# Extracting Decision Trees from Interval Pattern Concept Lattices

Zainab Assaghir[1], Mehdi Kaytoue[2], Wagner Meira Jr.[2] and Jean Villerd[3]

[1] INRIA Nancy Grand Est / LORIA, Nancy, France
[2] Universidade Federal de Minas Gerais, Belo Horizonte, Brazil
[3] Institut National de Recherche Agronomique / Ensaia, Nancy, France
Zainab.Assaghir@loria.fr, {kaytoue,meira}@dcc.ufmg.br,
Jean.Villerd@nancy.inra.fr

**Abstract.** Formal Concept Analysis (FCA) and concept lattices have shown their effectiveness for binary clustering and concept learning. Moreover, several links between FCA and unsupervised data mining tasks such as itemset mining and association rules extraction have been emphasized. Several works also studied FCA in a supervised framework, showing that popular machine learning tools such as decision trees can be extracted from concept lattices. In this paper, we investigate the links between FCA and decision trees with numerical data. Recent works showed the efficiency of ”pattern structures” to handle numerical data in FCA, compared to traditional discretization methods such as conceptual scaling.

## 1 Introduction

Decision trees (DT) are among the most popular classification tools, especially for their readability [1]. Connexions between DT induction and FCA have been widely studied in the context of binary and nominal features [2], including structural links between decision trees and dichotomic lattices [8], and lattice-based learning [7]. However the numerical case faces issues regarding FCA and numerical data. In this paper, we investigate the links between FCA and decision trees with numerical data and a binary target attribute. We use an extension of Formal Concept Analysis called *interval pattern structures* to extract sets of positive and negative hypothesis from numerical data. Then, we propose an algorithm thats extract decision trees from minimal positive and negative hypothesis.

The paper is organised as follows. Section 2 presents the basics of FCA and one of its extensions called interval pattern structureq for numerical data. Section 3 recalls basic notions of decision trees. Then, we introduce some definitions in section 4 showing the links between interval pattern structures and decision trees, and a first algorithm for building decision trees from minimal positive and negative hypothesis extracted from the pattern structures.

## 2 Pattern structures in formal concept analysis

**Formal contexts and concept lattices.** We assume that the reader is familiar with FCA, and recall here most important definitions from [3]. Basically, data are represented as a binary table called *formal context* $(G, M, I)$ that represents a relation $I$ between a set of objects $G$ and a set of attributes $M$. The statement $(g, m) \in I$ is interpreted as "the object $g$ has attribute $m$". The two operators $(\cdot)'$ define a Galois connection between the powersets $(2^G, \subseteq)$ and $(2^M, \subseteq)$, with $A \subseteq G$ and $B \subseteq M$:

$$\begin{aligned} A' &= \{m \in M \mid \forall g \in A : gIm\} \qquad && for\ A \subseteq G, \\ B' &= \{g \in G \mid \forall m \in B : gIm\} \qquad && for\ B \subseteq M \end{aligned}$$

For $A \subseteq G$, $B \subseteq M$, a pair $(A, B)$, such that $A' = B$ and $B' = A$, is called a *(formal) concept*. In $(A, B)$, the set $A$ is called the *extent* and the set $B$ the *intent* of the concept $(A, B)$. The set of all concepts is partially ordered by $(A_1, B_1) \leq (A_2, B_2) \Leftrightarrow A_1 \subseteq A_2$ $(\Leftrightarrow B_2 \subseteq B_1)$ and forms a complete lattice called the *concept lattice* of the formal context $(G, M, I)$.

In many applications, data usually consist in complex data involving numbers, intervals, graphs, etc. (e.g. Table 1) and require to be conceptually scaled into formal contexts. Instead of transforming data, leading to representation and computational difficulties, one may directly work on the original data. Indeed, to handle complex data in FCA, Ganter & Kuznetsov [4] defined pattern structures: it consists of objects whose descriptions admit a *similarity operator* which induces a semi-lattice on data descriptions. Then, the basic theorem of FCA naturally holds. We recall here their basic definitions, and present interval pattern structures from [5] to handle numerical data.

**Patterns structures.** Formally, let $G$ be a set of objects, let $(D, \sqcap)$ be a meet-semi-lattice of potential object descriptions and let $\delta : G \longrightarrow D$ be a mapping. Then $(G, (D, \sqcap), \delta)$ is called a *pattern structure*. Elements of $D$ are called *patterns* and are ordered by a subsumption relation $\sqsubseteq$ such that given $c, d \in D$ one has $c \sqsubseteq d \Longleftrightarrow c \sqcap d = c$. A pattern structure $(G, (D, \sqcap), \delta)$ gives rise to the following derivation operators $(\cdot)^\square$, given $A \subseteq G$ and an interval pattern $d \in (D, \sqcap)$:

$$A^\square = \bigsqcap_{g \in A} \delta(g)$$

$$d^\square = \{g \in G | d \sqsubseteq \delta(g)\}$$

These operators form a Galois connection between $(2^G, \subseteq)$ and $(D, \sqsubseteq)$. *(Pattern) concepts* of $(G, (D, \sqcap), \delta)$ are pairs of the form $(A, d)$, $A \subseteq G$, $d \in (D, \sqcap)$, such that $A^\square = d$ and $A = d^\square$. For a pattern concept $(A, d)$, $d$ is called a *pattern intent* and is the common description of all objects in $A$, called *pattern extent*. When partially ordered by $(A_1, d_1) \leq (A_2, d_2) \Leftrightarrow A_1 \subseteq A_2$ $(\Leftrightarrow d_2 \sqsubseteq d_1)$, the set of all concepts forms a complete lattice called a *(pattern) concept lattice*.

