Université de Tunis
Institut Supérieur de Gestion de Tunis

THÈSE POUR L'OBTENTION DU TITRE
DE DOCTEUR EN GESTION

Belief Decision Tree: A New Decision Tree Approach under Uncertainty

par

Zied Elouedi

Directeurs de Thèse:

Pr. Khaled Mellouli (IHEC, Tunisie)
Pr. Philippe Smets (IRIDIA-ULB, Belgique)

Année de Soutenance 2002

Acknowledgements

I would like to express my sincere gratitude to my advisor, Professor Khaled Mellouli, for introducing me to the area of research. I thank him for his constant encouragements and his valuable advices. His observations and comments for improvement have always been interesting and constructive.

I owe a debt of gratitude to my co-advisor, Professor Philippe Smets for his confidence and his continuous guidance. He has been patient in his supervision and generous in his time and support. This thesis has certainly benefited from his wisdom and his insightful and thoughtful suggestions.

I am also grateful to Thierry Denœux from *Université de Compiègne* for spending time discussing my research and for his useful remarks.

Many thanks to the members of *IRIDIA* in *Université Libre de Bruxelles* for their hospitality. In particular, I would like to thank Patrice Latinne for his help in getting databases used in simulation.

Special thanks are due to Salem Benferhat (*IRIT*), Jean-Charles Pomerol (*LIP6*) and Jean-Yves Jaffray (*LIP6*) for their hospitality and their help in my trainings abroad.

I thank all the members of the laboratory *LARODEC* in *Institut Supérieur de Gestion de Tunis* for their support. A particular acknowledgement goes to Nahla Ben Amor and Fédia Khalfallah for their continuous encouragements and moral support.

Finally, my thanks go to all the persons in *Institut Supérieur de Gestion de Tunis* for their help.

Contents

List of Figures

List of Tables

Abstract

In this thesis, we have developed a new decision tree approach under uncertainty. This so-called belief decision tree is a new classification method adapted to uncertain data where the uncertainty is represented by belief functions. We have developed two major procedures regarding a belief decision tree. The first procedure concerns its construction from a given training set characterized by uncertainty in the classes of its objects, and represented by the means of belief functions. In order to build the tree, many parameters adapted to this uncertain context, are defined such as the attribute selection measure, the partitioning strategy, the stopping criteria and the structure of leaves. Once the belief decision tree is constructed, the second procedure is related to the classification of instances, using the belief function formalism. Our objective is to provide a method able to classify objects even those characterized by uncertain attribute values. Simulations are performed in order to show the feasibility of our belief decision tree approach. Finally, some improvements are added to belief decision trees in order to better handle the uncertainty in the training set. Hence, methods for assessing beliefs, given by experts, on the classes of training instances are proposed. Furthermore, our belief decision tree approach is extended to deal with the case where uncertainty also occurs in the attributes of the training instances.

Résumé

Dans cette thèse, nous avons développé une nouvelle approche d'arbre de décision dans un environnemnet incertain. Cette nouvelle méthode de classification, appelée arbre de décision crédibiliste, est adaptée aux données incertaines où l'incertitude est représentée par des fonctions de croyance. Nous avons développé deux procédures majeures relatives aux arbres de décision crédibilistes. La première procédure concerne sa construction à partir d'un ensemble d'apprentissage caractérisé par une incertitude dans les classes de ses objets, et qui est représentée par des fonctions de croyance. Afin de construire cet arbre, plusieurs paramètres adapatés à cet environnment incertain, sont définis à savoir la mesure de sélection d'attributs, la stratégie de partitionnement, les critères d'arrêt et la structure des feuilles. Une fois l'arbe de décision crédibiliste construit, la seconde procédure est relative à la classification des instances et ce en utilisant le formalisme des fonctions de croyance. Notre objectif est d'offrir une méthode capable de classer les objets, même ceux caractérisés par des valeurs incertaines d'attributs. Des simulations ont été effectuées afin de montrer la faisabilté de notre approche d'arbre de décision crédibiliste. Finalement, des améliorations sont ajoutées aux arbres de décision crédibilistes afin de mieux manipuler l'incertitude dans l'ensemble d'apprentissage. Ainsi, des méthodes d'évaluation des croyances, données par les experts, sur les classes des instances d'apprentissage, sont propoées. Par ailleurs, notre approche est étendue, pour traiter le cas d'incertitude dans les attributs des instances d'apprentissage.

Introduction

Classification techniques have the major objective to find for each object its corresponding class by taking into account their attributes (characteristics). These techniques are widely used in different fields such as medicine, industry, marketing, finance, etc.

The classification techniques are divided into two groups: those working under a **supervised** mode where classes of instances in the training set are known a priori, and those working under an **unsupervised** mode where classes are not fixed a priori.

In addition to statistical classification techniques where the widely used method is the discriminant analysis, several non parametric classification techniques have been developed (Weiss & Kulikowski, 1991). The best known in the supervised mode are decision trees, k-nearest neighbors, neural networks, etc, whereas in the unsupervised mode, the most classical method, is the clustering.

These *'new'* techniques are characterized by their application to complex fields like the artificial intelligence, the pattern recognition, the speech recognition, the control of the industrial process, and so on.

However, the standard versions of these different techniques are inadequate and badly adapted to ensure their role of classification in an environment characterized by a lot of uncertain and imprecise parameters. Hence, having an uncertain, or imprecise or even missing information relative to any parameters of the classification problem may affect the correctness of classification results.

That is why researches are oriented to the improvement and the extension of these techniques, in order to adapt them to this kind of environment. Thus, the idea is to develop classification methods based on theories managing uncertainty and imprecision such as probability theory, fuzzy set theory, and belief function theory.

In this thesis, we are interested to the decision tree method. This non parametric technique, working under a supervised mode, is one of the most used classification methods in artificial intelligence (Quinlan, 1986, 1993), due notably to its simple structure and its accurate results *'easy'* to interpret by experts and even by ordinary users.

In order to overcome the limitations encountered in standard decision trees, and due to the uncertainty and the imprecision in some classification parameters, two kinds of decision tree were developed: **probabilistic decision trees** (Quinlan, 1987a, 1990b) and **fuzzy decision trees** (Umano et al., 1994), (Zeidler & Schlosser, 1996), (Marsala, 1998).

Probabilistic decision trees try to classify new instances where one or several needed attribute values are missing or uncertain. However, this kind of tree is operational only if there is no uncertainty in the training set, but where pruning is applied to the induced tree. Fuzzy decision trees have been defined to cope with data where both classes and attributes are represented by fuzzy values.

Inspite of its several advantages, the belief function theory had not yet been applied to decision trees when we started this work.

The objective of this thesis is to develop this new concept that we have called **belief decision trees**. This new approach is based on both the decision tree technique and the belief function theory in order to cope with uncertainty in the classification problem parameters.

Recently, Denœux and Skarstein-Bjanger (2000) have studied a similar problem with another interpretation different from the one that will be presented in this thesis. They consider the data as a *'random sample'*, but what is developed, is limited to binary classes.

Belief function theory as interpreted in the Transferable Belief Model (TBM) (Smets, 1988, 1998b; Smets & Kennes, 1994; Smets & Kruse, 1997) provides a mathematical tool to treat subjective, and personal judgments on the different parameters of any classification problem and can be easily extended to deal with objective probabilities. It allows experts to express partial beliefs in a much more flexible way than probability functions do. Besides, it permits to handle partial or even total ignorance concerning classification parameters.

Belief function theory offers interesting tools to combine several pieces of evidence, like the conjunctive and the disjunctive rules of combination. Decision making is solved through the pignistic transformation. In addition to these advantages, this theory is easily applied to reasoning based systems like decision support systems and expert systems.

Hence, the belief function theory provides a convenient framework to handle uncertainty in the decision tree technique and consequently to develop what we call a belief decision tree approach.

Belief decision trees should cope with uncertain classification parameters. This uncertainty occurs at two levels:

- **Classes:** Suppose the class assigned to an object in the training set is not known with certainty. It can be a disjunction of classes (the object belongs either to the first or the second class, etc). The general case consists in having degrees of belief on the actual class of the object.

- **Attributes:** Suppose one or several attribute values of some objects are not known with certainty. So, it is interesting to assign for each attribute a basic belief assignment representing beliefs on its values. In this thesis, we only deal with symbolic attributes.

In our belief decision tree approach, we will develop two important procedures:

- The construction of the belief decision tree: The structure of the training set[1] must be defined in order to take into account the uncertainty of some parameters.

[1]The structure of a *'standard'* training set in a certain environment is composed of elements represented as pairs (attributes, class) where for each object, we know exactly the value of each one of its attributes and also its assigned class which is unique.

In this thesis, we will basically handle uncertainty in the classes of the training objects. The attribute values of objects in the training set are supposed to be known with certainty.

Given this structure of the training set, the second step consists in defining an attribute selection measure allowing to find the root of the belief decision tree and those of the induced sub-trees. This measure has to take into account the basics of the belief function theory in order to deal with this uncertain context.

Then, we have to develop the algorithm for building the belief decision tree relative to the structure of the training set. To ensure this objective, and in addition to the attribute selection measure, other parameters have to be defined like, the partitioning strategy, the stopping criteria, and the structure of leaves.

- The classification of new instances: Once the belief decision tree is constructed, the second procedure is the inference procedure allowing the classification of new instances. Three cases will be studied:

1. The object to classify is characterized by attribute values which are known with certainty.

2. The object to classify is characterized by some (or all) disjunctive attribute values. This case involves the total ignorance of the value of one (or more) attribute(s).

3. The object to classify is characterized by some (or all) uncertain attribute values. Hence, each attribute of the object is defined by a basic belief assignment representing beliefs on its values. Obviously, this latter case is considered as the generalized case involving the simple and the disjunctive ones.

In practice, uncertainty occurs frequently in either attributes or classes. Therefore, in almost all fields where decision trees are applied and whenever uncertainty happens, the belief decision tree technique will be useful. We can mention several fields of classification where uncertainty is encountered like medicine where diseases (classes) of some patients or even their symptoms (attributes) may be uncertain.

Furthermore, belief decision trees can be applied to marketing, finance, or management problems. Moreover, the illustrative example used in this thesis deals with the classification of clients in a bank in order to plan a loan policy.

Uncertainty can also be present in other fields and hence, the use of belief decision trees is required, such that in military problems dealing for example with radar sensors, or in strategic problems like classification of scenarios according to their hypotheses relative to a strategic domain.

So, belief decision trees provide a tool to classify data pervaded with uncertainty, thus widening the scope of application of the classification procedures to those new domains where uncertainty cannot be avoided. Problems neglected in the past because of the lack of an adequate tool might become manageable thanks to our method.

Since we deal with subjective judgments allowing the treatment of expert opinions on classes relative to training objects, we will also develop methods for tuning the opinions of the experts by assessing their corresponding reliability factors and consequently evaluating their beliefs.

In this work, we study the case where uncertainty encountered in the training set concerns only the classes of the training objects. Some improvements related to uncertainty in attributes in the training set will also be defined.

Our thesis is organized in eight chapters distributed on three parts:

- *Part 1* presents the necessary theoretical aspects concerning the belief function theory and the decision trees.

- *Part 2* details the construction of the belief decision tree when the knowledge about the classes of the training objects is represented by a belief function. Two approaches are proposed for the attribute selection procedure: the averaging approach based on the extension of Quinlan method, and the conjunctive approach more based on the basics of the transferable belief model itself. The belief decision tree method can be adapted to handle the classification of new objects when uncertainty occurs in some (or all) the values of their attributes. Implementation and simulation are performed in order to judge of the feasibility of our method.

- *Part 3* provides some improvements that can be added to the belief decision tree approach. It treats the problems of assessing the beliefs encountered in the training objects. Experts produce the knowledge about the classes of the training objects. Their reliability is assessed either individually or in group. Our belief decision tree method is also extended to handle the case where uncertainty pervades the attributes of the training instances.

Finally, a general conclusion summarizes the major achievements of this thesis and presents possible future developments.

Three appendices complete this thesis. The first one describes a small example of the use of the belief decision tree approach within the scenario method. The second appendix presents the different data manipulated in the simulation. The third appendix presents an example of artificial bba's on classes used for performing simulations.

Part I

Theoretical Aspects

In this first part, we present the theoretical aspects used in our thesis. Our thesis deals with belief decision trees which permits the application of the decision tree technique to an uncertain context, and where the uncertainty is represented by belief functions.

This part of preliminaries presents the necessary background concerning the belief function theory and the decision tree method. This first part is composed of two chapters:

- The first chapter deals with the belief function theory as understood in the transferable belief model. This theory is a useful tool to represent uncertain knowledge.

  This chapter describes the basic notions of this theory and some illustrative examples. In fact, definitions of different functions provided by this theory are given. Besides, useful concepts within the belief function theory are detailed namely the combination, the Generalized Bayes Theorem, the discounting, the projection and the extension of belief functions, the coarsening and the refinement, and also the decision process.

- The second chapter presents the classification method called the decision tree which is among the well known machine learning techniques.

  This chapter details the basic characteristics of the decision tree method and some of its improvements related notably to decision tree under probability and fuzzyness. Illustrative examples are also provided in this chapter in order to more explain the different basics presented.

These two chapters are useful to understand the developments made in this thesis concerning the belief decision tree approach.

Chapter 1

Belief function theory

1.1 Introduction

The theory of belief functions is considered as a useful theory for representing and managing uncertain knowledge. This theory is introduced by Shafer (1976) as a model to represent quantified beliefs.

The belief function theory is widely applied to artificial intelligence where it is usually called **the Dempster-Shafer theory** (Gordon & Shortliffe, 1984). However, several interpretations of such terms are proposed (Smets, 1991): in particular **the lower probability model** (Walley, 1991), **the Dempster's model** (Dempster, 1967, 1968), **the theory of hints** (Kohlas & Monney, 1995), and **the Transferable Belief Model (TBM)** (Smets, 1988, 1998b; Smets & Kennes, 1994; Smets & Kruse, 1997). The first three interpretations are somehow based on probability theory, whereas the last one is not.

In this thesis, we deal with the interpretation of the belief function theory as explained by the TBM. In this chapter, we present the basics of this theory. Next, some special belief functions are described. Then, several concepts of the belief function theory are detailed like the combination, the Generalized Bayes Theorem, the discounting, the projection and the extension of belief functions, the coarsening and the refinement. Finally, we present some criteria used to solve the decision process within this theory.

1.2 Basic concepts

1.2.1 Frame of discernment

Let Θ be a finite non empty set including all the elementary events related to a given problem. These events are assumed to be exhaustive and mutually exclusive. Such set Θ is called **the frame of discernment**. It is also referred to as **the universe of discourse** or **the domain of reference** (Smets, 1988).

Generally, we handle all the subsets of Θ which belong to the power set of Θ, denoted by 2^Θ and defined as follows:

$$2^\Theta = \{A : A \subseteq \Theta\} \tag{1.1}$$

Every element of 2^Θ is called **a proposition** or **an event**. It can also be seen as a possible answer to a given question.

Note that the empty set $\emptyset$ belongs to the power set 2^Θ and it corresponds to the impossible proposition (the contradiction), whereas the set Θ corresponds to the certain proposition (the tautology).

Example 1.1 *Let's treat a classification problem of aerial targets. Suppose the frame of discernment related to this problem is defined as follows:*
$\Theta = \{Airplane, Helicopter, Missile\}$

The corresponding power set of Θ *is:*
$2^\Theta = \{\emptyset, \{Airplane\}, \{Helicopter\}, \{Missile\}, \{Airplane, Helicopter\}, \{Airplane, Missile\}, \{Helicopter, Missile\}, \{Airplane, Helicopter, Missile\}\}$

1.2.2 Basic belief assignment

The impact of a piece of evidence on the different subsets of the frame of discernment Θ is represented by the so-called **basic belief assignment** (bba), called initially by Shafer (1976) basic probability assignment, an expression that has unfortunately created serious confusion in the past.

The bba is defined as follows:

$$m : 2^{\Theta} \rightarrow [0, 1]$$

$$\sum_{A \subseteq \Theta} m(A) = 1 \tag{1.2}$$

The value $m(A)$, named **a basic belief mass** (bbm), shows the part of belief exactly committed to the event A of Θ given a piece of evidence. Due to the lack of information, this quantity cannot be apportioned to any strict subset of A. So, it represents the direct specific support of evidence on A.

The quantity $m(\Theta)$ measures the portion of belief assigned to the whole frame Θ. It represents the beliefs not assigned to the different subsets of Θ.

Shafer (1976) has initially proposed a normality condition expressed by:

$$m(\emptyset) = 0 \tag{1.3}$$

Such bba is called **a normalized basic belief assignment**.

Smets (1990) relaxes this condition and interprets $m(\emptyset)$ as the amount of conflict between the pieces of evidence or as the part of belief given to the fact that none of the hypotheses in Θ is true, in other words the hypotheses making up the frame of discernment of hypotheses are not exhaustive. This last interpretation refers to the so-called **open-world assumption** (Smets, 1990) (by opposition to the exhaustive frame of discernment which is referred to as the **closed-world assumption**).

Example 1.2 *Assume* $\Theta = \{Airplane, Helicopter, Missile\}$

Suppose a sensor expresses a piece of evidence concerning the nature of a detected aerial target. The bba related to the sensor's evidence is defined as follows:
$m(\{Airplane\}) = 0.6$;
$m(\{Airplane, Helicopter\}) = 0.2$;
$m(\Theta) = 0.2$;

For example, 0.6 represents the part of belief exactly supporting that the detected aerial target is an airplane.

1.2.3 Focal elements, body of evidence, core

The subsets A of the frame of discernment Θ such that $m(A)$ is strictly positive, are called **the focal elements** of the bba m.

The pair (F, m) is called **a body of evidence** where F is the set of all the focal elements relative to the bba m.

The union of all the focal elements of m are named **the core** and are defined as follows:

$$\varphi = \bigcup_{A:m(A)>0} A \tag{1.4}$$

Example 1.3 *Let's continue with the Example 1.2, the subsets* $\{Airplane\}$, $\{Airplane, Helicopter\}$, *and* Θ *are the focal elements of the bba* m.

So, (F, m) *is called the body of evidence such that:*
$F = \{\{Airplane\}, \{Airplane, Helicopter\}, \Theta\}$

The core of this bba m *is defined as follows:*
$\varphi = \{Airplane\} \cup \{Airplane, Helicopter\} \cup \Theta = \Theta$

1.2.4 Belief function

A belief function, denoted *bel*, corresponding to a specific bba m, assigns to every subset A of Θ the sum of the masses of belief committed exactly to every subset of A by m (Shafer, 1976).

Contrary to the bba which expresses only the part of belief committed exactly to A, the belief function *bel* represents the total belief that one commits to A without being also committed to $\overline{A}$.

The belief function *bel* is defined as follows:

$$bel : 2^{\Theta} \rightarrow [0, 1] \text{ such that:}$$

$$bel(A) = \sum_{\emptyset \neq B \subseteq \Theta} m(B) \tag{1.5}$$

The bbm $m(\emptyset)$ is not included in $bel(A)$ as $\emptyset$ is both a subset of A and $\overline{A}$. We want in $bel(A)$ only the bbm given to the subsets that support A without supporting $\overline{A}$.

Theorem 1.1 *(Shafer, 1976) Let Θ be a frame of discernment, the function $bel : 2^{\Theta} \rightarrow [0,1]$ is a belief function if and only if it satisfies the following conditions:*

$$bel(\emptyset) = 0 \tag{1.6}$$

For all $A_1, \ldots, A_n \in 2^{\Theta}$,

$$bel(A_1 \cup ... \cup A_n) \geq \sum_i bel(A_i) -$$

$$\sum_{i>j} bel(A_i \cap A_j) - ... - (-1)^n bel(A_1 \cap ... \cap A_n) \tag{1.7}$$

Usually, $bel(\Theta) = 1$ is assumed (Shafer, 1976) corresponding to a closed-world assumption. It can be ignored and we only require $\text{bel}(\Theta) \leq 1$.

Properties

- *Sub-additivity:*

$$bel(A) + bel(\overline{A}) \leq 1 \tag{1.8}$$

  This rule shows that the knowledge of the belief given to a proposition A does not necessarily give us an information about the degree of belief of the proposition $\overline{A}$ (Lopez De Mantaras, 1990).

  Contrary to the theory of probability, in the belief function theory increasing beliefs on a proposition A does not necessary require the decrease of beliefs on $\overline{A}$.

- *Monotonicity:*

$$A \subseteq B \Longrightarrow bel(B) \geq bel(A) \tag{1.9}$$

  Hence, Θ will get the highest value of *bel* (the upper limit), whereas $\emptyset$ will get the lowest value (the lower limit).

- For A, $B \subseteq \Theta$, $A \cap B = \emptyset$,

$$bel(A \cup B) \geq bel(A) + bel(B) \tag{1.10}$$

- $m(A)$ may be expressed by the values of *bel* as follows (Smets, 2002):

$$m(A) = \sum_{B \subseteq A} (-1)^{|A|-|B|} bel(B), \forall A \subseteq \Theta,\ A \neq \emptyset \tag{1.11}$$

- the bbm $m(\emptyset)$ is computed as follows:

$$m(\emptyset) = 1 - bel(\Theta) \tag{1.12}$$

Example 1.4 *The belief function bel corresponding to the bba m (see Example 1.2) is defined as follows:*
$bel(\emptyset) = 0$;
$bel(\{Airplane\}) = 0.6$;
$bel(\{Helicopter\}) = bel(\{Missile\}) = bel(\{Helicopter, Missile\}) = 0$;
$bel(\{Airplane, Helicopter\}) = 0.6 + 0.2 = 0.8$;
$bel(\{Airplane, Missile\}) = 0.6$;
$bel(\Theta) = 0.6 + 0.2 + 0.2 = 1$;

For example, 0.8 is the total belief committed to the proposition $\{Airplane, Helicopter\}$. *It represents the sum of the bbm's assigned to this set and also to its subsets.*

1.2.5 Plausibility function

The plausibility function *pl* quantifies the maximum amount of belief that could be given to a subset A of the frame of discernment. It is equal to the sum of the bbm's relative to subsets B compatible with A. In other words, it contains those parts of belief that do not contradict A ($B \cap A \neq \emptyset$).

$$pl : 2^{\Theta} \rightarrow [0, 1] \text{ such that:}$$

$$pl(A) = \sum_{A \cap B \neq \emptyset} m(B) \tag{1.13}$$

$$= bel(\Theta) - bel(\overline{A}) \tag{1.14}$$

$$= \sum_{B \subseteq \Theta} m(B) - \sum_{B \subseteq \overline{A}} m(B) \tag{1.15}$$

where $\overline{A}$ is the complement of the subset A relative to Θ.

Properties

- $pl(A)$ can be expressed by $bel(A)$, we get:

$$pl(A) = bel(A) + \sum_{A \cap B \neq \emptyset, B \not\subset A} m(B) \tag{1.16}$$

- *Over additivity:*

$$pl(A) + pl(\overline{A}) \geq 1 \tag{1.17}$$

- *Monotonicity:*

$$A \subseteq B \Longrightarrow pl(B) \geq pl(A) \tag{1.18}$$

- For A, $B \subseteq \Theta$, $A \cap B = \emptyset$,

$$pl(A \cup B) \leq pl(A) + pl(B) \tag{1.19}$$

- For $A \subseteq \Theta$,

$$bel(A) \leq pl(A) \tag{1.20}$$

Example 1.5 *The plausibility function pl corresponding to the bba m (see Example 1.2) is defined as follows:*
$pl(\emptyset) = 0$;
$pl(\{Airplane\}) = 0.6 + 0.2 + 0.2 = 1$;
$pl(\{Helicopter\}) = 0.2 + 0.2 = 0.4$;
$pl(\{Missile\}) = 0.2$;
$pl(\{Airplane, Helicopter\}) = 0.6 + 0.2 + 0.2 = 1$;
$pl(\{Airplane, Missile\}) = 0.6 + 0.2 + 0.2 = 1$;
$pl(\{Helicopter, Missile\}) = 0.2 + 0.2 = 0.4$;
$pl(\Theta) = 0.6 + 0.2 + 0.2 = 1$;

For example, 0.2 represents the maximum degree of belief that the proposition $\{Missile\}$ may have.

1.2.6 Commonality function

The commonality function q has no immediate intuitive interpretation. However, it may represent the total mass that is free to move to every element of A (Barnett, 1991). It is defined as follows:

$q : 2^{\Theta} \rightarrow [0, 1]$ such that:

$$q(A) = \sum_{A \subseteq B} m(B) \tag{1.21}$$

We have to notice that the commonality function is an interesting and useful tool simplifying the computations in the belief function theory. Its interpretation is given in details in (Smets, 1998b). Indeed, it can be understood as the amount of uncertainty in the context where we accept as true that the actual world belongs to A and can also be defined as follows:

$$q(A) = m[A](A) \text{ for all } A \subseteq \Theta \tag{1.22}$$

where $m[A](A)$ is the conditional bbm allocated to A in the context where A is (accepted as) true.

Properties

- The commonality value relative to the empty set is defined as follows:

$$q(\emptyset) = 1 \tag{1.23}$$

- The commonality value relative to the whole frame of discernment is defined as follows:

$$q(\Theta) = m(\Theta) \tag{1.24}$$

Example 1.6 *The commonality function q corresponding to the bba m (see Example 1.2) is defined as follows:*
$q(\emptyset) = 1$;
$q(\{Airplane\}) = 0.6 + 0.2 + 0.2 = 1$;
$q(\{Helicopter\}) = 0.2 + 0.2 = 0.4$;
$q(\{Missile\}) = 0.2$;
$q(\{Airplane, Helicopter\}) = 0.2 + 0.2 = 0.4$;
$q(\{Airplane, Missile\}) = q(\{Helicopter, Missile\}) = 0.2$;
$q(\Theta) = 0.2$;

Remark:

Note that the basic belief assignment (m), the belief function (bel), the plausibility function (pl) and the commonality function (q) are in one to one correspondence with each other. They can be considered as different expressions of the same information (Denœux, 1999).

1.3 Special belief functions

1.3.1 Introduction

In the literature, several kinds of belief functions are proposed. Such functions are used to express particular situations related generally to uncertainty. In this section, we present the vacuous belief function, the categorical belief function, the certain belief function, the simple support function, the Bayesian belief function, the consonant belief function, the dogmatic and non-dogmatic belief functions.

1.3.2 Vacuous belief function

A vacuous belief function is a normalized belief function defined such that (Shafer, 1976):

$$m(\Theta) = 1 \text{ and } m(A) = 0 \text{ for } A \neq \Theta \tag{1.25}$$

In other words,

$$bel(\Theta) = 1 \text{ and } bel(A) = 0 \text{ for } A \neq \Theta \tag{1.26}$$

Such basic belief assignment where Θ is the unique focal element, quantifies the state of **total ignorance** since there is no support given to any strict subset of Θ.

Example 1.7 *Assume a sensor S_0 was not able to detect the nature of the aerial target. Hence, we get a state of total ignorance where the corresponding bba m_0 is a vacuous bba defined as follows:*

$m_0(\Theta) = 1$ *and* $m_0(A) = 0$ *for* $A \neq \Theta$

1.3.3 Categorical belief function

A categorical belief function is a normalized belief function such that its bba is defined as follows (Mellouli, 1987):

$$m(A) = 1 \text{ for some } A \subset \Theta \text{ and } m(B) = 0, \text{ for } B \subseteq \Theta, B \neq A \tag{1.27}$$

Such function has a unique focal element A (which is not imperatively a singleton event) different from the frame of discernment Θ.

Example 1.8 *We get a piece of evidence ensuring that the aerial target can not be a missile. So, the corresponding bba m presents a categorical belief function characterized by:*

$m(\{Airplane, Helicopter\}) = 1$;

1.3.4 Certain belief function

A certain belief function is a categorical belief function such that its focal element is a singleton. Its corresponding bba is defined as follows:

$$m(A) = 1 \text{ and } m(B) = 0 \text{ for all } B \neq A \text{ and } B \subseteq \Theta \tag{1.28}$$

where A is a singleton event of Θ.

Such function represents a state of total certainty as it assigns all the belief to a unique elementary event.

Example 1.9 *Assume a sensor S_c affirms that the detected aerial target is an airplane. The corresponding bba m_c is a certain basic belief assignment defined as follows:*
$m_c(\{Airplane\}) = 1$;

1.3.5 Simple support function

A belief function is called **a simple support function** (ssf) if it has at most one focal element different from the frame of discernment Θ. This focal element is called **the focus** of the ssf.

A simple support function is defined as follows (Smets, 1995):

$$m(X) = \begin{cases} w & \text{if } X = \Theta \\ 1 - w & \text{if } X = A \text{ for some } A \subseteq \Theta \\ 0 & \text{otherwise} \end{cases} \tag{1.29}$$

where A is the focus and $w \in [0, 1]$.

This ssf describes a belief function induced by a piece of evidence supporting A (with $1 - w$) and leaving the remaining beliefs for Θ.

Note that the empty set $\emptyset$ can be considered as a focus of a simple support function.

Example 1.10 *Let's continue with the Example 1.1. Assume we have a bba defined as follows:*
$m(\{Helicopter, Missile\}) = 0.7$;
$m(\Theta) = 0.3$;

m is called a simple support function where the focus is the proposition $\{Helicopter, Missile\}$.

1.3.6 Bayesian belief function

A Bayesian belief function is a particular case of belief functions defined as follows (Shafer, 1976):

$$bel(\emptyset) = 0 \tag{1.30}$$

$$bel(\Theta) = 1 \tag{1.31}$$

$$bel(A \cup B) = bel(A) + bel(B) \text{ whenever } A, B \subset \Theta \text{ and } A \cap B = \emptyset \tag{1.32}$$

Properties

- *bel* is a Bayesian belief function if all its focal elements are singletons. Hence, *bel* becomes a probability distribution.
- As in the probability theory:

$$bel(A) + bel(\overline{A}) = 1 \text{ for } A \subset \Theta \tag{1.33}$$

- In the case of Bayesian belief functions we get:

$$bel = pl \tag{1.34}$$

Example 1.11 *Let's consider* $\Theta = \{Airplane, Helicopter, Missile\}$.

We get a piece of evidence expressed by the following bba m_b*:*
$m_b(\{Airplane\}) = 0.3$;
$m_b(\{Helicopter\}) = 0.4$;
$m_b(\{Missile\}) = 0.3$;
$m_b(\Theta) = 0$;

The bba m_b *is a Bayesian bba since all its focal elements are singletons.*

1.3.7 Consonant belief function

A belief function is said to be **consonant** if all its focal elements (A_1, A_2, ... , A_n) are nested, that is $A_1 \subseteq A_2 \subseteq \ldots \subseteq A_n$.

Properties

- Every simple support function is a consonant belief function.
- *bel* is a necessity measure:

$$bel(A \cap B) = \min(bel(A), bel(B)) \tag{1.35}$$

- pl is a possibility measure:

$$pl(A \cup B) = \max(pl(A), pl(B)) \tag{1.36}$$

Example 1.12 *Let's consider the same bba defined in the Example 1.2:*
$m(\{Airplane\}) = 0.6$;
$m(\{Airplane, Helicopter\}) = 0.2$;
$m(\Theta) = 0.2$;

We note that the focal elements of this bba m are nested. Hence, it is a consonant bba.

1.3.8 Dogmatic and non-dogmatic belief functions

A belief function is said to be **dogmatic** if and only if its corresponding bba m is such that $m(\Theta) = 0$. This case involves some previous cases (certain belief functions, Bayesian belief functions, categorical belief functions). **A non-dogmatic belief function** is defined such that $m(\Theta) > 0$ (Smets, 1995).

1.4 Combination

Handling information induced from different experts requires an evidence gathering process in order to get the fused information. The belief function theory, as understood in the TBM framework, offers interesting rules for aggregating the basic belief assignments (bba's) induced from distinct pieces of evidence and provided by two (or more) sources of information.

1.4.1 Combination of two information sources

Let m_1 and m_2 be two bba's defined on the same frame of discernment Θ. These two bba's are collected by two *'distinct'* pieces of evidence and induced from two experts (information sources). These bba's can be combined either conjunctively or disjunctively.

The conjunctive rule of combination

The conjunctive rule of combination handles the case where both sources of information are fully reliable. The result of combination is a joint bba presenting the conjunction of the two pieces of evidence induced from the

two sources. Hence, the induced bba quantifies the combined impact of the two pieces of evidence. It is defined as follows (Smets, 1998a):

$$(m_1 \textcircled{\cap} m_2)(A) = \sum_{B,C \subseteq \Theta : B \cap C = A} m_1(B) m_2(C) \tag{1.37}$$

This rule can be simply computed in terms of the commonality functions as follows:

$$(q_1 \textcircled{\cap} q_2)(A) = q_1(A) q_2(A) \tag{1.38}$$

where q_1 and q_2 are respectively the commonality functions corresponding respectively to the bba's m_1 and m_2.

The conjunctive rule is considered as the unnormalized Demspter's rule of combination dealing with the closed world assumptions, defined as follows (Shafer, 1976, 1986):

$$(m_1 \oplus m_2)(A) = K(m_1 \textcircled{\cap} m_2)(A) \tag{1.39}$$

where

$$K^{-1} = 1 - (m_1 \textcircled{\cap} m_2)(\emptyset) \tag{1.40}$$

$$\text{and } (m_1 \oplus m_2)(\emptyset) = 0 \tag{1.41}$$

K is called **the normalization factor**.

Properties

The conjunctive rule of combination is characterized by the following properties:

- *Compositionality:*
  $(m_1 \textcircled{\cap} m_2)$(A) is function of A, m_1 and m_2.
- *Commutativity:*

$$m_1 \textcircled{\cap} m_2 = m_2 \textcircled{\cap} m_1 \tag{1.42}$$

- *Associativity:*

$$(m_1 \textcircled{\cap} m_2) \textcircled{\cap} m_3 = m_1 \textcircled{\cap} (m_2 \textcircled{\cap} m_3) \tag{1.43}$$

- *Non-idempotency:*

  The conjunctive rule of combination is not idempotent. So usually:

$$m \textcircled{\cap} m \neq m \tag{1.44}$$

- *Neutral element:*

  The neutral element within the conjunctive rule of combination is the vacuous basic belief assignment representing the total ignorance. Hence,

$$m \textcircled{\cap} m_0 = m \tag{1.45}$$

  where m_0 is a vacuous bba.

- *Separable support function:*

  The combination of simple support functions using the conjunctive rule leads to the so-called **separable support function** (Smets, 1995).

Example 1.13 *Let's consider the frame of discernment given in the Example 1.1. Assume we have two distinct sensors S_1 and S_2 providing evidence on the nature of the detected aerial target.*

Two bba's m_1 and m_2 are relative to respectively S_1 and S_2.
$m_1(\{Airplane\}) = 0.6$;
$m_1(\{Airplane, Helicopter\}) = 0.2$;
$m_1(\Theta) = 0.2$;

$m_2(\{Helicopter\}) = 0.4$;
$m_2(\{Airplane, Helicopter\}) = 0.3$;
$m_2(\Theta) = 0.3$;

Applying the conjunctive rule of combination, we get:
$(m_1 \textcircled{\cap} m_2)(\emptyset) = 0.24$;
$(m_1 \textcircled{\cap} m_2)(\{Airplane\}) = 0.36$;
$(m_1 \textcircled{\cap} m_2)(\{Helicopter\}) = 0.16$;
$(m_1 \textcircled{\cap} m_2)(\{Airplane, Helicopter\}) = 0.18$;
$(m_1 \textcircled{\cap} m_2)(\Theta) = 0.06$;

$m_1 \textcircled{\cap} m_2$ *represents the joint bba induced from the combination of m_1 and m_2 by using the conjunctive rule of combination.*

The disjunctive rule of combination

The dual of the conjunctive rule is the disjunctive rule of combination that builds the bba representing the impact of two pieces of evidence when we only know that at least one of them is to be accepted, but we do not know which one. This rule is defined as follows (Smets, 1998a):

$$(m_1 \textcircled{\cup} m_2)(A) = \sum_{B,C \subseteq \Theta : B \cup C = A} m_1(B) m_2(C) \tag{1.46}$$

Example 1.14 *Let's continue with the same example (Example 1.13). However, we assume at least one of the two sensors (S_1 and S_2) is accepted, but we do not know which one. Hence, we will apply the disjunctive rule of combination. We get:*

$(m_1 \textcircled{\cup} m_2)(\{Airplane, Helicopter\}) = 0.56$;

$(m_1 \textcircled{\cup} m_2)(\Theta) = 0.44$;

Remark:

We have to mention that the disjunctive rule of combination (as the conjunctive rule of combination) is commutative and associative. So,

$$m_1 \textcircled{\cup} m_2 = m_2 \textcircled{\cup} m_1 \text{ and} \tag{1.47}$$

$$(m_1 \textcircled{\cup} m_2) \textcircled{\cup} m_3 = m_1 \textcircled{\cup} (m_2 \textcircled{\cup} m_3) \tag{1.48}$$

1.4.2 Combination of several information sources

Since the conjunctive and the disjunctive rules of combination are both commutative and associative, combining several pieces of evidence induced from distinct information sources (either conjunctively or disjunctively) may be easily ensured by applying repeatedly the chosen rule.

1.5 Generalized Bayes Theorem

Smets (1993a) has generalized the Bayesian theorem (GBT) within the TBM framework. Assume a vacuous a priori belief on a frame Θ where for each element θ_i of Θ you know what would be your beliefs on another frame X if θ_i happened ($bel^X[\theta_i]$, what is usually denoted as $bel(.|\theta_i)$).

Suppose you learn that the actual value of X is in $x \subseteq X$, then the GBT allows you to derive the conditional belief over Θ given x. In particular, one has (Smets, 1993a):

$$pl^{\Theta}[x](\theta) = 1 - \prod_{\theta_i \in \Theta} (1 - pl^{X}[\theta_i](x)) \tag{1.49}$$

Furthermore, if we assume we have only some beliefs on the value of x, and these beliefs are represented by a belief function bel^X over X (m^X its bba), then the GBT becomes:

$$pl^{\Theta}[m^X](\theta) = \sum_{x \subseteq X} m^X(x) pl^{\Theta}[x](\theta) \tag{1.50}$$

$$\text{where } pl_{\Theta}[\emptyset](\theta) = 0 \tag{1.51}$$

The bba's and the belief functions on Θ are computed from these plausibility functions.

1.6 Discounting

Dealing with evidence expressed by experts requires to take into account the level of expertise of each information source. In fact, experts are not fully reliable and a method of discounting seems imperative to update experts' beliefs by taking into account their reliability. The idea is to weight most heavily the opinions of the best experts and conversely for the less reliable ones.

Let $(1 - \alpha)$ be the degree of trust assigned to the expert (Ling & Rudd, 1989). In fact, it quantifies the strength of reliability given to this expert. There would be a bba defined on the set {reliable, not reliable} such that (Smets, 1992):

$$m(reliable) = 1 - \alpha \text{ and } m(not\ reliable) = \alpha \tag{1.52}$$

Updating the expert's opinions leads to:

$$m^{\alpha}(A) = (1 - \alpha) m(A) \text{ for } A \subset \Theta \tag{1.53}$$

$$m^{\alpha}(\Theta) = \alpha + (1 - \alpha) m(\Theta) \tag{1.54}$$

This operation is called by Shafer (1976) **a discounting** and the coefficient α is named **the discounting factor** (Guan & Bell, 1993). The larger α, the closer m^{α} is from the vacuous belief function.

Properties

- $\alpha = 0$ means that the expert is totally reliable.
- $\alpha = 1$ means that the expert is not reliable at all. His opinions have to be totally ignored.

Example 1.15 *Let's discount the bba m (given in the Example 1.2). We assume the degree of reliability given to the expert is equal to 0.8. So we get:*
$m^{\alpha}(\{Airplane\}) = 0.8 * 0.6 = 0.48$;
$m^{\alpha}(\{Airplane, Helicopter\}) = 0.8 * 0.2 = 0.16$;
$m^{\alpha}(\Theta) = 0.2 + (0.8 * 0.2) = 0.36$;

1.7 Projection and extension of belief functions

1.7.1 Introduction

Generally the bba's induced from experts are defined on different frames of discernment. In order to allow the combination of information, two concepts are proposed: **the projection** and **the extension of belief functions**.

In this section, we introduce the notions of variables, configurations and valuations. Then, we define the concepts of projection and extension of subsets of variables and those of belief functions.

1.7.2 Variables and configurations

Let ξ be a finite set where each element is represented by a variable. Variables are denoted by upper-case Latin alphabets, X, Y, Z, ..., whereas subsets of variables are denoted by lower-case Latin alphabets g, h, ... Each variable X is associated to its frame of discernment Θ_X which is the set of all the possible values related to this variable.

A frame of discernment Θ_h can be defined for a subset h of ξ. Such frame is the cross product of the different frames of discernment of the different variables.

$$\Theta_h = \times\{\Theta_X : X \in h\} \tag{1.55}$$

The elements of Θ_h are called **configurations** of h.

1.7.3 Valuations

For each subset of variables h, is assigned a set V_h. The elements of V_h are called **valuations** of h. The set of all the valuations is defined as follows (Shenoy, 1997):

$$V = \cup\{V_h : h \subseteq \xi\} \tag{1.56}$$

where ξ is the set involving all the variables.

In the case of belief functions, the valuation function is considered as a non negative numeric function defined on the frame of discernment of a subset of configurations of h (Θ_h) such that basic belief assignments, belief functions, plausibility functions, ...

1.7.4 Projection

Let g and h be two sets of variables, $h \subseteq g$. Let x be a configuration of g and δ be a non empty subset of the frame of discernment Θ_g.

The projection of configurations consists simply in dropping the extra coordinates. The projection of x to h is denoted by $\mathrm{x}^{\downarrow h}$.

The projection of the set δ on h is defined as follows (Shenoy & Shafer, 1990):

$$\delta^{\downarrow h} = \{x^{\downarrow h} : x \in \delta\} \tag{1.57}$$

1.7.5 Extension

The extension of subsets is accomplished from a little frame of discernment to a larger one.

Let g and h be two sets of variables, $h \subseteq g$, $g \neq h$ and λ be a non empty set of the frame of discernment Θ_h.

The extension of λ to g, denoted by $\lambda^{\uparrow g}$ is called **the cylindric extension**. It is obtained by applying $\lambda \times \Theta_{g-h}$.

1.7.6 Projection and extension of belief functions

Let g and h be two distinct sets of variables and let bel_g and bel_h be two belief functions, defined respectively on Θ_g and Θ_h and characterized by their corresponding bba's m_g and m_h.

If $h \subseteq g$, we can define the projection of bel_g on Θ_h denoted by $(bel_g)^{\downarrow h}$ which consists in dropping information, it is also called **marginalization**, whereas the extension of bel_h on Θ_g denoted by $(bel_h)^{\uparrow g}$ allows the information in bel_h to be extended to a larger frame.

Three cases related to the concepts of projection and extension are presented (Mellouli, 1987):

- **Case 1:** If $g \subseteq h$ then

$$(m_g)^{\uparrow h}(A \times \Theta_{h-g}) = m_g(A) \text{ for all A} \subseteq \Theta_g \tag{1.58}$$

$$(m_g)^{\uparrow h}(B) = 0 \text{ for all } B \text{ which has not the form of } A \times \Theta_{h-g} \tag{1.59}$$

  In this case, $(m_g)^{\uparrow h}$ is called **the vacuous extension** of m_g to Θ_h.

- **Case 2:** If $h \subseteq g$ then for all $A \subseteq \Theta_g$

$$Proj(A, \Theta_h) = \{\theta : \theta \in \Theta_h, A \cap (\{\theta\} \times \Theta_{g-h}) \neq \emptyset\} \tag{1.60}$$

$$(m_g)^{\downarrow h}(B) = \sum_{A \subseteq \Theta_g,\ Proj(A,\Theta_h)=B} m_g(A) \text{ for all } B \subseteq \Theta_h \tag{1.61}$$

  In this case, $(m_g)^{\downarrow h}$ is called **the marginal** of m_g to Θ_h.

- **Case 3:** If h $\not\subset$ g and g $\not\subset$ h then

$$(m_g)^{\downarrow h} = ((m_g)^{\downarrow g \cap h})^{\uparrow h} \tag{1.62}$$

  To compute $(m_g)^{\downarrow h}$, we project m_g on the set composed of the intersection between g and h, then we extend them to the set h.

Example 1.16 *Let's continue with the example of the aerial target and let's try to more specify each type of its targets.*

Let ξ be a set of variables such that:
$\xi = \{Type\_airplane, Type\_helicopter, Type\_missile\}$

We associate a frame of discernment including all the possible values of each variable. We get three frames of discernment relative respectively to Type_airplane, Type_helicopter, Type_missile. In order to facilitate the notations, we assign 1 for the variable Type_airplane, 2 for the variable Type_helicopter, 3 for the variable Type_missile.

So, $\xi = \{1, 2, 3\}$ *and let:*
$\Theta_{\{1\}} = \{Transport\_airplane, Fighter\_airplane\}$
$\Theta_{\{2\}} = \{Transport\_helicopter, Fighter\_helicopter\}$
$\Theta_{\{3\}} = \{Ground\_missile, Air\_missile\}$

Let $\Theta_{\{1,2\}}$ *be the frame of discernment relative to both Type_airplane and Type_helicopter. Using the Equation (1.55), we get:*

$$\begin{aligned}\Theta_{\{1,2\}} &= \Theta_{\{1\}} \times \Theta_{\{2\}} \\ &= \{(Transport\_airplane, Transport\_helicopter), (Transport\_airplane, \\ &\quad Fighter\_helicopter), (Fighter\_airplane, Transport\_helicopter), (Fighter\_airplane, Fighter\_helicopter)\}\end{aligned}$$

Let $bel_{\{1,2\}}$ *be a belief function defined on* $\Theta_{\{1,2\}}$ *and characterized by a bba* $m_{\{1,2\}}$ *such that:*
$m_{\{1,2\}}(\{(Transport\_airplane, Transport\_helicopter)\}) = 0.6;$
$m_{\{1,2\}}(\{(Transport\_airplane, Transport\_helicopter), (Transport\_airplane, Fighter\_helicopter)\}) = 0.2;$
$m_{\{1,2\}}(\Theta_{\{1,2\}}) = 0.2;$

- ***Case 1:*** *: Let's define* $(m_{\{1,2\}})^{\uparrow\{1,2,3\}}$*. Applying the Equations (1.58) and (1.59), we get:*
  $(m_{\{1,2\}})^{\uparrow\{1,2,3\}}(\{(Transport\_airplane, Transport\_helicopter)\} \times \Theta_{\{3\}}) = 0.6;$

  $(m_{\{1,2\}})^{\uparrow\{1,2,3\}}(\{(Transport\_airplane, Transport\_helicopter), (Transport\_airplane, Fighter\_helicopter)\} \times \Theta_{\{3\}}) = 0.2;$

  $(m_{\{1,2\}})^{\uparrow\{1,2,3\}}(\Theta_{\{1,2,3\}}) = 0.2;$

  The bba $(m_{\{1,2\}})^{\uparrow\{1,2,3\}}$ *is the vacuous extension of* $m_{\{1,2\}}$ *to* $\Theta_{\{1,2,3\}}$*.*

- ***Case 2:*** *: Let's define* $(m_{\{1,2\}})^{\downarrow\{1\}}$*. Applying the Equations (1.60) and (1.61), we get:*
  $(m_{\{1,2\}})^{\downarrow\{1\}}(\{Transport\_airplane\}) = 0.6 + 0.2 = 0.8;$

$(m_{\{1,2\}})^{\downarrow\{1\}}(\Theta_{\{1\}}) = 0.2;$

The bba $(m_{\{1,2\}})^{\downarrow\{1\}}$ *is the marginal of* $m_{\{1,2\}}$ *to* $\Theta_{\{1\}}$.

- ***Case 3:*** *: Let's define* $(m_{\{1,2\}})^{\downarrow\{1,3\}}$. *Applying the Equations (1.58), (1.59), (1.60), (1.61) and (1.62), we get:*
  $(m_{\{1,2\}})^{\downarrow\{1,3\}} = ((m_{\{1,2\}})^{\downarrow\{1\}})^{\uparrow\{1,3\}}$ *defined as follows:*
  $(m_{\{1,2\}})^{\downarrow\{1,3\}}(\{Transport\_airplane\} \times \Theta_{\{3\}}) = 0.6 + 0.2 = 0.8;$
  $(m_{\{1,2\}})^{\downarrow\{1,3\}}(\Theta_{\{1,3\}}) = 0.2;$

1.8 Coarsening and refinement

1.8.1 Introduction

In practice, it is common to deal with sources of information having different frames of discernment but compatible.

In order to deal with these sources and basically to be able to aggregate their beliefs, relations should be established between their different frames of discerment. Two operations are defined: **the refinement** and **the coarsening**.

1.8.2 Refinement and coarsening

Let Ω and Θ be two finite sets. The idea behind the refinement consists in obtaining one frame of discernment Ω from the set Θ by splitting some or all of its events (Shafer, 1976). On the other hand, the coarsening corresponds to a grouping together the events of a frame of discernment Θ to another frame compatible but which is more larger Ω (Smets, 1993b, 1997).

Let's define a mapping $\omega : 2^{\Theta} \rightarrow 2^{\Omega}$ such that (Shafer, 1976):

$$\omega(\{\theta\}) \neq \emptyset \text{ for all } \theta \in \Theta \tag{1.63}$$

$$\omega(\{\theta\}) \cap \omega(\{\theta'\}) = \emptyset \text{ if } \theta \neq \theta' \tag{1.64}$$

$$\bigcup_{\theta \in \Theta} \omega(\{\theta\}) = \Omega \tag{1.65}$$

So, given a disjoint partition $\omega(\{\theta\})$, we may set (Shafer, 1976):

$$\omega(A) = \bigcup_{\theta \in A} \omega(\{\theta\}) \tag{1.66}$$

where $\omega(A)$ represents all the possibilities in the frame Ω by splitting the elements of A.

Such mapping ω is called **a refining**, Ω is named **a coarsening** of Θ and Ω is **a refinement** of Θ.

Example 1.17 *Let's continue with the Example 1.1, a refinement of the frame of discernment Θ is:*

$\Omega = \{Transport\_airplane, Fighter\_airplane, Transport\_helicopter, Fighter\_helicopter, Ground\_missile, Air\_missile\}$ *where*

$\omega(Airplane) = \{Transport\_airplane, Fighter\_airplane\}$

$\omega(Helicopter) = \{Transport\_helicopter, Fighter\_helicopter\}$

$\omega(Missile) = \{Ground\_missile, Air\_missile\}$

On the other hand, Θ is considered as the coarsening of Ω.

1.8.3 Definition of bba's

Due to refinement and regarding bba's, it is easy to update a bba m^Θ defined on the frame of discernment Θ to a refinement Ω, we get the bba m^Ω defined as follows:

$$m^\Omega(B) = \begin{cases} m^\Theta(A) & \text{if } B = \omega(A) \text{ for some } A \subseteq \Theta \\ 0 & \text{otherwise} \end{cases} \tag{1.67}$$

The inverse operation is not obvious since it may exist some subsets A of Ω that are not be discerned by Θ and consequently are not equal for any $B \subseteq \Theta$. In (Shafer, 1976), we have the following definitions:

$$\overline{\theta} : 2^\Omega \to 2^\Theta$$

$$\overline{\theta}(A) = \{\theta \in \Theta : \omega(\{\theta\}) \cap A \neq \emptyset\} \tag{1.68}$$

Therefore, the bbm given to A belonging to Ω, by a bba m^Ω, can be transferred to $\overline{\theta}(A)$, so we get:

$$m^\Theta(B) = \sum_{\{A \subseteq \Omega, B = \overline{\theta}(A)\}} m^\Omega(A) \text{ for all } B \subseteq \Theta \tag{1.69}$$

This last equation results from the condition $bel^\Theta(B) = bel^\Omega(\omega(B))$ for all $B \subseteq \Theta$

1.9 Decision process

1.9.1 Introduction

The theory of belief functions is characterized by its ability to handle uncertainty and to ensure the combination of evidence induced from different sources.

