Interval Temporal Logic Decision Tree Learning

Andrea Brunello[2 [0000−0003−2063−218X]],
Guido Sciavicco[1 [0000−0002−9221−879X]], and
Ionel Eduard Stan[2 [0000−0001−9260−102X]]

[1] Dept. of Mathematics and Computer Science
University of Ferrara (Italy)
guido.sciavicco@unife.it
[2] Dept. of Mathematics, Computer Science, and Physics
University of Udine (Italy)
andrea.brunello@uniud.it
stan.ioneleduard@spes.uniud.it

Abstract. Decision trees are simple, yet powerful, classification models used to classify categorical and numerical data, and, despite their simplicity, they are commonly used in operations research and management, as well as in knowledge mining. From a logical point of view, a decision tree can be seen as a structured set of logical rules written in propositional logic. Since knowledge mining is rapidly evolving towards temporal knowledge mining, and since in many cases temporal information is best described by interval temporal logics, propositional logic decision trees may evolve towards interval temporal logic decision trees. In this paper, we define the problem of interval temporal logic decision tree learning, and propose a solution that generalizes classical decision tree learning.

Keywords: Decision trees · Interval temporal logics · Symbolic learning.

1 Introduction

It is commonly recognized that modern decision trees are of primary importance among classification models [30]. They owe their popularity mainly to the fact that they can be trained and applied efficiently even on big data sets, and that they are easily interpretable, meaning that they are not only useful for prediction per se, but also for understanding the reasons behind the predictions. Interpretability is of extreme importance in domains in which understanding the classification process is as important as the accuracy of the classification itself, such in the case of production business systems or in the computer-aided medicine domain. A typical decision tree is constructed in a recursive manner, following the traditional *Top Down Induction of Decision Trees* (TDIDT) approach [26]: starting from the root, at each node the attribute that best partitions the training data, according to a predefined score, is chosen as a test to guide the partitioning of instances into child nodes. The process continues until a sufficiently high degree of purity (with respect to the target class), or a minimum cardinality constraint (with respect to the number of instances reaching

the node), is achieved in the generated partitions. This is the case of the well-known decision tree learning algorithm ID3 [26], which is the precursor of the commonly-used C4.5 [27]. A decision tree can be seen as a structured set of rules: every node of the tree can be thought of as an *if-then-else* statement, and, in this way, each branch becomes a conjunction of such conditional statements, that is, a *rule*, whose right-hand part is the class. A conditional statement may have many forms: it can be a yes/no statement (for binary categorical attributes), a categorical value statement (for non-binary categorical attributes), or a splitting value statement (for numerical attributes); the ariety of the resulting tree is two if all attributes are binary or numerical, or more, if there are categorical attributes with more than two categories. Each statement can be equivalently represented with *propositional letters*, so that a decision tree can be also seen as a structured set of *propositional logic* rules.

Temporal classification: static solutions. Just focusing on the static aspects of data is not always adequate for classification; for example, in the medical domain, one may want to take into account which symptoms a patient was experiencing at the same time, or whether two symptoms were overlapping. That is, in some application domains, the temporal aspects of the information may be essential to an accurate prediction. Within static decision tree learning, temporal information may be aggregated in order to circumvent the absence of explicit tools for dealing with temporal information (for example, a patient can be labelled with a natural number describing how many times he/she has been running a fever during the observation period); the ability of a decision tree to perform a precise classification based on such processed data, however, strongly depends on how well data are prepared, and therefore on how well the underlying domain is understood. Alternatively, decision trees have been proposed that use *frequent patterns* [15, 19, 22] in nodes, considering the presence/absence of a frequent pattern as a categorical attribute [13, 18]. Nevertheless, despite being the most common approach to (explicit) temporal data classification, frequent patterns in sequences or series have a limited expressive power, as they are characterized by being *existential* and by intrinsically representing temporal information with instantaneous events.

Our approach: interval temporal logic decision trees. A different approach to temporal classification is mining temporal logic formulas, and since temporal databases universally adopt an interval-based representation of time, the ideal choice to represent temporal information in data is *interval* temporal logic. The most representative propositional interval temporal logic is Halpern and Shoham's Modal Logic of Allen's Relations [20], also known as HS. Its language encompasses one modal operator for each interval-to-interval relation, such as *meets* or *before*, and the computational properties of HS and its fragments have been studied in the recent literature (see, e.g. [10–12]). The very high expressive power of HS, as well as its versatility, make HS the ideal candidate to serve as the basis of a temporal decision tree learning algorithm. Based on these premises, we propose in this paper a decision tree learning algorithm that produces HS-based trees. Our proposal, Temporal ID3, is a direct generalization of

the ID3 algorithm [26], founded on the logical interpretation of tree nodes, and focuses on data representation and node generation; we borrow other aspects, such as splitting based on information gain and the overall learning process from the original algorithm. The accuracy of a decision tree and its resilience to over-fitting also depends on the stopping criterion and possible post-pruning operations, but we do not discuss these aspects here.