**Interval pattern structures.** Pattern structures allow us to consider complex data in full compliance with FCA formalism. This requires to define a meet operator on object descriptions, inducing their partial order. Concerning numerical data, an interesting possibility presented in [5] is to define a meet operator as an interval convexification. Indeed, one should realize that "similarity" or "intersection" between two real numbers (between two intervals) may be expressed in the fact that they lie within some (larger) interval, this interval being the smallest interval containing both two. Formally, given two intervals $[a_1, b_1]$ and $[a_2, b_2]$, with $a_1, b_1, a_2, b_2 \in \mathbb{R}$, one has:

$$\begin{aligned}[a_1, b_1] \sqcap [a_2, b_2] &= [min(a_1, a_2), max(b_1, b_2)] \\ [a_1, b_1] \sqsubseteq [a_2, b_2] &\Leftrightarrow [a_1, b_1] \supseteq [a_2, b_2]\end{aligned}$$

The definition of $\sqcap$ implies that smaller intervals subsume larger intervals that contain them. This is counter intuitive referring to usual intuition, and is explained by the fact that $\sqcap$ behaves as an union (actually convex hull is the union of intervals, plus the holds between them).

These definitions of $\sqcap$ and $\sqsubseteq$ can be directly applied component wise on vectors of numbers or intervals, e.g. in Table 1 where objects are described by vectors of values, each dimension corresponding to an attribute. For example, $\langle[5, 7.2], [1, 1.8]\rangle \sqsubseteq \langle[5, 7], [1, 1.4]\rangle$ as $[5, 7.2] \sqsubseteq [5, 7]$ and $[1, 1.8] \sqsubseteq [1, 1.4]$.

Now that vectors of interval forms a $\sqcap$-semi-lattice, numerical data such as Table 1 give rise to a pattern structure and a pattern concept lattice. An example of application of concept forming operators $(.)^\square$ is given below. The corresponding pattern structure is $(G, (D, \sqcap), \delta)$ with $G = \{p_1, ..., p_4, n_1, ..., n_3\}$ and $d \in D$ is a vector with $i^{th}$ component corresponding to attribute $m_i$.

$$\begin{aligned}\{p_2, p_3\}^\square &= \delta(p_2) \sqcap \delta(p_3) = \langle[5, 5.9], [2, 3.2], [3.5, 4.8], [1, 1.8]\rangle \\ \{p_2, p_3\}^{\square\square} &= \{p_2, p_3, p_4\}\end{aligned}$$

As detailed in [5], vectors of intervals can be seen as hyperrectangles in Euclidean space: first $(.)^\square$ operator gives the smallest rectangle containing some object descriptions while second $(.)^\square$ operator returns the set of objects whose descriptions are rectangles included in the rectangle in argument. Accordingly, $(\{p_2, p_3, p_4\}, \langle[5, 5.9], [2, 3.2], [3.5, 4.8], [1, 1.8]\rangle)$ is a pattern concept. All pattern concepts of an interval pattern structure form a concept lattice. Intuitively, lowest concepts have few objects and "small" intervals while higher concepts have "larger" intervals. An example of such lattice is given later.

## 3 Decision trees

Among all machine leaning tools, decision trees [6, 1] are one of the most widely used. They belong to the family of supervised learning techniques, where data consist in a set of explanatory attributes (binary, nominal or numerical) that describe each object, called example, and one target class attribute that affects each example to a nominal class. Many extensions have been proposed, e.g. to consider a numerical class attribute (regression trees) or other particular cases depending on the nature of attributes. In this paper we focus on data consisting

| | $m_1$ | $m_2$ | $m_3$ | $m_4$ | $\epsilon$ |
|---|---|---|---|---|---|
| $p_1$ | 7 | 3.2 | 4.7 | 1.4 | + |
| $p_2$ | 5 | 2 | 3.5 | 1 | + |
| $p_3$ | 5.9 | 3.2 | 4.8 | 1.8 | + |
| $p_4$ | 5.5 | 2.3 | 4 | 1.3 | + |
| $n_1$ | 6.7 | 3.3 | 5.7 | 2.1 | - |
| $n_2$ | 7.2 | 3.2 | 6 | 1.8 | - |
| $n_3$ | 6.2 | 2.8 | 4.8 | 1.8 | - |

Table 1: Example 1: numerical context with an external target attribute $\epsilon$

of numerical explanatory attributes and a binary class attribute. The aim of decision tree learning is to exhibit the relation between explanatory attributes and the class attribute through a set of decision paths. A decision path is a sequence of tests on the value of explanatory attributes that is a sufficient condition to assign a new example to one of the two classes. A decision tree gathers a set of decision paths through a tree structure where nodes contain tests on explanatory attributes. Each node has two branches, the left (resp. right) corresponds to the next test if the new example passed (resp. failed) the current test. When there is no more test to perform, the branch points to a class label, that represents a leaf of the tree. The links between FCA and decision tree learning have been investigated in the case where explanatory attributes are binary [7–10, 2]. However, to our knowledge, no research has been carried out until now in the case of numerical explanatory attributes. In the next section, we show how pattern structures can be used to extract decision trees from numerical data with positive and negative examples.

## 4 Learning in interval pattern structures

In [7], S. Kuznetsov considers a machine learning model in term of formal concept analysis. He assumes that the cause of a target property resides in common attributes of objects sharing this property. In the following, we adapt this machine learning model to the case of numerical data.