In order to make a decision, we hope to select the most likely hypothesis which may be difficult to realize directly with the basics of the belief function theory where bbm's are given not only to singletons but also to subsets of hypotheses.

In this section, we present some solutions allowing to ensure the decision making within the belief function theory. The best known is the pignistic probability proposed by the Transferable Belief Model (TBM) (Smets, 1988, 1998b; Smets & Kennes, 1994; Smets & Kruse, 1997). Other criteria will be presented like the maximum of credibility and the maximum of plausibility (Janez, 1996).

1.9.2 Pignistic probability

The TBM is based on a two level mental models:

- *The credal level* where beliefs are entertained and represented by belief functions.
- *The pignistic level* where beliefs are used to make decisions and represented by probability functions called **the pignistic probabilities**.

When a decision must be made, beliefs held at the credal level induce a probability measure at the pignistic measure denoted $BetP$ (Smets, 1998b). The link between these two functions is achieved by the pignistic transformation.

$$BetP(A) = \sum_{B \subseteq \Theta} \frac{|A \cap B|}{|B|} \frac{m(B)}{(1 - m(\emptyset))}, \text{ for all A} \subseteq \Theta \qquad (1.70)$$

It is the only transformation between belief functions and probability functions that satisfies some natural rationality requirements. The major one is described as follows: Suppose two contexts C_1 and C_2, suppose your beliefs in context C_i is represented by m_i and that the choice of the context obeys to some random process, with $P(C_1) = p$ and $P(C_2) = q$ with $p+q = 1$.

Let Γ denotes the operator that transforms a bba into a probability function. We want that it satisfies:

$$\Gamma(p\ m_1 + q\ m_2) = p\ \Gamma(m_1)\ +\ q\ \Gamma(m_2) \tag{1.71}$$

This translates the property that transforming the belief held before knowing the context that will be selected is the same as combining the conditional probability functions one would have obtained if the context had been known. Full details can be found in (Smets, 1998b, 2002; Smets & Kennes, 1994; Smets & Kruse, 1997). The probability function so obtained is then used to compute the expected utilities needed for optimal decision making.

Note that all over this thesis, we will use the pignistic transformation as a process to help us to make a decision.

Example 1.18 *Let* $\Theta = \{Airplane, Helicopter, Missile\}$.

Assume at the credal level, we have the following bba m defined as follows:
$m(\{Helicopter\}) = 0.4$;
$m(\{Airplane, Helicopter\}) = 0.3$;
$m(\Theta) = 0.3$;

In order to make a decision, we have to compute the pignistic probability BetP corresponding to the bba m, we get:
$BetP(\{Airplane\}) = 0.25$;
$BetP(\{Helicopter\}) = 0.65$;
$BetP(\{Missile\}) = 0.1$;

It is more probable that the detected aerial target is a helicopter.

1.9.3 Maximum of credibility

It consists in choosing the hypothesis having the highest value of the belief function *bel*, that is the most credible hypothesis.

Decision based on the maximum of credibility is considered as a pessimistic approach since it chooses the *'best'* hypothesis based on the minimum *'chance'* to realize (Janez, 1996).

Example 1.19 *Let's continue with the Example 1.18, and try to make a decision by using the criterion of the maximum of credibility. We get:*
$bel(\{Airplane\}) = 0$;
$bel(\{Helicopter\}) = 0.3$;
$bel(\{Missile\}) = 0$;

The largest value of bel is the one assigned to the hypothesis $\{Helicopter\}$.

According to the maximum of credibility, the helicopter is the detected aerial target.

1.9.4 Maximum of plausibility

It consists in choosing the hypothesis having the highest value of the plausibility function *pl* which means that we support the hypothesis that gives the less evidence for the contrary hypothesis.

Contrary to the maximum credibility criterion, this criterion is considered as optimistic since it takes into account of the maximum of *'chance'* of realization of each hypothesis.

Example 1.20 *Let's continue with the Example 1.18, and try to make a decision by using the criterion of the maximum of plausibility. We get:*
$pl(\{Airplane\}) = 0.6$;
$pl(\{Helicopter\}) = 1$;
$pl(\{Missile\}) = 0.3$;

The largest value of pl is the one assigned to the hypothesis $\{Helicopter\}$. *So according to the maximum of plausibility, the helicopter is the detected aerial target.*

1.10 Conclusion

In this chapter, we have presented the basic concepts of the belief function theory as understood in the transferable belief model. The different notions are illustrated by examples.

Through this presentation, this theory seems appropriate to handle uncertainty in classification problems especially within the decision tree technique. The following chapter will deal with this technique of decision tree where its basics will be described.

Chapter 2

Decision trees

2.1 Introduction

Decision trees are considered as one of the most widely used classification techniques. They represent a sequential procedure for deciding the class membership of a given instance.

Decision trees are especially used in artificial intelligence since their ability to express classification knowledge in a formalism easy to interpret. So, they are applied successfully to many areas such as expert systems, medical diagnoses, speech recognition, etc. They can also be used in other fields like marketing, finance, industry, and so on.

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 partition corresponds to a classification sub-problem.

In this chapter, we are interested to the basics of decision trees. We focus on standard decision trees where their representation, objectives and procedures will be described, then we present some decision tree algorithms. Finally, advantages and drawbacks of decision trees will be detailed.

In the second part of this chapter, two kinds of this classification technique in an uncertain and imprecise environment will be briefly exposed: probabilistic decision trees and fuzzy decision trees.

2.2 Terminology

A typical problem of classification is described as follows (Yuan & Shaw, 1995):

- **Object universe (U):** It contains all the objects (instances) already classified or to classify.

  $U = \{I_1, I_2, ...\}$

- **Attributes (A):** They describe each instance of the object universe U. In fact, each attribute is considered as one property of the object and can be represented by either a qualitative variable (**Eg:** color) or a quantitative variable (**Eg:** height, old, etc) which may have either discrete or continuous values (Baim, 1988).

  $A = \{A_1, A_2, ..., A_m\}$

- **Classes (C):** They are represented by the set of classes in which each object of the universe U has to belong.

  $C = \{C_1, C_2, ..., C_n\}$

- **Training set (T):** It includes all the objects whose classes are known. Hence, the elements of this set are $(m + 1)$-tuples where each element is composed by the attributes' values of the object, and its assigned class which is unique.

Hence, the classification task is to find a general classification rule that works well on the objects in a given training set (Quinlan, 1990a) in order to be useful to correctly classify new objects. A high rate of correct classification remains the main objective of decision trees.

2.3 Decision tree representation

A decision tree is a representation of a decision procedure allowing to determine the class of an object. It is composed of three basic elements:

1. **A decision node** specifying a test attribute.
2. **An edge** or **a branch** corresponding to the one of the possible attribute values which means one of the test attribute outcomes. It leads generally to a sub decision tree or to a leaf.
3. **A leaf** which is also named **an answer node**, including objects that, typically, belong to the same class, or at least are very similar.

In order to classify a new instance, 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 instance. This process will be repeated until a leaf is encountered.

In a decision tree, each path from the root to a leaf corresponds to a conjunction of test attributes and the tree is considered as a disjunction of these conjunctions.

Example 2.1 *Let's give a simple example of a classification problem that may be treated by the decision tree technique. Assume a bank wants to develop a classification of its clients by taking into account a number of their attributes. This classification will be useful since it allows to the bank to plan its loan policy.*

- *The object universe will be composed by the different clients:*

  $U = \{Client1, Client2, ...\}$

- *The attributes are the different characteristics allowing to distinguish one category of the bank's clients from another. In this example, for sake of simplicity, we only consider three attributes which are:*

  $A = \{Income, Property, Unpaid\_credit\}$

  Each attribute may take the following values:

  - Income with possible values $\{No, Low, Average, High\}$,

  - Property with possible values $\{Less, Greater\}$, *that is to express if the property's value of the client is less or greater than the loan expected by him,*

  - Unpaid_credit with possible values $\{Yes, No\}$ *in order to know if the client has an unpaid credit or not.*

- *Three classes are defined by the bank and one of them will be assigned to each client asking for a loan:*

  - C_1 *including good clients, i.e., reliable clients, for whom the bank accepts to give the whole loan.*

  - C_2 *to which belongs moderate clients, for whom the bank accepts to give a part of the loan.*

  - C_3 *regrouping 'bad' clients for whom the bank refuses to give the loan.*

  $C = \{C_1, C_2, C_3\}$;

Now, we have to define the different classification rules allowing to find for each client the appropriate class regarding its attributes.

For instance, we could have this kind of decision tree (see Figure 2.1):

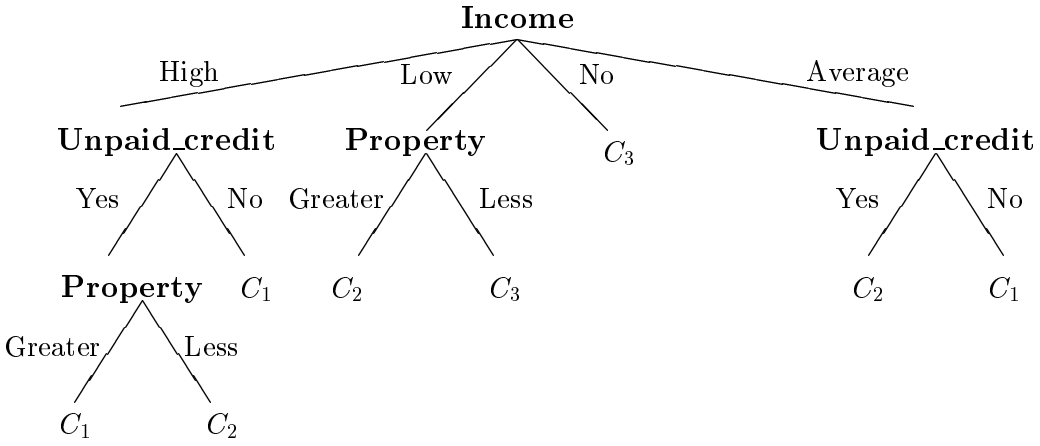

Figure 2.1: Decision tree

2.4 Objectives

The major objectives of a decision tree can be summarized as follows (Safavian & Landgrebe, 1991):

1. Classifying correctly as much as possible the objects of the training set.
2. Classifying correctly the new objects (not belonging to the training set) with a high rate, i.e., with a high percent of correct classification.
3. Having a simple structure easy to understand.
4. Updating easily the training set.

2.5 Decision tree procedures

A decision tree is made of two different procedures:

1. The first for building the tree.
2. The second for the classification of objects.

2.5.1 Building decision tree procedure

Description

The building of a decision tree is based on a given training set. It consists in selecting for each decision node the *'appropriate'* test attribute and also to define the class labeling each leaf of the induced tree.

Generally, the first task is to generate the different tests on each variable (attribute). For quantitative variables (either discrete or continuous), we get binary tests ($X \geq$, $X <$, ...), whereas for the qualitative variables these tests have to take into consideration all the different values of the variable. However, a pooling of certain values into one alternative may be possible (Decaestecker, 1997).

Then, the objective is to find in each decision node of the tree, the best test attribute allowing to diminish as much as possible the mixture of classes between each subset created by the test. In other words, the main idea is therefore to find the test attribute in order to get disjoint data facilitating the determination of objects' classes. This process will continue for each sub decision tree until reaching leaves and fixing their corresponding classes.

More details relative to building decision tree algorithms will be found in Section 2.6.

Pruning step

The pruning step is intimately related to the induced tree and consequently to the construction procedure. It consists in removing some edges that seem useless to forecast new objects' classes. It allows to simplify a decision tree already built and which may be complex and characterized by several levels. Hence, it represents a good way to control the growth of the tree, but it remains an optional procedure in decision tree algorithms.

Instead of a full-size tree, we will get a smaller tree which should give a better classification performance with a minimum error rate.

Two kinds of pruning can be applied (Esposito, Malerba, & Semeraro, 1997), (Furnkranz, 1997):

1. **The pre-pruning:** applied while the decision tree is built. This pre-pruning affects strongly the stopping criteria of the growth of the tree.

2. **The post-pruning:** considered as the common pruning which is applied after the construction of the decision tree.

2.5.2 Classification procedure with decision trees

In this section, we present an inference procedure for decision making regarding the classification task. In fact, the classification of a new instance is ensured as follows (Quinlan, 1987b):

- If the root of the decision tree is a leaf

  then the instance's class will be the one labeling this leaf.

- Else if the root is a test attribute

  then each test's outcome will be a sub decision tree and the process continues.

To classify an object, we start with the root of the decision tree, we test the attribute specified by this node. The result of this test allows us to move down the tree branch according to the attribute value of the given instance. This process is repeated until a leaf is encountered, the instance is being then classified in the same class as the one characterizing the reached leaf.

We notice that a path in a decision tree is equivalent to a production rule with the standard form:

If [premiss] then [conclusion]

where the premiss is the conjunction of the different tests, whereas the conclusion contains the class mentioned by the leaf found when at least one of the stopping criteria is verified.

Therefore, a decision tree is considered as a rule basis where each new instance can be classified by using the classical inference method of the deduction reasoning (as in artificial intelligence).

2.6 Algorithms

2.6.1 Introduction

Several algorithms have been developed in order to ensure the construction of decision trees. These algorithms are among the most well known and applied of all classification techniques especially in a supervised learning.

In fact, there are two ways to construct a decision tree:

1. The entire training set is available, thus the tree will be built by taking into account this whole set. Such used procedure is referred to as **a non-incremental algorithm**.
2. The training set arrives in a stream. For this case, two methods may be applied:
   - Once the new training set becomes available, we will discard the current tree and replace it by taking into account the enlarged training set.
   - Based on the new information, we will revise the existing tree. This is known as **an incremental algorithm**.

Among the non-incremental algorithms, we mention the **ID3** and **C4.5** algorithms developed by Quinlan (1986, 1993) which are probably the most popular ones. We can also mention the **CART** algorithm of Breiman et al. (1984).

In addition to these non-incremental algorithms, many incremental building decision tree algorithms have been proposed such as the **ID4** of Schlimmer and Fisher (1986) and notably **ID5** and **ID5R** of Utgoff (1988, 1989b).

2.6.2 Hunt's procedure

The original idea of an algorithm allowing the building of a decision tree relative to a training set, was initially proposed by Hunt in the late 1950's (Hunt, Marin, & Stone, 1966).

Let T be a training set of objects which may belong to one of the classes C_1, C_2,..., C_n.

1. If all the objects belong to the same class, the decision tree relative to the training set will be a leaf labeled by this class.

2. Otherwise (T contains objects belonging to a mixture of classes), let an attribute A_k be a test with possible values (v_1, v_2, ... , v_f). The training set will be partitioned into subsets $T_1, T_2, \ldots, T_f$ where each set T_t ($t = 1, \ldots, f$) corresponds to the value v_t of the test A_k. Then, the same procedure will be applied for each subset.

As a conclusion, a decision tree relative to a training set consists in a decision node identifying the test attribute, then one branch will be created for each possible outcome. The same process will be repeated recursively.

2.6.3 Decision tree algorithm parameters

The majority of building decision tree algorithms proceeds identically, by using a descendent way (from the root to the leaves). To ensure this approach, many parameters have to be defined and they can be considered as generic parameters of the algorithm.

The building decision tree formalism is also referred to as **T**op **D**own **I**nduction of **D**ecision **T**ree (**TDIDT**) (Van de Merckt, 1995) since it proceeds to successive divisions of the training set where each division represents a question about an attribute value.

A generic decision tree algorithm is characterized by the next properties:

1. **The attribute selection measure:** An attribute is chosen in order to partition the training set in an *'optimized'* manner. A decision node relative to this attribute is created. It becomes the root of the corresponding (sub) decision tree. The idea is to use an attribute selection measure taking into account the discriminative power of each attribute over classes. In other words, considering the ability of each attribute to determine training objects' classes.

   The selection measure is generally based on the information theory (Shannon, 1948), we can for instance mention those suggested by Quinlan: **the information gain** (Quinlan, 1986) and **the gain ratio** (Quinlan, 1993). Other measures are also developed namely **the Lopez De Mantaras measure** (Lopez De Mantaras, 1991), **the normalized gain** (Jun & Kim, 1997). More details can be found in (Mingers, 1989a), (Liu & White, 1994), (White & Liu, 1994).

2. **The partitioning strategy:** The current training set will be divided by taking into account the selected test attribute. In the case of symbolic attributes (with a finite number of values), this strategy consists in testing all the possible attribute values, whereas in the case of numeric attributes, a discretization step is generally needed (Fayyad & Irani, 1992), (Wehenkel, 1997).

3. **The stopping criteria:** They deal with the condition(s) of stopping the growth of a part of the decision tree (or even all the decision tree). In other words, they determine whether or not a training subset will be further divided.

   It is generally fulfilled when all the remaining objects belong to only one class. Therefore, the part of the decision tree verifying this criterion will be declared as a leaf. Note that other stopping criteria can be proposed like the case where there is no further attribute to test.

These different steps are applied recursively to the training subsets that do not verify any stopping criteria.

The major difference between these algorithms lies basically on the choice of these parameters (the attribute selection measure, the partitioning strategy, the stopping criteria).

2.6.4 Examples of building decision tree algorithms

Definitions

Let's denote by T a training set and $C = \{C_1, C_2, \ldots, C_n\}$ be the set of n mutually exclusive and exhaustive classes so that each instance in T belongs to one and only one class. Let A_k be one attribute which domain $D(A_k)$ is finite, $D(A_k)$ contains all the values relative to the attribute A_k.

The algorithms that will be presented in this section use **the information gain criterion** of Quinlan (1986, 1993) defined as follows:

$$Gain(T, A_k) = Info(T) - Info_{A_k}(T) \tag{2.1}$$

$$\text{where } Info(T) = -\sum_{i=1}^{n} \frac{freq(C_i, T)}{|T|} log_2 \frac{freq(C_i, T)}{|T|} \tag{2.2}$$

$$\text{and } Info_{A_k}(T) = \sum_{v \in D(A_k)} \frac{|T_v^{A_k}|}{|T|} Info(T_v^{A_k}) \tag{2.3}$$

$freq(C_i, T)$ denotes the number of objects in the set T belonging to the class C_i and $T_v^{A_k}$ is the subset of objects for which the attribute A_k has the value v.

Note that $Info(T)$ represents the classical formula for the entropy relative to the training set T and $Info_{A_k}(T)$ represents a weighted average of the entropies relative to all the subsets $T_v^{A_k}$ $(v \in D(A_k))$.

The best attribute relative to the training set T is the one that maximizes $Gain(T, A_k)$.

ID3 algorithm

ID3 is an inductive learning algorithm that constructs classification rules in the form of a decision tree. The input of the ID3 is the training set composed of objects and their classes, and produces a classification procedure of objects as an output (Maher & Clair, 1993).

The different steps of the ID3 algorithm are summarized as follows:

1. If all instances belong to one class, then the decision tree is a leaf containing that class.
2. Otherwise,
   - Define the test attribute by using the information gain criterion. The best attribute is the one that maximizes $Gain(T, A_k)$.
   - Divide the training set T into several subsets, one for each value of the selected attribute.
   - Apply the same procedure for each subset using only the data that belong to them.

Example 2.2 *This a simple example usually found in the literature (Quinlan, 1983) in order to explain the unfolding of the ID3.*

Let T be the training set composed by eight objects which are characterized by three attributes:

- ***Eyes:*** *Brown or Blues.*
- ***Hair:*** *Blond or Black or Red.*
- ***Height:*** *Tall or Short.*

Two classes are possible either, C_1 or C_2. T is described as follows (see Table 2.1):

Table 2.1: Training set

Hair	Eyes	Height	Class
Blond	Brown	Short	C_1
Blond	Blue	Tall	C_2
Blond	Brown	Tall	C_2
Blond	Blue	Short	C_1
Red	Blue	Tall	C_2
Dark	Brown	Short	C_1
Dark	Blue	Tall	C_1
Dark	Brown	Tall	C_2

Let's compute the value of $Info(T)$ relative to the training set T using the Equation (2.2):

$$\begin{aligned} Info(T) &= -\frac{freq(C_1,T)}{|T|}log_2\frac{freq(C_1,T)}{|T|} - \frac{freq(C_2,T)}{|T|}log_2\frac{freq(C_2,T)}{|T|} \\ &= -\frac{4}{8}log_2\frac{4}{8} - \frac{4}{8}log_2\frac{4}{8} \\ &= 1; \end{aligned}$$

We have to apply the Equation (2.1) in order to compute the information gain of each attribute. Let's firstly compute $Info_{Hair}(T)$, $Info_{Eyes}(T)$, and $Info_{Height}(T)$ by using the Equation (2.3):

$$\begin{aligned} Info_{Hair}(T) &= \frac{4}{8}Info(T^{Hair}_{Blond}) + \frac{1}{8}Info(T^{Hair}_{Red}) + \frac{3}{8}Info(T^{Hair}_{Dark}) \\ &= \frac{4}{8}(-\frac{2}{4}log_2\frac{2}{4} - \frac{2}{4}log_2\frac{2}{4}) + \frac{1}{8}(-\frac{1}{1}log_2\frac{1}{1}) + \frac{3}{8}(-\frac{2}{3}log_2\frac{2}{3} - \frac{1}{3}log_2\frac{1}{3}) \\ &= 0.844; \end{aligned}$$

$$Info_{Eyes}(T) = \frac{4}{8}Info(T^{Eyes}_{Blue}) + \frac{4}{8}Info(T^{Eyes}_{Brown})$$
$$= 1;$$

$$Info_{Height}(T) = \frac{5}{8}Info(T^{Height}_{Tall}) + \frac{3}{8}Info(T^{Height}_{Short})$$
$$= 0.451;$$

Hence,

$$Gain(T, Hair) = Info(T) - Info_{Hair}(T)$$
$$= 1 - 0.844$$
$$= 0.156;$$

$$Gain(T, Eyes) = 0;$$

$$Gain(T, Height) = 0.549;$$

According to the information gain criterion, the height attribute will be selected as the root of the decision tree relative to the training set. So, we get (see Figure 2.2):

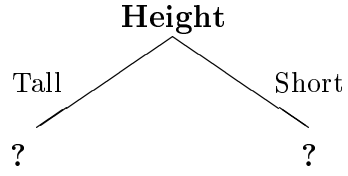

Figure 2.2: Decision tree (Level 1)

We have two branches in level 1, we have to look for the test attribute for each node induced by each branch.

- *Let's start by the first branch, for simplifying notations let's denote by $T1$ the subset T^{Height}_{Tall} containing the objects of T having tall as the height value:*

$$Info(T1) = -\frac{1}{5}log_2\frac{1}{5} - \frac{4}{5}log_2\frac{4}{5}$$
$$= 0.722;$$

$$Info_{Hair}(T1) = \frac{2}{5}Info(T1^{Hair}_{Blond}) + \frac{1}{5}Info(T1^{Hair}_{Red}) + \frac{2}{5}Info(T1^{Hair}_{Dark})$$
$$= 0.4;$$

$$Info_{Eyes}(T1) = \frac{3}{5}Info(T1^{Eyes}_{Blue}) + \frac{2}{5}Info(T1^{Eyes}_{Brown})$$
$$= 0.551;$$

Hence, $Gain(T1, Hair) = 0.322$ *and* $Gain(T1, Eyes) = 0.171$*;*

The hair attribute will therefore be the root of the sub decision tree relative to the branch of tall height.

We have to apply the same algorithm in order to continue with the induced sub decision tree.

- *For the second branch which corresponds to a short height, we notice that one of the stopping criteria is fulfilled. In fact, the three instances having short as a value of the height attribute are belonging to the same class* C_1*, so the induced node from the short branch is declared as a leaf and labeled by the class* C_1*.*

Continuing the same process, the final decision tree will be presented by (see Figure 2.3):

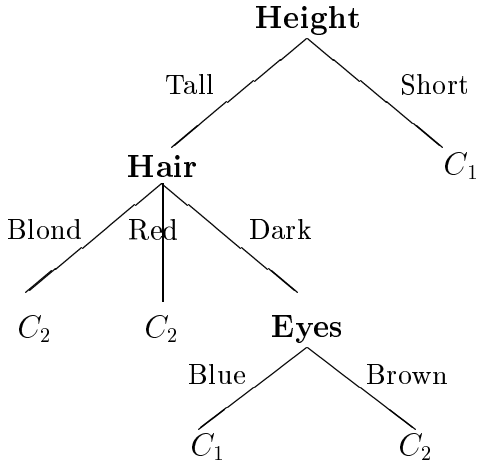

Figure 2.3: Final decision tree

Remark:

Although it has shown good results, the information gain criterion presents a serious limitation. It favors attributes with a large number of values over those with few number of values (Quinlan, 1993).

To overcome this drawback, Quinlan has proposed a kind of normalization known as **the gain ratio criterion**. In this manner, the attributes with many values will be adjusted.

$$Gain\ ratio(T, A_k) = \frac{Gain(T, A_k)}{Split\ Info(T, A_k)} \tag{2.4}$$

$$\text{where } Split\ Info(T, A_k) = -\sum_{v \in D(A_k)} \frac{|T_v^{A_k}|}{|T|} log_2 \frac{|T_v^{A_k}|}{|T|} \tag{2.5}$$

$Split\ Info(T, A_k)$ measures the information in the attribute due to the partition of the training set T into training subsets. This quantity describes the information content of the attribute itself.

The idea is to compute the gain ratio of each test attribute, the one presenting the highest value will be selected as the attribute test.

A more complete version of induction of decision trees called **C4.5** algorithm is available in (Quinlan, 1993). It presents in details the gain ratio as the attribute selection measure to be used to choose attributes in the decision nodes. Besides, it offers some improvements related notably to the pruning step.

ID4 algorithm

ID4 is one of the incremental decision tree algorithms developed by Schlimmer and Fisher (1986) which incrementally builds a decision tree. In fact, instead of building a tree using a set of objects, ID4 will update the induced decision tree based on each individually observed instance (Utgoff, 1988).

For each new instance, ID4 applies the following steps:

1. For each node, recompute the values of the attribute selection measure for the different attributes.

2. If the node is a leaf, and the new object does not affect it, then no change will be induced on this node.

3. Otherwise, change the leaf node to a decision node such that the chosen attribute is the one having the highest attribute selection measure value.

4. If the current node is a decision node such that its current attribute root won't be the best one, change it by the appropriate attribute, then discard all the sub-trees below.

5. Update the tree recursively and when necessary grow a branch.

This algorithm presents an important drawback concerning the discard of a part (or all) of the tree when the test attribute should be replaced since it constitutes a lost training effort.

ID5R algorithm

Another known incremental algorithm is the ID5R[1]. Contrary to the ID4, where we have to discard the subtrees below the old attribute, in the ID5R algorithm the idea is to restructure the tree (by a pull-up process), so that the desired test attribute becomes the new root. This is allowing to re-calculate the proportions of classes without needing to re-examine training objects. For more details see (Utgoff, 1988, 1989b).

This algorithm has the interesting property to induce the same tree as when all the training objects had been given in one batch (Utgoff, 1989a).

Remarks:

There are several decision tree algorithms where the differences reside generally in the used attribute selection measure and the applied pruning technique. Other parameters like the stopping criteria and the partitioning strategy may distinguish between these algorithms.

As exposed in this section, another aspect may differentiate between decision tree algorithms concerning the incremental algorithms and the non-incremental ones. We have to mention that in this thesis, we will be interested only with this latter kind of algorithms and notably those of Quinlan: the ID3 and C4.5 algorithms (Quinlan, 1986, 1993).

[1] The first version was developed under the name of ID5 in 1988 (Utgoff, 1988, 1989b).

2.7 Advantages and drawbacks

2.7.1 Advantages

Decision trees are characterized by their capability to break down a complex decision problem into several simpler decisions. They provide a powerful formalism for representing comprehensible, accurate classifiers (Quinlan, 1990b).

Furthermore, they allow to express knowledge in a simple manner easily understandable by the users and which facilitates the participation of experts in the production of rules that may exist between attributes.

Besides, they offer the possibility to integrate qualitative and quantitative variables in the model, they also select the most informative variables.

The different tests on each attribute value allow to reduce the computational complexity. In fact, unlike some classification techniques where each attribute value should be tested on the different classes, in decision trees these tests will be applied only to some classes. Thus, it will make the classification process faster.

Finally, this technique seems able to adapt to environment where lies uncertainty and imprecision. Hence, **probabilistic decision trees** (Quinlan, 1990b) and **fuzzy decision trees** (Umano et al., 1994), (Zeidler & Schlosser, 1996), (Marsala, 1998) have been developed and will be described in the next section (see Section 2.8).

2.7.2 Drawbacks

Despite their advantages, the decision trees present some drawbacks. We notably mention the overlap related to the large number of classes which may increase the number of leaves in the tree and consequently complicate the comprehension and the interpretation of the tree.

We also notice the accumulation of errors from level to level especially in a large tree. Besides, there is a difficulty to find the optimal decision tree, this difficulty is strongly related to the design of the decision tree.

2.8 Decision trees under probability and fuzzyness

The results provided by decision trees are categorical and do not convey potential uncertainties in classification (Quinlan, 1987a). Hence, small changes in the attribute value may affect the membership class. Furthermore, imprecise or missing information may prevent the unfolding of the classification task.

Hence, two kinds of decision trees have been developed: **probabilistic decision trees** and **fuzzy decision trees**.

2.8.1 Probabilistic decision trees

Definition

A probabilistic decision tree (Quinlan, 1987a, 1990b) deals with statistical uncertainty. It is interested basically to the classification of objects characterized by missing or uncertain attribute values. A probabilistic decision tree is constructed based on a training set having the same structure as in an ordinary decision tree, but where the induced tree is pruned.

Imperfect leaves

A leaf produced by pruning may generally misclassify some objects in the training set (Quinlan, 1987a). Hence, some leaves will represent more than one class: the objects really belonging to the leaf's class and those which are misclassified (due to generally the application of pruning) . These leaves are referred to as **imperfect leaves**.

Let nb be the number of objects in a leaf and e be the number of objects that are associated to this leaf but do not really belong (due to the pruning). So, the probability that a new object has as a class the leaf's class is $\frac{nb-e}{nb}$. This ratio is referred to as central estimation.

A more pessimistic estimation is equal to $\frac{nb-e-0.5}{nb}$, this latter estimation is invoked when a leaf contains a very small value of nb. So, the first estimation may not give a reliable probability of error of new objects.

An object at a leaf L has generally a probability $P(C|L)$ representing that the object at L belongs to the class C and which is equal to the previous estimation (either central or pessimistic).

Unknown or imprecise attributes

A good classifier must be able to give the object's class even when some information needed to ensure the classification task are not available or uncertain.

The general idea preconized by Quinlan (1990b) is to take into account all the possible alternatives for the unknown attribute value. Then for each possible value, we have to look for the assigned class. Finally, an aggregation of the different results has to be done.

Probability of belonging to a leaf

A path from the root of the decision tree to a leaf L passes through branches (edges) B_1, B_2, ... where each branch corresponds to the result of a particular test. Hence, the probability that an object I will reach a leaf L is defined as (Quinlan, 1990b) :

$$P_I(L) = P_I(B_1)P_I(B_2|B_1)P_I(B_3|B_2\&B_1)... \tag{2.6}$$

In the simple case where all the attribute values are known, each of these probabilities will be either 0 or 1 and consequently P_I will be equal to 1 if the object I will reach the leaf L. Otherwise, it will be equal to 0.

In this case, the path followed by taking into account the tests' outcomes will be unique and the object will be associated to only one leaf. However in the case where there are unknown attribute values, the probability $P_I(L)$ may be non-zero for more than one leaf and may also be less than one.

Classification procedure

When a new instance is presented with some unknown attribute values, it may belong to many classes with different probabilities. The probability that an object I belongs to a class C is given by (Quinlan, 1990b):

$$\sum_L P_I(L)P(C|L) \tag{2.7}$$

where $P(C|L)$ is the probability that the class C is associated to the leaf L (either central or pessimistic estimate). Remember that this probability is equal to 1 if the leaf is represented by just only one class.

Remarks:

The same approach can be used when we deal with partial information regarding an attribute value. We are just interested to the values shown by the partial information.

For continuous attributes, the value might be specified in a range $[X, Y]$. If we assume the attribute value is uniformly distributed in this interval, the probability that the attribute has a value less than or equal to some threshold V is (Quinlan, 1990b):

$$\textit{If } V < X \quad \text{Probab} = 0 \tag{2.8}$$

$$\textit{If } V > Y \quad \text{Probab} = 1 \tag{2.9}$$

$$\textit{Otherwise: } \text{Probab} = \frac{V - X}{Y - X} \tag{2.10}$$

These probabilities will be used to compute $P_X(B_i|B_1\&....)$.

2.8.2 Fuzzy decision trees

Introduction

The work started by Quinlan (1987a, 1990b) concerning the probabilistic decision trees remains insufficient since it basically emphasizes on the classification task and it is only based on the treatment of statistical uncertainty dealing with information presented in a probabilistic manner.

In fact, two types of uncertainty may occur: statistical and cognitive. This latter which may be presented by imprecise or missing information can also be treated by applying fuzzy theory. Thus, the idea is to develop the so-called **fuzzy decision trees**.

Definitions

Fuzzy concepts can be introduced in a classification problem into two levels (Yuan & Shaw, 1995):

1. **Object:** An object is said to be **fuzzy** if at least one of its attributes is fuzzy.

2. **Class:** A class is said to be **fuzzy** if it can be represented in fuzzy terms.

Like the classical one, a fuzzy decision tree is composed by three kinds of elements (Marsala & Meunier, 1997):

1. **Nodes:** for testing attributes.

2. **Edges:** associated to the results of the test attributes and which their values are represented by fuzzy sets.

3. **Leaves:** as terminal nodes, labeling classes with membership degrees.

We notice that in a fuzzy decision tree, each leaf may be labeled by more than one class.

Hence, fuzzy decision trees are characterized by their ability to take into account imprecision and fuzziness of the knowledge either presented in attributes or even in classes.

Methods of construction of fuzzy decision trees

Several methods to build fuzzy decision trees have been proposed, most of them are based on the ID3 algorithm (Janikow, 1998). The major difference between ID3 and the fuzzy ID3 lies basically on the computation of the information gain. For the ID3 algorithm, this measure is based on the proportions of the attribute values. However in the fuzzy algorithm, it uses the membership degrees of each attribute values.

As in classical decision trees, the difference between these algorithms in the fuzzy context resides especially in the choice of the attribute selection measure which has to take into account the discrimination power and also the fuzzy modalities assigned to the attributes.

The fuzzy ID3 algorithm (Umano et al., 1994): This algorithm, developed by Umano et al. (1994) is considered as a new version of the ID3 algorithm treating numeric attributes presented by fuzzy modalities.

In fact, the fuzzy ID3 algorithm is the extension of the ID3 applied to a fuzzy set of data and where the objective is to generate a fuzzy decision tree where the attributes are fuzzy sets defined by the user.

This algorithm differs from the classical ID3 on the way to compute the information gain based on the probability of membership values for data (Umano et al., 1994).

Let T be the training set where each object has m numerical values related to the attributes A_1, A_2, ..., A_m, and where the set of classes is defined by $C = \{C_1, C_2, ..., C_n\}$. Assume the fuzzy sets $F_{k1}, F_{k2}, ..., F_{kf}$ for the attribute A_k, and where the value of f varies on every attribute. Let T^{C_i} be a fuzzy subset of T whose class is C_i and $|T|$ is the sum of the membership values in a fuzzy set of data T.

The steps of the algorithm are presented as follows (Umano et al., 1994):

1. Generate the root node including all the objects belonging to the training set (a fuzzy set of all data with the membership value 1).

2. If a node (with the objects) satisfies one of the following conditions :

   (a) the proportion of objects belonging to a class C_i is defined as follows: $\frac{|T^{C_i}|}{|T|} \geq \theta_r$ where θ_r is a threshold.

   (b) The number of objects is the training set is such that $|T| < \theta_n$ where θ_n is a threshold.

   (c) There are no attributes for more classification.

   Hence, the node is declared as a leaf.

3. If one of the above conditions is not satisfied, it is a test node defined as follows:

   (a) For each attribute A_k, compute the information gain $Gain(T, A_k)$, then choose the attribute presenting the highest information gain attribute A_{max}.

   (b) Divide T into fuzzy subsets T_1, T_2, ..., T_f by taking into account the A_{max}'s values. The membership degrees in each T_t are equal to the product of the membership value in T and the value of $F_{max,t}$ of the value of A_{max} in T.

   (c) Generate f new nodes relative to the fuzzy subsets T_1, T_2, ..., T_f where the edges are labeled by the different $F_{max,t}$.

   (d) Replace T by T_t $(t = 1, 2, ..., f)$ and repeat from 2 recursively.

The information gain $Gain(T, A_k)$ for the attribute A_k by a fuzzy set of data T is defined by:

$$Gain(T, A_k) = Info(T) - Info_{A_k}(T) \tag{2.11}$$

$$\text{where } Info(T) = -\sum_{i=1}^{n} p_i log_2 p_i \tag{2.12}$$

$$Info_{A_k}(T) = \sum_{t=1}^{f} p_{kt} Info(T_{F_{kt}}) \tag{2.13}$$

$$p_i = \frac{|T^{C_i}|}{|T|} \tag{2.14}$$

$$p_{kt} = \frac{|T_{Fkt}|}{\sum_{t=1}^{f} |T_{F_{kt}}|} \tag{2.15}$$

To assign a class name to a leaf, two methods are proposed by (Umano et al., 1994):

1. The object is assigned to the class name having the highest membership value (the other classes are ignored).
2. The object is assigned to all class names with their membership values.

Classification procedure: In order to classify a new instance, Umano et al. (1994) have suggested to calculate the membership values of each leaf by taking into account the attributes' values of the given object. The idea consists in the aggregation of membership values on each path by applying the multiplication. Then to get the total membership value of the path attached to each leaf, we also adopt the multiplication. Finally, for the aggregation of the different values for the same class, we adopt the addition. That's why this technique is often referred to as *'* * + method'*.

The Zeidler's algorithm: Other researchers such as Zeidler and Schlosser (1996) have suggested a fuzzy decision tree algorithm derived from the one developed by Umano et al. (Umano et al., 1994).

The idea of Zeidler and Schlosser lies on developing a method for handling continuous-valued attributes with automatically generated membership functions. This contribution is achieved by using the so-called **cut points** allowing to compute the information gain of each attribute.

Remark:

Other algorithms for constructing fuzzy decision trees have been proposed, we can especially mention the one suggested by Marsala (1998) and in which he basically proposes another selection measure.

2.9 Conclusion

Decision trees are considered as one of the best classification techniques especially in artificial intelligence applications. It is now applied to many new computer science applications, notably those related to knowledge discovery such as expert systems, data mining...

Despite the advantages provided by decision trees and the improvements given as probabilistic and fuzzy decision trees, many researches are still needed in order to deal with the uncertainty especially the cognitive one, that may occur in the different parameters related to any classification problem.

The belief function theory as understood in the Transferable Belief Model (TBM) seems to be one of the appropriate formalism to cope with this kind of uncertainty. Thus, our objective will be to develop what we call **a belief decision tree approach** that will be presented in the following part of this thesis.

Part II

Belief Decision Tree

In this second part, considered as the major part of our thesis, we detail the developments that we have proposed in order to build belief decision trees, then to use them for classifying new objects.

This part is composed of four chapters:

- Chapter 3 presents the definition of a belief decision tree, its objectives and its representation. It also explains the nature of the uncertainty encountered in the data of the training set.

- Chapter 4 details the parameters that will be used to construct a belief decision tree. These parameters are the attribute selection measure, the partitioning strategy, the stopping criteria and the leaf structure, all of which have to be adapted to the uncertain context.

- Chapter 5 deals with the construction procedure of the belief decision tree and its use for the classification of new instances. For this latter procedure, several cases are treated. They depend on the values of the attributes of the instances to classify which may be certain, disjunctive or even uncertain.

- Chapter 6 presents implementation and simulation issues. The major algorithms are detailed. Simulations on data sets are performed in order to judge the feasibility of our algorithms.

Chapter 3

Presentation

3.1 Introduction

Decision trees are considered as an efficient machine learning method for expressing classification knowledge and using it. That's why, they are widely applied to a variety of fields notably in artificial intelligence.

Despite their accuracy and efficiency when precise and certain data are available, the standard decision tree algorithms show serious limitations when dealing with uncertainty. Such uncertainty may affect the parameters of any classification problem and can appear either in the construction or in the classification phase.

Faced to uncertain parameters, the standard decision trees seem to be insufficient to provide significant classification results. In fact, their results are categorical and do not convey the uncertainty that may occur in the attribute values or in the objects' classes.

To overcome this limitation, we propose to develop what we call **a belief decision tree**, a new classification technique based on the decision tree within the belief function theory in order to deal with uncertainty that may pervade any classification problem.

This theory for uncertainty representation, as understood in the Transferable Belief Model (TBM), provides a convenient framework for managing and manipulating uncertain knowledge, especially the cognitive uncertainty.

In this chapter, we present the definition of a belief decision tree. Then, we define the structure of the training set which will be illustrated by an example. Finally, the belief decision tree objectives and representation will be described.

3.2 Definition

A belief decision tree is a decision tree in an uncertain environment. The uncertainty will be represented and handled by the means of the belief function theory as explained in the Transferable Belief Model (TBM).

Contrary to a classical decision tree where objects' classes and attribute values are known with certainty, in a belief decision tree these two fundamental parameters may be uncertain. Such uncertainty can appear either in the construction or the classification phase.

3.3 Structure of the training set

3.3.1 Definition

Any decision tree is constructed from a training set of objects using successive refinements. This set is the basis leading to the induction of the tree and consequently to the classification of new instances in the inference phase.

This training set is generally composed of elements (objects) represented as pairs (attributes, class) where for each object, we know exactly the value of each one of its attributes and also its assigned class which is unique.

However, due to the uncertainty introduced here, the structure of the training set may be different from the traditional one.

Unlike the standard training set, we assume it may contain data where there is some uncertainty in the knowledge of the classes. In other words, each class of the training instances may be uncertain or even unknown, whereas the values of the attributes characterizing each training instance are supposed to be known with certainty.

We propose to represent the uncertainty on the classes of the training instances by a basic belief assignment (bba) defined on the set of possible classes considered in the classification problem. This bba, generally given by an expert (or several experts), represents the opinions-beliefs of this expert about the actual value of the class for each object in the training set.

Note that all over this thesis, we only deal with symbolic attributes.

3.3.2 Notations and assumptions

In this thesis, we use the following notations:

- T: a given training set,
- S: a given set of objects,
- I_j: an instance named also an object, or a case, or an example,
- $A = \{A_1, A_2 ... A_m\}$: a set of m attributes,
- $D(A_k)$: the domain of the attribute $A_k \in A$,
- $A_k(I_j)$: the value of the attribute A_k for the object I_j,
- $S_v^{A_k} = \{I_j : I_j \in S \text{ and } A_k(I_j) = v\}$: the subset of objects belonging to S and for which the value of the attribute $A_k \in A$ is $v \in D(A_k)$,
- $\Theta = \{C_1, C_2, ..., C_n\}$: the frame of discernment involving all the possible classes related to the classification problem. The classes C_i's are assumed to be mutually exclusive and exhaustive,
- C(I_j): the actual class of the object I_j,
- $m_g^{\Theta}\{I_j\}[B](C)$: the conditional bbm given to $C \subseteq \Theta$ relative to an object I_j given by an agent g that accepts that the information B is true. If the corresponding bba is discounted by a discounting factor α, the notation becomes $m_g^{\Theta,\alpha}\{I_j\}[B](C)$.

  Useless indices are omitted, generally when the missing elements are clearly defined from the context.

Example 3.1 *This is a simple example, presented in the previous chapter, (see Example 2.1) to illustrate our 'new' structure of the training set T within the belief function framework.*

We assume there are eight objects (clients) I_j ($j \in \{1,2,3,4,5,6,7,8\}$) that may belong to one of the three possible classes C_i ($i \in \{1,2,3\}$) related to the given classification problem.

Assume a bank wants to classify its clients by taking into account a number of their attributes. This classification may help the bank to plan its client policy loan.

The training set instances are characterized by three symbolic attributes defined as follows:

- *Income with possible values $\{No, Low, Average, High\}$,*
- *Property with possible values $\{Less, Greater\}$,*
- *Unpaid_credit with possible values $\{Yes, No\}$.*

Three classes may be assigned to clients ($\Theta = \{C_1, C_2, C_3\}$):

- *C_1 including good clients, i.e., reliable clients, for whom the bank accepts to give the whole loan.*
- *C_2 to which belongs moderate clients, for whom the bank accepts to give a part of the loan.*
- *C_3 regrouping 'bad' clients for whom the bank refuses to give the loan.*

For each client I_j belonging to the training set T, we assign a bba $m^{\Theta}\{I_j\}$ expressing beliefs on its assigned classes. These functions are defined on the same frame of discernment $\Theta = \{C_1, C_2, C_3\}$.

The structure of the training set T can be defined as follows (see Table 3.1):

Table 3.1: Training set T relative to a belief decision tree

Income	Property	Unpaid_credit	Class
High	Greater	Yes	$m^{\Theta}\{I_1\}$
Average	Less	No	$m^{\Theta}\{I_2\}$
High	Greater	Yes	$m^{\Theta}\{I_3\}$
Average	Greater	Yes	$m^{\Theta}\{I_4\}$
Low	Less	Yes	$m^{\Theta}\{I_5\}$
No	Less	No	$m^{\Theta}\{I_6\}$
High	Greater	No	$m^{\Theta}\{I_7\}$
Average	Less	Yes	$m^{\Theta}\{I_8\}$

where

$m^{\Theta}\{I_1\}(C_1)$=0.7; $m^{\Theta}\{I_1\}(\Theta)$ =0.3;
$m^{\Theta}\{I_2\}(C_2)$=0.5; $m^{\Theta}\{I_2\}(C_1 \cup C_2)$=0.4; $m^{\Theta}\{I_2\}(\Theta)$=0.1;
$m^{\Theta}\{I_3\}(C_1)$=0.6; $m^{\Theta}\{I_3\}(\Theta)$ =0.4;
$m^{\Theta}\{I_4\}(C_2)$=0.6; $m^{\Theta}\{I_4\}(C_3)$ =0.3; $m^{\Theta}\{I_4\}(\Theta)$=0.1;
$m^{\Theta}\{I_5\}(C_3)$=0.7; $m^{\Theta}\{I_5\}(C_2 \cup C_3)$=0.2; $m^{\Theta}\{I_5\}(\Theta)$=0.1;
$m^{\Theta}\{I_6\}(C_3)$=0.95; $m^{\Theta}\{I_6\}(\Theta)$ =0.05;
$m^{\Theta}\{I_7\}(C_1)$=0.95; $m^{\Theta}\{I_7\}(\Theta)$ =0.05;
$m^{\Theta}\{I_8\}(C_2)$=0.4; $m^{\Theta}\{I_8\}(C_3)$ =0.4; $m^{\Theta}\{I_8\}(\Theta)$=0.2;

For instance, for the client I_1, 0.7 of beliefs are exactly committed to the class C_1 showing that when the belief is 0.7 then the client I_1 is a good client (for whom the bank accepts to give the whole loan) , whereas 0.3 is assigned to the disjunction of classes $C_1 \cup C_2 \cup C_3$, i.e., 0.3 is assigned to the whole frame of discernment (ignorance).

3.3.3 Special cases

Among the advantages of working under the belief function framework, we notice that the two extreme cases namely the total knowledge and the total ignorance regarding training instance classes, can be easily expressed:

- When the class of the object I_j is perfectly known and is unique, it will be represented by a certain basic belief assignment (see Section 1.3.4)

having as the only focal element this class. Therefore, if the class of the object is C_i then its corresponding bba is defined as:

$$\begin{cases} m^{\Theta}\{I_j\}(C_i) = 1 \text{ for some } C_i \subseteq \Theta, C_i \neq \Theta, |C_i| = 1 \\ \text{and } m^{\Theta}\{I_j\}(C) = 0 \text{ for all } C \neq C_i, C \subseteq \Theta \end{cases} \tag{3.1}$$

Such case is referred to as **total knowledge** and corresponds to the classical '*certain*' context. Hence, our representation is also appropriate to describe the standard case, handled by Quinlan (1986, 1993), where all the classes of the training instances are known with certainty.

- When no information concerning the class of the object I_j is available which means that the expert is not able to give any judgment about the instance's classes. Thus, the bba will be a vacuous basic belief assignment (see Section 1.3.2) defined by:

$$\begin{cases} m^{\Theta}\{I_j\}(\Theta) = 1 \\ m^{\Theta}\{I_j\}(C) = 0 \text{ for } C \subset \Theta \end{cases} \tag{3.2}$$

Such case is referred to as **total ignorance**.

We have to mention that this case is not generally taken into account (ignored) in a training set since it does not provide any information regarding the instances' classes used to ensure the learning and the construction of the tree.

In addition to these two special cases, the case of disjunctive classes may be easily described by the so-called **categorical belief function** (see Section 1.3.3). In practice, this latter case often happens and corresponds to a disjunction of classes that will be assigned to a training instance by an expert.

Such opinion will be represented by a bba characterized by only one focal element representing the union of classes assigned to this training instance I_j and expressed as follows:

$$\begin{cases} m^{\Theta}\{I_j\}(C_i) = 1 \text{ for some } C_i \subseteq \Theta, C_i \neq \Theta, |C_i| > 1 \\ \text{and } m^{\Theta}\{I_j\}(C) = 0 \text{ for } C \subseteq \Theta, C \neq C_i \end{cases} \tag{3.3}$$

3.4 Objectives

In addition to the objectives of a standard decision tree (see Section 2.4), a belief decision tree aims at realizing two major objectives:

1. Building a decision tree from a given set of training instances characterized by uncertainty in their classes. In other words, ensuring the induction of the belief decision tree.

2. Ensuring the classification of new instances that may be described by uncertain or even unknown attribute values. Such procedure is also called the inference procedure.

3.5 Belief decision tree representation

Once the structure of the training set is defined, the representation of our belief decision tree is composed by the same elements as in the traditional decision tree:

- **Decision nodes** for testing attributes.
- **Branches** for specifying attribute values.
- **Leaves** dealing with classes of the training instances.

Due to the uncertainty related to training instances' classes, the structure of leaves will change. Instead of assigning a unique class to each leaf, it will be labeled by a bba expressing a belief on the actual class of objects belonging to the leaf. The computation of each leaf's bba will be shown in the next chapter.

3.6 Conclusion

In this presentation chapter, we have defined the belief decision tree approach as a new technique associating the decision tree technique with the belief function theory. Hence, we have detailed the characteristics of this new approach namely its definition, the structure of the training set, its objectives and also its representation.

The objective of the next chapter is to define the basic parameters making up the belief decision tree. We basically mention the attribute selection measure, the partitioning strategy, the stopping criteria and also the structure of leaves.

Chapter 4

Belief decision tree parameters

4.1 Introduction

Ensuring the construction of a belief decision tree and consequently the classification of new instances require the definition of fundamental parameters. These parameters have to cope with the uncertainty held in the training set.

As defined in the previous chapter, the structure of the training set is characterized by an uncertainty affecting the classes of the training instances, whereas the values of their attributes are known with certainty.

Hence, the attribute selection measure, the partitioning strategy, the stopping criteria and even the structure of leaves have to take into account the uncertainty expressed by the means of belief functions.

