduce the tree once it is fully grown. Both procedures strive to make the tree more readable and to reduce the risk of overfitting of the tree model without diminishing too much its predictive accuracy.

In our methodology we retained as measure of informative content the usual entropy of the distribution α_h of a leaf t_h, defined by

$$i(t_h) = -\sum_{k=1}^{K} \alpha_h^k \log(\alpha_h^k).$$

The quality $\Delta i(\theta_H(s), \theta_H)$ of the split s is then measured as

$$\Delta i(\theta_H(s), \theta_H) = i(t_h) - \frac{\pi_L}{\pi_L + \pi_R} i(t_L) - \frac{\pi_R}{\pi_L + \pi_R} i(t_R) \qquad (3)$$

C. Belief function

Let Ω_W be a finite set representing all the possible values of a variable W whose value is uncertain. In the case of uncertainty over a single sample or datum, this uncertainty concerns a fixed but ill-known value, hence is of epistemic nature (as opposed to aleatory uncertainty). As said in the Introduction, such type of uncertainty may be modelled by a variety of representations, including subjective probabilities but also sets, possibility distributions, p-boxes, etc. DST offers a generic and unifying framework to model all these representations.

A *mass function* or a basic belief assignment (*bba*) m^W on W, the basic tool of DST, is a function from the *powerset* 2^{Ω_W} of W to $[0,1]$ verifying $\sum_{B \in 2^{\Omega_W}} m^W(B) = 1$. A set $A \in 2^{\Omega_W}$ is a *focal element* of m^W if $m^W(A) > 0$. The particular case of probabilities is retrieved when all the focal elements are singletons (i.e., $m^W(A) > 0$ iff $|A| = 1$), while an imprecise observation A is modelled by committing all the mass to the set A (i.e., $m^W(A) = 1$). Ignorance or missing data are just an extreme case of such imprecision, and a missing data sample W can be modelled by $m^W(\Omega_W) = 1$. Another interesting possibility is to assign a reliability value $1-\gamma$ to an information source. $1-\gamma$ can then be considered as the belief degree that the source is relevant, and the information can be modelled by the following equation:

$$m^W(\{w\}) = 1 - \gamma; \quad m^W(\Omega_W) = \gamma \qquad (4)$$

The belief and plausibility functions are defined by:

$$Bel^W(A) = \sum_{B \subseteq A} m^W(B), \quad Pl^W(A) = \sum_{B \cap A \neq \emptyset} m^W(B) \qquad (5)$$

$Bel^W(A)$ measures the amount of information that implies $W \in A$, and is a measure of certainty, while $Pl^W(A)$ measure the amount of information that does not conflict with $W \in A$, and is a measure of plausibility. We naturally have $Bel^W(A) \leq Pl^W(A)$ with the two being equal in the specific case of probabilities.

In our classification context, an evidential dataset ED will be of the form

$$ED = m^{X,Y} = \begin{pmatrix} m_1^{X,Y} \\ \vdots \\ m_N^{X,Y} \end{pmatrix} = \begin{pmatrix} m_1^{X^1} & \cdots & m_1^{X^J} & m_1^Y \\ \vdots & \ddots & \vdots & \vdots \\ m_N^{X^1} & \cdots & m_N^{X^J} & m_N^Y \end{pmatrix}$$

where $m_i^{X,Y}$ describes the i^{th} sample with its uncertainty . Note that precise data set D is retrieved when $m(\{x_i^j\}) = 1$ and $m(y_i) = 1$ for all i, j.

D. Related work

Uncertainty in classification problems can be related to data quality (imperfection of data), on which this paper focuses, or to the classification prediction model (such as the frequentist estimations of the leaves and class probabilities in CART purity gain computations).

Globally, works have been done in classical probabilistic framework, in the fuzzy set one, in the imprecise probabilistic one, or in the belief function one.