Let us consider an interval pattern structure $(G, (D, \sqcap), \delta)$ with an external target property $\epsilon$. The set of objects $G$ (the training set) is partitioned into two disjoints sets: positive $G_+$ and negative $G_-$. Then, we obtain two different pattern structures $(G_+, (D, \sqcap), \delta)$ and $(G_-, (D, \sqcap), \delta)$.

**Definition 1 (Positive hypothesis).** *A positive hypothesis $h$ is defined as an interval pattern of $(G_+, (D, \sqcap), \delta)$ that is not subsumed by any interval pattern of $(G_-, (D, \sqcap), \delta)$, i.e. not subsumed by any negative example. Formally, $h \in D$ is a positive hypothesis iff*

$$h^\square \cap G_- = \emptyset \quad \textit{and} \quad \exists A \subseteq G_+ \quad \textit{such that} \quad A^\square = h$$

**Definition 2 (Negative hypothesis).** *A negative hypothesis $h$ is defined as an interval pattern of $(G_-, (D, \sqcap), \delta)$ that is not subsumed by any interval pattern* *of $(G_+, (D, \sqcap), \delta)$, i.e. not subsumed by any positive example. Formally, $h \in D$ is a negative hypothesis iff*

$$h^{\square} \cap G_+ = \emptyset \quad and \quad \exists A \subseteq G_- \quad such\ that \quad A^{\square} = h$$

**Definition 3 (Minimal hypothesis).** *A positive (resp. negative) hypothesis $h$ is minimal iff there is no positive (resp. negative) hypothesis $e \neq h$ such that $e \sqsubseteq h$.*

Going back to numerical data in Table 1, we now consider the external binary target property and split accordingly the object set into $G_+ = \{p_1, p_2, p_3, p_4\}$ and $G_- = \{n_1, n_2, n_3\}$. The pattern concept lattice of $(G_+, (D, \sqcap), \delta)$, where $D$ is the semi-lattice of intervals and $\delta$ is a mapping associating for each object its pattern description is given in Figure 1 where positive hypothesis are marked. Note that neither the interval pattern $\langle[5.5, 7], [2.3, 3.2], [4, 4.8], [1.3, 1.8]\rangle$ nor $\langle[5, 7], [2, 3.2], [3.5, 4.8], [1, 1.8]\rangle$ are positive hypothesis since they are both subsumed by the interval pattern $\delta(n_3) = \langle[6.2, 6.2], [2.8, 2.8], [4.8, 4.8], [1.8, 1.8]\rangle$. Therefore, there are two minimal positive hypothesis: $P_1 = \langle[5, 7], [2, 3.2], [3.5, 4.7], [1, 1.4]\rangle$ and $P_2 = \langle[5, 5.9], [2, 3.2], [3.5, 4.8], [1, 1.8]\rangle$. From $(G_-, (D, \sqcap), \delta)$ (not shown), we obtain the unique minimal negative hypothesis: $N_1 = \langle[6.2, 7.2], [2.8, 3.3], [4.8, 6], [1.8, 2.1]\rangle$.

Now, we consider decision trees more formally. Let the training data be described by $\mathbb{K}_{+-} = (G_+ \cup G_-, (D, \sqcap), \delta)$ with the derivation operator denoted by $(.)^{\square}$. This operator is called *subposition* in term of FCA.

**Definition 4 (Decision path).** *A sequence $\langle(m_1, d_1), (m_2, d_2), \ldots, (m_k, d_k)\rangle$, for different attributes $m_1, m_2, \cdots, m_k$ chosen one after another, is a called decision path of length $k$ if there is no $m_i$ such that $(m_i, d_i), (m_i, e_i)$ and $d_i$ and $e_i$ are not comparable, and there exists $g \in G_+ \cup G_-$ such that $\langle d_1, d_2, \ldots, d_k\rangle^{\square} \sqsubseteq \delta(g)$ (i.e. there is at least one example $g$ such that $d_i \sqsubseteq \delta(g)$ for each attribute $m_i$). For instance, $\langle(m_3, [4.8, 6]), (m_1, [6.2, 7.2])\rangle$ is a decision path for Example 1.*

If $i \leq k$ (respectively $i < k$), the sequence $\langle(m_1, d_1), (m_2, d_2), \ldots, (m_i, d_i)\rangle$ is called subpath (proper subpath) of a decision path $\langle(m_1, d_1), (m_2, d_2), \ldots, (m_k, d_k)\rangle$.

**Definition 5 (Full decision path).** *A sequence $\langle(m_1, d_1), (m_2, d_2), \ldots, (m_k, d_k)\rangle$, for different attributes $m_1, m_2, \ldots, m_k$ chosen one after another, is called full decision path of length $k$ if all object having $(m_1, d_1), (m_2, d_2), \ldots, (m_k, d_k)$ (i.e. $\forall g \in G, d_i \sqsubseteq \delta(g)$ for the attribute $m_i$) are either positive or negative examples (i.e. have either $+$ or $-$ value of the target attribute).*

We say that a full decision path is non-redundant if none of its subpaths is a full decision path. The set of all chosen attributes in a full decision path can be considered as a sufficient condition for an object to belong to a class $\epsilon \in \{+, -\}$. A decision tree is then defined as the set of full decision paths.