In the first part of this chapter, we develop the attribute selection measure in this uncertain context using the belief function framework. In fact, two approaches will be proposed: the first one is an averaging approach, considered as the extension of Quinlan approach to an uncertain framework, the second is a conjunctive approach based on the distance between training objects.

Next, we focus on the definition of the partitioning strategy, the stopping criteria and also the structure of leaves that will be used in the construction of a belief decision tree.

The major results of this chapter are developed in (Elouedi, Mellouli, & Smets, 2000a, 2000b, 2001a, 2001b).

4.2 The averaging and the conjunctive approaches for building belief decision trees

The training set in a belief decision tree is characterized by the fact that our knowledge about the value of the actual class of its instances is represented by a bba on Θ, the set of possible classes. The classical algorithms must be adapted to cope with such a context poisoned with uncertainty.

Before describing our algorithms in details (in the next chapter), we explain how we derive two of them. Of course, these are many possible algorithms, but the interest of the chosen ones comes from their close link to:

- the classical approach developed by Quinlan (1986, 1993) for our so-called **averaging approach**,
- the ideas behind the TBM itself (Smets, 1988, 1998b; Smets & Kennes, 1994; Smets & Kruse, 1997) for our so-called **conjunctive approach**.

4.2.1 The averaging approach

Suppose the decision tree is built, and consider one leaf, denoted S, and suppose 80% of the objects in S are C_1 objects. Why does this influence our knowledge about the class to which belongs a new instance that falls in the same leaf S ? There are (at least) two possible answers.

The sampling answer

We implicitly consider that the new object is an instance selected at random from the same population as the one from which the instances in S were selected. We assume the observed frequencies in S are *'good estimators'* of the distribution of the classes in the population. Then, the proportion 80% is equated (may not be in a very rigorous way) to the probability that an object randomly selected in the population represented by S is C_1, i.e., belongs to class C_1. We observe a new instance, and we can say that the probability that the new instance is a C_1, is 0.80. Therefore, the 80% proportion of C_1 in S becomes finally the probability that the new instance that falls in S is a C_1.

The largest the dominant proportion in S, the most confident we will feel in our claim about the new object's class. So, we would like that most objects in a leaf belong to the same class. If this ideal is not achievable, we would like to be able to assert with confidence that the new object's class is one among a few of the possible classes, and that many of the possible classes can be excluded. The worst case for a leaf is encountered when every class is equally represented, as assigning then a class to a new object falling in that leaf would be unjustified and unsupported.

The heterogeneity of the probabilities is what entropy of Equation (2.2) is supposed to quantify. The entropy is maximal when the classes are equi-probable, and becomes smaller when the distribution of the classes becomes further and further away from the equi-probability. The smallest entropy (of value 0) is reached when the probability is 1 for one class, and 0 for all the other classes.

Quinlan's algorithm is based on that idea, and thus tries to minimize the entropy at the leaf's level. Unfortunately, the justification for using the proportions instead of the probabilities can seriously be criticized. It would be acceptable if the number of objects in S was really large, but in practice this is not the case. Leaves with one element are even considered. So, the suggested justification is hard to defend, and we feel the second one is more appropriate.

The finite population answer

Suppose the data of S correspond to a population, not to a sample selected from a larger population. We assume one object is selected at random, with equi-probability, i.e., with probability $1/|S|$, from S, and that the new object is a duplicate of this selected instance, so it is exactly equal to the selected one. If we knew which object had been selected, the class of the new object would be the class of the selected one, but we do not know which one in S was actually selected. All we can say is that the probability that the new object's class is C_i, is equal to the probability that the selected object is a C_i, and this probability is equal, thanks to the equi-probable sampling method, to the proportions of C_i objects in S.

Entropy becomes then perfectly meaningful for the same reasons as given in the previous analysis.

Table 4.1: Training subset S (Standard case)

I_j	p_j	C_1	C_2	C_3
I_1	0.2	1	0	0
I_2	0.2	1	0	0
I_3	0.2	0	1	0
I_4	0.2	0	1	0
I_5	0.2	0	0	1
Mean		2/5	2/5	1/5

For instance, let Table 4.1 represents the five objects in the subset S of the training set. The class of each object is defined by the indicator function. Instances I_1 and I_2 are C_1's, etc ... If each instance has a probability $p_j = 0.2$ of being selected, then the probability of selecting a C_1 object is just the sum of the indicators weighted by the 0.2 probabilities. In fact, the indicator can be understood as the probability that the selected object is a C_i object given the selected object is the instance I_j. So, the probability that the selected object is C_i becomes:

$$P(C_i) = \sum_{I_j \in S} P(C_i| \text{ selected object is } I_j)\ p_j \tag{4.1}$$

Entropy could then be computed from these *'expected values'*.

The Equation (4.1) allows us to shift directly to the case where the classes are uncertain, and *the uncertainty is represented by a probability measure*. Table 4.2 presents the kind of data that could be collected from the five objects in S. Here object I_1 has a high probability of being a C_1, but might also be a C_2 or a C_3, even though these last two options are individually less probable than the first. The expected values are computed as in the previous cases, and the probability that the randomly selected object is a C_1 is 0.2 in this object. Entropy could then be computed from these *'expected values'*.

What about the entropy in this context? We would like that there would be as few ambiguity as possible when we classify a new object falling in a leaf. So, we would like that the probability in a leaf points essentially to one class, and entropy is an excellent measure to quantify this tendency. Hence, the use of the entropy computed from the average probability function in a leaf is plainly justified.

Table 4.2: Training subset S (Probabilistic case)

I_j	p_j	C_1	C_2	C_3
I_1	0.2	0.5	0.2	0.3
I_2	0.2	0.4	0.6	0
I_3	0.2	0	1	0
I_4	0.2	0.1	0.7	0.2
I_5	0.2	0	0.5	0.5
Mean		1/5	3/5	1/5

Suppose now *the uncertainty is represented by a bba*, with $m^{\Theta}\{I_j\}$ the bba about the actual class to which the instance I_j belongs (Θ is the set of the possible classes). If one asks what is the bba of the class to which belongs the randomly selected object, the answer happens to be the average of the $m^{\Theta}\{I_j\}$'s. This results from the fact that the bba's $m^{\Theta}\{I_j\}$ are just conditional bba's and if an object is selected according to a probability distribution, then the resulting bba, denoted $\overline{m}$, is :

$$\overline{m} = \sum_{I_j \in S} m^{\Theta}\{I_j\}\ p_j$$

This formula results from the conjunctive combination of the probability function p_j and the conditional bba's $m^{\Theta}\{I_j\}$'s, followed by a marginalization of Θ. It uses the relation (Smets, 1993a): $m_1 \textcircled{\cap}_2(A) = \sum_{B \subseteq \Omega} m_1[B](A)\, m_2(B)$, where m_1 and m_2 are defined on Ω. The p_j's are the m_2, and the $m^{\Theta}\{I_j\}$'s are the $m_1[B]$ where $B = \{(I_j, C_i) : C_i \in \Theta\}$, $\Omega = S \times \Theta$, and $A \subseteq \Theta$.

The probability used to compute the entropy is now replaced by the pignistic probability computed from $\overline{m}$. It just happens that the pignistic probability computed form $\overline{m}$ is the same as the average of the pignistic probabilities computed for each object:

$$BetP = \Gamma(\sum_{I_j \in S} m^{\Theta}\{I_j\}\ p_j) = \sum_{I_j \in S} \Gamma(m^{\Theta}\{I_j\})\ p_j$$

where Γ is the operator that transforms a bba into a pignistic probability function. This linearity property is even the major property that justifies the use of the pignistic probabilities (Smets, 1998b).

In consequence, when uncertainty is represented by bba's, it is enough to compute the pignistic probability over Θ, the set of possible classes, for each case, and proceeds as done in Table 4.2. The use of the entropy at the leaf level is justified just as in the probabilistic case.

Using bba's instead of probabilities on the classes is really not a real issue. One may then raise the question: why to use the TBM in such a case?

The answer is to be found in dynamic contexts where the beliefs about the classes for each individual can vary with time. New pieces of information could be collected about the data in the training set, in which case the bba's will be adapted by applying the appropriate Dempster's rules, and the change in the pignistic probabilities will not be those one obtained if the impact of the new information was handled within the probability model. So using the TBM, even though not essential when the training set is fixed, becomes interesting when our knowledge about the classes of the data in the training set can vary.

Besides, suppose the value of a needed attribute of a new object is itself uncertain, like the value being v_1 or v_2. As it will be explained in Section 5.3, we will compute the bba m_1 from the data in the leaf reached if v_1 was the object, and the bba m_2 from the data in the leaf reached if v_2 was the object. The combination of these two bba's is obtained by the disjunctive rule of combination, something that brings us far away from the pignistic probabilities, and requires the whole TBM apparatus.

So even though in simple cases, the need for the TBM was not essential, it becomes really so in more complicated contexts.

4.2.2 The conjunctive approach

The second method we considered is conceptually much closer to the TBM itself. Let us first reconsider what can be done at the leaf's level. In a given leaf S, we suppose we have several objects and ideally they all belong to the same class. So, every instance in S belongs to the same class, but we don't know which one. Each instance provides a bba $m^{\Theta}\{I_j\}$, $j = 1, \ldots, |S|$, that represents what is known from the instance I_j about the class to which it belongs, hence to the class that characterizes S. The belief we can build on the class *'common'* to those who belong to S is obtained by combining the $m^{\Theta}\{I_j\}$, for $I_j \in S$, by the conjunctive combination rule. This idea is based on similar approaches developed by Denoeux (Denœux, 1995), (Smets, 1999).

So, if $m^{\Theta}\{I_j\}$ is the bba of an object I_j in the considered leaf S, we compute:

$$m^{\Theta}\{S\} = \textcircled{\cap}_{I_j \in S} m^{\Theta}\{I_j\}$$

The bba $m^{\Theta}\{S\}$ is what all the objects in a leaf S jointly express about the class of those objects that belong to the leaf S.

So, the classification of a new instance that falls in a leaf S is based on this joint bba $m^{\Theta}\{S\}$, and the class is decided using the pignistic probabilities computed from this bba $m^{\Theta}\{S\}$.

Knowing what will be done once the tree is built, let us now shift to its construction. What *'nice'* property should be satisfied by the instances in a leaf. Ideally, they should belong to the same class, but their actual classes are unknown. Suppose then two objects with the same bba on Θ, the set of possible classes. It seems reasonable to be satisfied if both objects fall in the same leaf.

So, what we would like is that all objects in a leaf have bba's that are *'close'* to each others. Thus, a distance between bba's, and in particular between two bba's, is required.

Distance between bba's

Let $m^{\Theta}\{I_j\}$ and $m^{\Theta}\{I_w\}$ be two bba's, both defined on Θ. These two bba's are vectors in a $2^{|\Theta|}$ dimension space. A natural distance is the euclidian distance between the two vectors. But why to use the bba's themselves, and not any vector that is in one to one correspondence with the bba, like the *bel* vector, or the q vector... So, let f_j be such a vector where f_j is a function of $m^{\Theta}\{I_j\}$ which value at $X \subseteq \Theta$ is denoted $f_j(X)$. We are going to show that $f_j(X) = -ln(q^{\Theta}\{I_j\}(X))$ is an appropriate choice. We define the distance between two instances I_j and I_w belonging to S as:

$$d_{jw}^2 = \sum_{X \subseteq \Theta} (f_j(X) - f_w(X))^2$$

We can then define the distance among the instance within one group S as the average of the distance between pairs of instances in S:

$$D_S^2 = \frac{1}{2s^2} \sum_{I_j, I_w \in S} d_{jw}^2$$

where $s = |S|$

Ideally, this distance should be minimized if the goal is that objects in S are similar to each others. This *'intra-group'* distance (because computed within one group), has the advantage that minimizing it is equivalent to maximizing the *'inter-groups'* distances (the one computed between groups), another criterion that could have been advocated.

The intra-group distance can be shown to be equal to:

$$\begin{aligned}
D_S^2 &= \frac{1}{2s^2} \sum_{I_j, I_w \in S} \sum_X (f_j(X) - f_w(X))^2 \\
&= \frac{1}{2s^2} \sum_X \sum_{I_j, I_w \in S} (f_j(X)^2 - 2 f_j(X) f_w(X) + f_w(X)^2) \\
&= \frac{1}{2s^2} \sum_X (2s \sum_{I_j \in S} f_j(X)^2 - 2(\sum_{I_j \in S} f_j(X))^2) \\
&= \frac{1}{s} \sum_X (\sum_{I_j \in S} f_j(X)^2 - \frac{1}{s} (\sum_{I_j \in S} f_j(X))^2 \ ) \\
&= \sum_X \frac{1}{s} \sum_{I_j \in S} (\ f_j(X) - \overline{f(X)}\ )^2 \\
&\propto \sum_X \text{variance}(f_j(X))
\end{aligned}$$

The distance D_S^2 depends thus of $\overline{f(X)}$. This average $\overline{f}$ can be seen as the function of a bba $\overline{m}$ that has the following property: when *'added'* s times, the result has the same *'weight'* as the *'addition'* of the s bba's $m^\Theta\{I_j\}$. As far as within one group, all bba's will be summarized by the result of their combination by the conjunctive rule of combination, the *'addition'* operation can be seen as an application of the conjunctive combination.

So, we want:

$$\textcircled{\cap}_{j=1,\ldots,s} \overline{m} = \textcircled{\cap}_{j=1,\ldots,s}\ m^\Theta\{I_j\}$$

Thanks to the property of the commonality functions, the conjunctive combination rule can be written as a product, and even better, the logarithm of the commonality function of the combination is the sum of the logarithms of the commonality functions entered into the conjunctive combination.

Let κ be defined as minus the logarithm (basis e as the choice of the basis is arbitrary) of q, so let $\kappa^{\Theta}\{I_j\}(X) = -\ln(q^{\Theta}\{I_j\}(X))$ for $X \subseteq \Theta$. Then, we define:

$$\kappa^{\Theta}\{S\} = \textcircled{\cap}_{j=1,2,\ldots,s}\ \kappa^{\Theta}\{I_j\}$$

So, we get:

$$\kappa^{\Theta}\{S\} = \sum_{j=1,2,\ldots,s} \kappa^{\Theta}\{I_j\}$$

If we take $f(m^{\Theta}\{I_j\}) = \kappa^{\Theta}\{I_j\}$, and $\overline{\kappa} = \frac{1}{s}\kappa^{\Theta}\{S\}$, we have a function f that satisfies the idea that the impact of the s instances in the group under consideration is equal to the impact of s times the *'average'* case.

Beware that usually $\overline{\kappa}$ is not the logarithm of a commonality function, even though $\textcircled{\cap}_{j=1\ldots s}\kappa^{\Theta}\{I_j\}$ is the logarithm of a commonality function.

In fact, the $\textcircled{\cap}$ operator can innocuously be extended to *'generalized belief functions'*, i.e., any real function on Θ which coefficients of its Möbius transform (the equivalent of the basic belief masses) add to 1.

So, our proposed intra-group distance of instances $j = 1, \ldots, s$ becomes:

$$D_S^2 = \frac{1}{|S|} \sum_{X \subseteq \Theta} \sum_{I_j \in S} (\ \kappa^{\Theta}\{I_j\}(X) - \overline{\kappa(X)}\ )^2 \tag{4.2}$$

where $\kappa^{\Theta}\{I_j\}(X) = -\ln q^{\Theta}\{I_j\}(X)$ and $\overline{\kappa(X)} = \frac{1}{|S|}\sum_{I_j \in S} \kappa^{\Theta}\{I_j\}(X)$.

The case where $q^{\Theta}\{I_j\}(\Theta) = 0$ is solved, thanks to the continuity of all involved functions, by putting a very small mass ϵ on Θ, proceeding with the computation and taking the limit for $\epsilon \to 0$.

4.3 Belief decision tree parameters

In this section, we define the major parameters leading to the construction of the belief decision tree in both the averaging and the conjunctive approaches. At first, we describe what we have developed as attribute selection measures to ensure the construction of a belief decision tree. Then, we present the partitioning strategy and the stopping criteria. Finally, we detail the structure of leaves in belief decision trees.

4.3.1 Attribute selection measures in a belief decision tree

Introduction

One of the fundamental parameters in a decision tree (and consequently in a belief decision tree) is **the attribute selection measure**. This measure is applied in order to choose *'the best'* test attribute in each decision node of the tree. In fact, it enables us to quantify the power of discrimination of each attribute relatively to each class. It also allows optimizing the tree.

The attribute selection measure has to provide a better division of the training set into small subsets that are more homogeneous and leading to simpler representation and comprehension of the decision tree.

The structure of the training set is characterized by data for which the class is uncertain. This uncertainty is expressed by a bba on the classes' domain.

Averaging approach

Under this approach, the attribute selection measure is based on the entropy computed from the average pignistic probability computed from the pignistic probabilities of each instance in the node. We propose the following steps to choose the appropriate attribute:

1. Compute the pignistic probability of each instance I_j in a set of objects S by:

$$BetP^{\Theta}\{I_j\}(C_i) = \sum_{C_i \in C \subseteq \Theta} \frac{1}{|C|} \frac{m^{\Theta}\{I_j\}(C)}{1 - m^{\Theta}\{I_j\}(\emptyset)}, \ \forall C_i \in \Theta \qquad (4.3)$$

2. Compute the average pignistic probability function $BetP^{\Theta}\{S\}$ taken over the set of objects S in order to get the average probability on each class.

$$BetP^{\Theta}\{S\}(C_i) = \frac{1}{|S|} \sum_{I_j \in S} BetP^{\Theta}\{I_j\}(C_i) \tag{4.4}$$

3. Compute the entropy of the average pignistic probabilities in S. This $Info(S)$ value is equal to:

$$Info(S) = -\sum_{i=1}^{n} BetP^{\Theta}\{S\}(C_i) log_2 BetP^{\Theta}\{S\}(C_i) \tag{4.5}$$

4. Select an attribute A_k. Collect the subset $S_v^{A_k}$ made with the cases of S having v as a value for the attribute A_k.

5. Compute the average pignistic probability for those cases in the subset $S_v^{A_k}$. Let the result be denoted $BetP^{\Theta}\{S_v^{A_k}\}$ for $v \in D(A_k)$, $A_k \in A$.

6. Compute $Info_{A_k}(S)$ using the same definition as suggested by Quinlan (see Equation 2.3), but using the pignistic probabilities instead of the proportions. We get:

$$Info_{A_k}(S) = \sum_{v \in D(A_k)} \frac{|S_v^{A_k}|}{|S|} Info(S_v^{A_k}) \tag{4.6}$$

   where $Info(S_v^{A_k})$ is computed from the Equation (4.5) using $BetP^{\Theta}\{S_v^{A_k}\}$. The term $Info_{A_k}(S)$ is equal to the weighed sum of the different $Info(S_v^{A_k})$ relative to the considered attribute. These $Info(S_v^{A_k})$ are weighted by the proportion of objects in $S_v^{A_k}$.

7. Compute the information gain provided by the attribute A_k in the set of objects S such that:

$$Gain(S, A_k) = Info(S) - Info_{A_k}(S) \tag{4.7}$$

8. Using the Split Info (see Equation 2.5), compute the gain ratio relative to the attribute A_k:

$$Gain\ Ratio(S, A_k) = \frac{Gain(S, A_k)}{Split\ Info(S, A_k)} \tag{4.8}$$

9. Repeat for every attribute $A_k \in A$ and choose the one that maximizes the gain ratio.

Example 4.1 *Let's illustrate our attribute selection measure based on the average by a simple example already presented in the previous chapter (see Example 3.1). Let's remind the training set (see Table 4.3):*

Table 4.3: Training set T relative to a belief decision tree

Income	Property	Unpaid_credit	Class
High	Greater	Yes	$m^{\Theta}\{I_1\}$
Average	Less	No	$m^{\Theta}\{I_2\}$
High	Greater	Yes	$m^{\Theta}\{I_3\}$
Average	Greater	Yes	$m^{\Theta}\{I_4\}$
Low	Less	Yes	$m^{\Theta}\{I_5\}$
No	Less	No	$m^{\Theta}\{I_6\}$
High	Greater	No	$m^{\Theta}\{I_7\}$
Average	Less	Yes	$m^{\Theta}\{I_8\}$

where

$m^{\Theta}\{I_1\}(C_1)$=*0.7;* $m^{\Theta}\{I_1\}(\Theta)$ =*0.3;*
$m^{\Theta}\{I_2\}(C_2)$=*0.5;* $m^{\Theta}\{I_2\}(C_1 \cup C_2)$=*0.4;* $m^{\Theta}\{I_2\}(\Theta)$=*0.1;*
$m^{\Theta}\{I_3\}(C_1)$=*0.6;* $m^{\Theta}\{I_3\}(\Theta)$ =*0.4;*
$m^{\Theta}\{I_4\}(C_2)$=*0.6;* $m^{\Theta}\{I_4\}(C_3)$ =*0.3;* $m^{\Theta}\{I_4\}(\Theta)$=*0.1;*
$m^{\Theta}\{I_5\}(C_3)$=*0.7;* $m^{\Theta}\{I_5\}(C_2 \cup C_3)$=*0.2;* $m^{\Theta}\{I_5\}(\Theta)$=*0.1;*
$m^{\Theta}\{I_6\}(C_3)$=*0.95;* $m^{\Theta}\{I_6\}(\Theta)$ =*0.05;*
$m^{\Theta}\{I_7\}(C_1)$=*0.95;* $m^{\Theta}\{I_7\}(\Theta)$ =*0.05;*
$m^{\Theta}\{I_8\}(C_2)$=*0.4;* $m^{\Theta}\{I_8\}(C_3)$ =*0.4;* $m^{\Theta}\{I_8\}(\Theta)$=*0.2;*

We have, to compute the average pignistic probability relative the training set T (see Table 4.4):

Table 4.4: Computation of $BetP^{\Theta}\{T\}$

	C_1	C_2	C_3
$BetP^{\Theta}\{I_1\}$	0.8	0.1	0.1
$BetP^{\Theta}\{I_2\}$	0.23	0.73	0.04
$BetP^{\Theta}\{I_3\}$	0.74	0.13	0.13
$BetP^{\Theta}\{I_4\}$	0.04	0.63	0.33
$BetP^{\Theta}\{I_5\}$	0.04	0.13	0.83
$BetP^{\Theta}\{I_6\}$	0.02	0.02	0.96
$BetP^{\Theta}\{I_7\}$	0.96	0.02	0.02
$BetP^{\Theta}\{I_8\}$	0.06	0.47	0.47
Mean = $BetP^{\Theta}\{T\}$	0.36	0.28	0.36

The results induced from the pignistic transformation mean that the average probability that a training instance chosen randomly from T belongs respectively to the classes C_1, C_2, and C_3 are respectively 0.36, 0.28 and 0.36.

These probabilities will be used to compute $Info(T)$ described as the entropy relative to the whole training set T (see Equation 4.5):

$$\begin{aligned} Info(T) &= -\sum_{i=1}^{3} BetP(C_i) log_2 BetP(C_i) \\ &= -0.36 * log_2 0.36 - 0.28 * log_2 0.28 - 0.36 * log_2 0.36 \\ &= 1.575; \end{aligned}$$

The value 1.575 represents the average amount of information needed to identify the class of an instance in the training set T.

Once the $Info(T)$ is calculated, we have to look for the $Info_{Income}(T)$, $Info_{Property}(T)$ and $Info_{Unpaid\_credit}(T)$. These computations will be ensured by applying the Equation (4.6).

Let us do the computation for the income attribute. Let $BetP^{\Theta}\{T^{Income}_{High}\}$, $BetP^{\Theta}\{T^{Income}_{Average}\}$, $BetP^{\Theta}\{T^{Income}_{Low}\}$ and $BetP^{\Theta}\{T^{Income}_{No}\}$, be the average pignistic probability functions relative respectively to the clients belonging to T and having respectively high income, average income, low income and no income (see Table 4.5):

Table 4.5: Computation of $BetP^{\Theta}\{T^{Income}_{High}\}$, $BetP^{\Theta}\{T^{Income}_{Average}\}$, $BetP^{\Theta}\{T^{Income}_{Low}\}$ and $BetP^{\Theta}\{T^{Income}_{No}\}$

	C_1	C_2	C_3
$BetP^{\Theta}\{T^{Income}_{High}\}$	0.84	0.08	0.08
$BetP^{\Theta}\{T^{Income}_{Average}\}$	0.11	0.61	0.28
$BetP^{\Theta}\{T^{Income}_{Low}\}$	0.04	0.13	0.83
$BetP^{\Theta}\{T^{Income}_{No}\}$	0.02	0.02	0.96

The next step consists in the computation of $Info_{Income}(T)$. By applying the Equation (4.6), we get:

$$\begin{aligned} Info_{Income}(T) = & -\frac{3}{8}\sum_{i=1}^{3} BetP^{\Theta}\{T^{Income}_{High}\}(C_i) log_2 BetP^{\Theta}\{T^{Income}_{High}\}(C_i) \\ & - \frac{3}{8}\sum_{i=1}^{3} BetP^{\Theta}\{T^{Income}_{Average}\}(C_i) log_2 BetP^{\Theta}\{T^{Income}_{Average}\}(C_i) \\ & - \frac{1}{8}\sum_{i=1}^{3} BetP^{\Theta}\{T^{Income}_{Low}\}(C_i) log_2 BetP^{\Theta}\{T^{Income}_{Low}\}(C_i) \\ & - \frac{1}{8}\sum_{i=1}^{3} BetP^{\Theta}\{T^{Income}_{No}\}(C_i) log_2 BetP^{\Theta}\{T^{Income}_{No}\}(C_i) \\ = & \ 0.921; \end{aligned}$$

Then, we compute the information gain (see Equation 4.7), we get respectively:

$$\begin{aligned} Gain(T, Income) &= Info(T) - Info_{Income}(T) \\ &= 1.575 - 0.921 \\ &= 0.654; \\ Split\ Info(T, Income) &= 1.811; \\ Gain\ Ratio(T, Income) &= 0.361; \end{aligned}$$

By applying the same process, we get:

$$\begin{aligned} Gain\ Ratio(T, Property) &= 0.276; \\ Gain\ Ratio(T, Unpaid\_credit) &= 0.004; \end{aligned}$$

The attribute that maximizes the gain ratio is the income attribute. It will be chosen as the root of the belief decision tree relative to the training set T and a branch is created for each one of the possible values of the income attribute (High, Average, Low, No). The same procedure is applied, iteratively, to the instances that fall in each subset that is created according to the values of the selected attribute.

Remark:

The classical case where there is no uncertainty relative to the classes of the training instances is perfectly ensured by our attribute selection measure based on the averaging approach. Therefore, we get the same results as with the gain ratio of Quinlan.

Conjunctive approach

Our objective remains to develop an attribute selection measure able to handle the uncertainty in the training set. Indeed, our second approach that we call a conjunctive approach, consists in looking for a criterion that states for each attribute value how much objects are close from each others. Thus, our measure is based on the computation of distances between objects for each attribute.

The conjunctive approach uses the intra-group distance D_S^2 (see Equation 4.2) that quantifies for each attribute value how much objects are close from each others.

The attribute selection measure to build a belief decision tree under the conjunctive approach is made of the following steps:

1. For each instance I_j in the training set, compute :

$$\kappa^{\Theta}\{I_j\}(C) = -\ln q^{\Theta}\{I_j\}(C), \forall C \subseteq \Theta \tag{4.9}$$

   from the bba $m^{\Theta}\{I_j\}$.

2. For each attribute value v of the attribute A_k, compute the joint $\kappa^{\Theta}\{S_v^{A_k}\}$ defined on Θ by:

$$\kappa^{\Theta}\{S_v^{A_k}\} = \sum_{I_j \in S_v^{A_k}} \kappa^{\Theta}\{I_j\} \tag{4.10}$$

3. Hence for each attribute value, the intra-group distance $SumD(S_v^{A_k})$ is defined by:

$$SumD(S_v^{A_k}) = \frac{1}{|S_v^{A_k}|} \sum_{I_j \in S_v^{A_k}} \sum_{C \subseteq \Theta} (\kappa^{\Theta}\{I_j\}(C) - \frac{1}{|S_v^{A_k}|}\kappa^{\Theta}\{S_v^{A_k}\}(C))^2 \quad (4.11)$$

4. Once the different $SumD(S_v^{A_k})$ are calculated, for each attribute $A_k \in A$, compute $SumD_{A_k}(S)$ representing the weighted sum of the different $SumD(S_v^{A_k})$ relative to each value v of the attribute A_k:

$$SumD_{A_k}(S) = \sum_{v \in D(A_k)} \frac{|S_v^{A_k}|}{|S|} SumD(S_v^{A_k}) \quad (4.12)$$

5. At this level, we may conclude which attribute will be chosen as a root relative to the set of objects S. It consists in selecting the one presenting the minimal $SumD_{A_k}(S)$. In other words, the attribute presenting a partition of objects in which objects are the closest from each others. Nevertheless, we can proceed in order to take into account the number of possible values for the domain of the attribute.

6. By analogy to our averaging approach, we may also compute $Diff(S, A_k)$ defined as the difference between $SumD(S)$ and $SumD_{A_k}(S)$:

$$Diff(S, A_k) = SumD(S) - SumD_{A_k}(S) \quad (4.13)$$

where

$$SumD(S) = \frac{1}{|S|} \sum_{I_j \in S} \sum_{C \subseteq \Theta} (\kappa^{\Theta}\{I_j\}(C) - \frac{1}{|S|}\kappa^{\Theta}\{S\}(C))^2 \quad (4.14)$$

7. Using the Split Info (see Equation 2.5), compute the diff ratio relative to the attribute A_k:

$$Diff\ Ratio(S, A_k) = \frac{Diff(S, A_k)}{Split\ Info(S, A_k)} \quad (4.15)$$

8. Repeat for every attribute $A_k \in A$ and choose the one that maximizes the diff ratio.

Example 4.2 *Let's use the same training set presented in the Example 4.1 and let's try to apply our attribute selection measure based on this conjunctive approach in order to find the test attribute.*

We start by computing $SumD(T)$, the sum of distances separating each training instance to the whole set T. We have:

$$\begin{aligned} m\{T\} &= m^{\Theta}\{I_1\}\textcircled{\cap}m^{\Theta}\{I_2\}\textcircled{\cap}\ldots\textcircled{\cap}m^{\Theta}\{I_8\} \\ SumD(T) &= \frac{1}{8}\sum_{I_j\in T}\sum_{C\subseteq\Theta}(\kappa^{\Theta}\{I_j\}(C) - \frac{1}{8}\kappa^{\Theta}\{S\}(C))^2 \\ &= 13.437; \end{aligned}$$

We proceed with the computation for the income attribute. Four values may be possible which are high, average, low and no income. We define the following bba's:

$$\begin{aligned} m^{\Theta}\{T^{Income}_{High}\} &= m^{\Theta}\{I_1\}\textcircled{\cap}m^{\Theta}\{I_3\}\textcircled{\cap}m^{\Theta}\{I_7\}; \\ m^{\Theta}\{T^{Income}_{Avearge}\} &= m^{\Theta}\{I_2\}\textcircled{\cap}m^{\Theta}\{I_4\}\textcircled{\cap}m^{\Theta}\{I_8\}; \\ m^{\Theta}\{T^{Income}_{Low}\} &= m^{\Theta}\{I_5\} \\ and\ m^{\Theta}\{T^{Income}_{No}\} &= m^{\Theta}\{I_6\} \end{aligned}$$

$$\begin{aligned} SumD_{Income}(T) = &\frac{|T^{Income}_{High}|}{|T|}SumD_(T^{Income}_{High}) + \frac{|T^{Income}_{Average}|}{|T|}SumD_(T^{Income}_{Average}) + \\ &\frac{|T^{Income}_{Low}|}{|T|}SumD_(T^{Income}_{Low}) + \frac{|T^{Income}_{No}|}{|T|}SumD_(T^{Income}_{No}) \\ &= 2.782; \end{aligned}$$

By applying the same process to the other attributes, we get:

$$\begin{aligned} SumD_{Property}(T) &= 7.730; \\ SumD_{Unpaid\_credit}(T) &= 9.353; \end{aligned}$$

We have also to compute $Diff(T, A_k)$ for $A_k \in \{Income, Property, Unpaid\_credit\}$. We get:

$$\begin{aligned} Diff(T, Income) &= 10.655; \\ Diff(T, Property) &= 5.707; \\ Diff(T, Unpaid\_credit) &= 4.084; \end{aligned}$$

Then, the computation of the Diff Ratio gives the following results:

$$Diff\ Ratio(T, Income) = 5.882;$$
$$Diff\ Ratio(T, Property) = 5.707;$$
$$Diff\ Ratio(T, Unpaid\_credit) = 4.279;$$

The application of the 'Diff Ratio' criterion leads to the choice of the income attribute as the test attribute relative to the training set T.

Particular cases: Some particular cases may be considered in the conjunctive approach:

- *The dogmatic case* where training objects' classes are described through dogmatic bba's characterized by a null bbm for Θ. This case covers all the following cases: Bayesian, certain, and categorical, and the computation of the distances must be fixed in order to handle the infinity terms that appear.

- *The Bayesian case* where the knowledge about the training objects' classes are described through Bayesian bba's, i.e., probability functions.

- *The certain case* where training objects' classes are precisely known with certainty. The class of such objects is represented by the so-called certain bba assigning a value 1 to the class, and 0 to all the other subsets of Θ.

- *The categorical case* where training objects' classes are imprecise but certain, and represented by a disjunction of possible classes. The bba on the object's class is a categorical bba having a unique focal element different from the frame of discernment Θ.

Before detailing these different cases, let's define some notations. We use the symbol ∞ to denote limit $-\ln\epsilon$ when $\epsilon \to 0$. It means in particular that $\infty - \infty = 0$ as $-\ln\epsilon + \ln\epsilon = 0$ for all ϵ, hence the limit is also 0.

What do we compute?
$\kappa^{\Theta}\{I_j\}(C) = -\ln q^{\Theta}\{I_j\}(C)$ if $q^{\Theta}\{I_j\}(C) > 0$,
or $\kappa^{\Theta}\{I_j\}(C) = \infty$ if $q^{\Theta}\{I_j\}(C) = 0$

For an attribute A_k, we define:

- p as the number of instances in the considered set.
- r_C^{v,A_k} as the number of instances in the considered set with the value of the attribute A_k equals to v, and $\kappa^\Theta\{I_j\}(C)$ is finite.
- s_C^{v,A_k} as the number of instances in the considered set with the value of the attribute $A_k = v$, and $\kappa^\Theta\{I_j\}(C)$ is infinite.
- $p^{v,A_k} = r_C^{v,A_k} + s_C^{v,A_k}$ which is equal for every C.
- $\sum_v p^{v,A_k} = p$ for all A_k.

Let's now detail these different cases:

- *The dogmatic case:*

  For each attribute value v of the attribute A_k, compute $\kappa^\Theta\{S_v^{A_k}\}$ defined on Θ as follows:

$$\begin{aligned}\kappa^\Theta\{S_v^{A_k}\}(C) &= \sum_{I_j \in S_v^{A_k}} \kappa^\Theta\{I_j\}(C) \\ &= \sum_{I_j \in S_v^{A_k}, \kappa^\Theta\{I_j\}(C) \neq \infty} \kappa^\Theta\{I_j\}(C) + s_C^{v,A_k}\infty \\ &= t_C^{v,A_k} + s_C^{v,A_k}\infty\end{aligned}$$

  Hence, for each attribute value, the intra-group distance $SumD(S_v^{A_k})$ is defined as follows:

$$\begin{aligned}SumD(S_v^{A_k}) &= \frac{1}{|S_v^{A_k}|} \sum_{I_j \in S_v^{A_k}} \sum_{C \subseteq \Theta} (\kappa^\Theta\{I_j\}(C) - \frac{1}{|S_v^{A_k}|}\kappa^\Theta\{S_v^{A_k}\}(C))^2 \\ &= \frac{1}{p^{v,A_k}} \sum_{C \subseteq \Theta} \sum_{I_j \in S_v^{A_k}} (\kappa^\Theta\{I_j\}(C)^2 - \frac{1}{p^{v,A_k}}(t_C^{v,A_k} + s_C^{v,A_k}\infty)^2)\end{aligned}$$

So we need,

$$\begin{aligned}
T_C^{v,A_k} &= \sum_{I_j \in S_v^{A_k}} \kappa\{I_j\}(C)^2 - \frac{1}{p^{v,A_k}}(t_C^{v,A_k} + s_C^{v,A_k}\infty)^2 \\
&= tt_C^{v,A_k} + s_C^{v,A_k}\infty^2 - \frac{1}{p^{v,A_k}}(t_C^{v,A_k} + s_C^{v,A_k}\infty)^2 \\
&= tt_C^{v,A_k} + s_C^{v,A_k}\infty^2 - \frac{1}{p^{v,A_k}}((t_C^{v,A_k})^2 + 2t_C^{v,A_k}s_C^{v,A_k}\infty + (s_C^{v,A_k})^2\infty^2) \\
&= (s_C^{v,A_k} - \frac{(s_C^{v,A_k})^2}{p^{v,A_k}})\infty^2 + something\ smaller \\
&= s_C^{v,A_k}(1 - \frac{s_C^{v,A_k}}{p^{v,A_k}})\infty^2 + something\ smaller
\end{aligned}$$

Hence,

$$SumD(S_v^{A_k}) = \frac{1}{p^{v,A_k}} \sum_{C \subseteq \Theta} (s_C^{v,A_k}(1 - \frac{s_C^{v,A_k}}{p^{v,A_k}})\infty^2 + something\ smaller) \tag{4.16}$$

Once the different $SumD(S_v^{A_k})$ are calculated, for each attribute $A_k \in A$, compute $SumD_{A_k}(S)$ representing the weighted sum of the different $SumD(S_v^{A_k})$ relative to each value v of the attribute A_k:

$$\begin{aligned}
SumD_{A_k}(S) &= \sum_{v \in D(A_k)} \frac{|S_v^{A_k}|}{|S|} SumD(S_v^{A_k}) \\
&= \sum_{v \in D(A_k)} \sum_{C \subseteq \Theta} (s_C^{v,A_k}(1 - \frac{s_C^{v,A_k}}{p^{v,A_k}})\infty^2 + something\ smaller)
\end{aligned} \tag{4.17}$$

where we neglect the term $|S|$, that is not needed. The comparison (based on $SumD_{A_k}(S)$) between two attributes is thus performed on the terms:

$$\sum_{v \in D(A_k)} \sum_{C \subseteq \Theta} \left(s_C^{v,A_k}(1 - \frac{s_C^{v,A_k}}{p^{v,A_k}}) \right)$$

The largest the term the worst the attribute. This is not a nice property as such, but special case (the Bayesian belief function) is great provided every singleton receives a positive probability.

- *The Bayesian case:*

  In that case, $s_C^{v,A_k} = p^{v,A_k}$ for all $|C| \neq 1$, and $s_C^{v,A_k} = 0$ for all $|C| = 1$. It implies that the coefficients of the term in ∞^2 are always null. Note that $\kappa^\Theta\{I_j\}(\emptyset) = 0$.

  The term $\kappa^\Theta\{S_v^{A_k}\}$ becomes:

$$\kappa^\Theta\{S_v^{A_k}\}(C) = \sum_{I_j \in S_v^{A_k}} \kappa^\Theta\{I_j\}(C)$$

$$\kappa^\Theta\{S_v^{A_k}\}(C) = \begin{cases} \sum_{I_j \in S_v^{A_k}} \kappa^\Theta\{I_j\}(C) & \text{if } |C| = 1 \\ p^{v,A_k}\infty & \text{if } |C| > 1 \end{cases} \tag{4.18}$$

  The term $SumD(S_v^{A_k})$ becomes:

$$\begin{aligned} SumD(S_v^{A_k}) &= \frac{1}{|S_v^{A_k}|} \sum_{I_j \in S_v^{A_k}} \sum_{C \subseteq \Theta} (\kappa^\Theta\{I_j\}(C) - \frac{1}{|S_v^{A_k}|}\kappa^\Theta\{S_v^{A_k}\}(C))^2 \\ &= \frac{1}{p^{v,A_k}} \sum_{I_j \in S_v^{A_k}} \sum_{c \in \Theta} (\kappa^\Theta\{I_j\}(c) - \frac{1}{p^{v,A_k}}\kappa^\Theta\{S_v^{A_k}\}(c))^2 \\ &\quad + \frac{1}{p^{v,A_k}} \sum_{I_j \in S_v^{A_k}} (2^{|\Theta|} - |\Theta| - 1)(\infty - \infty)^2 \\ &= \frac{1}{p^{v,A_k}} \sum_{c \in \Theta} \sum_{I_j \in S_v^A} (\kappa^\Theta\{I_j\}(c) - \frac{1}{p^{v,A_k}}\kappa^\Theta\{S_v^{A_k}\}(c))^2 \\ &= \frac{1}{p^{v,A_k}} \sum_{c \in \Theta} variance_{S_v^{A_k}}(\kappa^\Theta\{I_j\}(c)) \end{aligned} \tag{4.19}$$

  Finally,

$$\begin{aligned} SumD_{A_k}(S) &= \sum_{v \in D(A_k)} \frac{|S_v^{A_k}|}{|S|} SumD(S_v^{A_k}) \\ &= \frac{1}{|S|} \sum_{v \in D(A_k)} \sum_{c \in \Theta} variance_{S_v^{A_k}}(\kappa^\Theta\{I_j\}(c)) \end{aligned} \tag{4.20}$$

  Therefore, the dogmaticity of the Bayesian belief function is not a problem anymore. The criterion is then based on the variance of the $-\log(p\{I_j\}(c))$ as on the singleton commonalities equal plausibilities and for a Bayesian belief function, plausibilities equal probabilities.

- *The certain case:*

  Suppose all the bba's in the training set are certain bba's. Each κ vector is made of ∞ 's, except for the singleton focal element where it is equal to 0. The intra-group distance is easier to compute when using the difference between two κ vectors. Among the objects in $S_v^{A_k}$, let G_c be the set of objects where the focal element is $c \in \Theta$.

  The difference between two κ's that share the same focal set is 0, and the one between two κ's with non equal focal sets is 2∞. The term $SumD(S_v^{A_k})$ is proportional to:

$$\begin{aligned}\sum_{I_j, I_w \in S_v^{A_k}} (\kappa^{\Theta}\{I_j\} - \kappa^{\Theta}\{I_w\})^2 &\propto \sum_{c\in\Theta} \sum_{I_j \in G_c} \sum_{c*\in\Theta} \sum_{I_w \in G_{c*}} (\kappa^{\Theta}\{I_j\} - \kappa^{\Theta}\{I_w\})^2 \\ &= \sum_{c\in\Theta} |G_c| \sum_{c*\in\Theta} |G_{c*}| \lambda(c, c*) \infty^2 \\ &= 2 \sum_{c\in\Theta} |G_c| (|S_v^{A_k}| - |G_c|) \infty^2\end{aligned}$$

  where

$$\lambda(c, c*) = \begin{cases} 2 & \text{if } c \neq c* \\ 0 & \text{otherwise} \end{cases}$$

  Finally, $SumD_{A_k}(S)$ is proportional to :

$$SumD_{A_k}(S) \propto \sum_{v \in D(A_k)} \sum_{c\in\Theta} |G_c| (|S_v^{A_k}| - |G_c|) \tag{4.21}$$

  which is maximal when $|G_c| = |S_v^{A_k}|/|\Theta|$ and minimal when $|G_c| = |S_v^{A_k}|$ for one c and 0 otherwise.

- *The categorical case:*

  Suppose all bba's in the training set are categorical bba's . Each κ vector is made of ∞'s, except for the subsets of the focal set where it is equal to 0.

  The intra-group distance is easier to compute when using the diference between two κ's vectors. The difference between two κ's that share the same focal set is 0. The difference between two κ's with non equal focal sets, for instance A and $B \subseteq \Theta$, is 2∞ multiplied by the number of subsets of A that are not subsets of B + the number of subsets of B that are not subsets of A. The computation proceeds as in the certain case.

4.3.2 Partitioning strategy

The partitioning strategy also known as the splitting strategy, defines the manner of the split of the training set according to the attribute values.

Since we deal with symbolic attributes, we create an edge for each value of the attribute chosen as a decision node. Thus, we get several training subsets where each one is relative to one branch and regrouping objects having the same attribute value.

The partitioning strategy for the construction of a belief decision tree is similar to the partitioning strategy used in the classical tree. This is due to the fact that the uncertainty is linked to the classes of the training instances and not to the values of their attributes.

4.3.3 Stopping criteria

The stopping criteria control the process of the construction of the belief decision tree. They allow to stop the development of a path and to declare the node as a leaf. In other words, the stopping criteria determine whether or not a training subset should be further divided.

Four strategies are proposed as stopping criteria:

1. If the treated node includes only one instance. Hence, the leaf will contain only this object.

2. If the treated node includes only instances for which the $m^{\Theta}\{I_j\}$'s are equal.

3. If there is no further attribute to test. In other words, if all the attributes are split.

4. If the value of the applied attribute selection measure (using either the gain ratio or the diff ratio) for the remaining attributes is less or equal than zero which means that the possible partition does not provide a better separation.

In such cases, a leaf will include one or several instances characterized by the same values for the selected attributes but generally having different bba's on their actual classes.

4.3.4 Structure of leaves

As mentioned, the leaf in a belief decision tree will be labeled by a basic belief assignment since the classes of the different training instances are expressed by the means of basic belief assignments.

The major question is how to compute each leaf's bba? In fact, Two cases must be treated:

In the averaging approach

Using the averaging approach in the attribute selection measure, the leaf's bba will be defined as follows:

1. When only one object belongs to the leaf L, the leaf's bba would be equal to this object's bba as defined in the training set.

2. When there are many objects attached to the leaf L, the leaf's bba would be equal to the average of the different basic belief assignments relative to these objects:

$$m^{\Theta}\{L\}(C) = \frac{\sum_{I_j \in L} m^{\Theta}\{I_j\}(C)}{|L|} \tag{4.22}$$

In the conjunctive approach

Using the conjunctive approach for the development of the attribute selection measure, the leaf's bba will be defined as follows:

1. When only one object belongs to the leaf L, the leaf's bba would be equal to this object's bba defined in the training set.

2. When there are many objects attached to the leaf L, the leaf's bba would be equal the result of the combination of these objects' bba by using the conjunctive rule:

$$m^{\Theta}\{L\} = \textcircled{\cap}_{I_j \in L}\ m^{\Theta}\{I_j\} \tag{4.23}$$

4.4 Conclusion

In this chapter, we have proposed the definition of the basic parameters useful for ensuring the construction and the classification procedures with the belief decision tree.

In fact, we have developed two attribute selection measures using the belief function formalism. Such parameter is considered as fundamental in the algorithm of the construction of belief decision trees . Then, we have defined the partitioning strategy and the stopping criteria regarding the structure of our training set. Finally, we have presented the structure of leaves in the belief decision tree, representing a major difference over the traditional decision tree.

In the next chapter, we present the construction of a belief decision tree where we detail the algorithm for building a decision tree in an uncertain case. Then, we develop the inference task ensuring the classification of new instances characterized basically by uncertain attribute values.

Chapter 5

Belief decision tree procedures

5.1 Introduction

Like the standard decision tree, the belief decision tree is composed of two principal procedures: the building or the construction of the tree from uncertain data and the classification of new instances that may be characterized by uncertain or even missing attribute values.

In the first part of this chapter, we are interested to the building procedure. In fact, the construction of a belief decision tree is based on the parameters defined in the previous chapter which are the attribute selection measure, the partitioning strategy, the stopping criteria and the structure of leaves.

Regarding these parameters, we present the procedure ensuring the construction of the belief decision tree. Examples explaining the unfolding of this procedure will be described.

In the second part of this chapter, we detail the classification procedure using belief decision trees. We develop three classification schema: 1) when all instances to classify have certain attribute values, 2) when some objects have disjunctive attribute values, and 3) the general case when the attribute values of the instances to classify are characterized by a belief function.

The different results found in this chapter are also detailed in (Elouedi et al., 2000a, 2001a).

5.2 Building procedure

5.2.1 Introduction

The building procedure is also called **the induction task**. It allows to induce a belief decision tree in order to use it for the classification task.

Even in an uncertain context, the building of belief decision trees requires a top down approach based on the conquer and divide principle.

5.2.2 Description

As mentioned, the algorithm to construct a decision tree (and consequently a belief decision tree) is based on four major parameters: the attribute selection measure, the partitioning strategy, the stopping criteria and the structure of leaves. These parameters must take into account the uncertainty encountered in the training set.

In fact, a belief decision is constructed from a training set of objects based on successive refinements. These refinements based on both the attribute selection measure and a partitioning strategy lead to small training subsets. The process is repeated until leaves are encountered. Such nodes have to satisfy at least one of the stopping criteria. Finally, for each leaf we have to compute its corresponding bba.

5.2.3 Algorithm of building a belief decision tree

Let T be a training set composed by objects I_j $(j = 1, \ldots, p)$ characterized by m symbolic attributes $(A_1, A_2, \ldots, A_m)$ and that belong to the set of classes $\Theta = \{C_1, C_2, \ldots, C_n\}$. For each object I_j of the training set will correspond a basic belief assignment $m^{\Theta}\{I_j\}$ expressing the belief about the value of $C(I_j)$, the actual class of the object I_j .

Our algorithm which uses a Top Down Induction of Decision Trees approach, will have the same skeleton as an ID3 and C4.5 algorithms (Quinlan, 1986, 1993). Besides, our algorithm is considered as generic since it provides two possibilities for selecting the attributes by using either the averaging approach or the conjunctive one.

Note that while using the averaging approach, we can use either the information gain or the gain ratio in an uncertain context. We will use the second criterion since it represents an improvement over the information gain selection measure. For the conjunctive approach, the diff ratio will be used.

The different steps of our algorithm leading to the construction of a belief decision tree are described as follows:

1. Generate the root node of the belief decision tree including all the objects of the training set T.
2. Choose which approaches will be applied: either the averaging approach or the conjunctive one.
3. Verify if this node satisfies or not at least one of the stopping criteria (see Section 4.3.3):
   - If yes, declare it as a leaf node and compute its corresponding bba according to the chosen approach (see Section 4.3.4).
   - If not, look for the attribute having the highest attribute selection measure (see Section 4.3.1). This attribute will be designed as the root of the belief decision tree related to the whole training set.
4. Apply the partitioning strategy (see Section 4.3.2) by developing an edge for each value of the chosen attribute as a root. This partition leads to several training subsets.
5. Create a root node relative to each training subset.
6. Repeat the same process for each training subset from the step 3, while verifying the stopping criteria.
7. Stop when all the nodes of the latter level of the tree are leaves.

Example 5.1 *Let's continue with the examples proposed in Chapter 4 (see Example 4.1 and Example 4.2). Our objective is to generate the two belief decision trees relative respectively to the average approach and the conjunctive approach.*

- *Averaging approach: As computed in the Example 4.1, we have found that:*

  $Gain\ Ratio(T, Income) = 0.361$;

  $Gain\ Ratio(T, Property) = 0.276$;

  $Gain\ Ratio(T, Unpaid\_credit) = 0.004$;

  We choose the income attribute as the root of the belief decision tree relative to the training set T, since it presents the highest gain ratio.

  Therefore, a branch is created below this root for each possible value (High, Average, Low, No) of the income attribute.