Most works dealing with uncertainty in classification (be it in probabilistic, fuzzy, imprecise probabilistic or evidential frameworks) concern uncertainties related to the classification model [14], [15], [16]. The few methodologies that handle uncertain attributes in the data are either fuzzy [17], [18] or probabilistic [19], [20] and rely heavily on considering "fractions" of examples falling in the tree leaves, something that cannot be done when the uncertainty model does not (always) bear on singletons, as is the case with belief functions.

In the probabilistic framework, Perinel [21] proposed a tree learning algorithm handling uncertain data modelled by subjective probabilities. It considers uncertainty on the attributes (not the class) and uses the EM algorithm on the conditional likelihood of the class given the input to estimate the set of probability densities of the classes in the leaves (our α_h^k parameters). Our work follows a similar line, but with a more generic uncertainty model (belief functions). Furthermore we consider that both the attributes and the class label can be uncertain.

III. Learning decision trees with uncertain data

In this section, the evidential likelihood and the E^2M algorithm basis are first recalled, then application of the E^2M to the tree parameter estimation is then presented and we finally give the general algorithm of our methodology. We only give the essential tools necessary to apply our method. Details about the general interpretation and implementation of E^2M can be found in [11].

A. Evidential likelihood and E^2M algorithm

The E^2M algorithm is an EM extension that handles uncertain data modelled by belief functions [11]. It maximizes

the *evidential likelihood*, which weights *incomplete* likelihood of each focal element by its mass. Given a parametric model $\{P_\theta \mid \theta \in \Theta\}$, the likelihood is $L(\theta;w) = P_\theta(W = w)$. The incomplete likelihood of a set-valued observation $w \in A$ is

$$L(\theta;A) \quad = \quad \sum_{w \in A} P_\theta(W = w) = P_\theta(w \in A)$$

and when uncertainty is described by m^W, evidential likelihood is given by:

$$L(\theta;m^W) \quad = \quad \sum_{i=1}^{z} m^W(A_i) L(\theta;A_i) \tag{6}$$

where $\{A_1,\ldots,A_z\}$ are the focal elements of m^W.

As for the classic EM algorithm, the maximisation of (6) is done through an iterative procedure alternating between two steps: *Expectation* (E) and *Maximisation* (M).

During the r^{th} iteration, at the E step, the expectation of the complete likelihood's logarithm $\log L(\theta;W)$ is computed with respect to a specific probability measure $P(. \mid \theta^{(r)}, m^W)$, that we denote $E[log(L(\theta,W)) \mid \theta^{(r)}, m^W]$. This probability measure is obtained by applying Dempster's combination [6] between $P_{\theta^{(r)}}$ (the current parametric model) and the uncertain observation m^W. This combination results, $\forall w \in \Omega_W$, in:

$$P(w \mid \theta^{(r)}, m^W) = \frac{P_{\theta^{(r)}}(w) Pl^W(\{w\})}{L(\theta^{(r)};m^W)} \tag{7}$$

The resulting mass function is a probability distribution (as one mass is a probability distribution). Eq. (7) can be applied if we assume cognitive independence between data uncertainty, that is learning the value of a given datum will not change our uncertainty about the value of other data (we refer to [11] for further explanations).

Once probability measure (7) is computed, we can proceed to the M step, in which the parameter $\theta^{(r+1)}$ maximising this expectation is found.

- E-step: $Q(\theta,\theta^{(r)}) = E[log(L(\theta;W)) \mid \theta^{(r)}, m^W]$
- M-step: $\theta^{(r+1)} = \arg\max_{\theta \in \Theta} Q(\theta,\theta^{(r)})$

As proved in [11], the obtained parameter happens to increase the value of the evidential likelihood (6). The E^2M algorithm therefore converges towards a local maximum of evidential likelihood estimator. Any likelihood maximisation problem where the data are evidential can thus be solved with the E^2M algorithm.

In this work, we use the E^2M algorithm considering that our model is a mixture of multinomial distribution, that is each leaf has a given probability and contains class items coming from a multinomial distribution. It can also be seen as a particular case of the general classification problem handled in [11].