Existing approaches to temporal logic decision trees. Learning temporal logic decision trees is an emerging field in the analysis of physical systems, and, among the most influential approaches, we mention learning of automata [3] and learning Signal Temporal Logic (STL) formulas [6, 14, 24, 28]. In particular, STL is a point-based temporal logic with *until* that encompasses certain metric capabilities, and learning formulas of STL has been focused on both the fine tuning of the metric parameters of a predefined formula and the learning the innermost structure of a formula; among others, decision trees have been used to this end [8]. Compared with STL decision tree learning, our approach has the advantage of learning formulas written in a interval-based, instead of point-based, well-known and highly expressive temporal logic language; because of the nature of the underlying language and of the interval temporal logic models, certain application domains fit naturally into this approach. Moreover, since our solution generalizes the classical decision tree learning algorithm ID3, and, particularly, the notion of information gain, it is not limited to binary classification only. Moreover, in [7] a first-order framework for TDIDT is presented with the attempt to make such paradigm more attractive to inductive logic programming (ILP). Such a framework provides a sound basis for logical decision tree induction; in opposition, we employ the framework to represent *modal*, instead of first-order, relational data. Additionally, our approach should not be confused with [23], in which the term *interval* indicates an uncertain numerical value (e.g., *the patient has a fever of 38 Celsius* versus *the patient has a fever between 37.5 and 38.5 Celsius*), and in which an algorithm for inducing decision trees on such uncertain data is presented that is based on the so-called Kolmogorov-Smirnov criterion, but the data that are object of that study are not necessarily temporal, and the produced trees do not employ any temporal (logical) relation. In [4, 29] and [21], the authors present two other approaches to a temporal generalization of decision tree learning. In the former, the authors provide a general method for building point-based temporal decision trees, but with no particular emphasis on any supporting formal language. In the latter, the constructed trees can be seen as real-time algorithms that have the ability to make decisions even if the entire description of the instance is not yet available. Finally, in [16], a generalization of the decision tree model is presented in which nodes are possibly labelled with a timestamp to indicate when a certain condition should be checked.

Summarizing, our approach is essentially different from those presented in the literature in several aspects. As a matter of fact, by giving a logical perspective to decision tree learning, we effectively generalize the learning model to a temporal one, instead of introducing a new paradigm. In this way, instances that present some temporal component are naturally seen as timelines, and, thanks

to the expressive power provided by HS, our algorithm can learn a decision tree based on the temporal relations between values, instead of the static information carried by the values.

2 Preliminaries

Decision trees. Decision tree induction is based on the following simple concepts [27]. Given a set of observable values $V = \{v_1, v_2, \ldots, v_n\}$, with associated probabilities $\Pi = \{\pi_1, \pi_2, \ldots, \pi_n\}$, the *information conveyed by* Π (or *entropy*), is defined as:

$$E(\Pi) = -\sum_{i=1}^{n} \pi_i \log(\pi_i).$$

Assume that a data set $\mathcal{T}$ has n instances, each characterized by the attributes $A_1, \ldots, A_l$, and distributed over s classes $C_1, \ldots, C_s$. Each class C can be seen as the subset of $\mathcal{T}$ composed of precisely those instances classified as C, so that the information needed to identify the class of an element of $\mathcal{T}$ is:

$$Info(\mathcal{T}) = E(\{\frac{|\mathcal{C}_1|}{|\mathcal{T}|}, \frac{|\mathcal{C}_2|}{|\mathcal{T}|}, \ldots, \frac{|\mathcal{C}_s|}{|\mathcal{T}|}\}).$$

Intuitively, the entropy is inversely proportional to the purity degree of $\mathcal{T}$ with respect to the class values. *Splitting*, which is the main operation in decision tree learning, is performed over a specific attribute A. If A is categorical and its domain $Dom(A)$ consists of m distinct values, we can split $\mathcal{T}$ into $\mathcal{T}_1, \ldots, \mathcal{T}_m$, each $\mathcal{T}_i$ being characterized by A having precisely the value a_i (i.e., $A = a_i$). The information of a categorical split, therefore, is:

$$InfoCat(A, \mathcal{T}) = \sum_{i=1}^{m} (\frac{|\mathcal{T}_i|}{|\mathcal{T}|}) Info(\mathcal{T}_i).$$

If, on the other hand, A is numerical, then the set $\{a_1 < \ldots < a_m\}$ of *actual* values for A that are present in $\mathcal{T}$ gives rise to $m-1$ possible splits, all of them binary, and the information conveyed by each possible split is, then, a function not only of the attribute but also of the chosen value:

$$InfoNum(A, a_i, \mathcal{T}) = (\frac{|\mathcal{T}_1|}{|\mathcal{T}|}) Info(\mathcal{T}_1) + (\frac{|\mathcal{T}_2|}{|\mathcal{T}|}) Info(\mathcal{T}_2),$$

where $\mathcal{T}_1$ (respectively, $\mathcal{T}_2$) encompasses all and only those instances with $A \leq a_i$ (respectively, $A > a_i$). The information conveyed by an attribute can be consequently defined as:

$$InfoAtt(A, \mathcal{T}) = \begin{cases} InfoCat(A, \mathcal{T}) & \text{if } A \text{ is categorical,} \\ \min\limits_{a_i \in Dom(A)} \{InfoNum(A, a_i, \mathcal{T})\} & \text{if } A \text{ is numerical,} \end{cases}$$