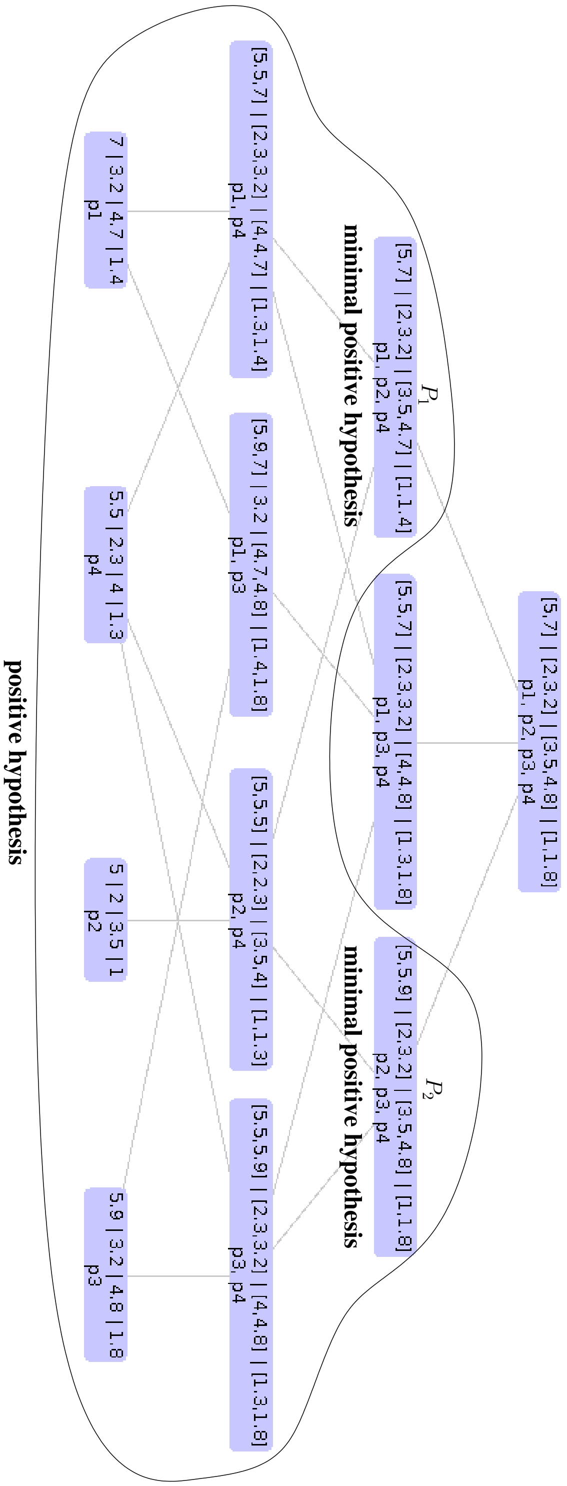


Fig. 1: Lattice of the pattern structure $(G_+, (D, \sqcap), \delta)$.

### 4.1 A first algorithm for building decision trees from interval pattern structures

In this section, we propose a first algorithm for extracting full decision paths from the sets of minimal positive hypothesis $\mathcal{P}$ and minimal negative hypothesis $\mathcal{N}$. Intuitively, minimal positive (resp. negative) hypothesis describe the largest areas in the attribute space that gathers the maximum number of positive (resp. negative) examples with no negative (resp. positive) example. Positive and negative areas may intersect on some dimensions. In Example 1 (see Table 1), $\mathcal{P} = \{P_1, P_2\}$ and $\mathcal{N} = \{N_1\}$ and we denote by $P_i \cap N_j$ the interval vector for which the $k$-the component is the intersection of the $P_i$ and $N_j$ intervals for the $k$-the component. Recall that $P_1 = \langle [5, 7], [2, 3.2], [3.5, 4.7], [1, 1.4] \rangle$, $P_2 = \langle [5, 5.9], [2, 3.2], [3.5, 4.8], [1, 1.8] \rangle$ and $N_1 = \langle [6.2, 7.2], [2.8, 3.3], [4.8, 6], [1.8, 2.1] \rangle$. Then we have:

$$P_1 \cap N_1 = \langle [6.2, 7], [2.8, 3.2], \emptyset, \emptyset \rangle$$

$$P_2 \cap N_1 = \langle \emptyset, [2.8, 3.2], [4.8], [1.8] \rangle$$

We note that $P_1$ and $N_1$ have no intersection for attributes $m_3$ and $m_4$. This means that any example that has a value for $m_3$ (resp. $m_4$) that is contained in $P_1$'s interval for $m_3$ (resp. $m_4$) can directly be classified as positive. Similarly, any example having a value for $m_3$ (resp. $m_4$) contained in $N_1$'s interval for $m_3$ (resp. $m_4$) can directly be classified as negative. The same occurs for $P_2$ and $N_1$ for $m_1$.

Therefore a full decision path for a minimal positive hypothesis $P$ is defined as a sequence $\langle (m_i, m_i(P)) \rangle_{i \in \{1 \ldots |\mathcal{N}|\}}$ where $m_i$ is an attribute such that $m_i(P \cap N_i) = \emptyset$[4]. A full decision path for a minimal negative hypothesis $N$ is defined as a sequence $\langle (m_j, m_j(N)) \rangle_{j \in \{1 \ldots |\mathcal{P}|\}}$ where $m_j$ is an attribute such that $m_j(N \cap P_i) = \emptyset$.