B. Tree parameters estimation through the E^2M algorithm

During the learning of a E^2M decision tree, the probabilities $(\pi_h,\alpha_h)_{h=1,\ldots,H}$ cannot be estimated by proportions because of the uncertain nature of the data.

For a given tree $\mathscr{P}_H$, the basic principle of our methodology it to estimate parameters $(\pi_h,\alpha_h)_{h=1,\ldots,H}$ by the maximisation of the evidential likelihood of the data in regard to the couple (Z,Y) with the E^2M algorithm. This likelihood maximisation requires some definitions:

- the *latent* or *hidden* variables (not directly observable) resulting from the data uncertainty: $y_i^k = 1$ if $y_i = \omega_k$, 0 otherwise and $z_{ih} = 1$ if the i^{th} sample is in the leaf t_h, 0 otherwise.
- the plausibility pl_{ih} that a sample $m_i^{X,Y}$ belong to a given leaf t_h :

  $$pl_{ih} = Pl_i^Z(z_{ih} = 1) = Pl_i^X(\times_{j=1}^J A_h^j) = \prod_{j=1}^J Pl_i^{X^j}(A^j)$$

  where $Pl_i^{X^j}$ is the plausibility computed from $m_i^{X^j}$ and $\times$ the cartesian product.
  By this mean we can compute plausibility on X from plausibility on Z.
  The last product is due to the *cognitive* independence assumption.
- the plausibility pl_i^k that a class of sample $m_i^{X,Y}$ is k:

  $$pl_i^k = Pl_i^Y(y^k = 1) = Pl_i^Y(\{\omega_k\})$$

  where Pl_i^Y is the plausibility computed from m_i^Y.

If we denote by $L(\theta_H, m^{X,Y})$ the evidential likelihood of the observed evidential data (X,Y) relatively to a tree $\mathscr{P}_H$, the results of applying E^2M algorithm with observation $m_i^{X,Y}$ and the complete likelihood of θ_H is

$$L(\theta_H;m^{X,Y}) \quad = \quad \prod_{i=1}^N \sum_{h=1}^H pl_{ih}\pi_h \sum_{k=1}^K pl_i^k \alpha_h^k \tag{8}$$

$$Q(\theta_H,\theta_H^{(q)}) \quad = \quad \sum_{i,h} t_{ih}^{(q)} \log \pi_h + \sum_{i,h,k} \beta_{ih}^{k(q)} \log \alpha_h^k \tag{9}$$

and the maximum of $Q(\theta_H,\theta_H^{(q)})$ is obtained for $\theta_H^{(q+1)} := (\alpha_h^{k(q+1)}, \pi_h^{(q+1)})_{\substack{h=1,\ldots,K \\ k=1,\ldots,K-1 \\ j=1,\ldots,H-1}}$

where $\alpha_h^{k(q+1)} = \frac{\sum_i \beta_{ih}^{k(q)}}{\sum_i t_{ih}^{(q)}}$ and $\pi_h^{(q+1)} = \frac{1}{N}\sum_{i=1}^N t_{ih}^{(q)}$

with $t_{ih}^{(q)} = E[z_{ih} \mid m^{X,Y};\theta_H^{(q)}] = \frac{\pi_h^{(q)} pl_{ih} \sum_{k=1}^K \alpha_h^{k(q)} pl_i^k}{\sum_{h=1}^H pl_{ih}\pi_h^{(q)} \sum_{k=1}^K pl_i^k \alpha_h^{k(q)}}$

and $\beta_{ih}^{k(q)} = E[z_{ih} y_i^k \mid m^{X,Y};\theta_H^{(q)}] = \frac{\alpha_h^{k(q)} \pi_h^{(q)} . pl_{ih} pl_i^k}{\sum_{h=1}^H pl_{ih}\pi_h^{(q)} \sum_{k=1}^K pl_i^k \alpha_h^{k(q)}}$

Algorithm 1 summarizes the E^2M algorithm applied to the maximisation of $L(\theta_H;m^{X,Y})$.