  We get the following belief decision tree (see Figure 5.1):

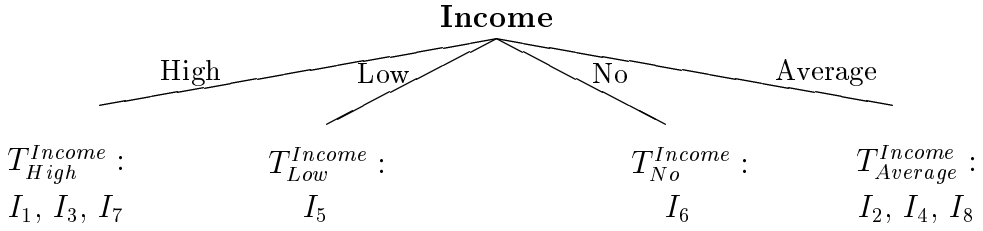

Figure 5.1: First generated belief decision tree

We notice that each of the training subsets T_{Low}^{Income} and T_{No}^{Income} contains only one object. Hence, one of the stopping criteria is fulfilled for these subsets. As a consequence, the nodes relative to T_{Low}^{Income} and T_{No}^{Income} are declared as leaves and their corresponding bba's will be respectively equal to $m^{\Theta}\{I_5\}$ and $m^{\Theta}\{I_6\}$.

For the training subsets T_{High}^{Income} and $T_{Average}^{Income}$, we apply the same process as we did for the training set T until at least one of the stopping criteria holds.

The final belief decision tree induced by our algorithm using the averaging approach is given by (see Figure 5.2):

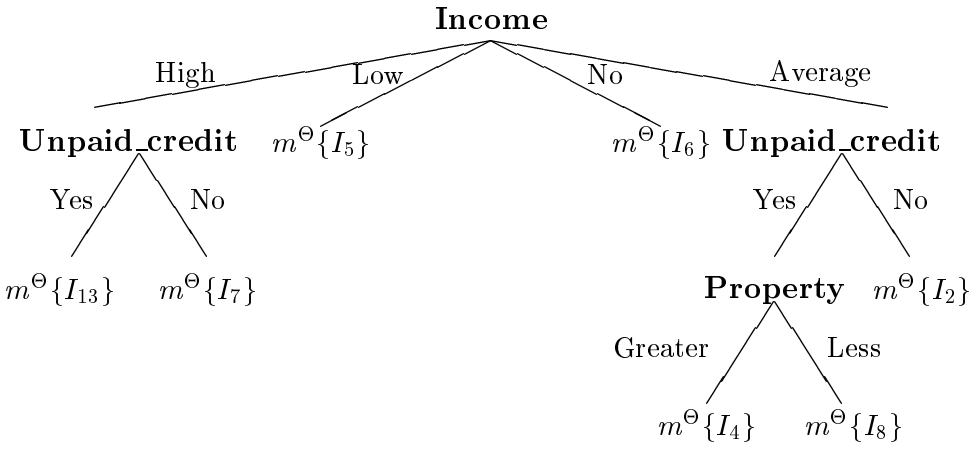

Figure 5.2: Final belief decision tree: Averaging approach

where $m^{\Theta}\{I_{13}\}$ is the average bba relative to the objects $\{I_1\}$ and $\{I_3\}$. So, we get (see Equation 4.22):

$m^{\Theta}\{I_{13}\}(C_1) = 0.65$;

$m^{\Theta}\{I_{13}\}(\Theta) = 0.35$;

- *Conjunctive approach: As computed in the Example 4.2, we have found that:*

  $Diff\ Ratio(T, Income) = 5.882$;

  $Diff\ Ratio(T, Property) = 5.707$;

  $Diff\ Ratio(T, Unpaid\_credit) = 4.279$;

  As noted, the income attribute presents the highest value of the Diff Ratio(T, A_k) where $A_k \in \{Income, Property, Unpaid\_credit\}$. Hence, it would be chosen as the root of the belief decision tree (like with the averaging approach).

By applying the same process to the induced training subsets, the final belief decision is (see Figure 5.3):

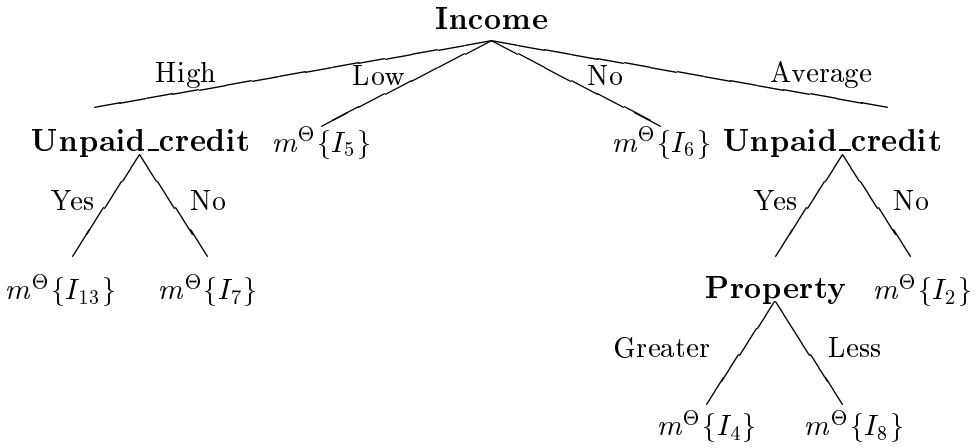

Figure 5.3: Final belief decision tree: Conjunctive approach

where $m^{\Theta}\{I_{13}\}$ *is defined as follows (see Equation 4.23):*
$m^{\Theta}\{I_{13}\} = m^{\Theta}\{I_1\} \textcircled{\cap} m^{\Theta}\{I_3\}$.

So, we get:

$m^{\Theta}\{I_{13}\}(C_1) = 0.88;$

$m^{\Theta}\{I_{13}\}(\Theta) = 0.12;$

Remark:

For this example, we get the same belief decision tree by using either the averaging approach or the conjunctive one (but some bba's leaves are different). This may not usually happen since these are two different approaches.

In fact, the induced belief decision trees may be identical or different according to the data composing the training set. In other words, it depends on the training instances and their corresponding classes' bba's.

5.3 Classification procedure

5.3.1 Introduction

Once the belief decision tree is constructed, the following procedure will be the classification of unseen instances referring to as new objects. Such task is also named **the inference task**.

As we deal with an uncertain environment, several cases regarding the knowledge of the attribute values have been studied in order to ensure classification using a belief decision tree.

5.3.2 Standard classification

Our method is able to ensure the standard classification where each attribute value (of the new instance to classify) is assumed to be exact and certain.

As in an ordinary tree, it consists in 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 belief decision tree, the new instance's class will be defined by a basic belief assignment related to the reached leaf. This bba defined on the set of classes, represents beliefs on the different subsets of classes (singletons and disjunctions) of the new instance to classify.

In order to make a decision and to get the probability of each singular class, we propose to apply the pignistic transformation.

Example 5.2 *Let's continue with the Example 5.1 (we use the results provided by the averaging approach*[1]*) and assume a new object to classify is characterized by the following values:*

- *Income = High;*
- *Property = Greater;*
- *Unpaid_credit = Yes;*

[1] The same process can be obviously applied to the conjunctive approach.

By tracing out the path from the root to one of its leaves of our generated belief decision tree (see Figure 5.2), the new instance will belong to the leaf labeled by $m^{\Theta}\{I_{13}\}$ *as a bba defined as follows:*
$m^{\Theta}\{I_{13}\}(C_1) = 0.65$;
$m^{\Theta}\{I_{13}\}(\Theta) = 0.35$;

Hence, the part of belief that this new instance belongs to the class C_1 *is 0.65, whereas the part of belief that it belongs to the class* C_1 *or* C_2 *or* C_3 *is 0.35.*

To make a decision, we have to apply the pignistic transformation, we get:
$BetP^{\Theta}\{I_{13}\}(C_1) = 0.76$;
$BetP^{\Theta}\{I_{13}\}(C_2) = 0.12$;
$BetP^{\Theta}\{I_{13}\}(C_3) = 0.12$;

Therefore, the probabilities that this instance belongs to C_1, C_2 *and* C_3 *are respectively equal to 0.76, 0.12 and 0.12. As a consequence, the new instance has more chances to belong to the class* C_1. *In other words, this client will be considered as a good client and the bank may accept to give him the whole asked loan.*

5.3.3 Disjunctive case

As we deal with an uncertain context, we propose, in our method, the classification of new instances characterized by uncertainty in the values of their attributes.

In this case, we assume new objects to classify are not only described by certain attribute values, but may also be described by the means of disjunctive values for some attributes. They may even have attributes with unknown values (missing values).

The idea to classify such objects is to look for the leaves that the given instance may belong to by tracing out all the possible paths induced by the different attribute values of the object to classify.

As a consequence, the new instance may belong to many leaves where each one is characterized by a basic belief assignment. These bba's must be combined in order to get beliefs on the instance's classes. The disjunctive rule of combination developed by Smets (see Equation 1.46) seems offering

a suitable context since it supposes at least one path is true (but we do not know which path).

When a decision has to be made, the induced bba from the disjunctive rule is transformed into a probability function by applying the pignistic transformation. This allows knowing the probability of the new instance to belong to each singular class related to the given problem.

Remark:

As mentioned, this case deals also with the total ignorance of some attribute values. When we handle unknown attribute values, all the branches relative to the considered attribute will be taken into account. Then, the same process (as in the case of the disjunctive values) will be applied.

Example 5.3 *Let's continue with the Example 5.1 and suppose the client I to classify is characterized by an average income, an unknown value regarding the property attribute and an unpaid credit.*

Since the value of the unpaid_credit attribute is unknown, then we have two possible paths to explore:

1. *(Income = Average, Property = Greater, Unpaid_credit = Yes),*
2. *(Income = Average, Property = Less, Unpaid_credit = Yes).*

By using our induced belief decision tree (see Figure 5.2), the first path leads to the leaf characterized by $m^{\Theta}\{I_4\}$*, whereas the second path leads to the one characterized by* $m^{\Theta}\{I_8\}$ *as a bba.*

Let $m^{\Theta}\{I\}$ *be the bba defining beliefs on the classes of this client I to classify. The bba* $m^{\Theta}\{I\}$ *is the result of the application of the disjunctive rule combining* $m^{\Theta}\{I_4\}$ *and* $m^{\Theta}\{I_8\}$*. So, we get:*

$m^{\Theta}\{I\} = m^{\Theta}\{I_4\} \textcircled{\cup} m^{\Theta}\{I_8\}$

where

$m^{\Theta}\{I\}(C_2) = 0.24$;
$m^{\Theta}\{I\}(C_3) = 0.12$;
$m^{\Theta}\{I\}(C_2 \cup C_3) = 0.36$;
$m^{\Theta}\{I\}(\Theta) = 0.28$;

By applying the pignistic transformation, we get:
$BetP^{\Theta}\{I\}(C_1) = 0.1$*;*
$BetP^{\Theta}\{I\}(C_2) = 0.51$*;*
$BetP^{\Theta}\{I\}(C_3) = 0.39$*;*

We notice that this instance I has the most probability to belong to the class C_2 which is equal to 0.51. So, this client may be considered by the bank as a moderate client and it is more probable that he will receive only a part of the asked loan.

5.3.4 General case

The uncertainty characterizing the new instances to classify is not necessarily presented by disjunctive values or missing values (total ignorance), but it may also be more complicated especially when the attribute values are described by several experts. Therefore, it would be interesting to extend our classification procedure based on a belief decision tree in order to handle more uncertainty.

The uncertainty about the value of the attribute can be defined by a bba on the set of all the possible values of the attribute. Such bba can be given by one expert or resulted from the combination (using the conjunctive rule) of several bba's, relative to many experts, on the attribute values. Let:

- m^{A_k} be the bba representing the part of belief committed exactly to the different values of the attribute A_k of the new instance to classify. This bba is defined on the frame of discernment Θ_{A_k} including all the possible values of the attribute A_k.

- Θ_A be the global frame of discernment relative to all the attributes belonging to the set of attributes A. It is equal to the cross product of the different Θ_{A_k}. We denote by:

$$\Theta_A = \times_{k=1,\dots,m}\Theta_{A_k} \tag{5.1}$$

Since an instance is described by a set of combination of values where each one is relative to an attribute, we have firstly to find the bba expressing beliefs on both the different attributes' values of the new instance to classify. In other words, we have to look for the joint bba representing beliefs on all the instance's attributes.

To ensure this objective, we have to apply the following steps:

1. Extend the different bba's m^{A_k} to the global frame of attributes Θ_A. As a result, we get the different bba's $m^{A_k\uparrow A}$.

2. Combine the different extended bba's by applying the conjunctive rule of combination.

$$m^A = \bigcirc\!\!\!\!\cap_{k=1,\ldots,m} m^{A_k\uparrow A} \tag{5.2}$$

Thus, we get a joint bba m^A representing beliefs on the different combinations of the attributes characterizing the given instance. Should there be some *'correlation'* between the bba's, the procedure could be adapted, but in any case the end product is a bba on Θ_A.

We then, consider individually the focal elements of the bba m^A. Let x be such a focal element. The next step in our classification task is to compute the belief functions $bel^{\Theta}[x]$, where Θ is the set of possible classes. The computation of this function depends on the subset x and more exactly on the focal elements of the bba m^A:

1. If the treated focal element x is a singleton (only one value for each attribute), then $bel^{\Theta}[x]$ is equal to the belief function corresponding to the leaf to which this focal element is attached.

2. If the focal element x is not a singleton (some attributes have more than one value), that is it contains a disjunction in some attribute values. Then, we have to explore all the possible paths relative to this combination of values. Two cases are possible:

   - If these paths lead to one leaf, then $bel^{\Theta}[x]$ is equal to this leaf's belief function.
   - If these paths lead to distinct leaves, then $bel^{\Theta}[x]$ is equal to the result of the disjunctive combination of each leaf's belief function by applying the disjunctive rule.

Finally, the belief functions computed with each focal element x are averaged using the m^A:

$$bel^{\Theta}[m^A](C) = \sum_{x\subseteq\Theta_A} m^A(x) bel^{\Theta}[x](C) \text{ for } C \subseteq \Theta \tag{5.3}$$

$bel^{\Theta}[m^A]$ represents total beliefs on the classes (subsets of classes) that the new instance may to belong.

To make a decision, this belief function is transformed to a probability function on singular classes via the pignistic transformation.

Remark:

Note that the cases of certain or disjunctive attribute values are included within this generalized case of classification of instances. These cases are easily expressed within the belief function framework.

For the standard case of certain attribute values, we get certain bba's, whereas for the case of missing values, the attribute is simply expressed by a vacuous bba. The attribute characterized by disjunctive values is represented by a categorical bba having only one focal element which is the disjunction of values.

Example 5.4 *In order to illustrate our generalized case, suppose we would classify a client characterized by certain and exact values for its income and unpaid_credit attributes which are respectively average and yes. However, there is some uncertainty in the value of the property attribute.*

The values of this client's attributes are defined as follows:

- *Income = Average which is equivalent to a certain bba* m^{Income} *having only the value average as a focal element:* $m^{Income}(\{Average\}) = 1$;
- *Unpaid_credit = Yes which is equivalent to a certain bba* m^{Unpaid_credit} *having only the value yes as a focal element:* $m^{Unpaid\_credit}(\{Yes\}) = 1$;
- *However, the value relative to the property attribute is uncertain and described by the bba* $m^{Property}$ *such that:* $m^{Property}(\{Greater\}) = 0.4$; $m^{Property}(\{Less\}) = 0.3$; $m^{Property}(\Theta_{Property}) = 0.3$;

Let's remind that:
$\Theta_{Income} = \{High, Average, Low, No\}$
$\Theta_{Property} = \{Greater, Less\}$
$\Theta_{Unpaid\_credit} = \{Yes, No\}$

Let $\Theta_A = \Theta_{Income} \times \Theta_{Property} \times \Theta_{Unpaid\_credit}$

So, Θ_A = { *(High, Greater, Yes), (High, Greater, No), (High, Less, Yes),(High, Less, No), (Average, Greater, Yes), (Average, Greater, No), (Average, Less, Yes), (Average, Less, No), (Low, Greater, Yes), (Low, Greater, No), (Low, Less, Yes), (Low, Less, No), (No, Greater, Yes), (No, Greater, No), (No, Less, Yes), (No, Less, No)* }

The extension of the different bba's to Θ_A*, the frame of discernment relative to the set of attributes A, gives as a result:*

- $m^{Income\uparrow A}(\{Average\} \times \Theta_{Property} \times \Theta_{Unpaid\_credit}) = 1$;
- $m^{Unpaid-credit\uparrow A}(\Theta_{Income} \times \Theta_{Property} \times \{Yes\}) = 1$;
- $m^{Property\uparrow A}(\Theta_{Income} \times \{Greater\} \times \Theta_{Unpaid\_credit}) = 0.4$;
  $m^{Property\uparrow A}(\Theta_{Income} \times \{Less\} \times \Theta_{Unpaid\_credit}) = 0.3$;
  $m^{Property\uparrow A}(\Theta_A) = 0.3$;

Once the attributes' bba's are extended to Θ_A*, then we can apply the conjunctive rule. The result of this combination will be a joint bba on singular instances or subsets of instances. So, we get:*
$m^A = m^{Income\uparrow A} \textcircled{\cap} m^{Property\uparrow A} \textcircled{\cap} m^{Unpaid\_credit\uparrow A}$ *such that:*
$m^A(\{(Average, Greater, Yes)\}) = 0.4$;
$m^A(\{(Average, Less, Yes)\}) = 0.3$;
$m^A(\{Average\} \times \Theta_{Property} \times \{Yes\}) = 0.3$;

Next, we have to find beliefs on classes (defined on Θ*) given the values of the attributes characterizing the new instance to classify. In fact, three belief functions have to be defined where for each one, we take into account one focal element of* m^A*.*

According to the belief decision tree generated (see Figure 5.2), we get:
$bel^{\Theta}[\{(Average, Greater, Yes)\}] = bel_4$;
$bel^{\Theta}[\{(Average, Less, Yes)\}] = bel_8$;
$bel^{\Theta}[\{Average\} \times \Theta_{Property} \times \{Yes\}] = bel_4 \textcircled{\cup} bel_8$;

Table 5.1: Beliefs on classes given the attributes' values

	C_1	C_2	C_3	$C_1 \cup C_2$	$C_1 \cup C_3$	$C_2 \cup C_3$	Θ
bel_4	0	0.6	0.3	0.6	0.3	0.9	1
bel_8	0	0.4	0.4	0.4	0.4	0.8	1
$bel_4 \textcircled{\cup} bel_8$	0	0.24	0.12	0.24	0.12	0.72	1

Hence, these belief functions will be averaged (see Equation 5.2), we get:
$bel^{\Theta}[m^A](C_1) = 0.4 * 0 + 0.3 * 0 + 0.3 * 0 = 0$*;*
$bel^{\Theta}[m^A](C_2) = 0.4 * 0.6 + 0.3 * 0.4 + 0.3 * 0.24 = 0.43$*;*
$bel^{\Theta}[m^A](C_3) = 0.4 * 0.3 + 0.3 * 0.4 + 0.3 * 0.12 = 0.28$*;*
$bel^{\Theta}[m^A](C_1 \cup C_2) = 0.4 * 0.6 + 0.3 * 0.4 + 0.3 * 0.24 = 0.43$;
$bel^{\Theta}[m^A](C_1 \cup C_3) = 0.4 * 0.3 + 0.3 * 0.4 + 0.3 * 0.12 = 0.28$*;*
$bel^{\Theta}[m^A](C_2 \cup C_3) = 0.4 * 0.9 + 0.3 * 0.8 + 0.3 * 0.72 = 0.82$*;*
$bel^{\Theta}[m^A](\Theta) = 1$;

Hence, we get:
$m^{\Theta}[m^A](C_1) = 0$;
$m^{\Theta}[m^A](C_2) = 0.43$;
$m^{\Theta}[m^A](C_3) = 0.28$;
$m^{\Theta}[m^A](C_1 \cup C_2) = 0$;
$m^{\Theta}[m^A](C_1 \cup C_3) = 0$;
$m^{\Theta}[m^A](C_2 \cup C_3) = 0.11$;
$m^{\Theta}[m^A](\Theta) = 0.18$;

Each basic belief mass represents the part of belief that the given instance may belong to such focal element.

Applying the pignistic transformation, the pignistic probability will be defined as follows:
$BetP(C_1) = 0.06$*;*
$BetP(C_2) = 0.55$*;*
$BetP(C_3) = 0.39$;

We notice that the probabilities that this instance belongs respectively to the classes C_1, C_2 and C_3 are respectively 0.06, 0.55 and 0.39.

As a consequence, it is most probable that this new instance (characterized by average income, an unpaid_credit, and uncertain value of the property attribute) will belong to the class C_2. In other words, this client may be considered, by the bank, as a moderate client and just a part of his asked loan will be given to him.

5.4 Conclusion

In this chapter, we have presented the two major procedures related to the belief decision tree technique. We have described the different steps of the procedure allowing the construction of the tree from uncertain data.

Next, we have detailed the inference task ensuring the classification of new instances using the constructed belief decision tree. Several cases, that depend on the nature of the attribute values, are analyzed in order to induce classification even with uncertain values.

The construction and the classification procedures are explained by an example dealing with the classification of clients in a bank in order to allow the bank to plan its loan policy. Another example relative to strategic problems will be presented in Appendix A. It will handle the classification of scenarios in the agriculture field in an uncertain context using the belief decision tree approach (Elouedi et al., 2000a).

In the next chapter, we will deal with the implementation of belief decision trees with both averaging and conjunctive approaches and their use for classification. Simulations will be performed in order to show the feasibility of our proposed methods.

Chapter 6

Implementation and simulation

6.1 Introduction

Implementing our belief decision tree approaches seems imperative since it allows us to have an idea concerning the feasibility of our proposed methods.

Hence, at first building belief decision trees in both averaging and conjunctive approaches will be ensured, then classification of new objects with both induced trees will be implemented.

Different results will be presented namely the different nodes of the tree, the number of decision nodes, leaves, and the different values of the criteria reflecting the classification results.

Once the different programs are implemented, for checking the feasibility of our approaches regarding belief decision trees and judging their qualities, we have performed several tests and simulations on data sets.

In fact, this chapter is composed of two parts:

- The first part deals with the implementation of belief decision trees where the major variables and programs are explained, then the important algorithms are detailed.
- The second one is interested to the simulation phase, the different parameters used in simulation are defined. Then, results over data from a real world problem are presented with an analysis of them. Note that the objective of simulation remains basically to show the well unfolding of our belief decision tree algorithms and not on comparison.

6.2 Implementation

6.2.1 The framework

In order to test our two approaches (averaging and conjunctive), we have developed programs in Matlab V6.0, implementing basically the building phase (according to the two approaches), then the classification phase taking into account the induced belief decision trees.

As detailed in the previous chapters, these programs are developed to handle symbolic attributes.

These programs have as input data sets with objects having certain attribute values but uncertain classes represented by basic belief assignments.

When there is no uncertainty in the training instances' classes, we can also apply the classical C4.5 algorithm of Quinlan which is then a particular case of our belief decision tree algorithm based on the averaging approach.

The outputs of our programs are basically:

1. The belief decision trees induced from the two approaches (averaging and conjunctive), and in particular some characteristics of these trees like the number of leaves and the number of decision nodes.

2. Results from the classification phase applied to the testing set (and even to the training set), namely the Percent of Correct Classification, denoted PCC, the *kappa* and the distance criterion, denoted $dist\_crit$, considered as criteria (that will be detailed in the following sections) to assess classification of objects according to the induced belief decision tree.

6.2.2 Major variables

In this section, we present the list of the major variables that are used in our developed algorithms (it is not an exhaustive list):

- *training_set* includes the attribute values of all the training instances, the beliefs on their classes and also their real classes (this latter value is useful to build the corresponding bba's on the training instances' classes).

- *number_attributes* is the number of attributes relative to the given classification problem.
- *number_classes* is the number of classes relative to the given classification problem.
- *number_value_attributes* is a vector representing the number of possible values of each attribute.
- *number_training_instances* is the number of instances in the training set.
- *node* (either a leaf or decision node) is composed of:
  - *.type* which may be a *'leaf'* or an *'attribute'*.
  - *.cases* which contains the remaining instances belonging to this node. Obviously, in the case of leaves, it represents the different objects belonging to such a leaf.
  - *.beliefs* is equal to 0 when it is an attribute, otherwise (when it is a leaf) computed as an average bba or as a joint bba of the instances belonging to the leaf (according to the applied approach).
  - *.name* is equal to 0 when it is a leaf, otherwise (when it is an attribute) contains the name of the attribute (its position).
  - *.forks* is equal to 0 when it is a leaf, otherwise (when it is an attribute), it contains the number of possible values of this attribute (in other words, the number of its branches).
  - *.level* is the level of the node in the belief decision tree.
  - *.path_attributes* represents the attributes leading to this node.
  - *.path_values* represents the values of the attributes leading to this node.
- *belief_classes* represents beliefs on classes relative to the training instances.
- *set_attribute_values* is the set of the different values of the attributes relative to the training instances.
- *remaining_attributes* is the list of the remaining attributes to be selected.
- *remaining_instances* is the list of the remaining instances to be treated.

- *belief_decision_tree_ave* is the belief decision tree induced from the application of the averaging approach. It is composed by nodes (decision nodes and leaves).
- *belief_decision_tree_conj* is the belief decision tree induced from the application of the conjunctive approach. It is composed by nodes (decision nodes and leaves).
- *number_non_empty_leaves* presents the number of non empty leaves in the induced belief decision tree.
- *number_empty_leaves* presents the number of empty leaves in the induced belief decision tree.
- *number_decision_nodes* presents the number of attribute nodes in the induced belief decision tree.
- *testing_set* includes the attribute values of all the testing instances, the beliefs on their classes (which will not be used, but it is mentioned just to have the same structure as the variable *training_set*) and also their real classes that will be compared with the classes given by the induced belief decision tree.
- *class_truth* represents the real class of the instances to classify.
- *instance_to_classify* is a matrix describing in each of its rows the value of each attribute of one instance to classify (from the testing set).
- PCC is the percent of correct classification of the instances to classify.
- *kappa* is the kappa criterion usually used to assess the classification of the instances to classify.
- *dist_crit* is the value of the distance criterion that we have developed to assess the classification of the instances to classify. It it is based on the computation of the distance between the pignistic probability given by the induced belief decision tree and the real class of each object to classify.

6.2.3 Belief decision tree programs

Introduction

We present the different programs that we have developed in order to build the belief decision trees using our two approaches, namely the averaging approach and the conjunctive approach. Then, the programs permitting to use the induced trees.

These developed programs can be regrouped according to their use. These groups of programs are described in the following subsections.

Execution

It includes programs ensuring the execution of the software allowing to get results of both approaches: the averaging and the conjunctive approaches.

- **Test:** allows to build a belief decision tree according to the averaging and conjunctive approaches on a data set, then it presents the classification results on the testing set for both approaches (averaging and conjunctive).
- **Test_Quinlan:** allows to build a belief decision tree, relative to a data set, in the certain case according to the Quinlan algorithm C4.5. Such algorithm represents a particular case of our averaging method with the belief decision tree approach. Once the tree is built, **Test_Quinlan** presents the classification results on the testing set.

In both programs, we can also present classification results on the training set.

Data creation

To create data that will be used to build belief decision trees, the idea is to deal with an existing database characterized by certain attributes' values and a certain and unique class for each one of its training instances, then we have to create bba's relative to the classes of the training instances.

These databases have to be divided into two parts: the training set and the testing set representing generally, respectively 90% and 10% from the whole generated database[1].

[1] These percentages may change according to the number of objects in the database, it may be for example 50% for each one of training and testing sets.

For creating data, Two kinds of programs have been developed:

- **Get_simulated_data:** generates training instances and testing instances relative to a database, then we create, artificially, bba's on their classes.
- **Get_simulated_data_Quinlan:** generates training instances and testing instances relative to a database that will be applied to the certain case. So, the bba's on classes are certain bba's, i.e., each one has only one focal element which is a singleton.

Definition of the training set

In this subsection, there are two procedures allowing to define the two parts of the training set in a belief decision tree, namely the attribute values of training instances and their classes' beliefs. Such programs are useful for getting more specific data from a training set to ensure the building of the tree.

- **Getting_attribute_values:** allows to get the different attribute values relative to training instances.
- **Getting_belief_classes:** allows to get beliefs on classes relative to training instances.

Building

The building phase is considered as an important phase, as in the averaging approach and in the conjunctive one, ensuring the building of belief decision trees. For the programs that we have developed, there are specific programs for each approach, and others common for both approaches.

1. **Averaging approach**
   - **Build_averaging:** is the principle program to build a belief decision tree according to the averaging approach. Besides, it presents the number of empty and non empty leaves, and the number of decision nodes relative to the induced tree.
   - **Build_bdt_averaging:** is a recursive function allowing to build a belief decision tree according to the averaging approach.
   - **Best_attribute_averaging:** allows to choose the best attribute according to the averaging approach and to fix its position. The chosen attribute is the one having the highest gain ratio.

- **Eval_gain_ratio:** allows to evaluate the gain ratio of each attribute belonging to the list of remaining attributes (the non selected attributes).
- **Compute_gain_ratio:** allows to compute the gain ratio of an attribute.
- **Info:** computes the Info value relative to a given training (sub) set (before partition). Such value represents the entropy of the treated training (sub) set.
- **InfoS:** computes the InfoS value of a given attribute relative to a given training (sub) set (after partition). The induced value from the procedure InfoS is considered as the weighted sum of the different Info values where each one is relative to a subset of one of the attribute values.
- **Average_bet:** computes the average pignistic probability $BetP$ relative to a given matrix (representing generally bba's on some training instances' classes).
- **Average_bba:** computes the average bba relative to a given matrix (representing generally bba's on some training instances' classes).

2. **Conjunctive approach**

- **Build_conjunctive:** is the principle program to build belief decision tree according to the conjunctive approach. Besides, it presents the number of empty and non empty leaves, and the number of decision nodes relative to the induced tree.
- **Build_bdt_conjunctive:** is a recursive function allowing to build a belief decision tree according to the conjunctive approach.
- **Best_attribute_conjunctive:** allows to choose the best attribute according to the conjunctive approach and to fix its position. The chosen attribute is the one having the highest diff ratio (the developed attribute selection measure according to the conjunctive approach).
- **Eval_diff_ratio:** allows to evaluate the diff ratio of each attribute belonging to the list of remaining attributes (the non selected attributes).
- **Compute_diff_ratio:** allows to compute the diff ratio of an attribute.

- **SumD:** computes the sum of distances between the training instances (before partition). It represents the intra-group distance in the given (sub) training set.
- **SumD_attribute:** computes the $SumD_{A_k}$ value of a given attribute A_k (after partition). The value induced by the procedure SumD_attribute is considered as the weighted sum of the different SumD (values) where each one is relative to a subset on one of the attribute values.
- **Dist:** computes the distance between two commonality functions relative to one instance and a training subset.
- **Conjunctive:** computes the conjunctive bba relative to a given matrix (representing generally bba's on training instances' classes).

3. **Common procedures for both approaches**

   These procedures are useful in both averaging and conjunctive approaches.

   - **SplitInfo:** computes the split info value of a given attribute. This quantity describes the information content of the attribute itself.
   - **Partition:** returns the class attribute values regrouping instances according to the values of the attribute.
   - **Freq_attribute_values:** computes the number of instances for each attribute value.
   - **Reset_freq:** initializes the number of instances for each attribute value with 0.
   - **Find_remaining_attributes:** allows to find the non selected attributes.
   - **Find_remaining_instances:** allows to find the remaining instances to be treated once the attribute is selected.

Classification

This set of procedures ensures the classification of testing instances[2]. These procedures are the following ones:

- **Classification_results:** computes the PCC, the *kappa* and the distance criterion ($dist\_crit$) enhanced from the classification of testing instances (or even training instances).

[2] Even training instances, if we would evaluate the classification on these instances.

- **Best_class:** allows to give the most probable class of the instances to classify and its corresponding pignistic probability.
- **Classify:** is used to classify an instance according to the induced tree. It gives its corresponding basic belief assignment and pignistic probability. Note that the objects to classify are characterized by certain attribute values since they can be either training or testing instances. So, all objects in the database have the same structure.

6.2.4 Belief decision tree algorithms

In this section, we present the major algorithms relative to both the belief decision tree approaches. Two phases will be considered: the building phase, then the classification phase.

Building phase

Averaging approach

Algorithm *Building_averaging*
Input: training_set, number_attributes, number_classes, number_value_attributes, number_training_instances
Output: belief_decision_tree_ave, number_non_empty_leaves, number_empty_ leaves, number_decision_nodes

```
1.  begin
2.  (* Getting beliefs on classes relative to the training instances *)
3.  belief_classes ← Getting_belief_classes(training_set, number_attributes,
    number_classes, number_training_instances);
4.
5.  (* Getting the different attribute values relative to the training instances *)
6.  set_attribute_values ← Getting_attribute_values(training_set,
    number_attributes, number_training_instances);
7.
8.  (* Getting the lists of instances and attributes in the whole training set *)
9.  list_instances ← [1:number_training_instances];
10. list_attributes ← [1:number_attributes];
11.
12. (* Initialization of useful variables *)
13. level ← 0;
14. belief_decision_tree_ave ← [ ];
15. path_attributes ← [ ];
16. path_values ← [ ];
17.
```

```
18.  (* Applying the recursive program build_belief_bdt_averaging *)
19.  belief_decision_tree_ave ← Build_bdt_averaging(belief_decision_tree_ave,
     number_classes, belief_classes, set_attribute_values, list_attributes,
     list_instances, number_value_attributes, level, path_attributes, path_values);
20.
21.  (* Finding the variables number_non_empty_leaves, number_empty_leaves, num-
     ber_decision_nodes *)
22.  number_non_empty_leaves ← 0;
23.  number_empty_leaves ← 0;
24.  number_decision_nodes ← 0;
25.  teta ← 2^number_classes;
26.  for i ← 1 to length(belief_decision_tree_ave)
27.      do if strcmp(belief_decision_tree_ave(i).type, "leaf")
28.           then if belief_decision_tree_ave(i).beliefs(teta) = 1
29.                   then number_empty_leaves ← number_empty_leaves + 1;
30.                   else number_non_empty_leaves ← number_non_empty_
                           leaves + 1;
31.                end if
32.           else number_decision_nodes ← number_decision_nodes + 1;
33.         end if
34.  end for
35.  end
```

Algorithm *Build_bdt_averaging*

Input: belief_decision_tree_ave, number_classes, belief_classes, set_attribute_values, remaining_attributes, remaining_instances, number_value_attributes, level, path_attributes, path_values

Output: belief_decision_tree_ave

```
1.   begin
2.   (* Updating the level *)
3.   l ← level + 1;
4.
5.   (* Defining the number of instances and attributes to be treated *)
6.   [number_instances_to_be_treated, number_attributes_to_be_treated] ← size(set_
     attribute_values(remaining_instances, remaining_attributes));
7.
8.   (* First test to know if the node is a leaf or not *)
9.   if number_instances_to_be_treated ≤ 1 or
     number_attributes_to_be_treated = 0
10.    then node.type ← "leaf";
11.         node.cases ← remaining_instances';
12.         node.beliefs ← Average_bba(belief_classes(remaining_instances,:));
13.         node.name ← 0;
14.         node.forks ← 0;
```

```
15.          node.level ← l;
16.          node.path_attributes ← path_attributes;
17.          node.path_values ← path_values;
18.          belief_decision_tree_ave ← [belief_decision_tree_ave node];
19.
20.          (* Position and gain ratio of the best_gain_ratio attribute according
             to the averaging approach *)
21.     else [best_gain_ratio, position] ← Best_attribute_averaging(belief_classes,
             set_attribute_values, remaining_attributes, remaining_instances,
             number_classes, number_value_attributes);
22.
23.          (* Second test to know if the node is a leaf or not *)
24.          if best_gain_ratio ≤ 0
25.            then node.type ← "leaf";
26.                 node.cases ← remaining_instances';
27.                 node.beliefs ← Average_bba(belief_classes(remaining_instan-
                    ces,:));
28.                 node.name ← 0;
29.                 node.forks ← 0;
30.                 node.level ← l;
31.                 node.path_attributes ← path_attributes;
32.                 node.path_values ← path_values;
33.                 belief_decision_tree_ave ← [belief_decision_tree_ave node];
34.
35.                 (* Partitioning *)
36.            else attribute_values ← set_attribute_values(:, position);
37.                 sub_remaining_attributes ← Find_remaining_attributes
                    (remaining_attributes, position);
38.                 remaining_attributes ← sub_remaining_attributes;
39.                 instances_before_for ← remaining_instances;
40.                 path_values_before_for ← path_values;
41.                 number_branches ← number_value_attributes(position);
42.
43.                 (* Defining elements of the node, in this case it is an at-
                    tribute *)
44.                 node.type ← "attribute";
45.                 node.cases ← remaining_instances';
46.                 node.beliefs ← 0;
47.                 node.name ← position;
48.                 node.forks ← number_value_attributes(position);
49.                 node.level ← l;
50.                 node.path_attributes ← path_attributes;
51.                 node.path_values ← path_values;
```

```
52.                belief_decision_tree_ave ← [belief_decision_tree_ave node];
53.
54.                (* Update the variable path_attributes *)
55.                path_attributes ← [path_attributes, position];
56.
57.                (* Treatment of each selected attribute value *)
58.                for v ← 1 to number_branches
59.                    do sub_remaining_instances ← Find_remaining_inst-
                           ances(attribute_values, v, remaining_instances);
60.                       remaining_instances ← sub_remaining_instances;
61.                       path_values ← [path_values, v];
62.
63.                       (* Applying a recursive program Build_bdt_averaging
                          for each training subset relative to the value v of the
                          selected attribute *)
64.                       belief_decision_tree_ave ← Build_bdt_averaging
                          (belief_decision_tree_ave, number_classes, belief_classes,
                          set_attribute_values, remaining_attributes,
                          remaining_instances, number_value_attributes, l,
                          path_attributes, path_values);
65.                       if v < number_branches
66.                          then remaining_instances ← instances_before_for;
67.                               path_values ← path_values_before_for;
68.                       end if
69.                end for
70.          end if
71.    end if
72.    end
```

Conjunctive approach

Algorithm *Building_conjunctive*

Input: training_set, number_attributes, number_classes, number_value_attributes, number_training_instances

Output: belief_decision_tree_conj, number_non_empty_leaves, number_empty_ leaves, number_decision_nodes

```
1.  begin
2.  (* Getting beliefs on classes relative to the training instances *)
3.  belief_classes ← Getting_belief_classes(training_set, number_attributes,
    number_classes, number_training_instances);
4.
5.  (* Getting the different attribute values relative to the training instances *)
6.  set_attribute_values ← Getting_attribute_values(training_set,
    number_attributes, number_training_instances);
```

```
7.
8.   (* Getting the lists of instances and attributes in the whole training set *)
9.   list_instances ← [1:number_training_instances];
10.  list_attributes ← [1:number_attributes];
11.
12.  (* Initialization of useful variables *)
13.  level ← 0;
14.  belief_decision_tree_conj ← [ ];
15.  path_attributes ← [ ];
16.  path_values ← [ ];
17.
18.  (* Applying the recursive program build_belief_bdt_conjunctive *)
19.  belief_decision_tree_conj ← Build_bdt_conjunctive(belief_decision_tree_conj,
     number_classes, belief_classes, set_attribute_values, list_attributes,
     list_instances, number_value_attributes, level, path_attributes, path_values);
20.
21.  (* Finding the variables number_non_empty_leaves, number_empty_leaves, num-
     ber_decision_nodes *)
22.  number_non_empty_leaves ← 0;
23.  number_empty_leaves ← 0;
24.  number_decision_nodes ← 0;
25.  teta ← 2^number_classes;
26.  for i ← 1 to length(belief_decision_tree_conj)
27.       do if strcmp(belief_decision_tree_conj(i).type, "leaf")
28.            then if belief_decision_tree_conj(i).beliefs(teta) = 1
29.                      then number_empty_leaves ← number_empty_leaves + 1;
30.                      else  number_non_empty_leaves ← number_non_empty_
                                leaves + 1;
31.                 end if
32.            else  number_decision_nodes ← number_decision_nodes + 1;
33.          end if
34.  end for
35.  end
```

Algorithm *Build_bdt_conjunctive*

Input: belief_decision_tree_conj, number_classes, belief_classes, set_attribute_values, remaining_attributes, remaining_instances, number_value_attributes, level, path_attributes, path_values

Output: belief_decision_tree_conj

```
1.   begin
2.   (* Update the level *)
3.   l ← level + 1;
4.
```

```
5.  (* Defining the number of instances and attributes to be treated *)
6.  [number_instances_to_be_treated, number_attributes_to_be_treated] ← size(set_
    attribute_values(remaining_instances, remaining_attributes));
7.
8.  (* First test to know if the node is a leaf or not *)
9.  if number_instances_to_be_treated ≤ 1 or
    number_attributes_to_be_treated = 0
10.   then node.type ← "leaf";
11.        node.cases ← remaining_instances';
12.        node.beliefs ← Conjunctive(belief_classes(remaining_instances,:));
13.        node.name ← 0;
14.        node.forks ← 0;
15.        node.level ← l;
16.        node.path_attributes ← path_attributes;
17.        node.path_values ← path_values;
18.        belief_decision_tree_conj ← [belief_decision_tree_conj node];
19.
20.        (* Position and gain ratio of the best_diff_ratio attribute according to
           the conjunctive approach *)
21.   else [best_diff_ratio, position] ← Best_attribute_conjunctive(belief_clas-
           ses, set_attribute_values, remaining_attributes, remaining_instances,
           number_classes, number_value_attributes);
22.
23.        (* Second test to know if the node is a leaf or not *)
24.        if best_diff_ratio ≤ 0
25.          then node.type ← "leaf";
26.               node.cases ← remaining_instances';
27.               node.beliefs ← Conjunctive(belief_classes(remaining_insta-
                  nces,:));
28.               node.name ← 0;
29.               node.forks ← 0;
30.               node.level ← l;
31.               node.path_attributes ← path_attributes;
32.               node.path_values ← path_values;
33.               belief_decision_tree_conj ← [belief_decision_tree_conj node];
34.
35.               (* Partitioning *)
36.          else attribute_values ← set_attribute_values(:,position);
37.               sub_remaining_attributes ← Find_remaining_attributes
                  (remaining_attributes, position);
38.               remaining_attributes ← sub_remaining_attributes;
39.               instances_before_for ← remaining_instances;
40.               path_values_before_for ← path_values;
```

```
41.                 number_branches ← number_value_attributes(position);
42.
43.                 (* Defining elements of the node, in this case it is an at-
                    tribute *)
44.                 node.type ← "attribute";
45.                 node.cases ← remaining_instances';
46.                 node.beliefs ← 0;
47.                 node.name ← position;
48.                 node.forks ← number_value_attributes(position);
49.                 node.level ← l;
50.                 node.path_attributes ← path_attributes;
51.                 node.path_values ← path_values;
52.                 belief_decision_tree_conj ← [belief_decision_tree_conj node];
53.
54.                 (* Update the variable path_attributes *)
55.                 path_attributes ← [path_attributes, position];
56.
57.                 (* Treatment of each selected attribute value *)
58.                 for v ← 1 to number_branches
59.                     do sub_remaining_instances ← Find_remaining_inst-
                        ances (attribute_values, v, remaining_instances);
60.                        remaining_instances ← sub_remaining_instances;
61.                        path_values ← [path_values, v];
62.
63.                        (* Applying a recursive program Build_bdt_conjunctive
                        for each training subset relative to the value v of the
                        selected attribute *)
64.                        belief_decision_tree_conj ← Build_bdt_conjunctive
                        (belief_decision_tree_conj, number_classes, belief_classes,
                        set_attribute_values, remaining_attributes, remaining_
                        instances, number_value_attributes, l, path_attributes,
                        path_values);
65.                        if v < number_branches
66.                          then remaining_instances ← instances_before_for;
67.                                path_values ← path_values_before_for;
68.                        end if
69.                 end for
70.         end if
71. end if
72. end
```

Classification phase

In the **Classification_results** algorithm, the different classification results are computed on the testing set with both averaging and conjunctive approaches. If necessary, we can also compute them for the training set.

Algorithm *Classification_results*
Input: class_truth, instance_to_classify, number_classes, belief_decision_tree
Output: PCC, kappa, dist_crit

```
1.  begin
2.  (* Initialization *)
3.  n_cases ← size(instance_to_classify,1);
4.  truth_prediction ← zeros(number_classes, number_classes+1);
5.  dist_class ← zeros(n_cases,1);
6.
7.  (* Treatment *)
    for i ← 1 to n_cases
8.      do [class_result(i), class_result_probab(i,:)] ← Best_class(instance_to_clas-
           sify(i,:), number_classes, belief_decision_tree);
9.         truth_prediction(class_truth(i), class_result(i)) ← truth_prediction
           (class_truth(i), class_result(i)) + 1;
10.        if class_result(i) ≤ number_classes
11.           then dist_class(i) ← sum(power(class_result_probab(i,:),2));
12.                dist_class(i) ← dist_class(i) - power(class_result_probab (i,class_
                   truth(i)),2) + power(1-class_result_probab(i,class_truth(i)),2);
13.        end if
14. end for
15.
16. (* Computing PCC *)
17. ttrace ← trace(truth_prediction);
18. n_classed_cases ← n_cases - sum(truth_prediction(:,number_classes+1));
19. PCC ← ttrace / n_classed_cases;
20.
21. (* Computing kappa *)
22. ssum ← sum( sum (truth_prediction(:,1:number_classes)).*
23. sum(truth_prediction(:,1:number_classes)'))/n_classed_cases;
24. kappa ← (ttrace -ssum )/(n_classed_cases - ssum);
25.
26. (* Computing the distance criterion *)
27. dist_crit ← sum(dist_class)/n_classed_cases;
28. end
```

6.3 Simulation and results

6.3.1 Experimental setup

For checking the feasibility of our approaches regarding belief decision trees and judging their qualities, we have performed several tests and simulations.

We have made experiments on real databases that already exist, for which we have created an artificial bba for each one of its training instances in order to get the same structure of the training set needed in building belief decision trees. In other words, we will get a training set where each one of its objects is characterized by exact attribute values but its corresponding class is described by a basic belief assignment.

For simulation, we have used a modified Wisconsin breast cancer database inspired by the Wisconsin breast cancer database[3], but modified in order to satisfy the prerequisites of our methods: symbolic attributes and uncertain classes.

6.3.2 Performance indicators

The evaluation of classification efficiency is not obvious in an uncertain context. Several indicators may be used to assess performance.

We present two major kinds of results related to simulations. One is directly relative to the building procedure of the belief decision tree in both approaches. It consists in finding the complexity or the size of the induced tree characterized by the number of leaves (non empty leaves) and also the number of decision nodes (including test attributes). These values are not interesting unless the pruning of the tree is applied but they allow to give an idea about the induced trees.

The second kind of results concerns the classification phase in both approaches. The performance of a classification method can be measured in terms of classification accuracy on new instances. In such a case, three types of results will be presented:

[3] See http://www.ics.uci.edu/~mlearn/MLRepository.html

- **The Percent of Correct Classification:** PCC

  It consists in computing the percent of correct classification, denoted PCC, of the instances belonging to the testing set which are classified according to the induced tree.

  Instead of the PCC criterion, some researchers use the error rate, which is exactly equivalent to the PCC, i.e., the proportion of incorrect predictions that the decision tree makes on the testing set. So, it is equal to $1 - PCC$.

  In the framework of belief decision trees, leaves are characterized by bba's. For each testing instance, we determine its corresponding leaf. Once it is found, we look for the most probable class corresponding to this leaf using the pignistic probability computed from the leaf's bba. If more than one class have the highest probability, one of them is chosen randomly. The obtained class is considered as the class of the testing instance.

  Hence, the PCC relative to the whole testing set is computed by making comparison, for each testing instance, between its real class (known by us) and the class obtained by the tree.

$$PCC = \frac{\text{number of well classified instances}}{\text{number of classified instances}} \tag{6.1}$$

  where the number of well classified instances is computed as the sum of testing instances for which the class obtained by the belief decision tree (the most probable class) is the same as their real class. The denominator represents only the number of classified instances since some testing instances may not be classified by the tree (belong to an empty leaf).

  Obviously a PCC equals to 100% qualifies the belief decision tree as an excellent classifier, whereas the minimal value of PCC is 0% corresponds to *'a null'* classifier.

  In order to avoid confusion, in the different formulas using PCC, we consider its values between 0 and 1. However, for making interpretations and illustrating representations (tables and figures), we use the corresponding values in percent (between 0% and 100%).

- **The kappa criterion:** *kappa*

  The *kappa* degree of agreement is a quantitative measurement to assess the quality of the classification.

  Contrary to the PCC criterion which only focalizes on the equality of the real class and the one obtained by the tree, the *kappa* criterion tries to consider all the factors in the confusion matrix by penalizing the PCC by the percent of objects that could be correctly classified by chance (Rosner, 1995), (Siegel & Castellan, 1988).

$$kappa = \frac{PCC - \text{ proportion correctly classified by chance}}{1 - \text{ proportion correctly classified by chance}} \tag{6.2}$$

  Note that if the prediction agrees perfectly with the supervision, then the value of $kappa = 1$, whereas when $kappa = 0$ it means that this agreement is obtained by chance (Latinne, Debeir, & Decaestecker, 2000). When $kappa < 0$, it is the case where the agreement is worse than the one obtained by chance.

- **The distance criterion:** *dist_crit*

  We have developed a criterion allowing to take into account all the beliefs characterizing the leaf's bba. More exactly, we propose to compare the pignistic probability induced from the testing instance's bba and its real class.

  The distance criterion for a testing instance I_j belonging to a leaf L (its bba is $m^{\Theta}\{L\}$) is defined as follows:

$$\begin{aligned} dist\_crit(I_j) &= Distance(BetP^{\Theta}\{L\}, C(I_j)) \\ &= \sum_{i=1}^{n} (BetP^{\Theta}\{L\}(C_i) - \delta_{j,i})^2 \end{aligned} \tag{6.3}$$

  where the real class of the testing instance I_j is $C(I_j)$, and $\delta_{j,i} = 1$ if $C(I_j) = C_i$ and 0 otherwise.

This distance verifies the following property:

$$0 \leq dist\_crit(I_j) \leq 2 \tag{6.4}$$

Next, we have to compute the average total distance relative to all the classified testing instances denoted $dist\_crit$. So, we get:

$$dist\_crit = \frac{\sum_{I_j \in \text{ classified instances}} dist\_crit(I_j)}{\text{number of classified instances}} \tag{6.5}$$

All these criteria (PCC, $kappa$, and $dist\_crit$) measure **the accuracy** of the results induced from classification based on the belief decision trees.

Remarks:

1. These three criteria are generally applied to testing sets in order to judge the efficiency of the belief decision trees. Besides, we can apply them to training sets which may give us some ideas about the behaviour of the induced belief decision trees according to the two approaches: averaging and conjunctive.

2. We sometimes deal with another criterion for judging a tree. It is related to the understandability. In fact, one of the major advantages and characteristics of decision trees is their possibility to represent knowledge in an explicitly manner easily understood by users. However, there is a general agreement that deeper trees are less comprehensible.

   The understandability of a decision tree, and consequently a belief decision tree, remains difficult to quantify. The number of leaves and the number of nodes are generally used for this purpose (especially when pruning is applied).

6.3.3 Constructing uncertainty in training set

In classification problems, training sets are generally composed with training objects described by known attribute values and classes. Instances with partially known classes are usually eliminated from the databases, probably because the users were in pain to know what to do with these *'messy'* objects.

As described before, the belief decision trees are essentially built to handle uncertain classes where their uncertainty is represented by a bba given on the set of possible classes. So, the question is how will we construct these bba's?

These bba's are created artificially. They take into account three basic parameters:

- The real classes of the training instances.
- A probability P used to build the focal elements of each bba.
- The number N of simple support functions (ssf) composing the bba on instance's class.