Here examples of such decision paths (built from $P_1, P_2$ and $N_1$ respectively) are:

$$\langle (m_3, [3.5, 4.7]) \rangle (P_1)$$

$$\langle (m_4, [1, 1.4]) \rangle (P_1)$$

$$\langle (m_1, [5, 5.9]) \rangle (P_2)$$

$$\langle (m_3, [4.8, 6]), (m_1, [6.2, 7.2]) \rangle (N_1)$$

$$\langle (m_4, [1.8, 2.1]), (m_1, [6.2, 7.2]) \rangle (N_1)$$

Decision paths built from $P_1$ and $P_2$ are sequences that contain a single element since $|\mathcal{N}| = 1$. Decision paths built from $N_1$ are sequences that contain two elements since $|\mathcal{P}| = 2$. Two distinct full decision paths can be built from $P_1$ since there are two attributes for which $P_1$ and $N_1$ do not intersect.

A positive (resp. negative) decision tree is therefore a set of full decision paths, one for each minimal positive (resp.negative) hypothesis. For instance:

[4] For any interval pattern $P$, the notation $m_i(P)$ denotes its interval value for the attribute $m_i$.

*"if* $m_3 \in [3.5, 4.7]$ *then +, else if* $m_1 \in [5, 5.9]$ *then + else -"* is an example of positive decision path. An example of negative decision path is *"if* $m_1 \in [6.2, 7.2]$ *and* $m_3 \in [4.8, 6]$ *then -, else +"*.

Algorithm 1 describes the computation of full decision paths for minimal positive hypothesis. The dual algorithm for minimal negative hypothesis is obtained by interchanging $\mathcal{P}$ and $\mathcal{N}$.

```
1 $Res \leftarrow$ empty array of size $|\mathcal{P}|$;
2 foreach $P \in \mathcal{P}$ do
3     foreach $N \in \mathcal{N}$ such that $\exists m_i, m_i(P \cap N) = \emptyset$ do
4         if $(m_i, m_i(P)) \notin Res[P]$ then
5             $Res[P] \leftarrow Res[P] \cup (m_i, m_i(P))$;
6     foreach $N \in \mathcal{N}$ such that $\nexists m_i, m_i(P \cap N) = \emptyset$ do
7         $Res[P] \leftarrow Res[P] \cup \{\bigvee_{m \in M}(m, m(P) \setminus m(P \cap N))\}$;
```

**Algorithm 1:** Modified algorithm for extracting full decision paths $Res$ (including non-redundant) for minimal positive hypothesis

The different steps of the algorithm are detailed below:

line 1: $Res$ will contain a full decision path for each minimal positive hypothesis.
line 2: Process each minimal positive hypothesis $P$.
line 3: For each minimal negative hypothesis $N$ that has at least one attribute $m$ such that $m(P \cap N) = \emptyset$, choose one of these attribute, called $m_i$ below.
line 4: Ensure that $m_i$ has not already been selected for another $N$, this enables to produce non redundant full decision paths (see Example 2).
line 5: Add the interval $m_i(P)$ in the full decision path of $P$. The test $m_i \in m_i(P)$ will separate between positive examples covered by $P$ and negative examples covered by $N$.
line 6: For each minimal negative hypothesis $N$ that has no attribute $m$ such that $m(P \cap N) = \emptyset$.
line 7: Positive examples covered by $P$ and negative examples covered by $N$ can be separated by a disjunction of tests $m \in m(P) \setminus m(P \cap N))$ on each attribute $m$. Hence, there is at least one attribute for which a positive example from $P$ belongs to $m(P)$ and not to $m(N)$. Otherwise, $N$ would not be a negative hypothesis.

Note that Example 1 is a particular case where all negative examples are gathered in a unique minimal negative hypothesis.

A few values have been modified in Table 2 in order to produce two minimal negative hypothesis.

| | $m_1$ | $m_2$ | $m_3$ | $m_4$ | $\epsilon$ |
|---|---|---|---|---|---|
| $p_1$ | 7 | 3.2 | 4.7 | 1.4 | + |
| $p_2$ | 5 | 2 | 3.5 | 1 | + |
| $p_3$ | 5.9 | 3.2 | 4.8 | 1.8 | + |
| $p_4$ | 5.5 | 2.3 | 4 | 1.3 | + |
| $n_1$ | 5.9 | 3.3 | 5.7 | 1.4 | - |
| $n_2$ | 7.2 | 3.2 | 6 | 1.8 | - |
| $n_3$ | 6.2 | 2.8 | 4.8 | 1.8 | - |

Table 2: Example 2: training set

Minimal positive hypothesis $P_1$ and $P_2$ remain unchanged while there are two minimal negative hypothesis:

$$N_1 = \langle [5.9, 7.2], [3.2, 3.3], [5.7, 6], [1.4, 1.8] \rangle$$

$$N_2 = \langle [6.2, 7.2], [2.8, 3.2], [4.8, 6], [1.8, 1.8] \rangle$$

This leads to the following intersections:

$$P_1 \cap N_1 = \langle [5.9, 7], [3.2], \emptyset, [1.4] \rangle$$

$$P_1 \cap N_2 = \langle [6.2, 7], [2.8, 3.2], \emptyset, \emptyset \rangle$$

$$P_2 \cap N_1 = \langle [5.9], [3.2], \emptyset, [1.4, 1.8] \rangle$$

$$P_2 \cap N_2 = \langle \emptyset, [2.8, 3.2], [4.8, 4.8], [1.8] \rangle$$

Examples of full decision path computed by Algorithm 1 from $P_1$ are

$$\langle (m_3, [3.5, 4.7]), (m_4, [1, 1.4]) \rangle (1)$$

$$\langle (m_3, [3.5, 4.7]), (m_3, [3.5, 4.7]) \rangle (2)$$

Note that neither $N_1$ nor $N_2$ intersect $P_1$ on $m_3$, therefore the full decision path (2) can be simplified as $\langle (m_3, [3.5, 4.7]) \rangle$. More generally, following previous definitions, $\langle (m_3, [3.5, 4.7]) \rangle$ is a non-redundant full decision path while $\langle (m_3, [3.5, 4.7]), (m_4, [1, 1.4]) \rangle$ and $\langle (m_3, [3.5, 4.7]), (m_3, [3.5, 4.7]) \rangle$ are not. A conditional test has been added in Algorithm 1 in order to also produce such non-redundant full decision paths.