C. Tree learning and best split selection

During the tree learning, for every leaf t_h and for every possible split s dividing t_h in t_L and t_R, $\theta_H(s)$ is estimated by the maximization of $L(\theta_H(s);m^{X,Y})$. Indeed, to compute the

Algorithm 1: θ_H estimation with the E^2M algorithm applied to a tree $\mathscr{P}_H$

Input: $\theta_H^{(0)}, \varepsilon$
Output: final θ_H

$r = 1$;
repeat
  Estimate $\theta^{(r+1)} = (\alpha_h^{k(q+1)}, \pi_h^{(q+1)})$ from $\theta_H^{(r)}$;
  Estimate $L(\theta^{(r+1)}; m^{X,Y})$ using (8) ;
  $r = r+1$;
until $\frac{L(\theta^{(r)};m^{X,Y}) - L(\theta^{(r-1)};m^{X,Y})}{L(\theta^{(r-1)};m^{X,Y})} > \varepsilon$;
$\theta_H = \theta^{(r)}$;

purity gain resulting from s, we use the parameters associated to leaves t_h, t_L and t_R, we thus need the initial tree parameters θ_H and the split tree $\theta_{H+1} = \theta_H(s)$ parameters has illustrated in formula (3).

The main difference between our trees and classical *CART* trees is the way parameters $\theta_H(s)$ are estimated at each split during the growing phase. In fact, with the E^2M estimation, when a leaf is split all the other leaf parameters change so computing the purity gain of a given split requires to re-estimate all the parameters of the tree (run Algorithm 1). In the precise case (classical CART), assessing the quality of a split does not require to re-estimate all the parameters. This means that our method is more computationally demanding, but has the advantage of handling uncertain data.

The global algorithm to build a E^2M decision tree is represented in algorithm 2. It results in a grown tree from which predictions can be done.

Algorithm 2: Learning of a E^2M decision tree general algorithm

Input: $\mathscr{P}_1 = \{t_1\} = \{\Omega_X\}$, data $m^{X,Y}$
Output: final tree $\mathscr{P}_H$

$H = 1$;
$\mathscr{P}_1 = \{t_1\} = \{\Omega_X\}$;
Apply Algorithm 1 to θ_1;
while *STOPPING CRITERIA not met* **do**
  foreach *Possible split s* **do**
    Compute $\theta_H(s)$;
    Estimate $\Delta i(\theta_H(s), \theta_H)$;
  $s_{optimal} = \arg\max_s \Delta i(\theta_H(s), \theta_H)$;
  $\mathscr{P}_{H+1} = \mathscr{P}_H(s_{optimal})$;
  $\theta_{H+1} = \theta_H(s_{optimal})$;
  $H = H+1$;

IV. Experiments

Though real world applications often have imperfect and uncertain data, there is a lack of benchmark datasets with those characteristics. Therefore, in order to evaluate our methodology, we use classical benchmark datasets and introduce data reliability into them. The main idea is that, if some data can be false (due to measurement error or expert mistake) and we are just aware of some reliability degree about the data, then we can build belief functions to model this awareness with Equation (4). In applications, the data reliability is often unknown. In this case a first step is advised in order to evaluate it. This can be done for example by verifying the rate of the corrupted data after a second measurement of a sample of the whole dataset. By this means we can get global reliability levels or local ones when repeating this verification separately for all variables (attributes and class labels), a complete, some other methods could be explored but this is not the topic of this paper.

The E^2M decision tree methodology is tested and compared to *CART* with a 10-fold cross validation procedure on several UCI datasets with the following characteristics:

data set	# features	# classes	# examples
Iris	4	3	150
Balance scale	4	3	625
Wine	13	3	178
Glass	9	7	214
E.Coli	8	7	336

For the stopping criteria, we set a maximum number of leaves at 5 and a relative minimum purity gain of 5%.