The idea is to try to *'disturb'* the certainty of the classes of the training instances while keeping the true class in the focal elements of the created bba.

For each object, we randomly generate a subset θ of Θ such that the actual class of the object under consideration belongs to θ, and every of the other class belongs to θ with probability P. We then build a simple support function with θ its focal element, its weight being a uniformly distributed random number in [0,1]. We build N simple support functions and combine them conjunctively. The resulting bba is the bba describing our belief about the value of the actual class to which the object belongs.

In the simple case where the frame is binary, the focal set of each ssf is either the actual class or the whole frame. The former case occurs with a probability $1 - P$ and the latter with a probability P. Furthermore, the latter case results in a vacuous belief function.

So for a given object, the final bba on its class is the result of the combination of $n \in [0, N]$ ssf which all focus on the actual value. The number n is 0 with probability P^N, 1 with probability $(1 - P)P^{N-1}$, etc ... The final ssf is thus also a ssf which weight is the product of n uniformly distributed random number in [0,1].

The larger P, the smaller n, thus the smaller the weight given to the actual value, thus the larger the uncertainty on the class.

6.3.4 Defining the training and the testing set: Cross validation

The construction of a belief decision tree requires a training set for building it and a testing set for evaluating its performance. In our study, the data set is composed of objects described by the value of each one of its attributes followed by a bba on its classes, then its real class.

The data set is divided into 10 parts. Nine parts are used as the training set, the last is used as the testing set. The procedure is repeated ten times, each time using another part as the testing set. This method, called **a cross-validation** permits an unbiased estimation of the *PCC* (and the other criteria) even when the data set is small.

6.3.5 Simulation on the modified Wisconsin breast cancer database

Introduction

We have used the so-called *'modified Wisconsin breast cancer'* database (see Appendix B) which is composed of symbolic attributes and a unique class relative to each one of its objects. Then, we have introduced uncertainty in these classes (see Appendix C).

The characteristics of the modified Wisconsin breast cancer database are presented in Table 6.1:

Table 6.1: The modified Wisconsin breast cancer database parameters

Parameters	Value
Total number of instances	690
Number of training instances	621
Number of testing instances	69
Number of classes	2
Number of attributes	8
Number of values for each attribute	[2 3 14 8 2 2 2 3]

Results

Our simulations are divided into two parts:

- The first one tests the efficiency of Quinlan algorithm which is obtained by applying the averaging approach, of our belief decision tree, when there is no uncertainty in classes.
- The second one tests the feasibility of our belief decision tree algorithms in both averaging and conjunctive approaches. Several cases will be handled according to the value of the probability P (used in the *'the class destruction'*) which will vary from 0 to 0.9. The number of ssf will be the same for all these cases.

Quinlan case: Certainty case
Our averaging approach for building belief decision trees is equivalent to Quinlan algorithm when there is no uncertainty in the classes of the training instances. Our first experiment is to apply our belief decision tree algorithm to such a situation.

Table 6.2: Quinlan case: Decision nodes and leaves

Parameters	Value
Mean number of decision nodes	96.8
Mean number of non empty leaves	165.5

Without applying a mechanism of pruning, the belief decision trees induced from the averaging approach in the case of total certainty present on the average 96.8 decision nodes and 165.5 leaves.

Next, Table 6.3 presents the values of PCC, *kappa* and *dist_crit*. We also present these data for the training set. We mention them just to show that the belief decision tree classifies correctly its training instances.

Table 6.3: Quinlan case: Classification results

Parameters	Testing set	Training set
Mean (%) PCC	83.6	94.3
Mean *kappa*	0.66	0.89
Mean *dist_crit*	0.28	0.07

As noted in Table 6.3, the mean values of PCC and *kappa* in the testing set are high (respectively 83.6% and 0.66). These results are confirmed by the value of the *dist_crit* which is small (0.28). The values given by the training set shows the same behavior as with the testing set.

Uncertainty case
We have applied our belief decision tree algorithm to both averaging and conjunctive approaches. The cross validation is applied for different values of the probability P which varies from 0 to 0.9.

Table 6.4 presents the mean number of non empty leaves (N. L.) and decision nodes (N. DN.) relative to each approach: averaging (ave) and conjunctive (conj).

Tables 6.5 to 6.7 present, respectively, the PCC, the *kappa* and the *dist_crit* in the testing set (TS) and training set (TR) when using the averaging (ave) and conjunctive (conj) approaches.

1. *Decision nodes and leaves:*

Table 6.4: Decision nodes and leaves

P	N. DN. ave	N. DN. conj	N. L. ave	N. L. conj
0	259.3	274.2	337.2	336.2
0.1	253.6	277.0	316.2	316.1
0.2	243.0	264.2	295.7	295.7
0.3	229.7	240.1	269.0	268.9
0.4	234.9	246.8	246.0	245.9
0.5	208.5	217.5	222.4	222.4
0.6	207.8	225.0	225.0	225.0
0.7	183.0	186.9	146.1	146.1
0.8	160.6	170.9	109.8	109.8
0.9	114.1	114.0	63.4	63.4
Mean	*209.5*	*221.7*	*220.2*	*220.1*

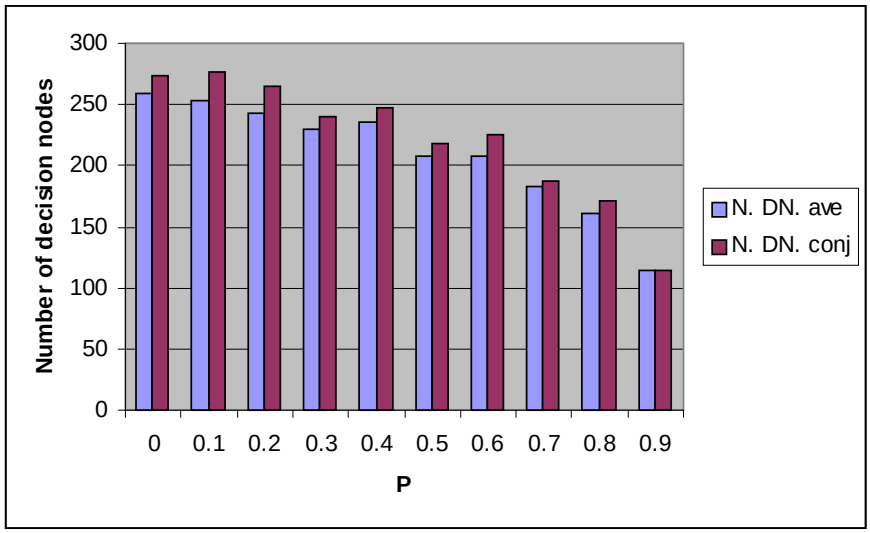

Figure 6.1: Decision nodes in averaging and conjunctive approaches

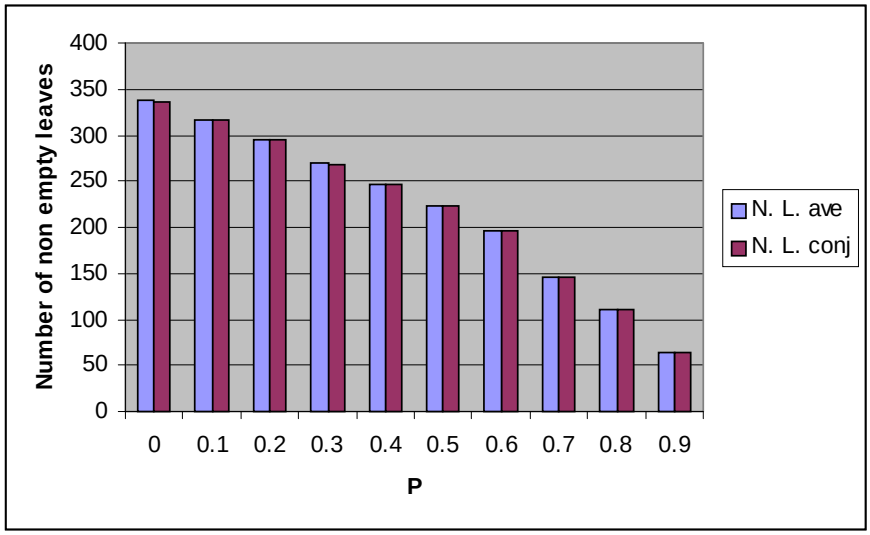

Figure 6.2: Leaves in averaging and conjunctive approaches

As shown in Table 6.4, Figure 6.1 and Figure 6.2, the mean numbers of decision nodes in both averaging and conjunctive approaches are respectively 209.5 and 221.7 and those of leaves in both averaging and conjunctive approaches are respectively 220.2 and 220.1.

Note that these numbers are approximately the same for either the averaging or conjunctive approach especially for the number of leaves. However, for the decision nodes, their mean number is a little bit larger within the conjunctive approach than with the averaging one.

It is interesting to notice that the larger the value of the probability P, in other words the larger the uncertainty in the training instances' bba's, the smaller the number of leaves and decision nodes. For example for the probability $P = 0.9$, the mean number of leaves for both approaches is 63.4 and the one relative to decision nodes for averaging and conjunctive approaches is respectively 114.1 and 114.

This may be explained by the fact that increasing uncertainty augments the chance for getting negative values for the gain ratios and the diff ratios for the remaining attributes, which leads to the declaration of leaves after only few levels, and consequently the diminution of number of leaves and decision nodes. In that case, nodes are declared leaves earlier during the tree construction.

2. *PCC criterion:*

Table 6.5: Classification results: *PCC*

P	**PCC TS ave** (%)	**PCC TS conj** (%)	**PCC TR ave** (%)	**PCC TR conj** (%)
0	84.4	83.3	94.2	94.2
0.1	83.1	82.1	94.1	94.0
0.2	84.8	83.9	94.0	94.0
0.3	86.9	86.3	93.6	93.6
0.4	83.9	83.3	92.4	92.2
0.5	87.0	85.9	93.0	93.1
0.6	83.0	82.5	91.8	91.8
0.7	85.6	86.9	92.9	92.9
0.8	81.3	81.5	90.9	90.9
0.9	80.4	81.2	88.2	88.2
Mean (%)	*84.0*	*83.7*	*92.5*	*92.5*

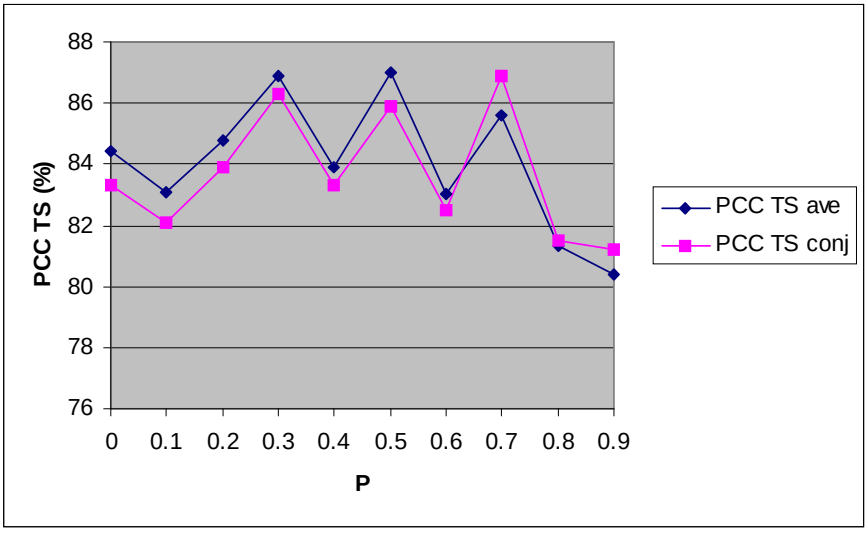

Figure 6.3: *PCC* of the testing set in averaging and conjunctive approaches

The mean PCC relative to the testing set for both approaches is equal to 84.0% for the averaging approach and 83.7% for the conjunctive one. The difference between the two approaches in terms of PCC is almost non existing.

In fact, we notice that in general the PCC of the averaging approach is just a little bit better than the conjunctive one when the probability P is less or equal than 0.6, whereas from P equals to 0.7 until 0.9, the PCC of the conjunctive approach becomes a little bit better than the one of the averaging approach.

It is interesting to note that for the different values of the probability P, the PCC for both approaches remains high (between 80.4% and 87.0% for the averaging approach and between 81.2% and 86.9% for the conjunctive approach).

Furthermore, we have to note that the minimal value of PCC for both approaches remains also high even when P is high, i.e, when it is equal to 0.8 and 0.9.

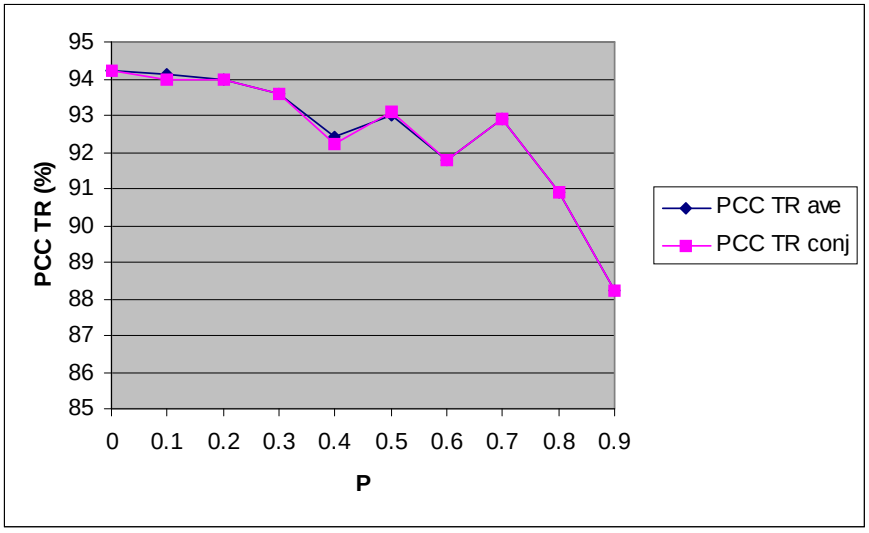

Figure 6.4: *PCC* of the training set in averaging and conjunctive approaches

As mentioned before, the PCC (as well as the *kappa* and *dist_crit* criteria) relative to the training set has not the same importance as the one relative to the testing set.

Nevertheless, it may confirm the results produced by the testing set. With the modified Wisconsin breast cancer database, the mean PCC relative to the training set is 92.5% for the averaging approach and also for the conjunctive approach.

As with the testing set, the values of PCC relative to the training set are high for all the values of P, and its minimal values are for the values of P equal to 0.8 and 0.9, but these latter PCC's remain high (90.9% and 88.2% for both approaches).

3. *kappa criterion:*

Table 6.6: Classification results: *kappa*

P	***kappa* TS ave**	***kappa* TS conj**	***kappa* TR ave**	***kappa* TR conj**
0	0.68	0.66	0.88	0.88
0.1	0.66	0.64	0.88	0.88
0.2	0.69	0.67	0.88	0.88
0.3	0.73	0.72	0.87	0.87
0.4	0.66	0.66	0.85	0.84
0.5	0.74	0.71	0.86	0.86
0.6	0.65	0.64	0.83	0.83
0.7	0.7	0.72	0.86	0.86
0.8	0.61	0.62	0.82	0.81
0.9	0.60	0.62	0.77	0.77
Mean	*0.67*	*0.66*	*0.85*	*0.85*

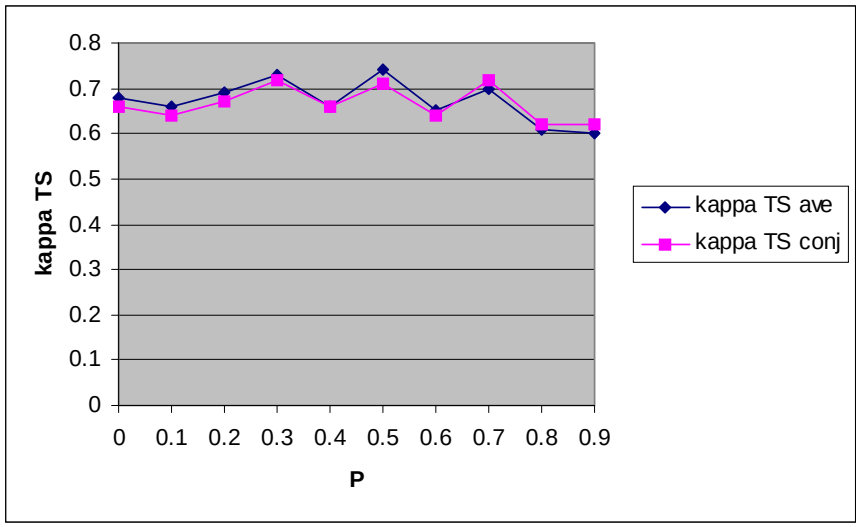

Figure 6.5: *kappa* of the testing set in averaging and conjunctive approaches

The second criterion for judging classification is the *kappa* one. In the testing set, the mean kappa value is almost the same for both approaches (0.67 and 0.66 for respectively the averaging approach and the conjunctive one).

Using this criterion, the values of *kappa* in both approaches for every value of P are high, and their minimal values, as with the PCC, are for P equals to 0.8 and 0.9, respectively 0.61 and 0.60 for the averaging approach and 0.62 (for $P = 0.8$ and $P = 0.9$) for the conjunctive approach.

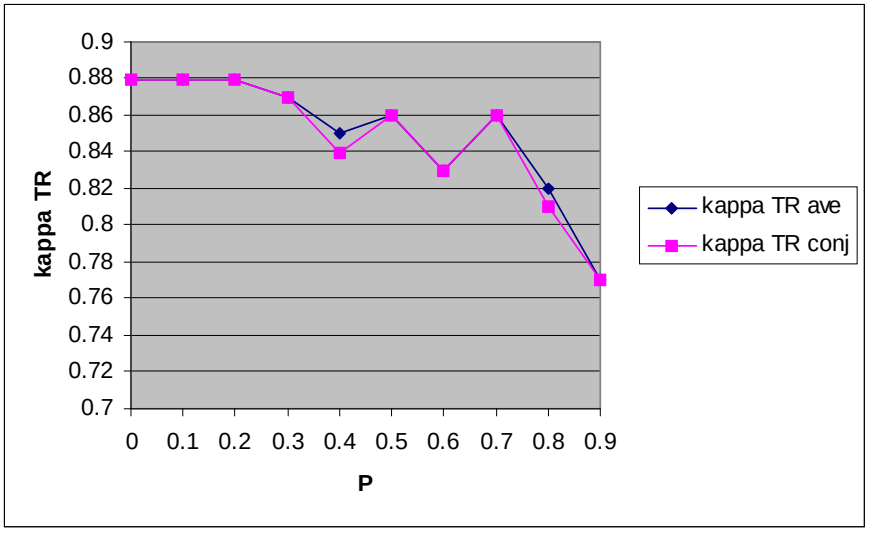

Figure 6.6: *kappa* of the training set in averaging and conjunctive approaches

The graph of the *kappa* criterion relative to the training set (see Figure 6.6) shows that for the different values of the probability P, the *kappa* values for both approaches are quasi identical.

Therefore and as noted in Table 6.6, the mean kappa value in the testing set for both the averaging and the conjunctive approach is the same and equals to 0.85 .

4. *Distance criterion (dist_crit):* As explained in Section 6.3.2, according to the distance criterion, the smaller the *dist_crit* value, the better the classifier.

Table 6.7: Classification results: *dist_crit*

P	***dist_crit* TS ave**	***dist_crit TS* conj**	***dist_crit TR* ave**	***dist_crit TR* conj**
0	0.38	0.31	0.32	0.22
0.1	0.38	0.33	0.33	0.22
0.2	0.38	0.31	0.34	0.23
0.3	0.38	0.31	0.35	0.25
0.4	0.40	0.33	0.37	0.26
0.5	0.39	0.32	0.37	0.26
0.6	0.42	0.35	0.39	0.29
0.7	0.42	0.33	0.40	0.28
0.8	0.44	0.38	0.41	0.32
0.9	0.45	0.39	0.42	0.36
Mean	*0.40*	*0.34*	*0.37*	*0.27*

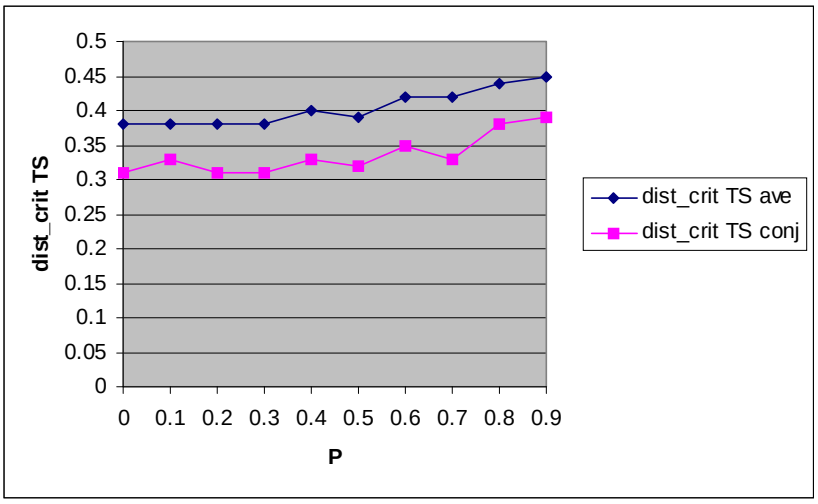

Figure 6.7: *dist_crit* of the testing set in averaging and conjunctive approaches

Contrary to the PCC and the *kappa* criteria, the mean value of the distance criterion (*dist_crit*) for the testing set obtained with the averaging approach is greater (0.40) than the one obtained with the conjunctive approach (0.34).

The difference is not enormous but it denotes that the belief decision tree obtained with the conjunctive approach presents a *'better'* classifier than the one obtained with the averaging approach and this holds for every value of P.

In such a context, the conjunctive approach seems better than the averaging one. This can be explained by the fact that the conjunctive rule of combination produces bba's that give more support to the best supported hypothesis than the averaging approach.

The conjunctive approach is *'bolder'* than the averaging approach, and as far as they have about the same and high PCC, the distance can be expected to be smaller with the conjunctive approach.

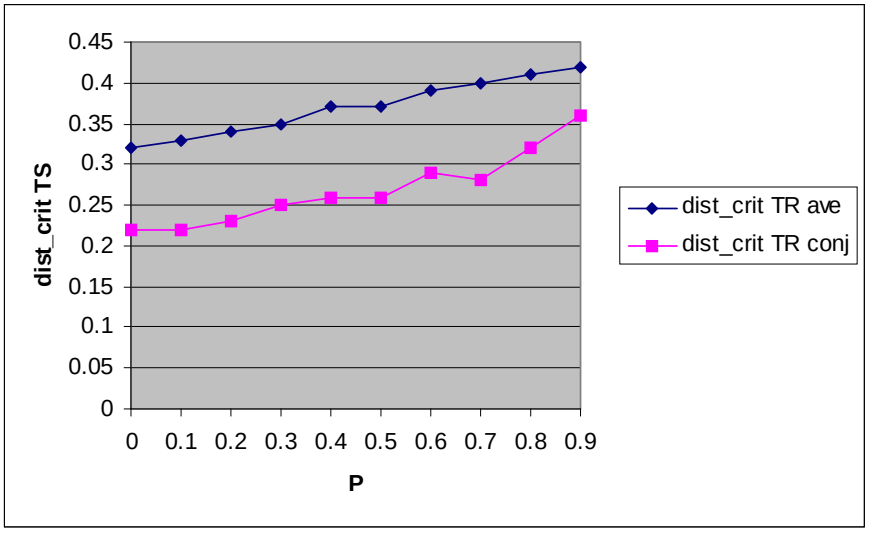

Figure 6.8: *dist_crit* of the training set in averaging and conjunctive approaches

The results of the *dist_crit* obtained with the training set can be interpreted as the same manner as those of the testing set.

Finally, let's sum up the different results of simulation in the following table:

Table 6.8: Summary of the results obtained from the testing sets

	$PCC(\%)$	$kappa$	$dist\_crit$
Quinlan	84.0	0.66	0.28
Averaging	80.4 - 87.0	0.60 - 0.74	0.38 - 0.45
Conjunctive	81.2 - 86.9	0.62 - 0.72	0.31 - 0.39

6.4 Conclusion

In this chapter, we have outlined our proposed method to build belief decision trees for both the averaging and the conjunctive approaches. Thus, we have basically detailed the major variables and also the main programs.

Next, we have shown results obtained from experiments and simulations of our developed approaches: the averaging and the conjunctive approaches.

The next part will deal with the improvements than can be added to belief decision trees. It focalizes on uncertainty pervaded in the training set. Hence beliefs on classes of training objects will be assessed. Besides, uncertainty in attributes in the training set will be handled.

Part III

Improvements for Handling Uncertainty in the Training Set

In the third part of this thesis, we present some improvements useful for handling uncertainty in the training set. Such set is the basis leading to the construction of a belief decision tree.

This part is composed of two chapters:

- Chapter 7 presents methods to assess objectively the beliefs on the classes of the objects in the training set. Knowledge about the classes are obtained from one or several experts. The idea consists in assessing the reliability of each expert.

  Two methods are proposed:

  - The first method produces the reliability of an expert when considered alone.
  - The second method considers a set of experts, and weights each of them so that together they produce the best predictor.

- Chapter 8 handles the case where there is some uncertainty not only in the classes of the objects of the training set, but also in their attributes.

  We redefine the basic parameters that must be adapted when constructing belief decision trees in such context. They are the attribute selection measure, the partitioning strategy, the stopping criteria and the leaf structure.

Chapter 7

Assessing beliefs in the training set

7.1 Introduction

In belief decision trees, knowledge about the actual class of each training instance is described by a basic belief assignment (bba) defined on the set of possible classes. Each training instance's bba, generally given by an expert (or several experts), represents the opinions-beliefs of this expert (or these experts) about the actual value of the class for each object in the training set.

Experts are usually not fully reliable and do not have the same level of expertise. Therefore, we must assess the reliability of each expert before using its opinions (Cooke, 1991). In the TBM, the partial reliability of an expert is handled by the discounting operation (see Section 1.6).

This chapter addresses basically the problem of assessing the discounting factors to be applied to the beliefs on training instances' classes generated by experts in the context of belief decision trees.

In the first part of this chapter, we present the role of the experts and why they may present different levels of expertise. Next, two major cases will be treated:

1. First, we develop a method for the evaluation of the reliability of an expert when considered alone. The method is based on finding the discounting factor that minimizes the distance between the pignistic probabilities computed from the discounted beliefs and the actual value of the class.

2. Next, we develop a method for assessing the reliability of several experts that are supposed to work together and their reports are aggregated. The discounting factors are computed on the basis of minimizing the distance between the pignistic probabilities computed from the combined discounted belief functions and the actual value of the class.

Particular cases related to experts will be treated by the two proposed methods.

All these results are summarized in (Elouedi, Mellouli, & Smets, 2001c, 2002).

7.2 Role of experts

The value of the classes of the instances in the training set is an essential component for building belief decision trees. They are usually assessed by one or several experts. These assessments can be represented by bba's.

In this thesis, an expert means anyone with special knowledge about an uncertain event related to the treated problem. So, it can be a human expert, a sensor, or any information source.

Several cases regarding the experts defining our structure of the training set can be analyzed:

1. *One expert for all the training objects:*
   There is only one expert which will give his belief about the actual class of each instance in the training set. Hence, all bba's are produced by only one expert.

2. *Several experts for each training object:*
   For each training object, we get the opinion of several experts on its actual class. Thus, each training object's bba will be equal to the combination of the different experts' bba's (on this object) obtained by using the conjunctive rule of combination (see Section 1.4).

3. *Several experts for different training objects:*
   Each expert provides his belief about the actual class of some training instances, and different experts do not have to consider the same instances.

In this chapter, we suppose the attributes are fixed and their values are known with full precision and certainty.

7.3 Levels of expertise

Experts differ in their levels of expertise, some of them are more reliable than others due to their better knowledge, training, experience, intelligence, ... To express their opinions, experts may use different background, methodology and even knowledge. Hence, the necessity to consider the experts' reliability when receiving their opinions, and consequently their judgments must be appropriately *'discounted'*.

Similarly, sensors do not have the same degree of reliability. This may be due not only to the same reasons as mentioned for experts, but also to other factors more specific to sensors. For instance, the measurements can differ in their nature, completeness, precision and certainty. Furthermore, working environment can also affect the reliability of sensors since some of them could be better adapted to the conditions encountered in the considered environment than other sensors.

This shows that many factors can induce differences between expert reliabilities. Thus, it is necessary to consider the reliability factor to weight the bba's.

7.4 Evaluation of the expert's reliability

7.4.1 Introduction

Usually the discounting factor applicable to an expert is unknown and users would like to find objective ways to assess it. The user's opinion might be useful, but in practice, users prefer to base their opinions on objective facts, not on *'hunches'* and *'whims'*.

In fact, finding an *'automatic'* method to assess the expert's reliability relative to a given problem requires information regarding the judgments given previously by the expert concerning *'past'* events (related to the same kind of problem) for which the truth is known by us and not by the expert. Then, a comparison between the truth and the expert's judgments allows to derive the reliability of the expert.

In practice, one domain where we can get this kind of information is represented by classification problems where objects will be presented to the expert in order to identify their classes, and the expert's report is compared to the reality.

In such problems, we can easily get the expert's opinions on the classes to which an object belongs to, a class otherwise well known by us. For example, we can use an old training set relative to the same problem where training instances' classes are known with certainty.

Several classification methods have been developed using belief function basics (Denœux, 1995, 1997, 2000), (Zouhal, 1997; Zouhal & Denœux, 1998), (De Smet, 1998), (Smets, 1998a) (Elouedi et al., 2000a, 2001a), (Delmotte & Smets, 2002a, 2002b), ... In the following subsections, we focus on classification problems especially those based on the belief decision trees. The method can be easily adapted to other domains, the underlying schema being quite general.

7.4.2 The framework

Our objective is to get a training set reflecting as much as possible the truth. Hence, in order to find the reliability factor to assign to the expert, the idea is to present to the considered expert a kind of training set $\mathcal{T}$ (used for example in the past) presenting the same characteristics (attributes and classes) as the one that will be used for building the belief decision tree. This set $\mathcal{T}$ contains objects such that their classes are known by us and not by the expert.

So, let $\mathcal{T}$ be a set composed of s objects denoted by I_j $(j = 1, 2, ..., s)$. Each object has to belong to one of the possible classes relative to the given problem. The set of possible classes is defined by $\Theta = \{C_1, C_2, ..., C_n\}$.

For each object I_j, we know its class, denoted $C(I_j)$ with $C(I_j) \in \Theta$, and the expert produces a bba, denoted $m^{\Theta}\{I_j\}$ on Θ, that represents its opinion on the actual value of $C(I_j)$ taking into account the different attribute values of I_j.

We emphasize that the expert does know not the real classes of the objects.

7.4.3 Evaluation of the discounting factor

Our first method for assessing the discounting factor considers one expert alone. Finding the expert's discounting factor is achieved by comparing the bba's $m^{\Theta}\{I_j\}$ produced by the expert about the class of each of the s objects in $\mathcal{T}$and their real classes.

Assume the discounting factor α relative to this expert is known, then its bba should be discounted taking into account the discounting factor α. So, we get the discounted bba $m^{\Theta,\alpha}\{I_j\}$ by using Equations (1.53) and (1.54).

In order to make a decision about the class to which the object belongs to, we apply the pignistic transformation to the bba $m^{\Theta,\alpha}\{I_j\}$. So, we get the pignistic probability, denoted $BetP^{\Theta,\alpha}\{I_j\}$ presenting the probability of the object I_j to belong to each singular class. This probability function has then to be compared with the actual class $C(I_j)$ of the object I_j.

Let the indicator function δ be defined as: $\delta_{j,i} = 1$ if $C(I_j) = C_i$ and 0 otherwise (δ is already defined in Chapter 6 for the *dist_crit*). The idea is to compute the distance between the pignistic probability computed from the discounted expert's bba and the indicator function δ. This distance is used as a measure of the reliability of the expert for what concerns the object I_j and it is defined as follows:

$$\begin{aligned} Distance(I_j, \alpha) &= Distance(BetP^{\Theta,\alpha}\{I_j\}, C(I_j)) \\ &= \sum_{i=1}^{n} (BetP^{\Theta,\alpha}\{I_j\}(C_i) - \delta_{j,i})^2 \end{aligned} \tag{7.1}$$

This distance verifies the following property:

$$0 \leq Distance(I_j, \alpha) \leq 2 \tag{7.2}$$

When the expert is correct with certainty 1 and $\alpha = 0$ (no discounting), the corresponding distance $Distance(I_j, 0) = 0$. In the case where the expert supports a wrong class with certainty and $\alpha = 0$ (no discounting), $Distance(I_j, 0) = 2$.

Judging the reliability of an expert by taking into account only one object is not sufficient, so the next step consists in computing the distance between the pignistic probability of each object I_j in $\mathcal{T}$ and its corresponding δ. The sum of these distances, denoted $TotalDistance$, reflects well the overall expert's reliability.

$TotalDistance$ is defined as follows:

$$TotalDistance = \sum_{j=1}^{s} Distance(I_j, \alpha)$$
$$= \sum_{j=1}^{s} \sum_{i=1}^{n} (BetP^{\Theta,\alpha}\{I_j\}(C_i) - \delta_{j,i})^2 \quad (7.3)$$

We then estimate the discounting factor as the coefficient $\alpha \in [0, 1]$ that minimizes $TotalDistance$. In other words, α that makes the values of $BetP^{\Theta,\alpha}\{I_j\}$ as close as possible to the truth represented by $\delta j, i$.

7.4.4 The case of normalized belief functions

In the special but common case, beliefs produced by experts are presented by normalized bba's, i.e., $m(\emptyset) = 0$. In such a case, it is possible to explicit the value of α that minimizes $TotalDistance$.

So, let $BetP^{\Theta}\{I_j\}$ be the pignistic probability function computed from $m^{\Theta}\{I_j\}$, before discounting. The explicited value of the factor α is defined in the following theorem:

Theorem 7.1 *Let a set of normalized bba's $m^{\Theta}\{I_j\}$ defined on the set of classes $\Theta = \{C_1, C_2, \ldots, C_n\}$ for objects $I_j, j = 1, \ldots, s$. Let the indicator function $\delta_{j,i} = 1$ if $C(I_j) = C_i$ ($C(I_j)$ is the actual class of the object I_j) and 0 otherwise. Let $BetP^{\Theta,\alpha}\{I_j\}$ be the pignistic probability function computed from the discounted bba $m^{\Theta,\alpha}\{I_j\}$. The discounting factor α that minimizes:*

$$TotalDistance = \sum_{j=1}^{s} \sum_{i=1}^{n} (BetP^{\Theta,\alpha}\{I_j\}(C_i) - \delta_{j,i})^2$$

is given by:

$$\alpha = \min(1, \max(0, \frac{\sum_{j=1}^{s} \sum_{i=1}^{n} (\delta_{j,i} - BetP^{\Theta}\{I_j\}(C_i)) BetP^{\Theta}\{I_j\}(C_i)}{s/n - \sum_{j=1}^{s} \sum_{i=1}^{n} BetP^{\Theta}\{I_j\}(C_i)^2})) \quad (7.4)$$

Proof. Given the bba $m^{\Theta}\{I_j\}$, its α-discounted bba is:

$$m^{\Theta,\alpha}\{I_j\}(C) = \begin{cases} (1-\alpha) m^{\Theta}\{I_j\}(C) & \text{if } C \subset \Theta \\ (1-\alpha) m^{\Theta}\{I_j\}(\Theta) + \alpha & \text{if } C = \Theta \end{cases}$$

Hence, the pignistic probability $BetP^{\Theta,\alpha}\{I_j\}$ computed from the discounted bba $m^{\Theta,\alpha}\{I_j\}$ can be defined as a function of the pignistic probability $BetP^{\Theta}\{I_j\}$ which is computed directly from the initial bba $m^{\Theta}\{I_j\}$. We get:

$$\begin{aligned} BetP^{\Theta}\{I_j\}(C_i) &= \sum_{C_i \in C} \frac{m^{\Theta}\{I_j\}(C)}{|C|} \\ BetP^{\Theta,\alpha}\{I_j\}(C_i) &= \sum_{C_i \in C} \frac{m^{\Theta,\alpha}\{I_j\}(C)}{|C|} \\ &= \sum_{C_i \in C} \frac{(1-\alpha)m^{\Theta}\{I_j\}(C)}{|C|} + \alpha/n \\ &= (1-\alpha)BetP^{\Theta}\{I_j\}(C) + \alpha/n \end{aligned}$$

Let $P_{ij} = BetP^{\Theta}\{I_j\}(C_i)$. The value of $TotalDistance$ to be minimized becomes:

$$\begin{aligned} TotalDistance &= \sum_{j=1}^{s} \sum_{i=1}^{n} (BetP^{\Theta,\alpha}\{I_j\}(C_i) - \delta_{j,i})^2 \\ &= \sum_{j=1}^{s} \sum_{i=1}^{n} ((1-\alpha)P_{ij} + \alpha/n - \delta_{j,i})^2 \end{aligned}$$

Its extremum is reached when its derivative is null, hence when:

$$\begin{aligned} 0 &= \frac{d\ TotalDistance}{d\alpha} \\ &= 2\sum_{j,i} ((1-\alpha)P_{ij} + \alpha/n - \delta_{j,i})(-P_{ij} + 1/n) \\ &\propto \sum_{j,i} -(1-\alpha)P_{ij}^2 - \alpha s/n + \sum_{j,i} \delta_{j,i}P_{ij} + (1-\alpha)s/n + \alpha s/n - s/n \\ &= \sum_{j,i} -(1-\alpha)P_{ij}^2 - \alpha s/n + \sum_{j,i} \delta_{j,i}P_{ij} \end{aligned}$$

Thus,

$$\alpha = \frac{\sum_{j,i}(\delta_{j,i} - P_{ij})P_{ij}}{s/n - \sum_{j,i} P_{ij}^2} \tag{7.5}$$

In order that $\alpha \in [0,1]$, we get the limit constraints. □

Hence, in the case of normalized bba's, we may easily find the discounting factor of an expert. The results can be used to discount the future opinions of this expert (on the real training set) or to order several experts according to their reliabilities. This latter case is useful when we have to choose only one expert for giving his opinions concerning the classes of the training instances that will be used to build the belief decision tree.

7.4.5 The simplified equivalent

A discounting factor of 0.7 is better than one of 0.8, but what represents intuitively an $\alpha = 0.7$ value. To get a hunch of what a particular α represents, we imagine a highly simplified schema with only two objects and two classes. Both objects belong to the class C_1.

The first object is classified correctly with certainty 1. The second object is classified as belonging to the class C_1 with a probability π and to the class C_2 with a probability $1 - \pi$. Given a value α, we can determine the value of π that would produce the same discounting factor by the above procedure. The value π corresponds to a probability, and is thus easier to interpret than the α discounting factor.

The link between π and α is given by:

$$\pi = \frac{3 - 2\alpha - \sqrt{1 + 4\alpha - 4\alpha^2}}{4 - 4\alpha}, \text{ for } \alpha \in [0, 1] \tag{7.6}$$

Proof. Let's remind the Equation (7.5):

$$\alpha = \frac{\sum_{j,i}(\delta_{j,i} - P_{ij})P_{ij}}{s/n - \sum_{j,i} P_{ij}^2}$$

Taking into account the probabilities given by the expert regarding the classes of the two objects, we get:

$$\begin{aligned} \alpha &= \frac{(1-\pi)\pi + (\pi - 1)(1-\pi)}{2/2 - 1 - \pi^2 - (1-\pi)^2} \\ &= \frac{-2\pi^2 + 3\pi - 1}{-2\pi^2 + 2\pi - 1} \end{aligned}$$

So, expressing the probability π in terms of α is given via the following steps:

$$(-2\pi^2 + 2\pi - 1)\alpha = -2\pi^2 - 3\pi - 1$$

$$(2 - 2\alpha)\pi^2 + (2\alpha - 3)\pi + 1 - \alpha = 0$$

So, the discriminative Δ is equal to:

$$\begin{aligned}\Delta &= (2\alpha - 3)^2 - 4(2 - 2\alpha)(1 - \alpha) \\ &= -4\alpha^2 + 4\alpha + 1\end{aligned}$$

For $\alpha \in [0, 1]$, the value Δ is strictly positive. Since the value π has to belong to $[0, 1]$, then only one solution will be retained which is:

$$\pi = \frac{3 - 2\alpha - \sqrt{1 + 4\alpha - 4\alpha^2}}{4 - 4\alpha}, \text{ for } \alpha \in [0, 1]$$

□

Note that π varies from 0 to .5 when α varies from 1 to 0.

Example 7.1 *Our example is already presented in previous chapters (see Example 2.1). It concerns the classification of clients asking for loans. As described before, three attributes characterize these clients namely the income of the client, the value of his property over the amount of the asked loan, and if he has or not unpaid credits.*

Let's remind the possible client's classes defined by the bank for whom asking for loans:

- C_1 *including good clients, i.e., reliable clients, for whom the bank accepts to give the whole loan.*
- C_2 *to which belongs moderate clients, for whom the bank accepts to give a part of the loan.*
- C_3 *regrouping 'bad' clients for whom the bank refuses to give the loan.*

$\Theta = \{C_1, C_2, C_3\}$

Assume two experts E_1 and E_2 are applied to present their opinions concerning the class of some clients. The objective is to find the reliability factor to assign to each expert in order to choose the more reliable one to present his opinions on the real training set that will be used to build the belief decision tree.

The judgments of experts on the classes are expressed by the bba's detailed in Table 7.1 where we consider a set $\mathcal{T}$ containing 4 clients which classes are known by us but not by the two experts.

At the first row of the table, we have the actual class of each client, then the bba's produced by the two experts about the classes of the four objects.

Table 7.1: The bba's produced by two experts about the classes to which belong four clients

Truth	$\mathbf{C}_1$	$\mathbf{C}_2$	$\mathbf{C}_1$	$\mathbf{C}_3$
E_1	I_1	I_2	I_3	I_4
$\emptyset$	0	0	0	0
C_1	0	0	0	0
C_2	0	0.5	0.4	0
C_3	0.5	0.2	0	0
$C_1 \cup C_2$	0	0	0	0
$C_1 \cup C_3$	0	0	0.6	0.6
$C_2 \cup C_3$	0.3	0	0	0.4
$C_1 \cup C_2 \cup C_3$	0.2	0.3	0	0
E_2	I_1	I_2	I_3	I_4
$\emptyset$	0	0	0	0
C_1	0	0.3	0.2	0
C_2	0	0	0	0
C_3	0	0	0	0
$C_1 \cup C_2$	0.7	0.4	0	0
$C_1 \cup C_3$	0	0	0	0
$C_2 \cup C_3$	0	0	0.6	1
$C_1 \cup C_2 \cup C_3$	0.3	0.3	0.2	0

Let α_1 be the discounting factor applicable to expert E_1. We discount the bba's produced by E_1 relative to the clients I_1, I_2, I_3 and I_4. We get:

$$\begin{aligned} m_{E_1}^{\Theta,\alpha_1}\{I_1\}(C_3) &= 0.5(1-\alpha_1) \\ m_{E_1}^{\Theta,\alpha_1}\{I_1\}(C_2 \cup C_3) &= 0.3(1-\alpha_1) \\ m_{E_1}^{\Theta,\alpha_1}\{I_1\}(\Theta) &= 0.2 + 0.8\alpha_1 \end{aligned}$$

$$
\begin{aligned}
m_{E_1}^{\Theta,\alpha_1}\{I_2\}(C_2) &= 0.5(1-\alpha_1)\\
m_{E_1}^{\Theta,\alpha_1}\{I_2\}(C_3) &= 0.2(1-\alpha_1)\\
m_{E_1}^{\Theta,\alpha_1}\{I_2\}(\Theta) &= 0.3+0.7\alpha_1
\end{aligned}
$$

$$
\begin{aligned}
m_{E_1}^{\Theta,\alpha_1}\{I_3\}(C_2) &= 0.4(1-\alpha_1)\\
m_{E_1}^{\Theta,\alpha_1}\{i_3\}(C_1\cup C_3) &= 0.6(1-\alpha_1)\\
m_{E_1}^{\Theta,\alpha_1}\{I_3\}(\Theta) &= \alpha_1
\end{aligned}
$$

$$
\begin{aligned}
m_{E_1}^{\Theta,\alpha_1}\{I_4\}(C_1\cup C_3) &= 0.6(1-\alpha_1)\\
m_{E_1}^{\Theta,\alpha_1}\{I_4\}(C_2\cup C_3) &= 0.4(1-\alpha_1)\\
m_{E_1}^{\Theta,\alpha_1}\{I_4\}(\Theta) &= \alpha_1
\end{aligned}
$$

The corresponding BetP's are presented in Table 7.2:

Table 7.2: *BetP*'s computed from the four discounted bba's produced by E_1

E_1	I_1	I_2	I_3	I_4
C_1	$0.06+0.26\alpha_1$	$0.1+0.24\alpha_1$	$0.3+0.03\alpha_1$	$0.3+0.03\alpha_1$
C_2	$0.22+0.12\alpha_1$	$0.6-0.27\alpha_1$	$0.4-0.06\alpha_1$	$0.2+0.13\alpha_1$
C_3	$0.72-0.38\alpha_1$	$0.3+0.03\alpha_1$	$0.3+0.03\alpha_1$	$0.5-0.16\alpha_1$

Using the different values of BetP's, the whole distance relative to the expert E_1 will be equal to:

$$TotalDistance = \sum_{j=1}^{4}\sum_{i=1}^{3}(BetP^{\Theta,\alpha_1}\{I_j\}(C_i)-\delta_{j,i})^2$$

Hence,

$$TotalDistance = 0.41\alpha_1^2 - 0.54\alpha_1 + 2.83$$

Minimizing TotalDistance under the constraint $0 \leq \alpha_1 \leq 1$ gives as a result $\alpha_1 = 0.66$.

Therefore, the discounting factor to be given to the expert E_1 by taking into account its opinions on the classes of the objects I_j, $j = 1, 2, 3, 4$, is equal to 0.66.

Applying the same procedure for the reports produced by the expert E_2, we get the discounting factor $\alpha_2 = 0.52$.

Thus, the expert E_2 is (a little) better than the expert E_1, in other words, it is (a little) more reliable than E_1.

Just to get an idea about what represents the two discounting factors (see Section 7.4.5), their equivalent in the highly simplified schema of 2 objects produce π values of 0.22 and 0.28, respectively, what can be understood as 'the experts are really not good, and the second is just a little better than the first'. This is indeed what the data also show.

So, if we had to choose one from the two experts it will be the second one E_2.

7.4.6 Categorical imprecise experts

Experts are qualified as categorical when there is only one focal element, so there is a $\tau_j \subseteq \Theta$ with $m^{\Theta}\{I_j\}(\tau_j) = 1$ for all objects I_j belonging to $\mathcal{T}$. Categorical experts are precise if $|\tau_j| = 1$ for all I_j.

We say that an expert is blind if the class τ_j is selected without regard of the object I_j. It means that it allocates the object to a class at random, using the same distribution of every object. It is unbiased if the distribution gives equal probability for each class. An expert is worth using if its results are better than those that a blind expert would achieve.

We consider the case of an imprecise but categorical expert i.e., an expert such that for all object I_j, there is a $\tau_j \subseteq \Theta$ with $m^{\Theta}\{I_j\}(\tau_j) = 1$ and $|\tau_j| \geq 1$, with a strict inequality for some objects. This is probably one of the most frequent cases to be encountered in real life when trying to assess an expert reliability. Thanks to the simplified structure of the data, it is possible to derive the value of the discounting factor to assign to the expert.

Let α be the discounting factor to determine. The pignistic probabilities after discounting are given by:

$$BetP^{\Theta,\alpha}\{I_j\}(C_i) = \begin{cases} \frac{\alpha}{n} + \frac{1-\alpha}{r} & \forall C_i \in \tau_j \\ \frac{\alpha}{n} & \forall C_i \notin \tau_j \end{cases}$$

where $r = |\tau_j|$ and $n = |\Theta|$.

The distance denoted $Distance(I_j, \alpha)$ becomes:

- If $C(I_j) \in \tau_j$ (possibly exact classification):

$$Distance(I_j, \alpha) = -\frac{1-\alpha^2}{r} + 1 - \frac{\alpha^2}{n} \tag{7.7}$$

Proof.

$$\begin{aligned} Distance(I_j, \alpha) =& (\frac{\alpha}{n} + \frac{1-\alpha}{r} - 1)^2 + (r-1)(\frac{\alpha}{n} + \frac{1-\alpha}{r})^2 + (n-r)(\frac{\alpha}{n})^2 \\ =& r(\frac{\alpha}{n} + \frac{1-\alpha}{r})^2 - 2(\frac{\alpha}{n} + \frac{1-\alpha}{r}) + 1 + (n-r)(\frac{\alpha}{n})^2 \\ =& r\frac{\alpha^2}{n^2} + 2r\frac{\alpha}{n}\frac{1-\alpha}{r} + r\frac{(1-\alpha)^2}{r^2} - 2\frac{\alpha}{n} - 2\frac{1-\alpha}{r} + 1 \\ & + \frac{\alpha^2}{n} - r\frac{\alpha^2}{n^2} \\ =& 2\frac{\alpha}{n}(1-\alpha) + \frac{(1-\alpha)^2}{r} - 2\frac{\alpha}{n} - 2\frac{1-\alpha}{r} + 1 + \frac{\alpha^2}{n} \\ =& -\frac{\alpha^2}{n} - \frac{1-\alpha^2}{r} + 1 \end{aligned}$$

□

- If $C(I_j) \notin \tau_j$ (wrong classification):

$$Distance(I_j, \alpha) = \frac{(1-\alpha)^2}{r} + 1 - \frac{\alpha^2}{n} \tag{7.8}$$

Proof.

$$\begin{aligned}
Distance(I_j,\alpha) =& r(\frac{\alpha}{n}+\frac{1-\alpha}{r})^2+(n-r-1)(\frac{\alpha}{n})^2+(\frac{\alpha}{n}-1)^2\\
=& r(\frac{\alpha}{n}+\frac{1-\alpha}{r})^2+(n-r)(\frac{\alpha}{n})^2-2\frac{\alpha}{n}+1\\
=& r\frac{\alpha^2}{n^2}+2r\frac{\alpha}{n}\frac{1-\alpha}{r}+r\frac{(1-\alpha)^2}{r^2}+\frac{\alpha^2}{n}-r\frac{\alpha^2}{n^2}-2\frac{\alpha}{n}+1\\
=& 2\frac{\alpha}{n}(1-\alpha)+\frac{(1-\alpha)^2}{r}+\frac{\alpha^2}{n}-2\frac{\alpha}{n}+1\\
=& -\frac{\alpha^2}{n}+\frac{(1-\alpha)^2}{r}+1
\end{aligned}$$

□

Let s be the total number of classified objects. Let $s_{i,C}$ be the number of instances which class $C(I_j)$ is C_i and classified as in C by the expert:
$s_{i,C} = |\{j : C(I_j) = C_i, \tau_j = C\}|$

Let $s_{.,C} = \sum_{i=1}^{n} s_{i,C}$ and $s_{i,.} = \sum_{C\subseteq\Theta} s_{i,C}$

The overall distance becomes:

$$\begin{aligned}
TotalDistance =& \sum_{i=1}^{n}(\sum_{C\subseteq\overline{C}_i} s_{i,C_i\cup C}\frac{-(1-\alpha^2)}{|C|+1}+\sum_{C\subseteq\overline{C}_i} s_{i,C}\frac{(1-\alpha)^2}{|C|})+\\
& s(1-\frac{\alpha^2}{n}) \qquad (7.9)\\
=& -(1-\alpha^2)\sum_{i=1}^{n}\sum_{C\subseteq\overline{C}_i}\frac{s_{i,C_i\cup C}}{|C|+1}+(1-\alpha)^2\sum_{i=1}^{n}\sum_{C\subseteq\overline{C}_i}\frac{s_{i,C}}{|C|}+\\
& s(1-\frac{\alpha^2}{n}) \qquad (7.10)
\end{aligned}$$

Define

$$BetP^{\Theta}\{I_j\}(C_i) = \sum_{C_i\in C\subseteq\Theta}\frac{\mu_i(C)}{|C|} = \sum_{C\subseteq\overline{C}_i}\frac{\mu_i(C_i\cup C)}{|C|+1}$$

The term $BetP^{\Theta}\{I_j\}(C_i)$ is the probability with which you would bet correctly if the object is a C_i object, thus the analogous of the Percent of Correct Classification PCC (scaled in [0,1]) for the C_i objects.