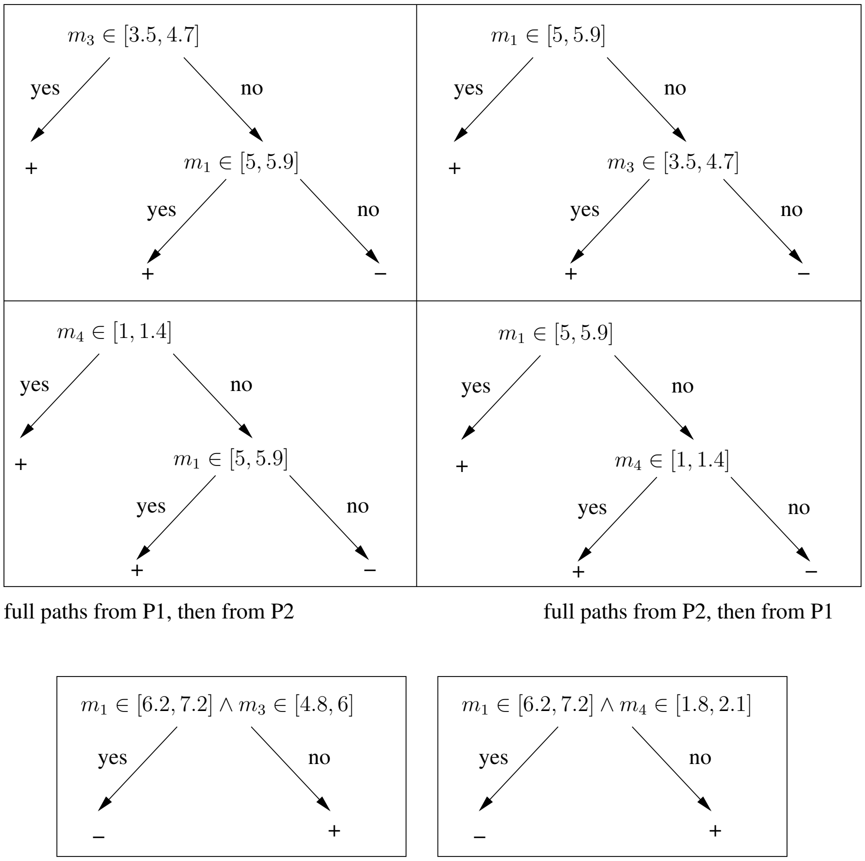
Finally a concrete positive decision tree is built from the set of full decision paths, each node corresponds to a minimal positive hypothesis $P_i$ and contains a test that consists in the conjunction of the elements of a full decision path. The left child contains + and the right child is a node corresponding to another minimal positive hypothesis $P_j$ or - if all minimal positive hypothesis have been processed.

An example of decision tree for example 2 is: *"if* $m_3 \in [3.5, 4.7]$ *and* $m_4 \in [1, 1.4]$ *then +, else (if* $m_3 \in [3.5, 4.8]$ *and* $m_1 \in [5, 5.9]$ *then +, else -)"*.

We detail below the complete process for examples 1 and 2.

### 4.2 Example 1

| | |
|---|---|
| $\mathcal{P}$ | $P_1 = \langle [5,7], [2,3.2], [3.5,4.7], [1,1.4] \rangle$<br>$P_2 = \langle [5,5.9], [2,3.2], [3.5,4.8], [1,1.8] \rangle$ |
| $\mathcal{N}$ | $N_1 = \langle [6.2,7.2], [2.8,3.3], [4.8,6], [1.8,2.1] \rangle$ |
| intersections | $P_1 \cap N_1 = \langle [6.2,7], [2.8,3.2], \emptyset, \emptyset \rangle$<br>$P_2 \cap N_1 = \langle \emptyset, [2.8,3.2], [4.8], [1.8] \rangle$ |
| full decisions paths from $\mathcal{P}$ | $\langle (m_3, [3.5,4.7]) \rangle (P_1)$<br>$\langle (m_4, [1,1.4]) \rangle (P_1)$<br>$\langle (m_1, [5,5.9]) \rangle (P_2)$ |
| full decisions paths from $\mathcal{N}$ | $\langle (m_3, [4.8,6]), (m_1, [6.2,7.2]) \rangle (N_1)$<br>$\langle (m_4, [1.8,2.1]), (m_1, [6.2,7.2]) \rangle (N_1)$ |