Following a procedure similar to the one used in [11], a simple noise injection procedure is performed as follows: for each observation x_i^j (when attributes are noised) or class y_i (when classes are noised), a noise level γ_i^j (γ_i in the case of classes) is sampled from a uniform distribution on $[0,1]$. To decide whether we replace (noise) an attribute/class value x_i^j or y_i by another value $\tilde{x}_i^j$ or $\tilde{y}_i$, a number u is uniformly simulated on [0,1], and the value is replaced if $u < \gamma_i^j(\gamma_i)$. Replacing values $\tilde{x}_i^j$ or $\tilde{y}_i$ are uniformly drawn from Ω_{Xj} or Ω_Y.

In the case of CART trees, noised observations are used to learn the model, while in the case of E^2M trees, data uncertainty is modelled by masses on the noised observations in regards to Equation (4) with the sampled levels γ_i^j.

As the application of the noise procedure is stochastic, we make all experiments five times and retain the mean error rates obtained by the cross-validation procedures. We also compare the model prediction accuracies with the accuracy obtained by predicting systematically the most represented class of the learning data set. We call this predictor *naive*.

Tables I, II and III provide the error rate means for *naive*, *CART* and the E^2M decision trees respectively with noise on all the attributes, on class labels and on both. A Wilcoxon rank sum test was performed with a significance level of 5% to test if the obtained error rates were statistically significatively different, and best significant error rates were put in bold in the tables.

algorithm	*naive*	*CART*	E^2M
iris	0.67	0.22	**0.14**
balance	0.54	0.54	**0.37**
wine	0.60	0.28	**0.19**
glass	0.65	**0.54**	0.58
E.Coli	0.57	0.34	**0.28**

TABLE I. Error rates with noised attributes

algorithm	*naive*	*CART*	E^2M
iris	0.67	**0.15**	0.16
balance	0.54	0.54	**0.39**
wine	0.63	0.23	0.20
glass	0.66	0.57	0.58
E.Coli	0.57	0.27	0.29

TABLE II. ERROR RATES WITH NOISED CLASS LABELS

algorithm	*naive*	*CART*	E^2M
iris	0.67	0.44	**0.17**
balance	0.54	0.54	**0.39**
wine	0.63	0.46	**0.22**
glass	0.65	0.65	**0.58**
E.Coli	0.39	0.38	**0.22**

TABLE III. ERROR RATES WITH NOISED ATTRIBUTES AND CLASS LABELS

Although these results concern a limited number of data sets, they are clearly promising, as the E^2M trees almost always perform better than CART trees, and perform almost as well in other cases. This means that our algorithm is able to take advantage of those data that are highly reliable, leaving aside data with high uncertainty. The difference is obvious when both attributes and classes are noised, in which case E^2M trees accuracy is much better than those of CART trees. This difference could be explained by the number of noised variables. In deed, when only class labels are noised, the advantage of the evidential modelling of the data reliability is less exploited as it only concerns a single variable.

V. CONCLUSION

A decision tree methodology was presented in the belief function framework. The trees are grown from evidential datasets and during the learning of the tree, the probabilities of the different leaves and of the classes in the leaves are estimated with the E^2M algorithm. This method allows us to take account of different kinds of uncertainty: probabilistic data, missing data, imprecise data, fuzzy data, ...

We have demonstrated with some first experiments the potential interest of our algorithm and of such a modelling. Therefore it seems to have a lot of potential for a whole range of applications. In particular, it appears quite better in a very noisy (i.e., noised attributes and classes) environment.

There are also a couple of perspective or possible ameliorations that we could mention:

- using the E^2M to evaluate classifiers when test data classes (and possibly attributes) are themselves uncertain. Indeed, in real application it will not always be possible to have a sufficient number of precise test data. Also, being able to do such an evaluation will be necessary to develop pruning procedures of learned trees;
- develop a local learning algorithm, that avoids having to evaluate the whole set of parameters $\theta_H(s)$ for each split, and compare its performances to the global method;
- extend the methodology to regression decision trees when classes are numerical.