This reflects the idea that:

- all objects with $C(I_j) = \tau_j$ are correctly classified,
- half of those with $C(I_j) \in \tau_j$ for $|\tau_j| = 2$ are correctly classified,
- $1/m$ of those with $C(I_j) \in \tau_j$ for $|\tau_j| = m$ are correctly classified.

A global PCC can be defined in this context of imprecise data:

$$PCC = \sum_{i=1}^{n} \frac{s_{i,\cdot}}{s} BetP^{\Theta}\{I_j\}(C_i) \tag{7.11}$$

Define:

$$S(i) = \frac{1}{s_{i,\cdot}} \sum_{\emptyset \neq C \subseteq \Theta} \frac{s_{i,C}}{|C|}, \qquad S = \frac{1}{s} \sum_{\emptyset \neq C \subseteq \Theta} \frac{s_{\cdot,C}}{|C|} = \sum_{i=1}^{n} \frac{s_{i,\cdot}}{s} S(i) \tag{7.12}$$

Using the same arguments as for the PCC, S can be seen as the maximal number the objects that we can expect to classify correctly when data are imprecise.

Then,

$$TotalDistance/s = -(1-\alpha^2)PCC + (1-\alpha)^2(S - PCC) + (1 - \frac{\alpha^2}{n}) \tag{7.13}$$

The value of α in [0,1] that minimizes $TotalDistance$ is obtained by finding the value that makes the derivate null, and satisfies the domain constraints.

$$\frac{d\ TotalDistance}{d\alpha} = 2\alpha s PCC - 2(1-\alpha)s(S - PCC) - \frac{2s\alpha}{n} = 0$$

Hence

$$\alpha = \min(1, \frac{S - PCC}{S - 1/n}) \tag{7.14}$$

where the $1/n$ can be seen as the expected number of objects correctly classified by a blind unbiased expert. The only case when the domain constraint is used, is when $PCC < 1/n$, thus when the expert is worse than a blind expert that allocates objects on pure chance.

When $\alpha < 1$, the expert is worth using. When $\alpha = 1$, all bba's produced by the classifier are so discounted that they become vacuous, in which case why to bother with such a expert.

When the classifier is categorical and precise, i.e., when $|\tau_j| = 1$ for all objects I_j, then $S = 1$ and

$$\alpha = \min(1, \frac{1 - PCC}{1 - 1/n}) \tag{7.15}$$

7.4.7 Comparing α and *kappa*

A classical criteria for evaluating a classifier quality is the *kappa* coefficient. It is normally defined for categorical and precise classifiers. It is then defined as follows:

$$\begin{aligned} kappa &= \frac{PCC - \text{ proportion correctly classified by chance}}{\text{max possible } PCC - \text{ proportion correctly classified by chance}} \\ &= \frac{\sum_{i=1}^{n} s_{i,C_i} - \sum_{i=1}^{n} s_{.,C_i} s_{i,.}/s}{s - \sum_{i=1}^{n} s_{.,C_i} s_{i,.}/s} \end{aligned} \tag{7.16}$$

Of course *'max possible PCC'* is 1 (100%) in the classical context.

Consider the following experts:

- Suppose the expert is categorical, precise and worth using. Further, suppose the expert allocated the same number of objects in each class. Then $1 - \alpha = kappa$. This illustrates the link between the two coefficients.
- Suppose the expert is categorical, imprecise and worth using. The proportion correctly classified by chance becomes S, what fits indeed with the explanation given for Relation (7.12).
- Suppose the expert is not categorical and worth using, then the equation is the generalization of the previous adapted *kappa* coefficients to the non categorical cases.

In order to better understand what means the discounting factor α, we can propose another link that might help. For categorical and precise experts, the PCC can be seen as the expected utility obtained from the use of the expert when utilities are 1 for a correct decision ($C(I_j) = \tau_j$), and 0 for a wrong decision.

For the categorical imprecise expert, use utilities $1/|\tau_j|$ when $C(I_j) \in \tau_j$, thus when the expert is not wrong, and 0 when it is wrong. The coefficient fits nicely with the natural idea that if for instance all we know is $\tau_j = \{C_1, C_2, C_3\}$, then such a decision is worth 1/3 as on the average one can expect that 1/3 of those objects classified as τ_j are correctly classified and 2/3 are wrongly classified. Relation (7.4) could be seen as generalizing this idea to non categorical experts.

7.4.8 Uncertainty about the truth

Suppose the actual classes of the objects used to assess the discounting factor are not exactly known, but we only have a bba $m_0^\Theta\{I_j\}$ that expresses what we know about the true class of the object I_j. The adaptation of the distance is immediate. Let $p_{j,k}$ be the value of the pignistic probability induced from the $m_0^\Theta\{I_j\}$ with $p_{j,k} = BetP_0^\Theta(\text{class of } I_j \text{ is } C_k)$. If we knew that the class of I_j was C_k, we would compute:

$$Distance(I_j, \alpha) = \sum_{i=1}^{n} (BetP^{\Theta,\alpha}\{I_j\}(C_i) - \delta_{j,i})^2$$

according to Equation (7.1), where $\delta_{j,i} = 1$ if $C(I_j) = C_i$ ($C(I_j)$ is the actual class of the object I_j) and 0 otherwise. The probability that the class of I_j was C_k is $p_{j,k}$, so we weight this distance by $p_{j,k}$ and compute its expectation taken over k.

$$Distance(I_j, \alpha) = \sum_{k=1}^{n} p_{j,k} \sum_{i=1}^{n} (BetP^{\Theta,\alpha}\{I_j\}(C_i) - \delta_{j,i})^2 \tag{7.17}$$

We then proceed as previously (see Section 7.4.3).

7.4.9 Unclassified data

Suppose an object I_j has not been classified by the expert, and still we add it to the database to assess the discounting factor. It seems that the presence of this case should not interfere with the assessment of α. And so, is it indeed. In that case, $BetP^{\Theta,\alpha}\{I_j\}(C_i) = 1/n$ whatever α. Equation (7.17) becomes:

$$Distance(I_j, \alpha) = \sum_{k=1}^{n} p_{j,k} \sum_{i=1}^{n} (1/n - \delta_{j,i})^2 \tag{7.18}$$

which does not depend on α and therefore the evaluation of α will not be affected by the inclusion of those instances unclassified by the expert. This property results from the fact that the discounting of a vacuous bba is the vacuous bba itself.

7.5 Assessing the discounting factors of several experts used jointly

7.5.1 Assessing the discounting factors

This second method developed in this chapter permits the evaluation of the discounting factors when there are several experts and their opinions are aggregated in order to get a joint bba relative to the class of each training instance of $\mathcal{T}$.

Pooling all the experts' opinions together is done in order to derive a better predictor. This pooling is achieved conjunctively by combining the bba's produced by each expert. Before combining them, they must be discounted appropriately in order to take into account their individual reliability.

Let $m^{\Theta}_{E_e}\{I_j\}$ be the bba collected from the expert E_e about the actual value of the class of the object I_j. In order to get the optimal set of discounting factors, the following steps are applied:

- For each bba $m^{\Theta}_{E_e}\{I_j\}$, discount it by its discounting factor α_e given to the expert E_e. We get $m^{\Theta,\alpha_e}_{E_e}\{I_j\}$. This process is applied for each expert and for each object.

- For each object I_j $(j = 1, ..., s)$, apply the conjunctive combination rule in order to compute the overall bba $m^{\Theta,\overline{\alpha}}\{I_j\}$ about the class to which I_j belongs (the $\overline{\alpha}$ enhances its dependence on the α_e's).

$$m^{\Theta,\overline{\alpha}}\{I_j\} = m^{\Theta,\alpha_1}_{E_1}\{I_j\} \textcircled{\cap} \ldots \textcircled{\cap} m^{\Theta,\alpha_h}_{E_h}\{I_j\} \tag{7.19}$$

  The bba $m^{\Theta,\overline{\alpha}}\{I_j\}$ is the bba representing the result of the aggregation of the individual bba's produced by the various experts relative to the classes of the object I_j.

- Compute the corresponding $BetP^{\Theta,\overline{\alpha}}\{I_j\}$ (relative to the bba $m^{\Theta,\overline{\alpha}}\{I_j\}$) representing the pignistic probability on the class of the object I_j.

- For each object I_j, compute the distance between $BetP^{\Theta,\overline{\alpha}}\{I_j\}$ and the real class of I_j. This distance is defined by:

$$Distance(I_j, \overline{\alpha}) = \sum_{i=1}^{n} (BetP^{\Theta,\overline{\alpha}}\{I_j\}(C_i) - \delta_{j,i})^2$$

  where $\delta_{j,i} = 1$ if $C(I_j) = C_i$ and 0 otherwise.

- Compute $TotalDistance$ as follows:

$$TotalDistance = \sum_{j=1}^{s} Distance(I_j, \overline{\alpha}) \tag{7.20}$$

  $TotalDistance$ is expressed on the terms of the experts' discounting factors $\alpha_1, \alpha_2, \ldots, \alpha_h$.

- In order to find the optimal discounting factors, we minimize $TotalDistance$ on the α's under the constraints $0 \leq \alpha_e \leq 1, \ \forall e \in \{1, \ldots, h\}$

Example 7.2 *Let's consider the same data in the Example 7.1 (see Table 7.1) but assume the two experts' beliefs will be taken into account together. So, let's apply our second method to the two experts' opinions in order to get their merged judgment.*

Once E_1'bba's and E_2'bba's are discounted, we get respectively $m_{E_1}^{\Theta,\alpha_1}\{I_j\}$ and $m_{E_2}^{\Theta,\alpha_2}\{I_j\}$ where $j = 1, 2, 3, 4$, which are linear functions of the discounting factors.

For each client I_j, we compute the joint bba $m^{\Theta,\overline{\alpha}}\{I_j\}$:

$$m^{\Theta,\overline{\alpha}}\{I_j\} = m_{S_1}^{\Theta,\alpha_1}\{I_j\} \textcircled{\cap} m_{S_2}^{\Theta,\alpha_2}\{I_j\}$$

where the terms containing the α_e's are at worst of the form $\prod_{e=1,\ldots,h} \alpha_e$ where h is the number of experts ($h = 2$ in the present case).

The corresponding discounted $BetP$'s relative to the these bba's are also linear functions of the same product terms. The value of $Distance(I_j, \overline{\alpha})$ relative to the objects, as well as $TotalDistance$ are quadratic functions of the previous product terms.

So its minimization on the α_e is simple and can be achieved by any minimization program. Even when we work with more than two experts, any minimization program can give the different values of α_e's.

In the present case, $\alpha_1 = 0.28$ and $\alpha_2 = 0.12$. It should be enhanced that the discounting factors computed in this second method should not be assimilated to those computed with the first one. Here we want the α_e's so that the expert pooling itself is 'optimal', whereas in the first method, we compute the α_e's in order to evaluate the quality of the individual experts.

7.5.2 Experts observing different data sets

Suppose an object I_j in $\mathcal{T}$ has not been classified by the expert E_e. This is equivalent to using a vacuous bba for $m^{\Theta}_{E_e}\{I_j\}$. As before, this instance does not interfere with the assessment of the α's. The term $m^{\Theta,\overline{\alpha}}\{I_j\}$ encountered in Equation (7.19) is not changed as $m^{\Theta,\alpha_e}_{E_e}\{I_j\}$ is vacuous. The same holds for $BetP^{\Theta,\overline{\alpha}}\{I_j\}$, $Distance(I_j,\overline{\alpha})$ and $TotalDistance$. Hence, the α_e's will be the same.

The only problematic case would be if E_e had not observed any data, and produced only vacuous bba's. In that case $TotalDistance$ becomes independent of α_e, and any value would be as good as any other, as it should be indeed.

How could we assess the quality of an expert that does not tell anything. The fact that the assessment of α_e's does not depend on the addition of any vacuous bba implies that we can apply the previous method to the case where the data sets observed by each expert differ from experts to experts. One just considers all possible cases, adds (fictitiously) vacuous bba's for all the missing bba's, and proceeds as before.

7.6 Remarks

1. When the experts are considered alone or jointly, the found reliability factor will be used to update the real training set used to build the belief decision tree.

   In such a case, the knowledge about the training instance's class is generally uncertain and represented by a bba.

2. for the case where there are many experts where each one is interested to a disjoint part of the training set leading to the construction of the belief decision tree, the first method will be applied. In other words, we will look for the reliability factor of each expert independently.

7.7 Conclusion

In this chapter, we have presented two methods for assessing the reliability factors of non ideal experts.

In the TBM, the expert's reliability is represented by a discounting factor that transforms the basic belief assignment produced by the expert into another bba that is less *'bold'* than the original one. We describe two methods for assessing the discounting factors.

The first method treats the case where each expert is considered alone and consists in finding the discounting factor that will make his opinions as close as possible from the reality.

The second method treats the case where we have several experts that must be used jointly. We assess the discounting factors that will make the combined opinions as close as possible from the reality.

These methods can be adapted to handle partially known data, that is data for which we are not sure about the actual values and we can only produce a belief function to express our opinion about this value.

In our two methods, we have dealt within a classification framework where we dispose of a set of objects for which the classes are perfectly known by us and not by the experts. Hence, bba's on classes of the training set allowing the construction of the belief decision tree will be updated according to the reliability factor(s) found for the expert(s).

These methods can be easily extended to other problems of prediction once we know the truth for some case (as in supervised learning) and we can define a *'distance'* between the expert's reports and the reality. The technique consists in finding the discounting factors that will minimize this distance.

In the next chapter, we present briefly another kind of improvement that can be studied in belief decision trees related to uncertainty that may occur in the attributes of training instances.

Chapter 8

Uncertainty in attributes in the training set

8.1 Introduction

In the previous chapters, we have dealt with training sets characterized by uncertainty in classes of training objects while their attribute values are supposed to be known with certainty.

However, uncertainty may not only appear in classes but also in attributes of instances belonging to the training set and that will be used to ensure the building of the belief decision tree.

Hence, in order to make improvements in our belief decision tree approach, we deal, in this chapter, with the uncertainty that may occur in attribute values of training instances.

Therefore, in the training set uncertainty is encountered in both the classes of the training instances and also in their attribute values.

In this chapter, we start by presenting the difficulties of handling uncertainty in attributes of training instances. Next, we define the *'new'* structure of the training set and the different parameters leading to the construction of belief decision trees in such a case.

8.2 Difficulties

Dealing with uncertainty, represented by belief functions, in attributes in the training set, leads to two major problems:

1. Regarding the attribute selection measure, how will we choose the best attribute in each induced (sub) belief decision tree either in the averaging or the conjunctive approach. In other words, which attribute selection measures will we use?

2. Regarding the sub-node allocation, in other words, once the best attribute is chosen, in which node the remaining objects that may be characterized by uncertain attribute values will be put in order to define the next attribute or the leaf?

Therefore, in such a context, the major parameters useful for building a belief decision tree have to be defined. These parameters are basically the attribute selection measure, the partitioning strategy, the stopping criteria, and the structure of leaves (more precisely the leaves' bba's).

Note that some of these parameters may be defined exactly as the same manner as with the case of only uncertainty in the classes of training instances (see Chapter 4).

8.3 Uncertainty related to the attributes in the training set

In this section, we develop a method for constructing a belief decision tree from a training set characterized by uncertainty in the attribute values, whereas training instances' classes may be either certain or uncertain.

As described in previous chapters, classes of training instances are presented through bba's. Thus, in the case of total certainty in classes we deal with certain bba's.

8.3.1 Structure of the training set

Contrary to the classical structure of the training set where attribute values and training instances' classes are perfectly known, in this case, uncertainty is introduced in both the values of attributes of training instances and also their classes.

This latter representation is considered as the general case of uncertainty in a training set. A particular case can be considered if only attribute values in the training set are uncertain and when classes of training objects are supposed to be known with certainty. The other particular case is the one treated all over this thesis consisting in having uncertainty in classes of the training instances, whereas the attributes values are certain.

Example 8.1 *Let's reconsider the Example 3.1 presented in Chapter 3 related to the classification of clients in a bank. Let's remind that the training set instances are characterized by three symbolic attributes defined as:*

- *Income with possible values* $\{No, Low, Average, High\}$,
- *Property with possible values* $\{Less, Greater\}$
- *unpaid_credit with possible values* $\{Yes, No\}$.

Three classes may be assigned to clients ($\Theta = \{C_1, C_2, C_3\}$*):*

- C_1 *including good clients, i.e., reliable clients, for whom the bank accepts to give the whole loan.*
- C_2 *to which belongs moderate clients, for whom the bank accepts to give a part of the loan.*
- C_3 *regrouping 'bad' clients for whom the bank refuses to give the loan.*

The new structure of the training set for this example can be represented as follows (see Table 8.1):

Table 8.1: New structure of training set: Uncertainty in attributes and classes

Income	Property	Unpaid_credit	Class
$m^{Income}\{I_1\}$	$m^{Property}\{I_1\}$	$m^{Unpaid\_credit}\{I_1\}$	$m^{\Theta}\{I_1\}$
$m^{Income}\{I_2\}$	$m^{Property}\{I_2\}$	$m^{Unpaid\_credit}\{I_2\}$	$m^{\Theta}\{I_2\}$
$m^{Income}\{I_3\}$	$m^{Property}\{I_3\}$	$m^{Unpaid\_credit}\{I_3\}$	$m^{\Theta}\{I_3\}$
$m^{Income}\{I_4\}$	$m^{Property}\{I_4\}$	$m^{Unpaid\_credit}\{I_4\}$	$m^{\Theta}\{I_4\}$
$m^{Income}\{I_5\}$	$m^{Property}\{I_5\}$	$m^{Unpaid\_credit}\{I_5\}$	$m^{\Theta}\{I_5\}$
$m^{Income}\{I_6\}$	$m^{Property}\{I_6\}$	$m^{Unpaid\_credit}\{I_6\}$	$m^{\Theta}\{I_6\}$
$m^{Income}\{I_7\}$	$m^{Property}\{I_7\}$	$m^{Unpaid\_credit}\{I_7\}$	$m^{\Theta}\{I_7\}$
$m^{Income}\{I_8\}$	$m^{Property}\{I_8\}$	$m^{Unpaid\_credit}\{I_8\}$	$m^{\Theta}\{I_8\}$

where Θ_{Income}, $\Theta_{Property}$, and Θ_{unpaid_credit} are the frames of discernment of respectively the income, property and unpaid_credit attributes.

Hence, as described above with the values of these attributes, these frames of discernment are defined as follows:

$\Theta_{Income} = \{No, Low, Average, High\}$

$\Theta_{Property} = \{Less, Greater\}$

$\Theta_{Unpaid\_credit} = \{Yes, No\}$

Besides, $m^{Income}\{I_j\}$, $m^{Property}\{I_j\}$, and $m^{Unpaid\_credit}\{I_j\}$ (for $j = 1, \ldots, 8$) are the bba's defined on respectively Θ_{Income}, $\Theta_{Property}$, and $\Theta_{Property}$ and relative to the attributes of the instance I_j in which its corresponding values may be uncertain.

Finally, $m^{\Theta}\{I_j\}$ (for $j = 1, 2, ..., 8$) is the bba expressing the beliefs on the classes of the instance I_j.

As noted, such structure of the training set can be easily transformed to the 'old' structure of the training set that we have handled in all over this thesis dealing with only uncertainty in classes. Such case is ensured when all the attributes' bba's of all the training instances are certain bba's, which means that all the attribute values of training instances are known with certainty.

If all the bba's on the classes relative to the training instances are also certain bba's, we get the classical structure of training sets used in decision trees where no uncertainty is handled.

Furthermore, we can get only uncertainty in attributes if all classes are known with certainty. We can also get training instances with disjunctive values in attributes and also in classes.

As a conclusion, this 'new' structure of the training set presents a more general framework for dealing with uncertainty in attributes and classes in order to induce belief decision trees.

8.3.2 Attribute selection measure

Introduction

In the case of uncertainty present only in the classes of the training instances, we have proposed, as an attribute selection measure, two solutions namely the averaging approach and the conjunctive one (see Section 4.3.1). By analogy, handling uncertain attribute values in the training set will also be solved by these two approaches that will be adapted to this *'new'* context of uncertainty.

Averaging approach

Its steps are similar to those defined in Chapter 4 with only uncertainty in classes of training instances.

These steps can also be applied to either training sets with uncertainty in both attributes and classes or uncertainty only in attributes.

1. Compute the pignistic probability (relative to classes) of each instance I_j belonging to the training set:

$$BetP^{\Theta}\{I_j\}(C_i) = \sum_{C_i \in C \subseteq \Theta} \frac{1}{|C|} \frac{m^{\Theta}\{I_j\}(C)}{1 - m^{\Theta}\{I_j\}(\emptyset)}, \ \forall C_i \in \Theta \qquad (8.1)$$

2. Compute the average pignistic probability function $BetP^{\Theta}\{S\}$ taken over the set of objects S in order to get the average probability on each class.

$$BetP^{\Theta}\{S\}(C_i) = \frac{1}{\sum_{I_j \in S} p_j^S} \sum_{I_j \in S} p_j^S BetP^{\Theta}\{I_j\}(C_i) \qquad (8.2)$$

   where p_j^S is the probability of the object I_j to belong to the node (subset) S. The probability p_j^S is the product of the different probabilities (pignistic probabilities) of the values of the attributes[1] relative to the object I_j, allowing I_j to belong to the subset S.

3. Compute the entropy of the average pignistic probabilities in S. This value $Info(S)$ is equal to:

$$Info(S) = -\sum_{i=1}^{n} BetP^{\Theta}\{S\}(C_i) log_2 BetP^{\Theta}\{S\}(C_i) \qquad (8.3)$$

[1] We assume the attributes are independent, the value of one attribute does not affect the value of another attribute.

4. Select an attribute A_k and for each corresponding value v, collect the subset $S_v^{A_k}$ made with the objects having v as a value for the attribute A_k. Since the A_k values may be uncertain, $S_v^{A_k}$ will contain objects I_j such that their pignsitic probability relative to the value v is as follows:

$$BetP^{A_k}\{I_j\}(v) \neq 0. \tag{8.4}$$

5. Compute the weighted average pignistic probability for those objects in the subset $S_v^{A_k}$. Let the result be denoted $BetP^{\Theta}\{S_v^{A_k}\}$ for $v \in D(A_k)$, $A_k \in A$. It will be equal to:

$$BetP^{\Theta}\{S_v^{A_k}\}(C_i) = \frac{1}{\sum_{I_j \in S_v^{A_k}} p_j^{S_v^{A_k}}} \sum_{I_j \in S_v^{A_k}} p_j^{S_v^{A_k}} BetP^{\Theta}\{I_j\}(C_i) \tag{8.5}$$

where $p_j^{S_v^{A_k}}$ is the probability of the object I_j to belong to the subset $S_v^{A_k}$ having v as the value of the attribute A_k (its computation is done as the same manner as the computation of p_j^S).

6. Compute $Info_{A_k}(S)$ using the same definition as suggested by Quinlan (1986), but using the pignistic probabilities instead of the proportions. We get:

$$Info_{A_k}(S) = \sum_{v \in D(A_k)} \frac{|S_v^{A_k}|}{|S|} Info(S_v^{A_k}) \tag{8.6}$$

where Info($S_v^{A_k}$) is computed from the Equation (8.3) using $BetP^{\Theta}\{S_v^{A_k}\}$ and we define $|S| = \sum_{I_j \in S} p_j^S$ and $|S_v^{A_k}| = \sum_{I_j \in S_v^{A_k}} p_j^{S_v^{A_k}}$.

The term $Info_{A_k}(S)$ is equal to the weighed sum of the different $Info(S_v^{A_k})$ relative to the considered attribute. These $Info(S_v^{A_k})$ are weighted by the *'proportion'* of objects in $S_v^{A_k}$.

7. Compute the information gain provided by the attribute A_k in the set of objects S such that:

$$Gain(S, A_k) = Info(S) - Info_{A_k}(S) \tag{8.7}$$

8. Using the Split Info, compute the gain ratio relative to the attribute A_k:

$$Gain\ Ratio(S, A_k) = \frac{Gain(S, A_k)}{Split\ Info(S, A_k)} \tag{8.8}$$

Let's remind that the Split Info value is defined as follows:

$$Split\ Info(S, A_k) = -\sum_{v \in D(A_k)} \frac{|S_v^{A_k}|}{|S|} log_2 \frac{|S_v^{A_k}|}{|S|} \tag{8.9}$$

Besides, let's remind that $|S| = \sum_{I_j \in S} p_j^S$ and $|S_v^{A_k}| = \sum_{I_j \in S_v^{A_k}} p_j^{S_v^{A_k}}$.

9. Repeat for every attribute $A_k \in A$ and choose the one that maximizes the gain ratio.

Example 8.2 *Let's illustrate our averaging approach for handling uncertain attributes with a simple example. In fact, let's deal with the same example of classification of clients.*

In order to simplify the computation, let's illustrate our method with a simple case where uncertainty involved only one attribute (for example the property attribute) and with uncertainty in training instances' classes, the structure of the training set T can be defined as follows (see Table 8.2):

Table 8.2: Training set -Uncertainty in one attribute-

Income	Property	Unpaid_credit	Class
High	$m^{Property}\{I_1\}$	Yes	$m^{\Theta}\{I_1\}$
Average	$m^{Property}\{I_2\}$	No	$m^{\Theta}\{I_2\}$
High	$m^{Property}\{I_3\}$	Yes	$m^{\Theta}\{I_3\}$
Average	$m^{Property}\{I_4\}$	Yes	$m^{\Theta}\{I_4\}$
Low	$m^{Property}\{I_5\}$	Yes	$m^{\Theta}\{I_5\}$
No	$m^{Property}\{I_6\}$	No	$m^{\Theta}\{I_6\}$
High	$m^{Property}\{I_7\}$	No	$m^{\Theta}\{I_7\}$
Average	$m^{Property}\{I_8\}$	Yes	$m^{\Theta}\{I_8\}$

where

$m^{\Theta}\{I_1\}(C_1)$=0.7; $m^{\Theta}\{I_1\}(\Theta)$ =0.3;
$m^{\Theta}\{I_2\}(C_2)$=0.5; $m^{\Theta}\{I_2\}(C_1 \cup C_2)$=0.4; $m^{\Theta}\{I_2\}(\Theta)$=0.1;
$m^{\Theta}\{I_3\}(C_1)$=0.6; $m^{\Theta}\{I_3\}(\Theta)$ =0.4;
$m^{\Theta}\{I_4\}(C_2)$=0.6; $m^{\Theta}\{I_4\}(C_3)$ =0.3; $m^{\Theta}\{I_4\}(\Theta)$=0.1;
$m^{\Theta}\{I_5\}(C_3)$=0.7; $m^{\Theta}\{I_5\}(C_2 \cup C_3)$=0.2; $m^{\Theta}\{I_5\}(\Theta)$=0.1;
$m^{\Theta}\{I_6\}(C_3)$=0.95; $m^{\Theta}\{I_6\}(\Theta)$ =0.05;
$m^{\Theta}\{I_7\}(C_1)$=0.95; $m^{\Theta}\{I_7\}(\Theta)$ =0.05;
$m^{\Theta}\{I_8\}(C_2)$=0.4; $m^{\Theta}\{I_8\}(C_3)$ =0.4; $m^{\Theta}\{I_8\}(\Theta)$=0.2;

and

$m^{Property}\{I_1\}(Greater) = 0.6;$ $m^{Property}\{I_1\}(Less) = 0.1;$
$m^{Property}\{I_1\}(\Theta_{Property}) = 0.3;$
$m^{Property}\{I_2\}(Greater) = 1;$
$m^{Property}\{I_3\}(Less) = 1;$
$m^{Property}\{I_4\}(Greater) = 0.7;$ $m^{Property}\{I_4\}(\Theta_{Property}) = 0.3;$
$m^{Property}\{I_5\}(Less) = 0.5;$ $m^{Property}\{I_5\}(Greater \cup Less) = 0.2;$
$m^{Property}\{I_5\}(\Theta_{Property}) = 0.3;$
$m^{Property}\{I_6\}(Less) = 1;$
$m^{Property}\{I_7\}(less) = 0.4;$ $m^{Property}\{I_7\}(\Theta_{Property}) = 0.6;$
$m^{Property}\{I_8\}(Greater \cup Less) = 1;$

For the whole training set T, the proportions relative to the three possible classes C_1, C_2 and C_3 are respectively 0.36, 0.28 and 0.36.

Hence,

$$\begin{aligned} Info(T) &= -0.36 * log_2 0.36 - 0.28 * log_2 0.28 - 0.28 * log_2 0.28 \\ &= 1.575; \end{aligned}$$

The next steps consists in computing the gain ratios relative to the three attributes (income, property and unpaid_credit) in order to choose the one presenting the highest gain ratio.

For the income and unpaid_credit attributes, the computation of the gain ratios has been already done in Chapter 4 (see Example 4.1), since these attributes have kept the same values for their training instances), so we get:
$Gain\ ratio(T, Income) = 0.361;$
$Gain\ Ratio(T, unpaid\_credit) = 0.004;$

For the property attribute and due to the uncertainty in its values, we have started by computing the pignistic probabilities relative to the property attribute of training objects (see Table 8.3):

Table 8.3: Pignistic probability relative to the property attribute

	Greater	**Less**
$BetP^{Property}\{I_1\}$	0.75	0.25
$BetP^{Property}\{I_2\}$	1	0
$BetP^{Property}\{I_3\}$	0	1
$BetP^{Property}\{I_4\}$	0.85	0.15
$BetP^{Property}\{I_5\}$	0.25	0.75
$BetP^{Property}\{I_6\}$	0	1
$BetP^{Property}\{I_7\}$	0.3	0.7
$BetP^{Property}\{I_8\}$	0.5	0.5
Total	3.65	4.35

Next, let's compute the pignistic probability for each value of the property attribute ($BetP^{\Theta}\{T^{Property}_{Greater}\}$ and $BetP^{\Theta}\{T^{Property}_{Less}\}$). The following computations are based on the pignistic probability relative to the classes already computed in Table 4.4 (see Chapter 4) and on respectively the probabilities $p_j^{T^{Property}_{Greater}}$ and $p_j^{T^{Property}_{Less}}$.

At this level, we look for the probabilities of each object to have respectively greater and less as values of the property attribute in the training set T (we do not care about the values of the other attributes). So, $p_j^{\{T^{Property}_{Greater}\}}$ $= BetP^{Property}\{I_j\}(Greater)$ and $p_j^{T^{Property}_{Less}}(Less) = BetP^{Property}\{I_j\}(Less)$ (for $j = 1, \ldots, 8$).

The computations of ($BetP^{\Theta}\{T^{Property}_{Greater}\}$ and $BetP^{\Theta}\{T^{Property}_{Less}\}$) give the following results:

$$\begin{aligned} BetP^{\Theta}\{T^{Property}_{Greater}\}(C_1) = {} & \frac{1}{0.75+1+0+0.85+0.25+0+0.3+0.5}(0.75*0.8+ \\ & 1*0.23+0*0.74+0.85*0.04+0.25*0.04+0*0.02+ \\ & 0.3*0.96+0.5*0.06) \\ = {} & \frac{1}{3.65}(1.192) \\ = {} & 0.33; \end{aligned}$$

$$\begin{aligned} BetP^{\Theta}\{T^{Property}_{Greater}\}(C_2) = {} & \frac{1}{3.65}(0.75*0.1+1*0.73+0*0.13+0.85*0.63+ \\ & 0.25*0.13+0*0.02+0.3*0.02+0.5*0.47) \\ = {} & 0.44; \end{aligned}$$

$$BetP^{\Theta}\{T^{Property}_{Greater}\}(C_3) = \frac{1}{3.65}(0.75 * 0.1 + 1 * 0.04 + 0 * 0.13 + 0.85 * 0.33 + 0.25 * 0.83 + 0 * 0.96 + 0.3 * 0.02 + 0.5 * 0.47)$$
$$= 0.23;$$

Following the same process, we get the results relative to $BetP^{\Theta}\{T^{Property}_{Less}\}$ *(see Table 8.4):*

Table 8.4: Computation of $BetP^{\Theta}\{T^{\mathbf{Property}}_{\mathbf{Greater}}\}$ and $BetP^{\Theta}\{T^{\mathbf{Property}}_{\mathbf{Less}}\}$

	C_1	C_2	C_3
$BetP^{\Theta}\{T^{Property}_{Greater}\}$	0.33	0.44	0.23
$BetP^{\Theta}\{T^{Property}_{Less}\}$	0.39	0.14	0.47

Let's compute the values of $Info(T^{Property}_{Greater})$ *and* $Info(T^{Property}_{Less})$

$$Info(T^{Property}_{Greater}) = -0.33 * log_2 0.33 - 0.44 * log_2 0.44 - 0.23 * log_2 0.23$$
$$= 1.536;$$

$$Info(T^{Property}_{Less}) = -0.39 * log_2 0.39 - 0.14 * log_2 0.14 - 0.47 * log_2 0.47$$
$$= 1.441;$$

Hence,

$$Info_{Property}(T) = \frac{3.65}{8} Info(T^{Property}_{Greater}) + \frac{4.35}{8} Info(T^{Property}_{Less})$$
$$= 1.484;$$

The computation of Split Info for the property attribute gives the following value:

$$Split\ Info(T, Property) = -\frac{3.65}{8} log_2 \frac{3.65}{8} - \frac{4.35}{8} log_2 \frac{4.35}{8}$$
$$= 0.994;$$

Therefore,

$$Gain\ Ratio(T, Property) = 0.091;$$

Hence, the gain ratios relative to these three attributes are summarized in the following table (see Table 8.5):

Table 8.5: Gain Ratio values of the attributes

Attribute	**Income**	**Property**	**Unpaid_credit**
Gain Ratio	0.361	0.091	0.004

Therefore, the income attribute is the best one since it has the highest gain ratio.

Conjunctive approach

Under the conjunctive approach, the attribute selection measure used to build a belief decision tree, in the case of uncertain attribute values and uncertain classes in the training set, is made of the following steps:

1. For each object in the training set, compute:

$$\kappa^{\Theta}\{I_j\}(C) = -\ln q^{\Theta}\{I_j\}(C)\ \forall C \subseteq \Theta \tag{8.10}$$

   from the bba $m^{\Theta}\{I_j\}$.

2. For each attribute value v of an attribute A_k, compute the joint $\kappa^{\Theta}\{S_v^{A_k}\}$ defined on Θ by:

$$\kappa^{\Theta}\{S_v^{A_k}\} = \sum_{I_j \in S_v^{A_k}} p_j^{S_v^{A_k}} \kappa^{\Theta}\{I_j\} \tag{8.11}$$

   where $p_j^{S_v^{A_k}}$ is the probability of the object I_j to belong to the node (subset) $S_v^{A_k}$. The probability $p_j^{S_v^{A_k}}$ is computed as the same manner as defined in the averaging approach.

3. Hence for each attribute value, the intra-group distance $SumD(S_v^{A_k})$ is defined as follows:

$$SumD(S_v^{A_k}) = \frac{1}{|S_v^{A_k}|} \sum_{I_j \in S_v^{A_k}} \sum_{X \subseteq \Theta} p_j^{S_v^{A_k}} (\kappa^{\Theta}\{I_j\}(X) - \frac{1}{|S_v^{A_k}|}\kappa^{\Theta}\{S_v^{A_k}\}(X))^2 \tag{8.12}$$

   where $|S_v^{A_k}| = \sum_{I_j \in S_v^{A_k}} p_j^{S_v^{A_k}}$.

4. Once the different $SumD(S_v^{A_k})$ are calculated, for each attribute $A_k \in A$, compute $SumD_{A_k}(S)$ representing the weighted sum of the different $SumD(S_v^{A_k})$ relative to each value v of the attribute A_k:

$$SumD_{A_k}(S) = \sum_{v \in D(A_k)} \frac{|S_v^{A_k}|}{|S|} SumD(S_v^{A_k}) \tag{8.13}$$

5. We compute $Diff(S, A_k)$ defined as the difference between $SumD(S)$ and $SumD_{A_k}(S)$:

$$Diff(S, A_k) = SumD(S) - SumD_{A_k}(S) \tag{8.14}$$

where

$$SumD(S) = \frac{1}{|S|} \sum_{I_j \in S} \sum_{X \subseteq \Theta} p_j^S (\kappa^\Theta \{I_j\}(X) - \frac{1}{|S|} \kappa^\Theta \{S\}(X))^2 \tag{8.15}$$

where p_j^S is the probability of the object I_j to belong to the node (subset) S.

6. Using the split info (see Equation 8.9), compute the diff ratio relative to the attribute A_k:

$$Diff\ Ratio(S, A_k) = \frac{Diff(S, A_k)}{Split\ Info(S, A_k)} \tag{8.16}$$

7. Repeat for every attribute $A_k \in A$ and choose the one that maximizes the diff ratio.

Example 8.3 *Let's treat data presented in the Example 8.2 with the conjunctive approach in order to choose the best attribute according to this approach.*

So, we have, at first, to compute the values of diff ratio relative to the three attributes (income, property, and unpaid_credit).

The diff ratios relative to the income and unpaid_credit have been already computed in Chapter 4 (see Example 4.2) and their values are:
$Diff\ Ratio(T, Income) = 5.882$;
$Diff\ Ratio(T, Unpaid\_credit) = 4.279$;

Then, we have to compute the diff ratio relative to the property attribute which values are characterized by uncertain values.

After computations, we get:
$SumD(S_{Greater}^{Property}) = 3.781$*;*
$SumD(T_{Less}^{Property}) = 6.851$*;*

Hence,

$$SumD_{Property}(T) = \frac{|T_{Greater}^{Property}|}{|T|} SumD(T_{Greater}^{Property}) + \frac{|T_{Less}^{Property}|}{|T|} SumD(T_{Less}^{Property})$$
$$= 5.451;$$

So, $Diff(T, Property) = 7.986$*, then*
$Diff\ Ratio(T, Property) = 8.034$;

The diff ratios relative to the three attributes are summarized in the following table (see Table 8.6):

Table 8.6: Diff Ratio values of the attributes

Attribute	**Income**	**Property**	**Unpaid_credit**
Diff Ratio	5.882	8.034	4.279

The property value represents the highest value of diff ratio. Hence, it will be chosen as the best attribute.

8.3.3 Partitioning strategy

As in the case of only uncertainty in the classes of training instances, it consists in creating an edge for each attribute value chosen as a decision node.

However, the difficulty is to define the instances belonging to each of the several training subsets relative to each branch. Due to uncertainty in attributes, after partition each training instance may belong to more than one training subset. Hence, it will be assigned to each subset with a probability (of belonging to this subset) computed according to the pignstic probability of the values of its attributes.

Note that when the pignstic probability of the chosen attribute value is null for an object, the considered object will not belong to the subset induced from the branch corresponding to this value.

Example 8.4 *Let's continue with the Example 8.3. The property attribute is chosen as the best attribute, and the instance belonging to the subsets* $T_{Greater}^{Property}$, *and* $T_{Less}^{Property}$ *are summarized in this Table 8.7 with their probabilities.*

Table 8.7: Probabilities of instances in $T_{\text{Greater}}^{\text{Property}}$, and $T_{\text{Less}}^{\text{Property}}$

Instance	$T_{\text{Greater}}^{\text{Property}}$	$T_{\text{Less}}^{\text{Property}}$
I_1	0.75	0.25
I_2	1	0
I_3	0	1
I_4	0.85	0.15
I_5	0.25	0.75
I_6	0	1
I_7	0.3	0.7
I_8	0.5	0.5

For instances having probabilities equal to 0, they can be obviously excluded from the training subset.

8.3.4 Stopping criteria

They will be the same stopping criteria that we have defined in Chapter 4 (see Section 4.3.3).

However, it is interesting to note that the stopping criterion permitting to declare the node as a leaf if the treated node includes only one instance, rarely occurs due to the uncertainty in the values of the attributes of training instances. Therefore, each leaf may contain several objects but generally with probabilities less than 1.

8.3.5 Structure of leaves

In belief decision trees, leaves are represented by bba's. Nevertheless, we have also to take into account the probability of instances belonging to these leaves. Since we can use either the averaging or the conjunctive approch, thus we get two formulas of a leaf's bba according to the used approach:

In the averaging approach

Using the averaging approach in the selection attribute measure, the bba relative to the leaf L is defined as follows:

$$m^{\Theta}\{L\}(C) = \frac{1}{\sum_{I_j \in L} p_j^L} \sum_{I_j \in L} p_j^L m^{\Theta}\{I_j\}(C) \tag{8.17}$$

where p_j^L is the probability of the instance I_j to belong to the leaf L.

In the conjunctive approach

The idea is to consider for each bba $m^{\Theta}\{I_j\}$, the probability p_j^L as a reliability factor.

So, for each instance belonging to the considered leaf L, its corresponding bba should be weighted by its probability of belonging to this leaf, denoted p_j^L.

Next, the leaf's bba will be equal to the conjunctive combination of the discounted bba's relative to its instances.

Therefore, using the attribute selection measure based on the conjunctive approach, the leaf's bba will be defined as following:

$$m^{\Theta}\{L\}(C) = \textcircled{\cap}_{I_j \in L}\, p_j^L m^{\Theta}\{I_j\} \text{ for } C \subset \Theta \tag{8.18}$$

$$m^{\Theta}\{L\}(\Theta) = 1 - \sum_{C \subset \Theta} m^{\Theta}\{L\}(C) \tag{8.19}$$

Example 8.5 *Once the different parameters of building belief decision trees are defined, such trees can be easily built according to the same process as defined in Chapter 5 (see Section 5.2.3).*

Thus, using the averaging approach, the induced belief decision tree is represented in the Figure 8.1:

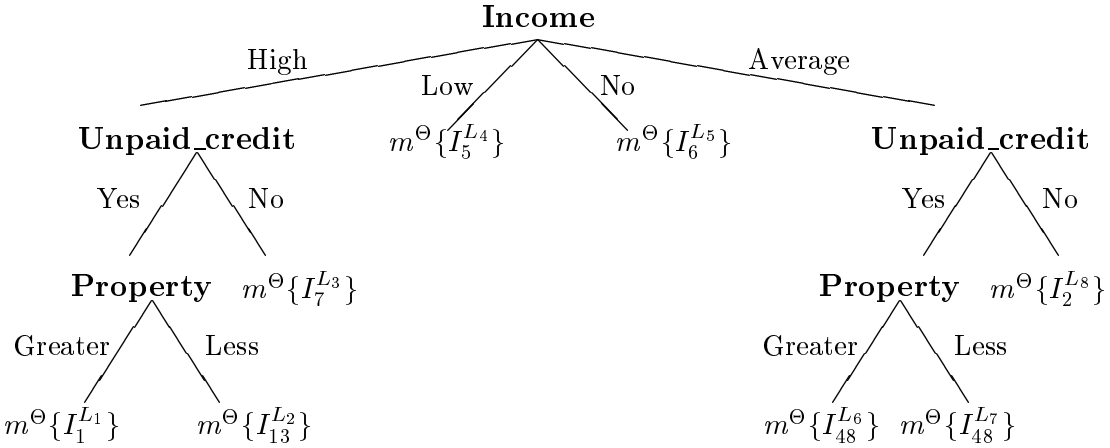

Figure 8.1: Belief decision tree: Averaging approach (Uncertain attributes)

where the leaves' bba's are defined as follows:

$$m^{\Theta}\{I_1^{L_1}\}(C_1) = 0.7;$$
$$m^{\Theta}\{I_1^{L_1}\}(\Theta) = 0.3;$$

$$m^{\Theta}\{I_{13}^{L_2}\}(C_1) = 0.62;$$
$$m^{\Theta}\{I_{13}^{L_2}\}(\Theta) = 0.38;$$

$$m^{\Theta}\{I_7^{L_3}\}(C_1) = 0.95;$$
$$m^{\Theta}\{I_7^{L_3}\}(\Theta) = 0.05;$$

$$m^{\Theta}\{I_5^{L_4}\}(C_3) = 0.7;$$
$$m^{\Theta}\{I_5^{L_4}\}(C_2 \cup C_3) = 0.2;$$
$$m^{\Theta}\{I_5^{L_4}\}(\Theta) = 0.1;$$

$$m^{\Theta}\{I_6^{L_5}\}(C_3) = 0.95;$$
$$m^{\Theta}\{I_6^{L_5}\}(\Theta) = 0.05;$$

$$m^{\Theta}\{I_{48}^{L_6}\}(C_2) = 0.53;$$
$$m^{\Theta}\{I_{48}^{L_6}\}(C_3) = 0.34;$$
$$m^{\Theta}\{I_{48}^{L_6}\}(\Theta) = 0.13;$$

$$m^{\Theta}\{I_{48}^{L_7}\}(C_2) = 0.45;$$
$$m^{\Theta}\{I_{48}^{L_7}\}(C_3) = 0.38;$$
$$m^{\Theta}\{I_{48}^{L_7}\}(\Theta) = 0.17;$$

$$m^{\Theta}\{I_{2}^{L_8}\}(C_2) = 0.5;$$
$$m^{\Theta}\{I_{2}^{L_8}\}(C_1 \cup C_2) = 0.4;$$
$$m^{\Theta}\{I_{2}^{L_8}\}(\Theta) = 0.1;$$

Using the conjunctive approach, we have an another induced belief decision tree (see Figure 8.2):

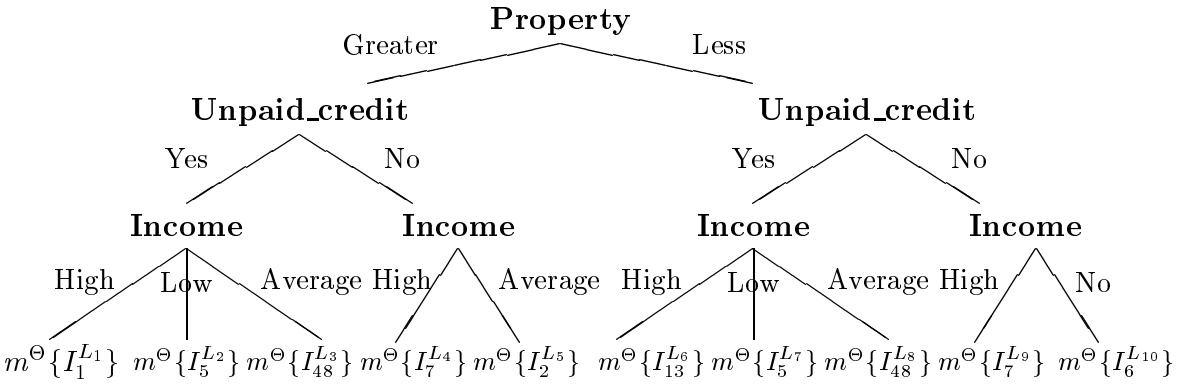

Figure 8.2: Belief decision tree: Conjunctive approach (Uncertain attributes)

where the leaves' bba's are defined according to the Equations (8.18) and (8.19) as follows:

$$m^{\Theta}\{I_1^{L_1}\}(C_1) = 0.53$$
$$m^{\Theta}\{I_1^{L_1}\}(\Theta) = 0.47;$$

$$m^{\Theta}\{I_5^{L_2}\}(C_3) = 0.18;$$
$$m^{\Theta}\{I_5^{L_2}\}(C_2 \cup C_3) = 0.05;$$
$$m^{\Theta}\{I_5^{L_2}\}(\Theta) = 0.83;$$

$$m^{\Theta}\{I_{48}^{L_3}\}(\emptyset) = 0.15;$$
$$m^{\Theta}\{I_{48}^{L_3}\}(C_2) = 0.46;$$
$$m^{\Theta}\{I_{48}^{L_3}\}(C_3) = 0.26;$$
$$m^{\Theta}\{I_{48}^{L_3}\}(\Theta) = 0.13;$$

$$m^{\Theta}\{I_{7}^{L_4}\}(C_1) = 0.29;$$
$$m^{\Theta}\{I_{7}^{L_4}\}(\Theta) = 0.71;$$

$$m^{\Theta}\{I_{2}^{L_5}\}(C_2) = 0.5$$
$$m^{\Theta}\{I_{2}^{L_5}\}(C_1 \cup C_2) = 0.4$$
$$m^{\Theta}\{I_{2}^{L_5}\}(\Theta) = 0.1;$$

$$m^{\Theta}\{I_{13}^{L_6}\}(C_1) = 0.67;$$
$$m^{\Theta}\{I_{13}^{L_6}\}(\Theta) = 0.33;$$

$$m^{\Theta}\{I_{5}^{L_7}\}(C_3) = 0.53;$$
$$m^{\Theta}\{I_{5}^{L_7}\}(C_2 \cup C_3) = 0.15;$$
$$m^{\Theta}\{I_{5}^{L_7}\}(\Theta) = 0.32;$$

$$m^{\Theta}\{I_{48}^{L_8}\}(\emptyset) = 0.03;$$
$$m^{\Theta}\{I_{48}^{L_8}\}(C_2) = 0.24;$$
$$m^{\Theta}\{I_{48}^{L_8}\}(C_3) = 0.21;$$
$$m^{\Theta}\{I_{48}^{L_8}\}(\Theta) = 0.52;$$

$$m^{\Theta}\{I_{7}^{L_9}\}(C_1) = 0.67;$$
$$m^{\Theta}\{I_{7}^{L_9}\}(\Theta) = 0.33;$$

$$m^{\Theta}\{I_{6}^{L_{10}}\}(C_3) = 0.95;$$
$$m^{\Theta}\{I_{6}^{L_{10}}\}(\Theta) = 0.03;$$

Note that for simplifying the representation, we have not presented empty leaves in the induced tree.

Remarks:

As noted, for building a belief decision tree in the case of uncertain attribute values, we use the same algorithm applied for constructing a tree when the uncertainty is only occurred in classes of training instances (see Section 5.2.3). What changes is the parameters composing this algorithm.

Regarding the classification, what we have developed in Chapter 5 (see Section 5.3), is directly applied to this situation. This is due to the fact that we have kept the same structure (representation) of the induced tree.

Thus, it would be possible to classify instances characterized by:

- only certain attribute values,
- or disjunctive or missing values for some (or all) attributes,
- or uncertain values for some (or all) attributes, where uncertainty is represented through bba's.

8.4 Conclusion

In this chapter, we have shown that our approach for building belief decision trees can be extended and applied to handle the case of uncertain attribute values in the training set.

Note that in this chapter, we have presented briefly how to handle uncertainty, described by belief functions, in attribute values of training instances. However, simulations, particular cases and other improvements related to this method are not studied in this thesis.

Conclusion

In this thesis, we have defined the belief decision tree which is a new approach associating the decision tree method to the belief function theory in order to handle uncertainty that can exist in classification problem parameters.

We consider the case where the knowledge about the class of each instance in the training set is represented by a basic belief assignment over the set of the possible classes, whereas the values of its attributes are known with certainty.