full paths from P1, then from P2

full paths from P2, then from P1

full paths from N1

Fig. 2: Decision trees built from Example 1

### 4.3 Example 2

| | |
|---|---|
| $\mathcal{P}$ | $P_1 = \langle [5,7], [2,3.2], [3.5,4.7], [1,1.4] \rangle$<br>$P_2 = \langle [5,5.9], [2,3.2], [3.5,4.8], [1,1.8] \rangle$ |
| $\mathcal{N}$ | $N_1 = \langle [5.9,7.2], [3.2,3.3], [5.7,6], [1.4,1.8] \rangle$<br>$N_2 = \langle [6.2,7.2], [2.8,3.2], [4.8,6], [1.8,1.8] \rangle$ |
| intersections | $P_1 \cap N_1 = \langle [5.9,7], [3.2], \emptyset, [1.4] \rangle$<br>$P_1 \cap N_2 = \langle [6.2,7], [2.8,3.2], \emptyset, \emptyset \rangle$<br>$P_2 \cap N_1 = \langle [5.9], [3.2], \emptyset, [1.4,1.8] \rangle$<br>$P_2 \cap N_2 = \langle \emptyset, [2.8,3.2], [4.8,4.8], [1.8] \rangle$ |
| full decisions paths from $\mathcal{P}$ | $\langle (m_3, [3.5,4.7]) \rangle (P_1)$<br>$\langle (m_3, [3.5,4.7]), (m_4, [1,1.4]) \rangle (P_1)$ (redundant)<br>$\langle (m_3, [3.5,4.8]), (m_1, [5,5.9]) \rangle (P_2)$ |
| full decisions paths from $\mathcal{N}$ | $\langle (m_3, [5.7,6]) \rangle (N_1)$<br>$\langle (m_3, [4.8,6]), (m_1, [6.2,7.2]) \rangle (N_2)$<br>$\langle (m_4, [1.8]), (m_1, [6.2,7.2]) \rangle (N_2)$ |

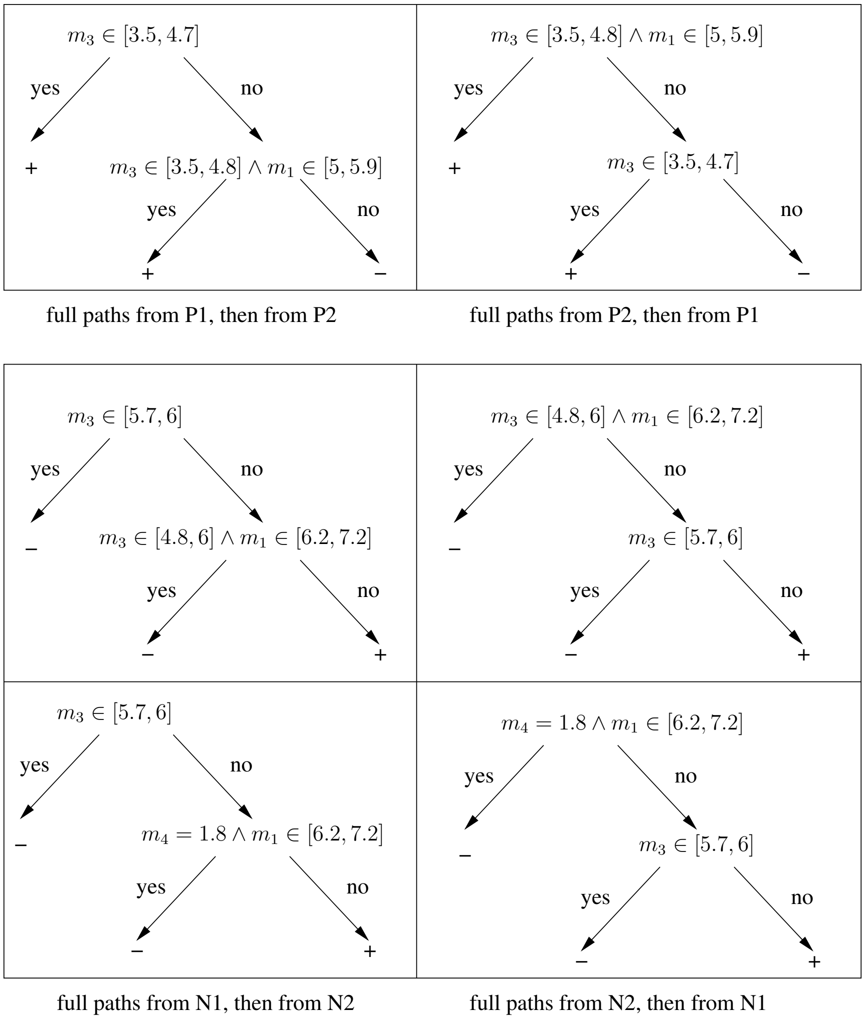
### 4.4 Comparison with traditional decision tree learning approaches

Standard algorithms such as C4.5 produce decision trees in which nodes contain tests of the form $a \leq v$, i.e. the value for attribute $a$ is less or equal to $v$, while our nodes contain conjunctions of tests of the form $a \in [a_1, a_2] \wedge b \in [b_1, b_2]$. A solution consists in identifying minimal and maximal values for each attribute in the training set, and by replacing them by $-\infty$ and $+\infty$ respectively in the resulting trees (see Figure 4). Moreover, common decision tree induction techniques use Information Gain maximization (or equivalently conditional entropy minization) to choose the best split at each node. The conditional entropy of a split is null when each child node is pure (contains only positive or negative examples). When this perfect split can not be expressed as an attribute-value test, it can be shown that the optimal split that minimize conditional entropy consists in maximizing the number of examples in one pure child node (proof is ommited due to space limitation). This optimal split exactly matches our notion of positive (resp. negative) minimal hypothesis, which corresponds to descriptions that gathers the maximum number of only positive (resp. negative) examples.