ACKNOWLEDGEMENT

The authors would like to greatly thank Brigitte Charnomordic and Nicolas Verzelen from the UMR MISTEA for their precious help and all their valuable and useful remarks and advices.

REFERENCES

[1] R. J. Little and D. B. Rubin, *Statistical analysis with missing data*. Wiley New York, 1987, vol. 539.

[2] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society, SERIES B*, vol. 39, no. 1, pp. 1–38, 1977.

[3] J. Josse, M. Chavent, B. Liquet, and F. Husson, "Handling missing values with regularized iterative multiple correspondence analysis," *Journal of classification*, vol. 29, no. 1, pp. 91–116, 2012.

[4] P. Walley, *Statistical reasoning with imprecise probabilities*. Chapman and Hall, 1991.

[5] D. Dubois and H. Prade, "Formal representations of uncertainty," *Decision-making Process: Concepts and Methods*, pp. 85–156.

[6] A. P. Dempster, "Upper and lower probabilities induced by a multivalued mapping," *Annals of Mathematical Statistics*, vol. 38, pp. 325–339, 1967.

[7] G. Shafer, *A mathematical theory of evidence.* Princeton-London: Princeton University Press.

[8] T. Denœux, "Conjunctive and disjunctive combination of belief functions induced by nondistinct bodies of evidence," *Artificial Intelligence*, vol. 172, no. 2, pp. 234–264, 2008.

[9] E. Lefevre, O. Colot, and P. Vannoorenberghe, "Belief function combination and conflict management," *Information fusion*, vol. 3, no. 2, pp. 149–162, 2002.

[10] D. Dubois, H. Fargie, and H. Prade, "Comparative uncertainty, belief functions and accepted beliefs," in *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 1998, pp. 113–120.

[11] T. Denœux, "Maximum likelihood estimation from uncertain data in the belief function framework," *IEEE Trans. on Know. and Data Eng.*, vol. 25, pp. 119–130, 2013.

[12] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen, *Classification And Regression Trees*, 1984.

[13] J. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, pp. 81–106, Oct. 1986. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/17050186

[14] J. Abellán and A. R. Masegosa, "An ensemble method using credal decision trees," *European journal of operational research*, vol. 205, no. 1, pp. 218–226, 2010.

[15] Z. Elouedi, K. Mellouli, and P. Smets, "Belief decision trees: theoretical foundations," *International Journal of Approximate Reasoning*, vol. 28, no. 2, pp. 91–124, 2001.

[16] N. Sutton-Charani, S. Destercke, and T. Denœux, "Classification trees based on belief functions," in *Belief Functions: Theory and Applications*. Springer, 2012, pp. 77–84.

[17] B. Bouchon-Meunier, M.-J. Lesot, and C. Marsala, "Modelling and management of subjective information in a fuzzy setting," *International Journal of General Systems*, vol. 42, no. 1, pp. 3–19, 2013.

[18] C. Olaru, "A complete fuzzy decision tree technique," *Fuzzy Sets and Systems*, vol. 138, no. 2, pp. 221–254, Sep. 2003. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0165011403000897

[19] S. Tsang, B. Kao, K. Y. Yip, W.-S. Ho, and S. D. Lee, "Decision trees for uncertain data," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 23, no. 1, pp. 64–78, 2011.

[20] A. Ciampi, E. Diday, J. Lebbe, E. Périnel, and R. Vignes, "Growing a tree classifier with imprecise data," *Pattern Recognition Letters*, vol. 21, no. 9, pp. 787–803, 2000.

[21] E. Périnel, "Construire un arbre de discrimination binaire à partir de données imprécises," *Revue de statistique Appliquée*, vol. 47, no. 1, pp. 5–30, 1999.