Based on this *'new'* structure of the training set characterized by uncertainty in the classes of the training instances, we present two attribute selection measures using the belief function formalism, one parallel to Quinlan's measure based on the entropy (the averaging approach), the other close in spirit to the transferable belief model (the conjunctive approach).

In fact, the first measure is the extension of the gain ratio criterion to the uncertain context, using essentially an averaging rule. The second attribute selection measure is based on the minimization of the intra-group distance, and consequently the maximization of the inter-group one. This developed distance between instances in the training set mainly uses the conjunctive rule of combination.

Partitioning strategy and stopping criteria are then provided, and the meaning of the data in leaves is detailed. Indeed, due to the uncertainty in the classes of the training objects, leaves will not be labeled by unique classes as in the standard decision trees. They will be characterized by bba's expressing beliefs on classes of objects belonging to the considered leaf. The computation of the leaves' bba's is done according to the approach used for selecting attributes.

Next, we present the different steps of the procedure allowing the construction of the tree in an uncertain context based on the parameters defined above.

After the construction procedure, we detail the inference task ensuring the classification of new instances using the constructed belief decision tree. We consider the case where the knowledge about the value of some attributes is represented by a bba, and show how to use the belief decision tree. All the leaves compatible with the knowledge are considered and their individual conclusions are combined by the disjunctive rule of combination. Classification is then based on the pignistic probabilities derived from the global bba.

This inference procedure includes different cases: it classifies instances with certain attributes values, those with disjunctive attribute values, and the more general case where the value of each attribute is represented by a bba. This latter case regroups the two first cases and also the case where the values of some attributes of the object to classify are missing.

The different developments leading to the construction of a belief decision tree and also the different cases of classification are illustrated with examples dealing with the classification of clients relative to a bank's loan policy.

In order to judge the feasibility of our method, implementation and simulation are performed. In fact, the two approaches allowing the construction of a belief decision tree namely the averaging approach and the conjunctive one are implemented. Then, several simulations are executed changing each time the degree of uncertainty encountered in the bba's relative to the classes of the training objects. This allows us to show that both approaches are feasible and that the standard case with total certainty in classes can be easily treated as a particular case of the averaging approach.

Evaluating classification is based on two known criteria usually used by classification methods namely the Percent of Correct Classification, denoted PCC, and the *kappa* criterion. We have developed an additional criterion, named $dist\_crit$, which is based on the computation of the distance between the pignistic probability given by the induced belief decision tree for each object to classify and the real class of this instance.

Then, we have proposed some improvements to our belief decision tree approach related basically to uncertainty in the training set.

At first, we have been interested to the assessment of beliefs in the training set. These beliefs are given by one or several experts on the classes of the training objects. Hence, it would be useful to weight these different bba's by taking into account the experts' reliabilities.

Indeed, we have developed two methods for assessing reliabilities. These methods are based on training sets having the same structure as the one used for building the belief decision tree, but where the classes of their objects are perfectly known by us, and not by the expert(s).

The first method deals with each expert independently. Since the reliability of an expert is represented by a discounting factor, this latter is chosen such that it minimizes the distance between the pignistic probabilities computed from the discounted beliefs given by the expert and the indicator function of the actual class of the considered objects.

The second method treats experts jointly when their opinions are merged in order to reach an aggregated opinion. They are computed so that together they minimize the same distance as above.

Once the reliability factors are found, they will be used to update the bba's in the *'real'* training set.

Note that particular cases of opinions presented by experts are handled in order to simplify formulas of the computation of reliability factors.

These proposed methods are presented for assessing beliefs in the training set used to build belief decision trees, but can be easily applied to any kind of classification problem or even to other prediction problems. Hence, what is required is a training set and a distance between the prediction and the actual values.

Another improvement of our belief decision tree approach is related to uncertainty in attributes in the training set. In addition to the uncertainty that pervades the classes of the training objects, we also handle the uncertainty that may occur in the attributes of these objects and that is represented by bba's.

We have presented how to deal with this *'new'* uncertainty in attribute values of training instances. So, we have defined the parameters allowing the construction of belief decision trees in such a case, namely the attribute selection measure (the averaging and the conjunctive approaches), the partitioning strategy, the stopping criteria and the leaves bba's.

Thus, several developments and adaptations have been proposed in order to handle the case of uncertainty in attributes in the training set.

Obviously, our research relative to belief decision trees is to be pursued, and contributions could be done in several contexts relative to this classification technique under uncertainty.

In fact, our proposed method provides a practical and useful tool to cope with uncertainty in classification parameters. So, handling problems ignored in the past, due to uncertain data, becomes now possible within our method. Several fields, where uncertainty is encountered, can be concerned. We notably think to management problems.

Moreover, the development relative to uncertainty in attributes in the training set can be more detailed and simulations can be performed for testing the feasibility of the proposed method.

Besides, pruning belief decision trees is an important aspect that should be developed either with uncertainty only in classes in the training set or with uncertainty in both attributes and classes of training objects. Pruning should be adapted to the uncertain context, in order to get less complex belief decision trees with a small size facilitating their comprehension.

As mentioned before, two kinds of pruning can be applied to decision trees, and consequently to belief decision trees:

- Pre-pruning methods consisting in controlling the decision tree during its growth.

- Post-pruning methods allowing to reduce the size of the tree once its growth is finished.

We can also combine together the pre-pruning and the post-pruning.

Several researches relative to pruning, especially the post one, of belief decision trees could be developed. We suggest to study the different pruning methods developed for classical decision trees like the error-complexity pruning (Breiman et al., 1984), the critical value pruning (Mingers, 1989b), the reduced error pruning (Quinlan, 1993), etc. Then, trying to adapt them to the uncertain context.

We may also try to develop pruning methods relative to belief decision trees, but which are more specific to the transferable belief model. Furthermore, pre-pruning methods for belief decision trees should be studied while improving stopping criteria declaring decision nodes as leaves.

In addition to these improvements, it would be interesting to extend our approach of belief decision tree to handle the continuous attributes. Indeed, all over this thesis, we have dealt with only symbolic attributes. Hence, it would be useful to develop the construction of belief decision trees and also the classification task based on continuous attributes.

Finally, we should mention that since decision trees under uncertainty using the belief function theory is considered as a *'new'* approach, several extensions can be projected. In fact, it would be interesting to study improvements proposed for classical decision trees, and adapt them to belief decision trees.

Appendix A

Scenario method vs belief decision tree approach

A.1 Introduction

A scenario is defined as a set of elementary hypotheses aiming at analyzing future events and anticipating what may happen. The task of classifying scenarios is one of the major preoccupation facing decision makers since its capability to assign similar scenarios to the same class, this will help finding the best strategic planning regarding a given problem.

Due to the uncertainty that may occur either in the configurations of scenario's hypotheses or even in the classes of the training scenarios, the classification task becomes more and more difficult. Ignoring this uncertainty or mistreating it, may lead to erroneous results.

In this appendix, we propose a method based on belief decision trees in order to ensure the classification of scenarios in an uncertain context. The results of this appendix are detailed in (Elouedi & Mellouli, 2000).

The belief decision tree approach offers a suitable framework to deal with the classification of scenarios in an uncertain environment. The use of the belief function theory as understood in the transferable belief model, allows a better representation of uncertainty characterizing the scenarios, especially the uncertainty expressed by experts.

A.2 Scenarios

A scenario describes a future situation. It allows to elicit and to anticipate about what may happen. Indeed, it is composed of a set of hypotheses related to the components of the given problem (field). Each hypothesis could take one or several configurations. Hence, a scenario is considered as a combination of these configurations.

These hypotheses and their configurations are defined through interviews and different questions given to experts and actors implicated in the given problem. To ensure this objective, several methods are proposed in the literature, the most used is the Delphi technique (Godet, 1991).

Two types of scenarios are defined:

- The exploratory scenarios based on past and present trends in order to elicit future. They are equivalent to the classical forecasting.
- The anticipatory scenarios built on the basis of different visions of future desired or redoubted. In fact, we have to fix the future objectives and try to find how to ensure them.

The scenario method plays an important role especially in decision problems given its capability to deal with various fields and consequently help decision makers to find the appropriate strategic planning.

The scenario method is mainly based on experts' opinions (Godet, 1991; Godet & Roubelat, 1996), (Mellouli & Elouedi, 1997), (Elouedi & Mellouli, 1998a, 1998b). It includes different steps, the major ones dealing with scenarios are their assessment and their classification.

The latter step related to the classification of scenarios allows to group scenarios sharing similar characteristics in the same class. By taking into account the class of the scenario, more reliable decisions could be taken.

A.3 Belief decision trees and scenarios

A.3.1 Scenarios vs training Set

A scenario is seen as a combination of hypotheses' configurations where each one is relative to a hypothesis H_k.

To represent scenarios within a belief decision tree, we consider the hypotheses as being the attributes, whereas the configurations corresponding to each hypothesis are assimilated to the attribute values.

As treated in the major part of this thesis (see Part 2), assume we have a training set of scenarios characterized by certain hypotheses' configurations, however there may be some uncertainty in their classes defined for each scenario by a bba on the set of classes. These bba's are given by experts.

Example A.1 *Let's consider a simple example presenting scenarios regarding the agriculture field. For simplicity sake, we define only three elementary hypotheses composing these scenarios:*
- *H_1: the rainfall which can be high or weak.*
- *H_2: the temperature which can be hot, mild, cold.*
- *H_3: the wind with values strong or weak.*

A possible scenario is for example having a high rainfall, with hot temperature and weak wind. In fact, there are twelve possible scenarios.

There are three classes to which the scenarios, related to this problem, may belong to:
- *C_1: regrouping the favorable scenarios for the agriculture field.*
- *C_2: regrouping the neutral scenarios for the agriculture field.*
- *C_3: regrouping the disastrous scenarios for the agriculture field.*

We have an expert's beliefs about the classes of scenarios that have occurred over the last six years, we get the following results:

Table A.1: Training set T

S_j	Rainfall	Temperature	Wind	Class
S_1	High	Mild	Weak	$m^{\Theta}\{S_1\}$
S_2	High	Cold	Strong	$m^{\Theta}\{S_2\}$
S_3	High	Hot	Weak	$m^{\Theta}\{S_3\}$
S_4	Weak	Hot	Weak	$m^{\Theta}\{S_4\}$
S_5	High	Mild	Weak	$m^{\Theta}\{S_5\}$
S_6	Weak	Hot	Strong	$m^{\Theta}\{S_6\}$

where

$m^{\Theta}\{S_1\}(C_1)$ *=0.8;* $m^{\Theta}\{S_1\}(\Theta)$ *= 0.2;*
$m^{\Theta}\{S_2\}(C_1 \cup C_2)$ *=0.9;* $m^{\Theta}\{S_2\}(\Theta)$ *=0.1;*
$m^{\Theta}\{S_3\}(C_2)$ *=0.2;* $m^{\Theta}\{S_3\}(C_3)$ *= 0.4;* $m^{\Theta}\{S_3\}(C_1 \cup C_2)$ *=0.2;*
$m^{\Theta}\{S_3\}(\Theta)$ *=0.2;*
$m^{\Theta}\{S_4\}(C_3)$ *=0.6;* $m^{\Theta}\{S_4\}(C_2 \cup C_3)$ *=0.2;* $m^{\Theta}\{S_4\}(\Theta)$ *= 0.2;*
$m^{\Theta}\{S_5\}(C_1)$ *=0.9;* $m^{\Theta}\{S_5\}(\Theta)$ *= 0.1;*
$m^{\Theta}\{S_6\}(C_3)$ *=0.9;* $m^{\Theta}\{S_6\}(\Theta)$ *= 0.1;*

As noted, there is some uncertainty concerning the classes of these training scenarios. It is difficult for experts to assure if these scenarios were favorable, neutral or disastrous for agriculture.

For example, 0.8 is the part of belief committed to the scenario S_1 to belong to the class of scenarios favorable for the agriculture, where 0.2 is the part of belief committed to the whole frame Θ (ignorance).

However, for the scenario S_6, 0.9 is the part of belief committed to the scenario S_1 showing that it is disastrous. The remaining belief equals to 0.1 is committed to Θ (ignorance).

This training set allows us to build the corresponding belief decision tree representing a learning taking into account these six training scenarios.

A.3.2 Construction procedure using scenarios

In order to construct a belief decision tree based on scenarios, we have to adapt the parameters used in the construction algorithm of a belief decision tree to the case handling scenarios instead of ordinary objects.

Therefore, the idea is to build a tree taking into account the scenarios belonging to the training set characterized by uncertain classes. In fact, the belief decision tree relative to the training scenarios will be built by employing a recursive divide and conquer strategy. Its steps are the same as defined in Section 5.2.3 and they can be summarized as follows:

- By using *the gain ratio measure* or *the diff ratio measure* (see Section 4.3.1), a hypothesis (the one having the highest attribute selection measure) will be chosen in order to partition the training set of scenarios. Therefore, the chosen hypothesis is selected as the root node of the current tree.

- Based on *a partitioning strategy* (see Section 4.3.2), the current training set will be divided into training subsets by taking into account the configurations of the selected hypothesis.
- When one of the *the stopping criteria* is satisfied (see Section 4.3.3), the training subset will be declared as a leaf.

As noted, the building of this kind of belief decision trees can be ensured using either the averaging or the conjunctive approach. Therefore, once the tree is built, this allows to classify new scenarios.

Example A.2 *Let's continue with the Example A.1, we will build the belief decision tree relative to the training set T using the averaging approach (the same process can be done for the conjunctive one).*

In order to find the root of the belief decision tree relative to the training set T, we have to compute the information gain of each hypothesis.

We start by computing Info(T):

$$Info(T) = -\sum_{i=1}^{3} BetP^{\Theta}\{T\}(C_i) log_2 BetP^{\Theta}\{T\}(C_i)$$

To compute the average pignistic probability $BetP\{T\}$, we have at first to calculate the different $BetP\{S_j\}$ where $j \in \{1, 2, ..., 6\}$ (see Table A.2):

Table A.2: Average BetP's

S_i	$\text{BetP}^{\Theta}\{S_i\}(C_1)$	$\text{BetP}^{\Theta}\{S_i\}(C_2)$	$\text{BetP}^{\Theta}\{S_i\}(C_3)$
S_1	0.86	0.07	0.07
S_2	0.48	0.48	0.04
S_3	0.17	0.37	0.46
S_4	0.07	0.17	0.76
S_5	0.94	0.03	0.03
S_6	0.03	0.03	0.94

$BetP^{\Theta}\{T\}$ the average pignistic probability taking over the whole training set T, is defined as follows:
$BetP^{\Theta}\{T\}(C_1) = \frac{1}{6} * (0.86 + 0.48 + 0.17 + 0.07 + 0.94 + 0.03) = 0.43$;
$BetP^{\Theta}\{T\}(C_2) = 0.19$;
$BetP^{\Theta}\{T\}(C_3) = 0.38$;
Hence, $Info(T) = 1.512$;

Then, we have to compute the information gain relative to each hypothesis related to the agriculture filed, we get:

$Gain(T, Rainfall) = 1.512 - 1.140 = 0.372$;
$Gain(T, Temperature) = 1.512 - 0.938 = 0.574$;
$Gain(T, Wind) = 1.512 - 1.469 = 0.043$;

Computing the split info for each attribute, we finally get as gain ratio values:

$Gain\ Ratio(T, Rainfall) = 0.405$;
$Gain\ Ratio(T, Temperature) = 0.393$;
$Gain\ Ratio(T, Wind) = 0.047$;

The rainfall hypothesis presents the highest gain ratio. Thus, it will be chosen as the root relative to the training set T.

So, we get the following decision tree (see Figure A.1):

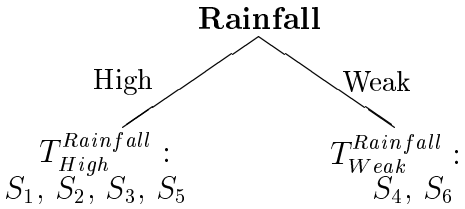

Figure A.1: Belief decision tree (First step)

Now, we have to apply the same process to the two training subsets: $T^{Rainfall}_{High}$ including the scenarios characterized by high rainfall and $T^{Rainfall}_{Weak}$ including those characterized by weak rainfall.

This process will be halt when the stopping criterion is fulfilled for all the training subsets.

For the problem related to the agriculture field, the final belief decision tree is presented in Figure A.2:

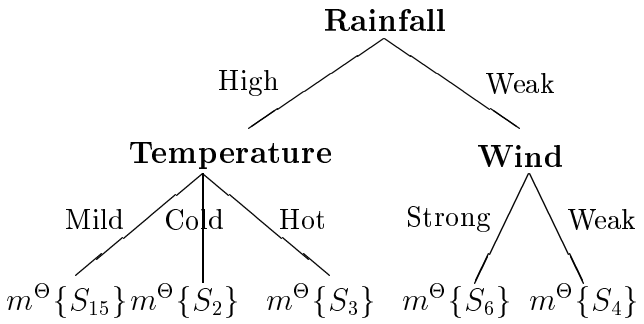

Figure A.2: Belief decision tree representing training scenarios

where $\mathrm{m}^{\Theta}\{S_{15}\}$ is the average bba of the subset including the scenarios S_1 and S_5. So, we get: $\mathrm{m}^{\Theta}\{S_{15}\}(C_1) = 0.85$; $\mathrm{m}^{\Theta}\{S_{15}\}(\Theta) = 0.15$.

A.3.3 Classification of new scenarios

This phase is very important since it allows to classify scenarios characterized by uncertain hypothesis configurations. This classification will be ensured by taking into account the constructed belief decision tree.

In fact, training scenarios (and their classes) represented by the means of a belief decision tree constitutes a convenient framework to classify new scenarios. As described in Chapter 5 (see Section 5.3), our classification method allows to handle not only uncertain hypotheses' configurations (described by basic belief assignments) but includes also hypotheses with certain or disjunctive or even unknown configurations.

The classification of a new scenario that may happen, provides a good capability to decision makers to fix the appropriate strategic planning according to beliefs assigned to the classes to which this scenario may belong.

Example A.3 *Let's continue with the Example A.2, we would like to classify the scenario S lying on the agriculture field that may occur in the future.*

Let's define by: $H = \{Rainfall, Temperature, Wind\}$
$\Theta_{Rainfall} = \{High, Weak\}$
$\Theta_{Temperature} = \{Hot, Mild, Cold\}$
$\Theta_{Wind} = \{Strong, Weak\}$
and $\Theta_H = \Theta_{Rainfall} \times \Theta_{Temperature} \times \Theta_{Wind}$

The expert is not sure about the 'future' configurations of some of the three elementary hypotheses related to the scenario S (to classify). He presents his opinions as follows:

$m^{Rainfall}(\{High\}) = 0.6;\ m^{Rainfall}(\Theta_{Rainfall}) = 0.4;$
$m^{Temperature}(\{Mild\}) = 1;$
$m^{Wind}(\Theta_{Wind}) = 1;$

In other words, the scenario S to classify is characterized by some uncertainty regarding the rainfall hypothesis, a mild temperature and a total ignorance concerning the wind hypothesis (presented by a vacuous bba).

So what will be the class of this 'uncertain' scenario S?

We start by extending the different hypotheses' bba's to Θ_H, we get:
$m^{Rainfall\uparrow H}(\{High\} \times \Theta_{Temperature} \times \Theta_{Wind}) = 0.6;$
$m^{Rainfall\uparrow H}(\Theta_{Rainfall} \times \Theta_{Temperature} \times \Theta_{Wind}) = 0.4;$
$m^{Temperature\uparrow H}(\Theta_{Rainfall} \times \{Mild\} \times \Theta_{Wind}) = 1;$
$m^{Wind\uparrow H}(\Theta_{Rainfall} \times \Theta_{Temperature} \times \Theta_{Wind}) = 1;$

In order to get the beliefs committed to the possible scenarios that may happen in the future, we have to combine these extended bba's by using the conjunctive rule of combination. We get:
$m^H = m^{Rainfall\uparrow H} \textcircled{\cap} m^{Temperature\uparrow H} \textcircled{\cap} m^{Wind\uparrow H}$ *such that:*
$m^H(\{(High, Mild, Strong), (High, Mild, Weak)\}) = 0.6;$
$m^H(\{(High, Mild, Strong), (High, Mild, Weak), (Weak, Mild, Strong),$
$(Weak, Mild, Weak)\}) = 0.4;$

We note that there are two focal elements with basic belief masses equal respectively to 0.6 and 0.4.

Then, we have to find beliefs on classes (defined on Θ) given the configurations of the hypotheses characterizing the new scenario S to classify. These beliefs have to take into account the two focal elements. According to the belief decision tree induced in the Example A.2 (see Figure A.2), we get:
$bel^{\Theta}[\{(High, Mild, Strong), (High, Mild, Weak)\}] = bel^{\Theta}\{S_{15}\};$
$bel^{\Theta}[\{(High, Mild, Strong), (High, Mild, Weak), (Weak, Mild, Strong),$
$(Weak, Mild, Weak)\}] = bel^{\Theta}\{S_{15}\} \textcircled{\cup} bel^{\Theta}\{S_4\} \textcircled{\cup} bel^{\Theta}\{S_6\}$

Let $bel_1 = bel^{\Theta}\{S_{15}\}$ *and* $bel_2 = bel^{\Theta}\{S_{15}\} \textcircled{\cup} bel^{\Theta}\{S_4\} \textcircled{\cup} bel^{\Theta}\{S_6\}$

The values (see Table A.3) of bel_1 *are induced from the bba* $m^{\Theta}\{S_{15}\}$*, whereas those of* bel_2 *are computed from the combination of* $bel^{\Theta}\{S_{15}\}$*,* $bel^{\Theta}\{S_4\}$ *and* $bel^{\Theta}\{S_6\}$ *using the disjunctive rule.*

Table A.3: Beliefs on classes given the hypotheses' configurations

	C_1	C_2	C_3	$C_1 \cup C_2$	$C_1 \cup C_3$	$C_2 \cup C_3$	Θ
bel_1	0.85	0	0	0.85	0.85	0	1
bel_2	0	0	0	0	0.46	0	1

Hence, these two belief functions will be averaged (using the values of the bba m^H*), we get:*

$bel^{\Theta}[m^H](C_1) = 0.6 * 0.85 + 0.4 * 0 = 0.51;$
$bel^{\Theta}[m^H](C_2) = 0;$
$bel^{\Theta}[m^H](C_3) = 0;$
$bel^{\Theta}[m^H](C_1 \cup C_2) = 0.6 * 0.85 = 0.51;$
$bel^{\Theta}[m^H](C_1 \cup C_3) = 0.6 * 0.85 + 0.4 * 0.46 = 0.69;$
$bel^{\Theta}[m^H](C_2 \cup C_3) = 0;$
$bel^{\Theta}[m^H](\Theta) = 1;$

Applying the pignistic transformation[1] *gives us the probability on each singular class, the pignistic probability will be defined as follows:*

$BetP^{\Theta}(C_1) = 0.70;$
$BetP^{\Theta}(C_2) = 0.11;$
$BetP^{\Theta}(C_3) = 0.19;$

Hence, the probability that the scenario S belongs respectively to the classes C_1*,* C_2 *and* C_3 *are respectively 0.70, 0.11 and 0.19. So, it seems that the scenario S has more chances (0.70) to be favorable for the agriculture field.*

A.4 Conclusion

In this appendix, we have presented a method for classifying scenarios using belief decision trees. The illustrative example is related to the scenarios regarding the agriculture field.

[1] From the different values of $bel^{\Theta}[m^H]$, we can easily deduce the corresponding basic belief masses and then apply the pignistic transformation

Our proposed method has the advantage to handle the uncertainty that may characterize either the classes of the training scenarios ensuring the construction of the belief decision tree or the configurations of the hypotheses making up the scenario to classify.

The result of the classification of scenarios provides a significant help to decision makers to conceive their strategic policy. Besides, it shows that belief decision trees can be applied to diverse fields in particular strategic problems.

Appendix B

Data used for simulation

B.1 Introduction

In this appendix, we present the modified Wisconsin breast cancer database that we have used for making simulation. This base is inspired by the Wisconsin breast cancer database[1], but modified in order to satisfy the prerequisites of our methods: symbolic attributes and uncertain classes.

It is composed of 690 instances characterized by 8 symbolic attributes and which can belong to two possible classes.

B.2 The modified Wisconsin breast cancer database

In Table B.1, we present the modified Wisconsin breast cancer database in the certain context. Let's denote by:

- j: the index of the object I_j, it varies from 1 to 690.
- A_k the attribute where k varies from 1 to 8, the column corresponding to each A_k represents the A_k's value taken by each instance. For sake of simplification in the following table each A_k will be denoted by **k** (so **1** for A_1, **2** for A_2, ...).
- C_j the class to assign to the object I_j in this data set. Its values are either 1 or 2.

[1] see http://www.ics.uci.edu/~mlearn/MLRepository.html

Table B.1: The modified Wisconsin breast cancer database

j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	C_j	j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	C_j
1	1	1	1	1	1	1	1	1	1	46	2	1	6	1	2	2	1	1	2
2	2	1	2	1	1	1	2	1	1	47	2	2	2	1	1	1	2	1	1
3	2	2	1	1	1	1	1	1	1	48	1	1	2	1	1	1	2	1	2
4	2	2	3	2	2	2	1	1	2	49	2	2	13	3	1	1	2	1	1
5	1	1	4	1	2	2	2	1	2	50	2	1	2	1	2	2	2	1	2
6	2	1	2	3	2	2	2	1	2	51	1	1	1	1	1	1	1	1	1
7	1	1	5	1	1	1	2	1	1	52	1	2	5	3	1	1	2	1	1
8	2	1	6	3	2	2	2	1	2	53	1	2	1	1	1	1	2	1	1
9	1	2	7	3	1	1	2	1	1	54	1	2	2	1	1	1	1	1	1
10	2	1	1	3	2	2	1	1	1	55	1	2	6	3	1	2	1	1	1
11	1	1	8	3	2	2	1	1	2	56	1	1	9	1	2	2	1	1	2
12	1	1	6	3	2	2	1	1	2	57	1	1	4	5	2	2	2	1	2
13	1	2	2	3	2	2	1	1	1	58	2	1	5	3	1	1	2	1	1
14	1	1	8	3	2	2	1	1	2	59	1	2	9	1	2	1	1	1	2
15	1	1	2	1	1	1	1	1	1	60	1	1	11	1	2	2	1	1	2
16	1	1	1	1	1	1	2	1	2	61	1	1	5	5	1	1	1	1	1
17	1	1	9	1	2	2	1	1	2	62	2	1	9	1	1	2	1	1	1
18	2	1	4	1	2	2	2	1	2	63	1	1	5	1	2	2	2	1	2
19	1	2	1	1	1	1	2	1	1	64	2	1	5	5	1	1	2	2	1
20	2	1	6	1	2	2	2	1	2	65	2	1	6	3	2	1	2	1	2
21	2	1	4	1	1	1	2	1	1	66	1	1	8	3	2	2	2	1	2
22	2	2	4	1	2	1	1	1	1	67	2	1	6	1	1	1	1	1	1
23	1	1	7	1	1	1	2	1	1	68	1	1	5	1	2	1	1	1	1
24	1	1	1	1	2	1	2	1	1	69	1	1	12	4	1	2	2	1	1
25	1	1	4	1	1	1	2	1	1	70	1	1	10	3	2	2	1	1	2
26	1	1	10	3	2	2	1	1	2	71	1	2	12	4	2	1	1	1	1
27	1	2	1	1	1	1	2	1	1	72	2	1	9	3	1	2	2	1	1
28	1	2	2	3	2	2	1	1	2	73	1	1	9	1	1	1	2	1	1
29	2	1	2	1	2	2	2	1	2	74	1	2	2	5	2	2	1	1	2
30	1	1	2	1	2	2	1	1	2	75	1	1	13	6	2	2	2	1	2
31	1	1	2	3	2	2	1	1	2	76	1	1	6	1	1	1	2	1	1
32	2	2	8	3	1	1	1	1	1	77	1	1	1	3	1	1	1	1	1
33	1	2	2	3	2	1	1	1	1	78	1	1	2	1	1	1	2	1	1
34	1	1	9	3	2	2	1	1	2	79	1	1	1	1	1	2	2	2	1
35	1	1	11	1	1	2	1	1	1	80	2	1	9	1	2	2	2	1	2
36	1	2	12	4	1	2	2	1	1	81	1	1	10	1	2	2	2	1	2
37	1	1	4	1	2	1	1	2	2	82	2	1	12	4	1	2	2	1	1
38	1	1	9	1	2	2	1	1	2	83	2	1	7	1	1	1	2	1	1
39	2	1	2	1	2	2	1	1	2	84	1	1	5	5	1	1	2	1	1
40	2	1	8	1	2	2	1	1	2	85	1	1	2	1	1	1	1	2	1
41	1	1	10	1	2	1	2	1	2	86	1	1	2	1	1	1	2	3	2
42	1	2	4	1	1	1	1	1	1	87	2	1	6	3	1	1	2	1	1
43	1	1	8	3	2	1	2	1	2	88	2	1	5	5	2	2	1	1	2
44	1	1	2	1	2	2	1	1	1	89	1	1	9	1	1	1	1	1	2
45	1	2	2	1	1	2	2	1	1	90	1	2	12	4	1	1	2	1	1

Table B.1: The modified Wisconsin breast cancer database (continue)

j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	C_j	j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	C_j
91	1	1	1	3	1	1	1	2	1	136	2	1	2	1	1	1	2	1	1
92	1	1	4	1	1	1	2	1	1	137	1	1	12	4	1	2	2	1	1
93	1	1	7	1	1	1	2	1	1	138	1	1	11	1	2	2	2	1	2
94	1	1	2	5	2	2	2	1	2	139	1	1	2	1	2	2	1	1	2
95	2	2	6	3	1	1	2	1	1	140	2	1	2	1	2	2	1	1	2
96	1	1	9	1	1	1	2	1	2	141	1	2	1	3	2	1	1	1	1
97	2	1	4	1	1	1	2	1	1	142	2	2	12	4	1	1	2	1	1
98	1	2	11	1	1	1	2	1	1	143	1	2	12	4	1	1	1	1	1
99	2	1	4	1	1	2	1	1	1	144	1	1	8	3	2	2	1	1	2
100	2	1	6	1	2	2	2	1	2	145	2	1	6	1	1	2	2	1	1
101	2	1	2	1	1	1	2	1	1	146	2	1	13	6	2	2	2	1	2
102	1	1	1	1	2	1	1	1	2	147	1	1	13	6	2	2	1	1	2
103	1	1	9	1	2	1	1	1	2	148	2	1	6	1	1	1	2	1	1
104	1	2	10	1	2	2	2	1	2	149	1	1	7	5	2	1	1	1	1
105	1	2	12	4	1	2	2	1	1	150	2	1	13	5	2	2	2	1	2
106	2	1	8	3	2	1	1	1	2	151	1	1	2	1	1	1	2	3	2
107	1	1	8	3	2	2	2	1	2	152	1	1	2	1	2	1	1	1	1
108	1	1	4	1	1	1	2	1	1	153	1	1	13	1	1	1	2	1	1
109	2	1	1	3	2	1	1	1	2	154	1	2	2	1	1	2	2	1	1
110	1	1	2	1	1	1	1	1	1	155	1	1	1	1	2	2	2	1	2
111	1	2	1	1	1	1	2	1	1	156	1	2	11	5	1	1	1	2	2
112	1	1	2	1	1	1	1	1	1	157	1	1	6	8	1	1	2	1	2
113	1	2	5	3	2	2	1	1	1	158	1	1	13	1	1	1	2	1	1
114	2	1	4	1	2	2	2	1	2	159	1	2	9	1	2	2	2	1	2
115	1	1	5	5	1	1	1	2	1	160	1	1	8	5	1	1	1	1	1
116	1	2	11	5	2	2	1	1	2	161	1	1	2	1	1	1	2	1	1
117	2	1	10	3	2	2	1	1	2	162	2	1	7	1	1	1	1	1	1
118	1	1	8	1	2	2	1	1	2	163	1	2	2	1	2	2	1	1	2
119	1	1	6	1	2	2	1	1	2	164	1	2	5	5	2	2	1	1	2
120	1	1	2	3	2	2	2	1	2	165	1	1	5	5	2	1	2	1	1
121	1	1	2	1	2	2	2	1	2	166	1	1	7	1	1	1	1	2	1
122	1	2	2	1	1	1	1	2	1	167	1	1	12	4	1	1	1	1	1
123	2	1	12	4	1	1	2	1	1	168	2	2	8	1	1	1	2	1	2
124	2	1	2	1	2	2	2	1	2	169	1	1	4	1	1	1	2	1	1
125	2	1	10	3	2	2	1	1	2	170	1	2	4	1	2	1	2	1	1
126	1	1	9	1	1	2	2	1	1	171	2	1	11	1	2	1	2	2	1
127	2	1	6	1	2	2	2	1	2	172	2	2	2	1	2	2	1	1	2
128	1	1	2	1	1	1	2	1	1	173	1	1	5	1	1	2	1	1	1
129	2	1	3	2	2	1	2	1	2	174	1	2	2	3	2	1	2	1	1
130	1	2	7	1	2	2	1	1	2	175	2	1	6	1	2	2	1	1	2
131	2	1	4	1	2	2	1	1	2	176	1	1	8	3	2	1	1	1	1
132	2	2	13	7	1	1	2	1	1	177	2	2	12	4	1	1	1	1	1
133	1	2	12	4	1	1	2	1	1	178	2	1	12	4	1	1	1	2	1
134	2	2	2	1	1	1	2	2	1	179	1	1	10	3	2	2	2	1	2
135	1	1	12	4	2	1	2	1	1	180	1	1	6	3	2	1	1	1	2

Table B.1: The modified Wisconsin breast cancer database (continue)

j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	C_j	j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	C_j
181	1	2	9	1	1	1	2	1	1	226	1	1	1	1	2	1	2	2	1
182	2	1	1	5	2	2	1	1	1	227	1	2	5	5	1	1	2	1	1
183	1	1	10	1	2	2	1	1	2	228	1	1	13	7	2	1	2	1	2
184	1	2	7	1	2	1	1	2	2	229	1	1	10	3	2	2	1	1	2
185	1	2	9	1	2	1	1	1	2	230	2	1	6	3	2	2	1	1	2
186	1	1	10	3	1	1	1	2	1	231	2	1	6	1	2	2	2	1	2
187	1	1	2	5	2	1	1	2	1	232	1	1	9	1	1	2	1	1	1
188	1	1	9	1	1	1	2	1	1	233	1	2	8	3	2	1	1	1	1
189	2	1	9	1	1	1	1	1	1	234	1	2	2	1	2	2	2	1	2
190	2	1	3	2	1	2	2	1	1	235	2	1	12	4	2	2	2	1	2
191	1	1	1	1	1	1	1	1	1	236	2	2	8	3	2	2	2	1	2
192	1	1	4	1	2	2	1	1	2	237	2	2	5	5	1	1	2	1	1
193	2	1	5	5	2	1	1	1	1	238	2	1	12	4	1	1	2	2	1
194	1	1	2	1	1	1	2	3	1	239	1	1	1	1	1	1	1	1	1
195	1	1	2	3	2	1	2	1	1	240	1	1	11	1	2	1	2	2	2
196	1	1	2	1	2	2	2	1	2	241	2	2	13	6	2	1	2	1	2
197	2	1	6	1	2	2	2	1	2	242	1	1	1	1	2	2	2	1	2
198	1	1	6	1	1	2	2	1	1	243	1	1	5	5	2	2	1	1	2
199	2	1	12	4	1	2	2	1	1	244	1	1	2	1	2	1	2	1	1
200	1	1	4	1	1	2	1	1	1	245	1	1	7	3	1	1	2	1	1
201	2	1	2	3	2	2	2	1	2	246	2	1	5	1	1	2	1	1	1
202	2	2	3	2	1	1	2	1	1	247	2	1	4	1	1	2	1	1	1
203	1	1	14	3	2	1	1	1	2	248	1	1	12	4	2	2	1	1	2
204	2	1	2	3	2	1	1	1	1	249	1	1	1	3	2	2	1	1	2
205	2	1	6	3	2	2	2	1	2	250	1	1	5	5	1	1	2	1	2
206	1	1	9	1	2	2	2	1	2	251	1	1	4	1	1	1	1	1	1
207	1	2	3	1	1	1	2	1	1	252	1	2	4	1	1	1	1	1	1
208	2	2	1	1	1	2	2	1	1	253	1	1	14	8	2	2	1	1	2
209	1	1	2	3	1	2	1	1	1	254	1	1	9	1	1	1	2	1	1
210	1	2	11	1	1	1	1	1	1	255	2	1	6	3	2	1	2	1	2
211	2	1	1	1	2	2	2	1	2	256	1	1	2	1	2	2	2	1	2
212	1	2	1	3	2	1	2	1	2	257	1	2	7	1	1	1	1	1	2
213	1	2	2	1	1	1	1	1	1	258	1	1	11	1	2	1	1	1	2
214	1	1	1	1	2	2	1	1	1	259	1	1	11	1	1	1	1	2	1
215	2	2	12	4	1	1	1	1	1	260	1	1	2	3	1	1	1	1	1
216	2	1	6	3	2	2	2	1	2	261	1	1	4	1	1	2	2	1	1
217	1	1	2	1	1	1	2	3	1	262	1	1	6	3	2	1	2	1	2
218	1	1	5	5	1	1	2	1	1	263	2	1	2	1	1	1	1	1	1
219	1	1	1	1	1	1	2	1	1	264	2	2	4	1	2	1	1	1	2
220	1	1	6	1	2	2	1	1	2	265	1	2	9	1	2	2	2	1	2
221	2	1	2	5	2	2	1	1	2	266	2	1	2	1	1	2	1	1	1
222	1	1	2	1	1	1	2	1	1	267	1	1	2	1	1	2	1	1	1
223	1	1	2	1	2	2	1	1	2	268	1	1	10	1	2	2	1	1	2
224	1	1	11	1	1	2	1	1	1	269	1	1	5	1	1	1	1	1	1
225	1	1	5	5	2	1	2	1	1	270	1	1	4	1	1	1	1	1	1

Table B.1: The modified Wisconsin breast cancer database (continue)

j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	C_j	j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	C_j
271	2	1	5	3	1	1	2	1	1	316	1	1	1	4	1	1	2	1	1
272	1	1	2	1	2	1	2	2	2	317	2	1	10	1	2	1	1	1	1
273	1	1	2	1	1	1	2	1	1	318	2	1	4	1	2	2	1	1	2
274	2	1	6	1	1	2	2	1	2	319	2	1	2	1	1	1	2	1	1
275	1	1	9	1	2	2	2	1	2	320	2	1	2	5	2	2	2	1	2
276	1	2	5	5	1	1	1	1	1	321	1	2	9	1	2	2	1	1	2
277	2	1	5	5	2	1	1	1	1	322	1	1	10	1	2	1	2	1	2
278	1	1	11	1	2	1	2	2	1	323	1	1	12	4	2	2	1	1	2
279	1	1	10	3	2	2	2	1	2	324	1	1	1	1	2	1	2	1	2
280	1	1	9	1	2	2	2	1	2	325	1	2	1	1	2	2	1	1	2
281	1	1	5	5	1	1	1	1	1	326	1	1	8	1	2	2	2	1	2
282	1	2	13	7	1	1	2	1	1	327	1	1	1	1	1	1	1	1	1
283	1	1	11	1	2	1	2	2	1	328	2	1	5	5	1	1	1	1	1
284	1	1	1	1	1	1	1	1	1	329	1	1	5	1	1	1	2	1	1
285	1	2	4	1	1	1	1	1	1	330	1	1	7	1	1	1	1	1	1
286	1	1	8	1	2	2	1	1	2	331	2	1	4	1	2	2	2	1	2
287	1	1	13	6	2	2	1	1	2	332	1	1	9	1	2	1	2	2	2
288	1	1	2	5	2	1	2	1	2	333	1	1	2	5	2	1	1	1	2
289	1	1	12	4	1	2	2	1	1	334	1	2	9	1	1	1	1	1	1
290	2	1	12	4	1	1	2	1	1	335	2	2	2	1	1	1	1	1	1
291	1	1	13	5	2	1	1	2	1	336	1	1	5	3	1	1	1	1	1
292	2	1	12	4	1	2	1	1	1	337	2	1	2	1	2	2	2	1	2
293	1	1	2	1	2	2	2	1	2	338	1	1	12	4	1	2	2	1	1
294	1	2	8	1	2	2	1	1	2	339	2	2	2	3	2	1	2	1	2
295	1	1	10	3	2	1	1	1	2	340	1	1	4	1	2	1	1	1	1
296	1	2	12	4	1	1	2	1	1	341	1	1	7	1	1	1	1	1	1
297	2	1	12	4	1	2	1	1	1	342	1	2	10	1	2	1	1	1	1
298	1	1	2	1	1	2	2	1	1	343	2	1	6	1	2	2	1	1	2
299	1	1	8	1	2	2	1	1	2	344	1	1	6	1	2	2	2	1	2
300	1	1	2	8	1	2	1	1	1	345	1	1	2	2	1	1	1	2	2
301	1	1	10	1	2	2	1	1	2	346	2	2	6	1	2	2	1	1	2
302	1	1	2	1	1	1	1	1	1	347	2	1	13	7	2	2	1	1	2
303	1	1	7	3	1	1	1	1	1	348	2	2	6	3	2	2	2	1	2
304	1	1	7	1	1	1	2	1	1	349	2	1	13	1	2	1	1	1	2
305	2	1	6	1	1	2	1	1	1	350	1	1	2	1	1	1	2	1	1
306	1	1	10	3	2	2	1	1	2	351	2	1	8	3	2	1	2	1	2
307	2	1	9	1	1	1	2	1	1	352	1	1	4	5	1	1	1	1	1
308	1	1	2	1	2	1	1	1	2	353	1	1	8	1	2	2	1	1	2
309	2	1	2	1	1	2	1	1	1	354	1	1	4	1	2	1	1	1	1
310	1	1	2	3	1	2	1	1	1	355	1	1	1	1	1	1	2	1	1
311	1	2	13	3	1	1	2	1	1	356	1	1	2	1	1	2	1	1	1
312	2	1	3	2	2	1	1	1	1	357	1	1	2	5	2	1	1	2	1
313	1	1	6	1	2	2	2	1	1	358	1	1	9	1	1	2	2	1	1
314	2	1	6	3	2	2	2	1	2	359	1	1	10	3	2	2	2	1	2
315	1	1	7	3	2	1	1	1	2	360	1	1	2	1	1	2	2	1	1

Table B.1: The modified Wisconsin breast cancer database (continue)

j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	C_j	j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	C_j
361	1	1	13	6	2	2	1	1	2	406	2	1	2	1	2	2	1	1	2
362	2	1	9	1	2	2	1	1	2	407	1	1	5	5	2	2	2	1	2
363	2	1	11	3	1	2	1	1	1	408	2	2	4	1	1	1	2	1	1
364	2	2	2	1	1	2	2	1	1	409	1	1	13	1	2	2	2	1	2
365	1	1	7	1	1	1	1	2	1	410	1	1	4	1	1	1	2	1	1
366	2	1	4	1	1	1	1	1	1	411	1	2	1	4	1	1	2	1	1
367	1	2	9	1	1	1	2	2	1	412	2	2	12	4	1	1	2	1	1
368	1	1	2	1	2	2	2	1	2	413	2	1	3	4	2	2	2	1	2
369	2	1	12	4	1	2	2	1	1	414	1	2	2	3	2	2	1	1	2
370	1	1	11	1	2	2	2	1	2	415	2	1	2	3	2	1	2	1	2
371	2	1	4	1	2	1	2	1	2	416	2	1	1	1	1	1	1	2	1
372	1	1	9	1	1	1	2	1	1	417	2	1	4	3	2	1	2	1	1
373	1	1	12	4	1	1	2	1	1	418	1	1	8	3	2	2	2	1	2
374	2	1	6	1	1	2	1	1	1	419	2	1	12	4	1	1	2	1	1
375	1	1	8	3	1	1	1	2	2	420	2	1	10	1	2	1	2	1	2
376	2	1	5	5	2	2	2	1	2	421	1	1	7	3	2	2	2	1	2
377	1	1	11	1	2	2	1	1	2	422	1	2	2	1	1	1	1	1	2
378	1	1	1	1	1	1	2	1	1	423	1	2	1	1	1	1	2	1	1
379	2	2	3	2	1	1	2	1	1	424	2	1	6	3	2	2	2	1	2
380	1	1	9	3	2	2	2	1	2	425	1	2	2	3	1	1	2	1	1
381	1	1	11	1	2	2	1	1	2	426	1	1	2	1	2	1	2	1	1
382	1	1	5	1	1	1	1	1	1	427	1	1	11	1	2	2	2	1	2
383	1	2	2	1	1	1	2	1	1	428	1	2	2	5	2	1	1	1	1
384	1	2	4	1	1	1	2	1	1	429	2	1	9	1	1	1	1	1	1
385	2	1	6	1	2	2	1	1	2	430	1	1	9	1	2	1	1	1	1
386	1	1	2	3	2	2	2	1	2	431	2	1	6	3	2	2	2	1	2
387	2	1	3	2	1	1	1	1	1	432	1	2	10	1	1	1	1	1	1
388	1	1	9	1	2	2	2	1	2	433	2	1	6	3	1	2	1	1	1
389	1	1	2	1	2	2	2	1	2	434	1	1	9	1	2	2	1	1	2
390	2	1	6	1	2	2	2	1	2	435	1	1	13	6	2	2	2	1	1
391	1	1	12	4	2	1	1	1	2	436	1	1	2	1	2	2	1	1	2
392	1	2	11	1	1	1	1	1	1	437	1	1	9	1	2	1	1	1	1
393	2	1	12	4	1	2	2	1	1	438	1	1	6	3	2	2	1	1	2
394	2	1	10	3	2	2	2	1	2	439	1	1	2	1	2	2	2	1	2
395	2	1	2	1	1	1	2	3	2	440	1	1	1	1	1	1	1	1	1
396	1	2	7	1	1	1	1	1	1	441	1	1	8	1	2	1	2	1	2
397	2	1	6	1	2	2	2	1	1	442	1	1	5	1	1	1	1	1	1
398	2	1	6	1	1	1	1	1	1	443	1	1	10	1	1	1	2	1	1
399	1	1	2	1	1	2	2	1	1	444	1	1	2	3	2	1	2	1	2
400	2	2	8	1	1	1	2	1	1	445	1	1	6	1	1	1	2	1	1
401	2	1	7	1	1	1	2	1	1	446	1	1	6	1	2	2	1	1	2
402	1	2	9	1	2	1	1	1	2	447	1	1	8	3	2	1	2	1	2
403	1	1	2	5	1	2	1	1	1	448	1	1	13	6	1	1	2	1	1
404	1	1	9	1	1	2	2	1	1	449	1	1	11	1	1	1	1	2	1
405	1	1	6	1	1	1	2	1	1	450	2	1	4	1	1	1	2	1	1

Table B.1: The modified Wisconsin breast cancer database (continue)

j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	C_j	j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	C_j
451	1	1	9	1	1	2	2	1	1	496	2	1	10	1	1	1	2	1	2
452	1	1	4	1	2	1	1	1	1	497	1	1	2	5	2	1	2	1	2
453	2	1	2	1	1	1	2	3	2	498	2	1	2	1	2	2	1	1	2
454	1	1	5	3	2	1	2	1	2	499	2	3	10	4	1	1	1	2	2
455	2	2	6	1	2	1	1	1	1	500	1	1	2	1	2	2	2	1	2
456	1	1	13	7	1	2	2	1	1	501	1	3	12	8	1	1	1	3	2
457	1	1	5	3	2	1	1	1	2	502	1	2	11	1	1	1	2	1	1
458	2	1	8	3	2	2	1	1	2	503	1	1	11	5	2	2	1	1	2
459	1	1	5	5	1	1	1	1	1	504	1	1	13	3	2	2	2	1	2
460	1	1	5	1	2	2	2	1	2	505	1	2	9	1	2	2	1	1	2
461	1	1	1	1	1	2	1	1	1	506	1	1	6	1	2	1	1	1	2
462	2	1	4	1	2	1	1	1	2	507	2	1	1	3	1	1	2	1	1
463	1	1	6	1	1	1	2	1	1	508	1	1	9	3	1	2	2	1	1
464	1	1	7	1	2	2	2	1	2	509	1	2	9	1	1	1	2	2	1
465	2	1	12	4	1	2	2	1	1	510	1	1	9	1	2	2	1	1	1
466	1	1	6	1	2	1	1	1	2	511	1	2	14	8	1	1	1	1	1
467	2	1	12	4	2	2	1	1	2	512	1	1	2	3	1	1	2	1	1
468	1	2	5	5	2	2	1	1	1	513	1	1	9	1	1	1	1	1	1
469	1	1	6	3	2	1	1	1	2	514	2	1	6	1	2	2	1	1	2
470	2	2	1	3	1	1	1	1	1	515	1	1	5	1	1	1	2	1	1
471	1	1	2	1	2	2	1	1	2	516	2	1	7	1	2	1	1	1	1
472	1	1	9	3	2	2	2	1	2	517	2	1	1	1	2	2	2	1	2
473	1	1	6	1	2	2	1	1	2	518	2	1	5	5	2	2	2	2	2
474	1	2	12	4	1	2	2	1	1	519	1	1	10	1	2	2	1	1	2
475	1	2	10	1	1	1	2	1	1	520	2	1	12	4	1	1	2	1	1
476	1	2	9	1	2	2	1	1	1	521	1	1	9	1	2	2	1	1	2
477	1	2	1	1	1	1	2	2	1	522	2	1	9	1	1	2	2	1	1
478	2	1	2	1	1	1	2	1	1	523	2	1	6	1	2	2	1	1	1
479	2	1	6	1	2	1	1	1	2	524	2	1	4	1	2	2	2	1	2
480	1	2	4	1	2	1	2	1	1	525	1	1	8	1	2	2	2	1	2
481	2	1	10	3	2	2	2	1	2	526	1	1	2	1	1	1	2	1	1
482	1	1	2	1	2	1	2	1	1	527	1	2	9	3	2	1	1	1	2
483	1	1	2	1	1	1	2	3	1	528	2	1	6	1	1	1	1	1	1
484	2	2	12	4	1	2	1	1	1	529	2	2	12	4	1	1	2	1	1
485	1	1	6	1	2	1	2	2	2	530	2	1	2	1	1	2	1	1	1
486	1	2	2	1	2	2	1	1	2	531	1	2	9	1	1	1	1	1	1
487	1	1	11	1	2	2	1	1	2	532	2	1	6	1	1	1	2	2	1
488	1	1	12	4	2	2	2	1	2	533	1	1	11	1	1	1	2	2	1
489	1	1	5	5	2	2	2	1	2	534	1	1	9	3	2	2	2	1	2
490	1	1	11	1	2	1	2	2	1	535	1	1	4	1	2	1	1	1	1
491	1	2	4	1	2	2	1	1	2	536	2	1	6	1	2	2	2	1	2
492	1	1	10	1	2	2	2	1	2	537	1	1	5	5	1	1	2	1	1
493	1	2	12	4	1	1	2	1	1	538	1	1	2	3	1	1	1	1	2
494	2	2	4	1	2	1	2	1	1	539	2	1	6	3	2	2	2	1	2
495	1	2	2	1	1	1	2	1	1	540	1	1	10	3	2	2	2	1	2

Table B.1: The modified Wisconsin breast cancer database (continue)

j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	C_j	j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	C_j
541	2	1	5	1	1	1	2	1	1	586	1	1	2	1	2	2	1	1	2
542	1	1	5	5	1	1	1	1	1	587	1	1	6	3	2	2	2	1	1
543	1	1	11	1	2	2	1	1	2	588	2	1	1	1	2	1	1	1	2
544	1	2	2	1	1	1	2	1	1	589	1	1	9	1	2	1	1	1	2
545	1	1	6	3	2	2	1	1	1	590	1	2	8	3	1	1	2	1	1
546	1	2	2	1	1	1	2	1	1	591	1	1	10	1	1	1	1	1	1
547	1	1	12	4	1	1	2	1	1	592	1	1	2	1	2	2	2	1	2
548	1	1	11	1	1	1	2	2	1	593	1	1	12	4	1	1	2	1	1
549	2	2	5	3	1	1	1	1	1	594	1	1	2	1	2	2	1	1	2
550	1	2	7	1	2	1	2	2	1	595	1	1	9	1	1	1	2	1	1
551	1	1	9	1	2	2	1	1	2	596	1	2	5	5	1	1	1	1	1
552	1	1	11	1	2	2	2	1	2	597	1	2	2	1	2	1	2	2	2
553	2	1	4	1	1	1	2	1	1	598	1	1	9	1	2	1	1	1	2
554	1	1	12	4	2	2	2	1	1	599	1	1	11	1	1	2	2	1	1
555	2	2	12	4	2	1	2	1	1	600	1	1	11	3	2	2	1	1	2
556	1	2	5	1	1	1	1	1	1	601	2	1	5	1	1	1	2	1	1
557	2	2	2	1	1	1	2	1	1	602	1	1	6	3	2	1	1	1	2
558	1	1	2	3	2	1	2	1	2	603	1	1	8	1	2	2	2	1	2
559	1	1	12	4	1	2	2	1	1	604	2	1	6	3	2	2	2	1	2
560	1	1	2	5	2	2	2	1	2	605	2	1	10	3	2	2	1	1	2
561	2	2	2	1	1	1	1	1	1	606	2	2	12	4	1	1	2	1	1
562	1	1	6	3	2	2	2	1	2	607	1	2	5	3	1	1	1	2	1
563	1	2	4	1	2	1	2	1	1	608	1	1	2	1	2	2	2	1	2
564	1	1	10	5	1	1	2	1	1	609	2	1	9	1	1	1	1	1	1
565	1	1	7	5	2	2	1	1	2	610	1	1	9	1	2	1	2	1	2
566	2	1	6	1	2	2	1	1	2	611	1	1	2	5	2	2	2	1	2
567	2	1	2	1	2	1	2	1	2	612	1	2	7	1	1	1	2	1	1
568	1	1	9	5	2	1	1	1	1	613	1	1	5	5	2	1	1	1	2
569	2	1	1	3	1	1	1	2	1	614	1	2	11	1	1	1	2	2	1
570	1	2	11	1	1	1	1	2	1	615	1	1	2	3	2	2	2	1	2
571	1	1	6	3	1	1	1	1	1	616	1	1	4	1	2	2	2	1	2
572	1	1	11	1	1	2	2	1	1	617	2	1	2	1	1	1	2	1	1
573	2	1	5	5	2	1	1	1	2	618	2	1	11	1	1	2	1	1	1
574	2	1	9	1	1	1	1	1	1	619	1	2	10	1	1	1	2	1	1
575	1	1	8	1	2	2	1	1	2	620	1	1	8	1	2	2	2	1	2
576	2	1	8	3	2	2	2	1	2	621	1	1	5	5	1	1	1	1	1
577	2	2	10	1	1	1	1	1	1	622	2	1	6	1	2	2	2	1	2
578	2	1	4	3	1	1	2	1	1	623	2	1	6	1	2	2	2	1	2
579	2	1	6	1	2	2	2	1	1	624	1	1	2	1	1	1	1	1	1
580	1	1	5	5	1	1	2	2	1	625	1	1	9	1	2	1	1	1	2
581	1	2	10	3	1	1	2	1	1	626	1	2	4	1	2	1	2	1	1
582	1	1	1	3	2	1	1	1	2	627	2	1	6	1	2	1	1	1	1
583	2	2	8	3	2	2	2	1	2	628	1	2	2	1	1	1	2	1	1
584	1	1	11	1	2	1	1	1	2	629	2	2	12	4	2	2	2	1	1
585	1	2	2	3	1	1	2	1	1	630	1	1	11	1	1	1	1	1	1

Table B.1: The modified Wisconsin breast cancer database (continue)

j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	C_j	j	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	C_j
631	1	2	13	3	2	1	1	1	2	661	1	2	10	1	2	2	2	1	2
632	1	1	5	5	1	1	2	1	1	662	2	1	5	1	1	1	1	2	1
633	2	1	9	1	2	1	2	2	2	663	1	2	4	1	2	2	2	1	2
634	1	1	9	1	1	2	2	1	1	664	2	1	4	1	2	1	2	1	2
635	1	1	12	4	1	1	2	1	1	665	1	1	7	1	1	1	2	1	1
636	1	2	1	1	2	2	2	1	1	666	1	2	5	1	1	1	1	1	1
637	1	1	2	1	1	1	2	1	1	667	1	2	2	5	1	1	2	1	1
638	1	1	5	3	1	1	1	1	1	668	2	1	12	4	2	1	2	1	1
639	1	1	7	1	1	1	1	1	1	669	2	1	2	1	1	1	1	1	1
640	1	1	4	1	2	1	2	2	2	670	1	1	2	1	1	1	2	2	1
641	1	1	12	4	1	1	2	1	1	671	2	1	6	1	2	1	1	1	2
642	2	2	6	3	2	2	1	1	2	672	1	2	2	3	2	1	2	1	2
643	1	1	8	3	2	2	1	1	2	673	2	1	12	4	1	1	2	1	1
644	1	1	2	1	1	1	2	2	1	674	1	1	6	3	2	1	1	1	2
645	1	1	6	1	2	2	2	1	1	675	2	1	2	3	2	1	1	1	2
646	1	1	7	3	1	1	1	2	1	676	1	2	1	3	1	1	1	1	1
647	2	1	2	1	2	2	1	1	1	677	2	1	4	1	1	1	2	1	1
648	1	1	10	1	1	1	2	1	1	678	1	1	11	1	2	1	1	2	1
649	1	2	2	1	2	1	1	1	1	679	1	2	2	1	1	1	1	1	1
650	1	1	6	3	2	2	2	1	2	680	1	2	1	1	2	2	2	1	2
651	1	1	6	1	2	1	1	1	2	681	1	1	2	1	1	1	1	2	1
652	1	2	1	1	1	2	1	1	1	682	1	2	2	1	2	1	2	1	1
653	1	1	13	7	1	1	1	1	1	683	2	1	10	1	2	1	2	1	2
654	1	1	2	3	2	2	1	1	2	684	2	1	5	8	1	1	2	1	1
655	1	1	7	1	2	1	1	2	1	685	1	2	10	3	2	2	2	1	2
656	2	1	7	1	1	1	1	1	1	686	1	1	8	1	2	1	2	1	2
657	1	2	8	1	2	2	1	1	2	687	1	1	2	1	1	1	2	1	1
658	1	1	2	1	1	1	2	1	1	688	2	1	4	1	2	1	2	1	2
659	2	2	3	4	1	1	2	1	1	689	2	1	8	3	2	2	2	1	2
660	1	1	2	1	1	1	2	1	2	690	1	1	13	1	1	2	2	2	2

B.3 Conclusion

In this appendix, we have presented the data used for simulation namely the modified Wisconsin breast cancer database. Note that for each simulation, 90% from the modified Wisconsin breast cancer database and its corresponding bba's (see Appendix C) will be chosen as the training set, the remaining instances will be used as testing objects. This operation is then repeated ten times where in each trial we change the 10% given to the testing set.