However we insist that our algorithm is only a first and naive attempt to produce decision trees from multi-valued contexts using pattern structures. Its aim is only to clarify the links between decision tree learning and pattern structures. Therefore it obviously lacks of relevant data structures and optimization. However we plan to focus our efforts on algorithm optimization and then on rigorous experimentations on standard datasets.

## 5 Concluding remarks

In this paper, we studied the links between decision trees and FCA in the particular context of numerical data. More precisely, we focused on an extension


$m_3 \in [3.5, 4.7]$
yes
no
+
$m_3 \in [3.5, 4.8] \wedge m_1 \in [5, 5.9]$
yes
no
+
−
$m_3 \in [3.5, 4.8] \wedge m_1 \in [5, 5.9]$
yes
no
+
$m_3 \in [3.5, 4.7]$
yes
no
+
−
$m_3 \in [5.7, 6]$
yes
no
−
$m_3 \in [4.8, 6] \wedge m_1 \in [6.2, 7.2]$
yes
no
−
+
$m_3 \in [4.8, 6] \wedge m_1 \in [6.2, 7.2]$
yes
no
−
$m_3 \in [5.7, 6]$
yes
no
−
+
$m_3 \in [5.7, 6]$
yes
no
−
$m_4 = 1.8 \wedge m_1 \in [6.2, 7.2]$
yes
no
−
+
$m_4 = 1.8 \wedge m_1 \in [6.2, 7.2]$
yes
no
−
$m_3 \in [5.7, 6]$
yes
no
−
+

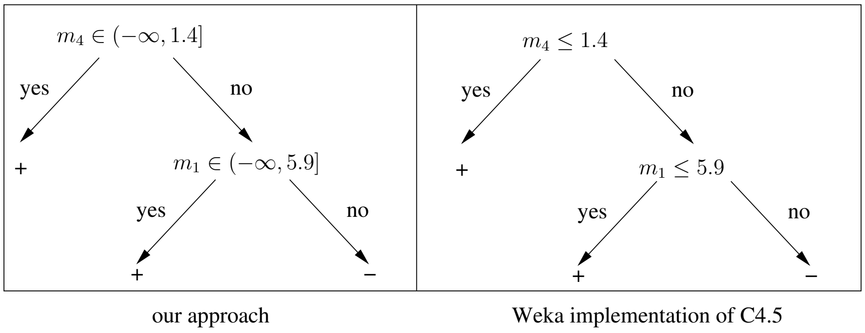
full paths from P1, then from P2

full paths from P2, then from P1

full paths from N1, then from N2

full paths from N2, then from N1

Fig. 3: Decision trees built from Example 2



our approach

Weka implementation of C4.5

Fig. 4: Comparison of decisions trees produced by our approach and by C4.5 for Example 1

of FCA for numerical data called interval pattern structures, that has recently gained popularity through its ability to handle numerical data without any discretization step. We showed that interval pattern structures from positive and negative examples are able to reveal positive and negative hypothesis, from which decision paths and decision trees can be built.

In future works, we will focus on a comprehensive and rigorous comparison of our approach with traditional decision tree learning techniques. Moreover, we will study how to introduce in our approach pruning techniques that avoid overfitting. We will also investigate solutions in order to handle nominal class attributes (i.e. more than two classes) and heterogeneous explanatory attributes (binary, nominal, ordinal, numerical). Finally, notice that interval patterns are closed since $(.)^{\square\square}$ is a closure operator. In a recent work [11], it has been shown that whereas a closed interval pattern represents the smallest hyper-rectangle in its equivalence class, interval pattern generators represent the largest hyper-rectangles. Accordingly, generators are favoured by minimum description length principle (MDL), since being less constrained. An interesting perspective is to test their effectiveness to describe minimal hypothesis in the present work.

## References

1. Quinlan, J.: Induction of decision trees. Machine learning **1**(1) (1986) 81–106
2. Fu, H., Njiwoua, P., Nguifo, E.: A comparative study of fca-based supervised classification algorithms. Concept Lattices (2004) 219–220
3. Ganter, B., Wille, R.: Formal Concept Analysis. Springer-Verlag (1999)
4. Ganter, B., Kuznetsov, S.O.: Pattern structures and their projections. In: ICCS '01: Proceedings of the 9th International Conference on Conceptual Structures, Springer-Verlag (2001) 129–142
5. Kaytoue, M., Kuznetsov, S.O., Napoli, A., Duplessis, S.: Mining gene expression data with pattern structures in formal concept analysis. Inf. Sci. **181**(10) (2011) 1989–2001
6. Breiman, L.: Classification and regression trees. Chapman & Hall/CRC (1984)

7. Kuznetsov, S.O.: Machine learning and formal concept analysis. Int. Conf. on Formal Concept Analysis, LNCS 2961, (2004) 287–312
8. Guillas, S., Bertet, K., Ogier, J.: A generic description of the concept lattices classifier: Application to symbol recognition. Graphics Recognition. Ten Years Review and Future Perspectives (2006) 47–60
9. Nijssen, S., Fromont, E.: Mining optimal decision trees from itemset lattices. In: Proceedings of the 13th ACM SIGKDD international conference on knowledge discovery and data mining, ACM (2007) 530–539
10. Nguifo, E., Njiwoua, P.: Iglue: A lattice-based constructive induction system. Intelligent data analysis **5**(1) (2001) 73
11. Kaytoue, M., Kuznetsov, S.O., Napoli, A.: Revisiting Numerical Pattern Mining with Formal Concept Analysis. In: International Joint Conference on Artificial Intelligence (IJCAI), Barcelona, Espagne (2011)