In the next appendix, we give an example of bba's created for the objects of this data set and presenting beliefs on their classes.

Appendix C

Artificial bba's

C.1 Introduction

In this appendix, we present one set of bba's given to the classes of the instances of the modified Wisconsin breast cancer database and which are created artificially as described in Chapter 6 in the part 2 of this thesis.

C.2 Artificial bba's

In Table C.1, we present the bba created for the class of each object of the modified Wisconsin breast cancer database. These bba's created artificially based on a probability $P = 0.2$ (used in the *'the class destruction'*).

For each instance in the modified Wisconsin breast cancer database, we present bbm's on the different possible classes of the power set of Θ.

Let $\Theta = \{C_1, C_2\}$
where C_1 and C_2 are denoted respectively by 1 and 2 in Appendix B.

So, $2^{\Theta} = \{\emptyset, C_1, C_2, \Theta\}$

Let's remind that j represents the index of the object I_j, it varies from 1 to 690.

Table C.1: Artificial bba's

j	$\emptyset$	C_1	C_2	Θ	j	$\emptyset$	C_1	C_2	Θ
1	0	0.39222	0	0.60778	46	0	0	0.084403	0.9156
2	0	0.050737	0	0.94926	47	0	0.11867	0	0.88133
3	0	0.32977	0	0.67023	48	0	0	0.46353	0.53647
4	0	0	0.22607	0.77393	49	0	0	0	1
5	0	0	0.39091	0.60909	50	0	0	0.17461	0.82539
6	0	0	0.057531	0.94247	51	0	0.24932	0	0.75068
7	0	0	0	1	52	0	0.42423	0	0.57577
8	0	0	0	1	53	0	0	0	1
9	0	0.32893	0	0.67107	54	0	0.073999	0	0.926
10	0	0	0	1	55	0	0.13112	0	0.86888
11	0	0	0	1	56	0	0	0.26904	0.73096
12	0	0	0.43695	0.56305	57	0	0	0.30744	0.69256
13	0	0	0	1	58	0	0.45192	0	0.54808
14	0	0	0.015643	0.98436	59	0	0	0.39548	0.60452
15	0	0.42087	0	0.57913	60	0	0	0	1
16	0	0	0.31665	0.68335	61	0	0.41501	0	0.58499
17	0	0	0.2112	0.7888	62	0	0.11284	0	0.88716
18	0	0	0.27189	0.72811	63	0	0	0.27685	0.72315
19	0	0.26393	0	0.73607	64	0	0.24714	0	0.75286
20	0	0	0	1	65	0	0	0.16897	0.83103
21	0	0.10549	0	0.89451	66	0	0	0.36364	0.63636
22	0	0.12414	0	0.87586	67	0	0.38051	0	0.61949
23	0	0.23966	0	0.76034	68	0	0.12477	0	0.87523
24	0	0.22423	0	0.77577	69	0	0.087605	0	0.9124
25	0	0.38958	0	0.61042	70	0	0	0.38982	0.61018
26	0	0	0.47579	0.52421	71	0	0	0	1
27	0	0	0	1	72	0	0.21763	0	0.78237
28	0	0	0	1	73	0	0.22759	0	0.77241
29	0	0	0.47764	0.52236	74	0	0	0	1
30	0	0	0.21587	0.78413	75	0	0	0.38621	0.61379
31	0	0	0.24071	0.75929	76	0	0	0	1
32	0	0	0	1	77	0	0.49969	0	0.50031
33	0	0.47712	0	0.52288	78	0	0.10545	0	0.89455
34	0	0	0.27806	0.72194	79	0	0.4911	0	0.5089
35	0	0.15528	0	0.84472	80	0	0	0.16001	0.83999
36	0	0.039658	0	0.96034	81	0	0	0	1
37	0	0	0.23437	0.76563	82	0	0.16508	0	0.83492
38	0	0	0.38999	0.61001	83	0	0.073789	0	0.92621
39	0	0	0.40482	0.59518	84	0	0.0039525	0	0.99605
40	0	0	0	1	85	0	0.43965	0	0.56035
41	0	0	0.016606	0.98339	86	0	0	0.15445	0.84555
42	0	0	0	1	87	0	0	0	1
43	0	0	0.023255	0.97675	88	0	0	0	1
44	0	0	0	1	89	0	0	0	1
45	0	0.20755	0	0.79245	90	0	0.26494	0	0.73506

Table C.1: Artificial bba's (continue)

j	$\emptyset$	C_1	C_2	Θ	j	$\emptyset$	C_1	C_2	Θ
91	0	0	0	1	136	0	0	0	1
92	0	0.20904	0	0.79096	137	0	0.4294	0	0.5706
93	0	0	0	1	138	0	0	0	1
94	0	0	0.28962	0.71038	139	0	0	0	1
95	0	0.32415	0	0.67585	140	0	0	0.089261	0.91074
96	0	0	0.4832	0.5168	141	0	0.35035	0	0.64965
97	0	0.30171	0	0.69829	142	0	0	0	1
98	0	0.22459	0	0.77541	143	0	0.46903	0	0.53097
99	0	0.20248	0	0.79752	144	0	0	0.45832	0.54168
100	0	0	0	1	145	0	0.33784	0	0.66216
101	0	0.30248	0	0.69752	146	0	0	0	1
102	0	0	0.38875	0.61125	147	0	0	0.43844	0.56156
103	0	0	0.12487	0.87513	148	0	0	0	1
104	0	0	0	1	149	0	0.27567	0	0.72433
105	0	0.40355	0	0.59645	150	0	0	0.058592	0.94141
106	0	0	0.00033529	0.99966	151	0	0	0.13652	0.86348
107	0	0	0.12333	0.87667	152	0	0	0	1
108	0	0.1938	0	0.8062	153	0	0.32652	0	0.67348
109	0	0	0.28287	0.71713	154	0	0	0	1
110	0	0.1922	0	0.8078	155	0	0	0.053004	0.947
111	0	0.17976	0	0.82024	156	0	0	0	1
112	0	0.16318	0	0.83682	157	0	0	0.016854	0.98315
113	0	0.18462	0	0.81538	158	0	0.10709	0	0.89291
114	0	0	0.36076	0.63924	159	0	0	0.33934	0.66066
115	0	0	0	1	160	0	0.066463	0	0.93354
116	0	0	0.43431	0.56569	161	0	0.40931	0	0.59069
117	0	0	0	1	162	0	0.27933	0	0.72067
118	0	0	0	1	163	0	0	0.4073	0.5927
119	0	0	0.43069	0.56931	164	0	0	0.098269	0.90173
120	0	0	0.28859	0.71141	165	0	0.11932	0	0.88068
121	0	0	0.4645	0.5355	166	0	0.030854	0	0.96915
122	0	0.31695	0	0.68305	167	0	0.3522	0	0.6478
123	0	0.21869	0	0.78131	168	0	0	0.38855	0.61145
124	0	0	0.012017	0.98798	169	0	0.45604	0	0.54396
125	0	0	0.096707	0.90329	170	0	0.12446	0	0.87554
126	0	0.3693	0	0.6307	171	0	0.088901	0	0.9111
127	0	0	0.04103	0.95897	172	0	0	0.45371	0.54629
128	0	0.088118	0	0.91188	173	0	0.2837	0	0.7163
129	0	0	0.3711	0.6289	174	0	0.24857	0	0.75143
130	0	0	0.3247	0.6753	175	0	0	0.36554	0.63446
131	0	0	0.39945	0.60055	176	0	0.23337	0	0.76663
132	0	0.11527	0	0.88473	177	0	0	0	1
133	0	0.087778	0	0.91222	178	0	0.084831	0	0.91517
134	0	0.27812	0	0.72188	179	0	0	0.19127	0.80873
135	0	0.23934	0	0.76066	180	0	0	0.13574	0.86426

Table C.1: Artificial bba's (continue)

j	$\emptyset$	C_1	C_2	Θ	j	$\emptyset$	C_1	C_2	Θ
181	0	0.42018	0	0.57982	226	0	0.45316	0	0.54684
182	0	0	0	1	227	0	0.14429	0	0.85571
183	0	0	0.22856	0.77144	228	0	0	0	1
184	0	0	0.12709	0.87291	229	0	0	0.45595	0.54405
185	0	0	0.31808	0.68192	230	0	0	0.38116	0.61884
186	0	0.45917	0	0.54083	231	0	0	0.37512	0.62488
187	0	0.43996	0	0.56004	232	0	0.16676	0	0.83324
188	0	0.21022	0	0.78978	233	0	0.29722	0	0.70278
189	0	0.083944	0	0.91606	234	0	0	0	1
190	0	0.18058	0	0.81942	235	0	0	0.46521	0.53479
191	0	0.071885	0	0.92812	236	0	0	0	1
192	0	0	0.19582	0.80418	237	0	0.13056	0	0.86944
193	0	0.44787	0	0.55213	238	0	0.12998	0	0.87002
194	0	0.23291	0	0.76709	239	0	0.15564	0	0.84436
195	0	0.23605	0	0.76395	240	0	0	0.16149	0.83851
196	0	0	0.031831	0.96817	241	0	0	0.38977	0.61023
197	0	0	0.34498	0.65502	242	0	0	0	1
198	0	0	0	1	243	0	0	0.41518	0.58482
199	0	0.12536	0	0.87464	244	0	0.074053	0	0.92595
200	0	0.49357	0	0.50643	245	0	0.28632	0	0.71368
201	0	0	0.3166	0.6834	246	0	0.30223	0	0.69777
202	0	0.22764	0	0.77236	247	0	0.13483	0	0.86517
203	0	0	0.46193	0.53807	248	0	0	0.16061	0.83939
204	0	0	0	1	249	0	0	0.13825	0.86175
205	0	0	0.05446	0.94554	250	0	0	0.30404	0.69596
206	0	0	0.046559	0.95344	251	0	0.28064	0	0.71936
207	0	0.43782	0	0.56218	252	0	0	0	1
208	0	0.045798	0	0.9542	253	0	0	0.38692	0.61308
209	0	0.10404	0	0.89596	254	0	0.032364	0	0.96764
210	0	0	0	1	255	0	0	0	1
211	0	0	0.38047	0.61953	256	0	0	0.24604	0.75396
212	0	0	0.20833	0.79167	257	0	0	0.04996	0.95004
213	0	0.32907	0	0.67093	258	0	0	0.38892	0.61108
214	0	0	0	1	259	0	0.092182	0	0.90782
215	0	0	0	1	260	0	0.25232	0	0.74768
216	0	0	0	1	261	0	0.42127	0	0.57873
217	0	0.024254	0	0.97575	262	0	0	0.23279	0.76721
218	0	0.23255	0	0.76745	263	0	0	0	1
219	0	0	0	1	264	0	0	0.11914	0.88086
220	0	0	0.25385	0.74615	265	0	0	0.30737	0.69263
221	0	0	0	1	266	0	0.30803	0	0.69197
222	0	0.11698	0	0.88302	267	0	0.37291	0	0.62709
223	0	0	0.015643	0.98436	268	0	0	0.42246	0.57754
224	0	0	0	1	269	0	0.07262	0	0.92738
225	0	0	0	1	270	0	0.27606	0	0.72394

Table C.1: Artificial bba's (continue)

j	$\emptyset$	C_1	C_2	Θ	j	$\emptyset$	C_1	C_2	Θ
271	0	0.28331	0	0.71669	316	0	0.4204	0	0.5796
272	0	0	0.16819	0.83181	317	0	0.20501	0	0.79499
273	0	0	0	1	318	0	0	0.39353	0.60647
274	0	0	0.093914	0.90609	319	0	0.04813	0	0.95187
275	0	0	0.048991	0.95101	320	0	0	0.3056	0.6944
276	0	0	0	1	321	0	0	0.18557	0.81443
277	0	0	0	1	322	0	0	0.33965	0.66035
278	0	0	0	1	323	0	0	0	1
279	0	0	0.16983	0.83017	324	0	0	0.20042	0.79958
280	0	0	0.07309	0.92691	325	0	0	0.37343	0.62657
281	0	0.016147	0	0.98385	326	0	0	0.26528	0.73472
282	0	0.025993	0	0.97401	327	0	0	0	1
283	0	0.18847	0	0.81153	328	0	0.40287	0	0.59713
284	0	0	0	1	329	0	0.4873	0	0.5127
285	0	0.2702	0	0.7298	330	0	0.047147	0	0.95285
286	0	0	0.028351	0.97165	331	0	0	0	1
287	0	0	0	1	332	0	0	0.35366	0.64634
288	0	0	0.44275	0.55725	333	0	0	0	1
289	0	0.12391	0	0.87609	334	0	0.25982	0	0.74018
290	0	0.42363	0	0.57637	335	0	0.0049808	0	0.99502
291	0	0.14452	0	0.85548	336	0	0.19176	0	0.80824
292	0	0	0	1	337	0	0	0.14852	0.85148
293	0	0	0.043314	0.95669	338	0	0.32591	0	0.67409
294	0	0	0.091194	0.90881	339	0	0	0.47097	0.52903
295	0	0	0.49189	0.50811	340	0	0.3248	0	0.6752
296	0	0.081709	0	0.91829	341	0	0	0	1
297	0	0.12215	0	0.87785	342	0	0.39135	0	0.60865
298	0	0.45486	0	0.54514	343	0	0	0	1
299	0	0	0.13114	0.86886	344	0	0	0	1
300	0	0.068733	0	0.93127	345	0	0	0	1
301	0	0	0.018467	0.98153	346	0	0	0	1
302	0	0	0	1	347	0	0	0.32942	0.67058
303	0	0.023578	0	0.97642	348	0	0	0.049205	0.95079
304	0	0	0	1	349	0	0	0.35444	0.64556
305	0	0.10115	0	0.89885	350	0	0.03709	0	0.96291
306	0	0	0.30959	0.69041	351	0	0	0.4036	0.5964
307	0	0	0	1	352	0	0	0	1
308	0	0	0.45647	0.54353	353	0	0	0.016046	0.98395
309	0	0	0	1	354	0	0.19247	0	0.80753
310	0	0	0	1	355	0	0.062149	0	0.93785
311	0	0	0	1	356	0	0.35917	0	0.64083
312	0	0.45627	0	0.54373	357	0	0.2895	0	0.7105
313	0	0.070902	0	0.9291	358	0	0	0	1
314	0	0	0	1	359	0	0	0.43548	0.56452
315	0	0	0.06358	0.93642	360	0	0.42898	0	0.57102

Table C.1: Artificial bba's (continue)

j	$\emptyset$	C_1	C_2	Θ	j	$\emptyset$	C_1	C_2	Θ
361	0	0	0.26172	0.73828	406	0	0	0.41232	0.58768
362	0	0	0.12621	0.87379	407	0	0	0.33642	0.66358
363	0	0	0	1	408	0	0.34139	0	0.65861
364	0	0.3086	0	0.6914	409	0	0	0.24904	0.75096
365	0	0.48352	0	0.51648	410	0	0	0	1
366	0	0.22402	0	0.77598	411	0	0.078872	0	0.92113
367	0	0	0	1	412	0	0.25798	0	0.74202
368	0	0	0.3925	0.6075	413	0	0	0.4854	0.5146
369	0	0	0	1	414	0	0	0.46546	0.53454
370	0	0	0	1	415	0	0	0.34073	0.65927
371	0	0	0.14775	0.85225	416	0	0.47258	0	0.52742
372	0	0.48071	0	0.51929	417	0	0	0	1
373	0	0.31755	0	0.68245	418	0	0	0.39584	0.60416
374	0	0	0	1	419	0	0	0	1
375	0	0	0	1	420	0	0	0.26423	0.73577
376	0	0	0.33262	0.66738	421	0	0	0	1
377	0	0	0	1	422	0	0	0.4321	0.5679
378	0	0.37778	0	0.62222	423	0	0.24422	0	0.75578
379	0	0.39878	0	0.60122	424	0	0	0.18497	0.81503
380	0	0	0.14469	0.85531	425	0	0.45646	0	0.54354
381	0	0	0.40877	0.59123	426	0	0.12054	0	0.87946
382	0	0.14869	0	0.85131	427	0	0	0.31401	0.68599
383	0	0.41139	0	0.58861	428	0	0.41231	0	0.58769
384	0	0	0	1	429	0	0.071138	0	0.92886
385	0	0	0.18998	0.81002	430	0	0.25638	0	0.74362
386	0	0	0.32904	0.67096	431	0	0	0.42092	0.57908
387	0	0.48803	0	0.51197	432	0	0	0	1
388	0	0	0.4015	0.5985	433	0	0.044507	0	0.95549
389	0	0	0.44776	0.55224	434	0	0	0.33185	0.66815
390	0	0	0.30144	0.69856	435	0	0.019734	0	0.98027
391	0	0	0.32337	0.67663	436	0	0	0.42937	0.57063
392	0	0.059526	0	0.94047	437	0	0	0	1
393	0	0.067756	0	0.93224	438	0	0	0.18215	0.81785
394	0	0	0.088842	0.91116	439	0	0	0.35173	0.64827
395	0	0	0.20641	0.79359	440	0	0	0	1
396	0	0.22992	0	0.77008	441	0	0	0.3898	0.6102
397	0	0.0744	0	0.9256	442	0	0.091924	0	0.90808
398	0	0.33279	0	0.66721	443	0	0.094819	0	0.90518
399	0	0	0	1	444	0	0	0.42137	0.57863
400	0	0.17445	0	0.82555	445	0	0.064288	0	0.93571
401	0	0.026915	0	0.97309	446	0	0	0.044142	0.95586
402	0	0	0.12069	0.87931	447	0	0	0.36595	0.63405
403	0	0.032516	0	0.96748	448	0	0	0	1
404	0	0.080649	0	0.91935	449	0	0.44152	0	0.55848
405	0	0.12441	0	0.87559	450	0	0.13409	0	0.86591

Table C.1: Artificial bba's (continue)

j	$\emptyset$	C_1	C_2	Θ	j	$\emptyset$	C_1	C_2	Θ
451	0	0.14816	0	0.85184	496	0	0	0.46841	0.53159
452	0	0.059309	0	0.94069	497	0	0	0	1
453	0	0	0.19722	0.80278	498	0	0	0	1
454	0	0	0	1	499	0	0	0.15032	0.84968
455	0	0.17179	0	0.82821	500	0	0	0.47472	0.52528
456	0	0.27883	0	0.72117	501	0	0	0.36194	0.63806
457	0	0	0.21804	0.78196	502	0	0.070085	0	0.92991
458	0	0	0.18288	0.81712	503	0	0	0.44966	0.55034
459	0	0	0	1	504	0	0	0.3608	0.6392
460	0	0	0.12367	0.87633	505	0	0	0	1
461	0	0.0064721	0	0.99353	506	0	0	0.46015	0.53985
462	0	0	0.18547	0.81453	507	0	0.35682	0	0.64318
463	0	0.40613	0	0.59387	508	0	0.35679	0	0.64321
464	0	0	0.068199	0.9318	509	0	0.29209	0	0.70791
465	0	0	0	1	510	0	0.063427	0	0.93657
466	0	0	0.47889	0.52111	511	0	0.0027626	0	0.99724
467	0	0	0	1	512	0	0.36175	0	0.63825
468	0	0.12365	0	0.87635	513	0	0	0	1
469	0	0	0.018593	0.98141	514	0	0	0.38298	0.61702
470	0	0.46513	0	0.53487	515	0	0	0	1
471	0	0	0.11242	0.88758	516	0	0.15952	0	0.84048
472	0	0	0.049602	0.9504	517	0	0	0.37456	0.62544
473	0	0	0	1	518	0	0	0.13045	0.86955
474	0	0	0	1	519	0	0	0.42239	0.57761
475	0	0.36858	0	0.63142	520	0	0.44076	0	0.55924
476	0	0	0	1	521	0	0	0.48933	0.51067
477	0	0.28238	0	0.71762	522	0	0.47958	0	0.52042
478	0	0.15381	0	0.84619	523	0	0.089666	0	0.91033
479	0	0	0.33859	0.66141	524	0	0	0.26998	0.73002
480	0	0.16273	0	0.83727	525	0	0	0.48222	0.51778
481	0	0	0.031696	0.9683	526	0	0.091649	0	0.90835
482	0	0	0	1	527	0	0	0.49549	0.50451
483	0	0.13143	0	0.86857	528	0	0	0	1
484	0	0.48207	0	0.51793	529	0	0.23947	0	0.76053
485	0	0	0.18343	0.81657	530	0	0.095585	0	0.90442
486	0	0	0.36393	0.63607	531	0	0.12809	0	0.87191
487	0	0	0.11255	0.88745	532	0	0.43592	0	0.56408
488	0	0	0.023577	0.97642	533	0	0.034413	0	0.96559
489	0	0	0.41981	0.58019	534	0	0	0	1
490	0	0	0	1	535	0	0	0	1
491	0	0	0.44745	0.55255	536	0	0	0.43592	0.56408
492	0	0	0.041084	0.95892	537	0	0.36974	0	0.63026
493	0	0.41525	0	0.58475	538	0	0	0.12289	0.87711
494	0	0.24102	0	0.75898	539	0	0	0.12905	0.87095
495	0	0.13241	0	0.86759	540	0	0	0.29904	0.70096

Table C.1: Artificial bba's (continue)

j	$\emptyset$	C_1	C_2	Θ	j	$\emptyset$	C_1	C_2	Θ
541	0	0.42892	0	0.57108	586	0	0	0.4345	0.5655
542	0	0.47648	0	0.52352	587	0	0.059733	0	0.94027
543	0	0	0.41995	0.58005	588	0	0	0.039853	0.96015
544	0	0	0	1	589	0	0	0.10594	0.89406
545	0	0	0	1	590	0	0.14099	0	0.85901
546	0	0.12013	0	0.87987	591	0	0	0	1
547	0	0.12429	0	0.87571	592	0	0	0.19128	0.80872
548	0	0.43024	0	0.56976	593	0	0.25201	0	0.74799
549	0	0.16795	0	0.83205	594	0	0	0.23152	0.76848
550	0	0.029464	0	0.97054	595	0	0.047396	0	0.9526
551	0	0	0	1	596	0	0	0	1
552	0	0	0.48069	0.51931	597	0	0	0.15746	0.84254
553	0	0.08093	0	0.91907	598	0	0	0.38062	0.61938
554	0	0.0725	0	0.9275	599	0	0	0	1
555	0	0.22623	0	0.77377	600	0	0	0.45547	0.54453
556	0	0.1321	0	0.8679	601	0	0.48343	0	0.51657
557	0	0.15305	0	0.84695	602	0	0	0.22925	0.77075
558	0	0	0	1	603	0	0	0.15894	0.84106
559	0	0	0	1	604	0	0	0.34056	0.65944
560	0	0	0.48224	0.51776	605	0	0	0.44733	0.55267
561	0	0.33058	0	0.66942	606	0	0.48563	0	0.51437
562	0	0	0.40404	0.59596	607	0	0.051277	0	0.94872
563	0	0.10791	0	0.89209	608	0	0	0.18953	0.81047
564	0	0.30123	0	0.69877	609	0	0.47623	0	0.52377
565	0	0	0.03029	0.96971	610	0	0	0.047619	0.95238
566	0	0	0.00025522	0.99974	611	0	0	0	1
567	0	0	0.2276	0.7724	612	0	0.22025	0	0.77975
568	0	0	0	1	613	0	0	0	1
569	0	0.11591	0	0.88409	614	0	0.24055	0	0.75945
570	0	0.25322	0	0.74678	615	0	0	0.14661	0.85339
571	0	0.25589	0	0.74411	616	0	0	0.0019404	0.99806
572	0	0.3621	0	0.6379	617	0	0.41687	0	0.58313
573	0	0	0.097985	0.90202	618	0	0	0	1
574	0	0.28907	0	0.71093	619	0	0	0	1
575	0	0	0.13852	0.86148	620	0	0	0.22727	0.77273
576	0	0	0.17664	0.82336	621	0	0.021485	0	0.97851
577	0	0.13498	0	0.86502	622	0	0	0.41116	0.58884
578	0	0.35614	0	0.64386	623	0	0	0.407	0.593
579	0	0.32659	0	0.67341	624	0	0.1103	0	0.8897
580	0	0	0	1	625	0	0	0	1
581	0	0.33225	0	0.66775	626	0	0.45919	0	0.54081
582	0	0	0.11957	0.88043	627	0	0.26326	0	0.73674
583	0	0	0.43323	0.56677	628	0	0.161	0	0.839
584	0	0	0.37475	0.62525	629	0	0.020484	0	0.97952
585	0	0.18137	0	0.81863	630	0	0.2155	0	0.7845

Table C.1: Artificial bba's (continue)

j	$\emptyset$	C_1	C_2	Θ	j	$\emptyset$	C_1	C_2	Θ
631	0	0	0.45345	0.54655	661	0	0	0.049759	0.95024
632	0	0.16123	0	0.83877	662	0	0.49905	0	0.50095
633	0	0	0.17179	0.82821	663	0	0	0.064157	0.93584
634	0	0	0	1	664	0	0	0.29198	0.70802
635	0	0.47622	0	0.52378	665	0	0.42954	0	0.57046
636	0	0	0	1	666	0	0.087466	0	0.91253
637	0	0.040406	0	0.95959	667	0	0.058504	0	0.9415
638	0	0.46007	0	0.53993	668	0	0.35039	0	0.64961
639	0	0.16708	0	0.83292	669	0	0	0	1
640	0	0	0.34709	0.65291	670	0	0.12734	0	0.87266
641	0	0.41861	0	0.58139	671	0	0	0.089757	0.91024
642	0	0	0.40357	0.59643	672	0	0	0	1
643	0	0	0	1	673	0	0.3247	0	0.6753
644	0	0.3462	0	0.6538	674	0	0	0.051594	0.94841
645	0	0.1599	0	0.8401	675	0	0	0.076136	0.92386
646	0	0.35979	0	0.64021	676	0	0.31746	0	0.68254
647	0	0.11887	0	0.88113	677	0	0.20413	0	0.79587
648	0	0.12032	0	0.87968	678	0	0.08175	0	0.91825
649	0	0.27261	0	0.72739	679	0	0	0	1
650	0	0	0.46008	0.53992	680	0	0	0.026377	0.97362
651	0	0	0.37272	0.62728	681	0	0.075249	0	0.92475
652	0	0.39567	0	0.60433	682	0	0.2671	0	0.7329
653	0	0.49911	0	0.50089	683	0	0	0.41465	0.58535
654	0	0	0.26081	0.73919	684	0	0.0054822	0	0.99452
655	0	0.496	0	0.504	685	0	0	0.21526	0.78474
656	0	0	0	1	686	0	0	0.072293	0.92771
657	0	0	0.40901	0.59099	687	0	0.31798	0	0.68202
658	0	0.4989	0	0.5011	688	0	0	0.4681	0.5319
659	0	0.19302	0	0.80698	689	0	0	0.49911	0.50089
660	0	0	0.35489	0.64511	690	0	0	0.10381	0.89619

C.3 Conclusion

In this appendix, we have detailed one set of bba's assigned to the classes of the instances of the modified Wisconsin breast cancer database that we have used for making simulation.

The set of bba's illustrated in this appendix is relative to the value of the probability P equals to 0.2. Obviously for performing simulations presented in Chapter 6, we have developed a set of 690 bba's, for each value of the probability P (from 0 to 0.9).

Bibiliography

Baim, P. W. (1988). A method for attribute selection in inductive learning systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *10*(6), 888–896.

Barnett, J. A. (1991). Calculating Dempster-Shafer plausibility. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *13*(6), 599–602.

Breiman, L., Friedman, J. H., Olshen, R., & Stone, C. J. (1984). *Classification and regression trees.* Monterey, CA: Wadsworth & Brooks.

Cooke, R. M. (1991). *Experts in uncertainty.* Oxford Univ. Press, New York.

De Smet, Y. (1998). *Application de la théorie des fonctions de croyances aux problèmes de classification.* Master's thesis, Université Libre de Bruxelles, Belgique.

Decaestecker, C. (1997). *Dévelopements méthodologiques pour la classification de données réelles: Application à l'aide au diagnostic et au pronostic de tumeurs gliales.* PhD thesis, Faculté des Sciences et Faculté de Medecine, Université Libre de Bruxelles, Belgique.

Delmotte, F., & Smets, P. (2002a). Target identification based on the transferable belief model interpretation of Dempster-Shafer model. Part I: Methodology. *Submitted for publication.*

Delmotte, F., & Smets, P. (2002b). Target identification based on the transferable belief model interpretation of Dempster-Shafer model. Part II: Applications. *Submitted for publication.*

Dempster, A. P. (1967). Upper and lower probabilities induced by a multiple valued mapping. *Annals of Mathematical Statistics*, *38*, 325–339.

Dempster, A. P. (1968). A generalization of Bayesian inference. *Journal of the Royal Statistical Society, Series B*, *30*, 205-247.

Denœux, T. (1995). A k-nearest neighbor classification rule based on Dempster-Shafer theory. *IEEE Transactions on Systems, Man and Cybernetics*, *25*(5), 804-813.

Denœux, T. (1997). Analysis of evidence theoretic decision rules for pattern classification. *Pattern Recognition*, *30*(7), 1095–1107.

Denœux, T. (1999). Reasoning with imprecise belief structures. *International*

Journal of Approximate Reasoning, *20*, 79-111.

Denœux, T. (2000). A neural network classifier based on Dempster-Shafer theory. *IEEE Transactions on Systems, Man and Cybernetics - Part A*, *30*(2), 131–150.

Denœux, T., & Skarstein-Bjanger, M. (2000). Induction of decision trees for partially classified data. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics* (pp. 2923–2928). Nashville, USA: IEEE.

Elouedi, Z., & Mellouli, K. (1998a). Evaluation and ranking scenarios using belief function model. In *Proceedings of Computational Engineering in Systems Applications* (pp. 102–106). Hammamet, Tunisia: CESA'98, IMACS Multiconference.

Elouedi, Z., & Mellouli, K. (1998b). Pooling dependent expert opinions using the theory of evidence. In *Proceedings of the Seventh International Conference on Information Processing and Management of Uncertainty* (Vol. 1, pp. 32–39). Paris, France: IPMU'98.

Elouedi, Z., & Mellouli, K. (2000). Classifying scenarios with belief decision trees. In S. Arikawa & S. Morishita (Eds.), *Proceedings of the Third International Conference on Discovery Science* (pp. 127–140). Kyoto, Japan: DS 2000, Lecture Notes in AI 1967, Springer-Verlag.

Elouedi, Z., Mellouli, K., & Smets, P. (2000a). Classification with belief decision trees. In S. A. Cerri & D. Dochev (Eds.), *Proceedings of the Nineth International Conference on Artificial Intelligence: Methodology, Systems, Architectures.* (pp. 80–90). Varna, Bulgaria: AIMSA 2000, Lecture Notes on AI 1904, Springer-Verlag.

Elouedi, Z., Mellouli, K., & Smets, P. (2000b). Decision trees using the belief function theory. In *Proceedings of the Eighth International Conference on Information Processing and Management of Uncertainty* (Vol. 1, pp. 141–148). Madrid, Spain: IPMU'00.

Elouedi, Z., Mellouli, K., & Smets, P. (2001a). Belief decision trees: Theoretical foundations. *International Journal of Approximate Reasoning*, *28*, 91-124.

Elouedi, Z., Mellouli, K., & Smets, P. (2001b). Induction of belief decision trees: a conjunctive approach. In G. Govaert, J. Janssen, & N. Limnios (Eds.), *Proceedings of the Tenth Conference of the Applied Stochastic Models and Data Analysis.* (pp. 404–409). Compiègne, France: ASMDA 2001.

Elouedi, Z., Mellouli, K., & Smets, P. (2001c). The evaluation of sensors' reliability and their tuning for multisensor data fusion within the transferable belief model. In S. Benferhat & P. Besnard (Eds.), *Proceedings of the Sixth European Conference on Symbolic and Quantitative*

Approaches to Reasoning with Uncertainty (pp. 350–361). Toulouse, France: ECSQARU 2001, Lecture Notes on AI 2143, Springer-Verlag.

Elouedi, Z., Mellouli, K., & Smets, P. (2002). Tuning of sensors' reliability for multisensor data fusion with the transferable belief model. *Submitted for publication.*

Esposito, F., Malerba, D., & Semeraro, G. (1997). A comparative analysis of methods for pruning decision trees. *IEEE Pattern Analysis and Machine Intelligence*, *19*(5), 476-491.

Fayyad, U. M., & Irani, K. B. (1992). On the handling of continuous-valued attributes in decision tree generation. *Machine Learning*, *8*, 87-102.

Furnkranz, J. (1997). Pruning algorithms for rule learning. *Machine Learning*, *27*, 139-171.

Godet, M. (1991). *De l'anticipation à l'action. Manuel de prospective et de stratgie.* Dunod, Paris.

Godet, M., & Roubelat, F. (1996). Creating the future: The use and misuse of scenarios. *Long Range Planning*, *29*(1), 164-171.

Gordon, J., & Shortliffe, E. H. (1984). The Dempster-Shafer theory of evidence. In B. G. Buchanan & E. H. Shortliffe (Eds.), *Rule-based expert systems: the MYCIN experiments of the Stanford Heuristic Programming Project* (pp. 272–292). Addison-Wesley, Reading, MA.

Guan, J. W., & Bell, D. A. (1993). A generalization of the Dempster-Shafer theory. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence* (pp. 10–25). Chambéry, France: IJCAI'93, Morgan Kaufman.

Hunt, E. B., Marin, J., & Stone, P. J. (1966). *Experiments in induction.* New York: Academic Press.

Janez, F. (1996). *Fusion de sources d'information définies sur des référentiels non exhaustifs différents.* PhD thesis, Université d'Angers, France.

Janikow, C. Z. (1998). Fuzzy decision trees: Issues and methods. *IEEE Transactions on Systems, Man and Cybernatics - Part B: Cybernatics*, *28*(1), 1-14.

Jun, B. H., & Kim, J. (1997). A new criterion in selection and discretization of attributes for the generation of decision trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*(12), 1371-1375.

Kohlas, J., & Monney, P. A. (1995). *A mathematical theory of hints: An approach to Dempster-Shafer theory of evidence.* Lecture Notes in Economics and Mathematical Systems 425, Springer-Verlag.

Latinne, P., Debeir, O., & Decaestecker, C. (2000). Different ways of weakening decision trees and their impact on classification accuracy of DT combination. In J. Kittler & F. Roli (Eds.), *Proceedings of the First International Workshop on Multiple Classifier Systems* (pp. 200–209).

Cagliari, Italy: MCS 2000, Lecture Notes on Computer Science 1857, Springer-Verlag.

Ling, X. N., & Rudd, W. G. (1989). Combining opinions from several experts. *Applied Artificial Inteligence*, *3*, 439–452.

Liu, W. Z., & White, A. P. (1994). The importance of attribute selection measures in decsion tree induction. *Machine Learning*, *15*, 25-41.

Lopez De Mantaras, R. (1990). *Approximate reasoning models.* Ellis Horwood, Chichester.

Lopez De Mantaras, R. (1991). A distance-based attribute selection measure for decision tree induction. *Machine Learning*, *6*, 81-92.

Maher, P. E., & Clair, D. S. (1993). Uncertain reasoning in an ID3 machine learning framework. In *Proceedings of the Second IEEE International Conference on Fuzzy Systems* (Vol. 1, pp. 7–12). San Francisco, USA: FUZZ-IEEE'93.

Marsala, C. (1998). *Apprentissage inductif en présence de données imprécises: Construction et utilisation d'arbres de decision flous.* PhD thesis, Université Paris6, LIP6, France.

Marsala, C., & Meunier, B. B. (1997). Forrests of fuzzy decision trees. In M. Mares, R. Mesiar, V. Novak, J. Ramik, & A. Stupnanova (Eds.), *Proceedings of the Seventh International Fuzzy Systems Association World Congress* (Vol. 1, pp. 369–374). Prague, Czech Republic: IFSA'97.

Mellouli, K. (1987). *On the propagation of beliefs in networks using the Dempster-Shafer theory of evidence.* PhD thesis, School of business, University of Kansas.

Mellouli, K., & Elouedi, Z. (1997). Pooling expert opinions using dempster-shafer theory of evidence. In *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics* (Vol. 4, pp. 1900–1905). Orlondo, USA: IEEE.

Mingers, J. (1989a). An empirical comparaison of selection measures for decision tree induction. *Machine Learning*, *3*, 319-342.

Mingers, J. (1989b). An empirical comparaison of pruning methods for decision tree induction. *Machine Learning*, *4*, 227–243.

Quinlan, J. R. (1983). Learning efficient classification and their application to chess end games. In R. S. Michalski, J. G. Carbonell, & T. M. Michell (Eds.), *Machnie Learning: An artificial intelligence approach* (pp. 463–482). Morgan Kaufmann.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, *1*, 81-106.

Quinlan, J. R. (1987a). Decision trees as probabilistic classifiers. In *Proceedings of the Fourth International Workshop on Machine Learning* (pp.

31–37). Morgan Kaufmann.

Quinlan, J. R. (1987b). Generating production rules from decision trees. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence* (pp. 304–307). Milan, Italy: IJCAI'87, Morgan Kaufmann.

Quinlan, J. R. (1990a). Decision trees and decision making. *IEEE Transactions on Systems, Man and Cybernatics*, *20*(2), 339-346.

Quinlan, J. R. (1990b). Probabilistic decision trees. In R. S. Michalski, J. G. Carbonell, & T. M. Michell (Eds.), *Machnie Learning, Vol. 3, Chap. 5* (pp. 267–301). Morgan Kaufmann.

Quinlan, J. R. (1993). *C4.5: Programs for machine learning.* Morgan Kaufmann, San Mateo, California.

Rosner, B. (1995). *Fundamentals of biostatistics.* Duxnury Press (ITP), Belmont, CA, USA, Fourth edition.

Safavian, S. R., & Landgrebe, D. (1991). A survey of decision tree classifier methodology. *IEEE Transactions on Systems, Man and Cybernatics*, *21*(3), 660-674.

Schlimmer, J. C., & Fisher, D. (1986). A case study of incremental concept induction. In *Proceedings of the Fifth National Conference on Artificial Intelligence* (pp. 496–501). AAAI'86, Morgan Kaufmann, Inc., Los Altos, Ca.

Shafer, G. (1976). *A mathematical theory of evidence.* Princeton Univ. Press. Princeton, NJ.

Shafer, G. (1986). The combination of evidence. *International Journal of Intelligent Systems*, *1*, 155–180.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell Systems Technical Journal*, *27*(3), 379-423,623-656.

Shenoy, P. P. (1997). Binary join trees for computing marginals in the shenoy-shafer architecture. *International Journal of Approximate Reasoning*, *17*, 239-263.

Shenoy, P. P., & Shafer, G. (1990). Axioms for probability and belief functions propagation. In R. D. Shachter, T. S. Levitt, L. N. Kanal, & J. F. Lemmer (Eds.), *Uncertainty in Artificial Intelligence 4* (pp. 159–198). North Holland, Amsterdam.

Siegel, S., & Castellan, N. J. (1988). *Non-parametric statistics for the behavioral sciences.* McGraw-Hill, Second edition.

Smets, P. (1988). Belief functions. In P. Smets, E. H. Mamdani, D. Dubois, & H. . Prade (Eds.), *Non standard logics for automated reasoning* (pp. 253–286). Academic Press, London.

Smets, P. (1990). The combination of evidence in the transferable belief model. *IEEE Pattern analysis and Machine Intelligence*, *12*(5), 447-458.

Smets, P. (1991). The transferable belief model and other interpretations of Dempster-Shafer's model. In P. P. Bonissone, M. Henrion, L. N. Kanal, & J. F. Lemmer (Eds.), *Uncertainty in Artificial Intelligence 6* (pp. 375–384). North Holland, Amsterdam.

Smets, P. (1992). The transferable belief model for expert judgments and reliability problems. *Reliability Engineering and System Safety*, *38*, 59–66.

Smets, P. (1993a). Belief functions: The disjunctive rule of combination and the generalized bayesian theorem. *International Journal of Approximate Reasoning*, *9*, 1-35.

Smets, P. (1993b). Quantifying beliefs by belief functions: An axiomatic justification. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence* (pp. 598–603). Chambéry, France: IJCAI'93, Morgan Kaufmann.

Smets, P. (1995). The canonical decomposition of a weighted belief. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence* (pp. 1896–1901). Montreal, Canada: IJCAI'95, Morgan Kaufmann.

Smets, P. (1997). The normative representation of quantified beliefs by belief functions. *Artificial Intelligence*, *92*, 229–242.

Smets, P. (1998a). The application of the transferable belief model to diagnostic problems. *International Journal of Intelligent Systems*, *13*, 127-157.

Smets, P. (1998b). The transferable belief model for quantified belief representation. In D. M. Gabbay & P. Smets (Eds.), *Handbook of defeasible reasoning and uncertainty management systems* (Vol. 1, pp. 267–301). Doordrecht, The Netherlands: Kluwer.

Smets, P. (1999). Practical uses of belief functions. In K. B. Laskey & H. Prade (Eds.), *Proceedings of the Fifteenth Uncertainty in Artificial Intelligence* (pp. 612–621). Morgan Kaufman, San Francisco, Ca.

Smets, P. (2002). Decision making in a context where uncertainty is represented by belief functions. In R. P. Srivastava (Ed.), *Belief functions in business decisions.* Physica-Verlag, Forthcoming.

Smets, P., & Kennes, R. (1994). The transferable belief model. *Artificial Intelligence*, *66*, 191–234.

Smets, P., & Kruse, R. (1997). The transferable belief model for belief representation. In A. Motro & P. Smets (Eds.), *Uncertainty in Information Systems: From needs to solutions* (pp. 343–368). Boston, MA: Kluwer.

Umano, M., Okamoto, H., Tamura, I. H. H., Kawachi, F., Umedzu, S., & Kinoshita, J. (1994). Fuzzy decision trees by fuzzy ID3 algorithm and its application to diagnosis systems. In *Proceedings of the Third IEEE*

International Conference on Fuzzy Systems (Vol. 3, pp. 2113–2118). Orlondo, USA: FUZZ-IEEE'94.

Utgoff, P. E. (1988). ID5: An incremental ID3. In *Proceedings of the Fifth International Conference on Machine Learning* (pp. 107–120). Morgan Kaufmann.

Utgoff, P. E. (1989a). Improved training via incremental learning. In *Proceedings of the Sixth International Workshop on Machine Learning* (pp. 362–365). Morgan Kaufmann.

Utgoff, P. E. (1989b). Incremental induction of decision trees. *Machine Learning*, *4*, 161-186.

Van de Merckt, T. (1995). *Towards the integration of concept learning into problem-solvers: Classification, evaluation and concept decription in numerical attribute spaces.* PhD thesis, Université Libre de Bruxelles.

Walley, P. (1991). *Statistical reasoning with imprecise probabilities.* Chapman and Hall, London.

Wehenkel, L. (1997). Discretization of continuous attributes for supervised learning variance evaluation and variance reduction. In M. Mares, R. Mesiar, V. Novak, J. Ramik, & A. Stupnanova (Eds.), *Proceedings of the Seventh International Fuzzy Systems Association World Congress* (Vol. 1, pp. 381–388). Prague, Czech Republic: IFSA'97.

Weiss, S. M., & Kulikowski, C. A. (1991). *Computer systems that learn.* San Mateo, California: Morgan Kofmann Publishers.

White, A. P., & Liu, W. Z. (1994). Bias in information-based measures in decsion tree induction. *Machine Learning*, *15*, 321-329.

Yuan, Y., & Shaw, M. J. (1995). Induction of fuzzy decision trees. *Fuzzy Sets and Systems*, *69*, 125-139.

Zeidler, J., & Schlosser, M. (1996). Continuous valued attributes in fuzzy decision trees. In *Proceedings of the Sixth International Conference on Information Processing and Management of Uncertainty* (Vol. 1, pp. 395–400). Granada, Spain: IPMU'96.

Zouhal, L. M. (1997). *Contribution à l'application de la théorie des fonctions de croyance en reconnaissance des formes.* PhD thesis, Université de Compiègne, France.

Zouhal, L. M., & Denœux, T. (1998). An evidence theoretic k-nn rule with parameter optimisation. *IEEE Transactions on Systems, Man and Cybernetics - Part C*, *28*(2), 263-271.