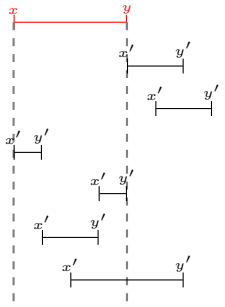
HS	Allen's relations	Graphical representation
$\langle A \rangle$	$[x,y]R_A[x',y'] \Leftrightarrow y = x'$	
$\langle L \rangle$	$[x,y]R_L[x',y'] \Leftrightarrow y < x'$	
$\langle B \rangle$	$[x,y]R_B[x',y'] \Leftrightarrow x = x', y' < y$	
$\langle E \rangle$	$[x,y]R_E[x',y'] \Leftrightarrow y = y', x < x'$	
$\langle D \rangle$	$[x,y]R_D[x',y'] \Leftrightarrow x < x', y' < y$	
$\langle O \rangle$	$[x,y]R_O[x',y'] \Leftrightarrow x < x' < y < y'$	

Fig. 1. Allen's interval relations and HS modalities.

and the *information gain* brought by A is defined as:

$$Gain(A, \mathcal{T}) = Info(\mathcal{T}) - InfoAtt(A, \mathcal{T}).$$

The information gain, which can be also seen as the reduction of the expected entropy when the attribute A has been chosen, is used to drive the splitting process, that is, to decide over which attribute (and how) the next split is performed. The underlying principle to decision tree building consists of recursively splitting the data set over the attribute that guarantees the greatest information gain until a certain stopping criterion is met. Each split can be seen as a propositional condition *if p then -, else -*. When splitting is performed over a numerical attribute, e.g., $A \leq a_i$, then the corresponding propositional statement is simply the condition itself (in our example, is a propositional letter $p_{A \leq a_i}$); when it is performed over a categorical attribute, e.g., $A = a_1$, $A = a_2$, ..., then each statement is a propositional statement (in our example, $p_{A=a_1}$, $p_{A=a_2}$,...) on its own.

Interval temporal logic. Let $\mathbb{D} \subseteq \mathbb{N}$. In the *strict* interpretation, an *interval* over $\mathbb{D}$ is an ordered pair $[x, y]$, where $x, y \in \mathbb{D}$ and $x < y$, and we denote by $\mathbb{I}(\mathbb{D})$ the set of all intervals over $\mathbb{D}$. If we exclude the identity relation, there are 12 different Allen's relations between two intervals in a linear order [1]: the six relations R_A (adjacent to), R_L (later than), R_B (begins), R_E (ends), R_D (during), and R_O (overlaps), depicted in Fig. 1, and their inverses, that is, $R_{\overline{X}} = (R_X)^{-1}$, for each $X \in \mathcal{A}$, where $\mathcal{A} = \{A, L, B, E, D, O\}$. Halpern and Shoham's modal logic of temporal intervals (HS) is defined from a set of propositional letters $\mathcal{AP}$, and by associating a universal modality $[X]$ and an existential one $\langle X \rangle$ to each Allen's relation R_X. Formulas of HS are obtained by

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid \langle X \rangle \varphi \mid \langle \overline{X} \rangle \varphi,$$

where $p \in \mathcal{AP}$ and $X \in \mathcal{A}$. The other Boolean connectives and the logical constants, e.g., $\rightarrow$ and $\top$, as well as the universal modalities $[X]$, can be defined in the standard way. For each $X \in \mathcal{A}$, the modality $\langle \overline{X} \rangle$ (corresponding to the

inverse relation $R_{\overline{X}}$ of R_X) is said to be the *transpose* of the modalities $\langle X \rangle$, and vice versa. The semantics of HS formulas is given in terms of *timelines* $T = \langle \mathbb{I}(\mathbb{D}), V \rangle$[3], where $\mathbb{D}$ is a linear order and $V : \mathcal{AP} \to 2^{\mathbb{I}(\mathbb{D})}$ is a *valuation function* which assigns to each atomic proposition $p \in \mathcal{AP}$ the set of intervals $V(p)$ on which p holds. The *truth* of a formula φ on a given interval $[x, y]$ in an interval model T is defined by structural induction on formulas as follows:

$$
\begin{array}{ll}
T, [x,y] \Vdash p & \text{if } [x,y] \in V(p), \text{ for } p \in \mathcal{AP}; \\
T, [x,y] \Vdash \neg\psi & \text{if } T, [x,y] \nVdash \psi; \\
T, [x,y] \Vdash \psi \vee \xi & \text{if } T, [x,y] \Vdash \psi \text{ or } T, [x,y] \Vdash \xi; \\
T, [x,y] \Vdash \langle X \rangle \psi & \text{if there is } [w,z] \text{ s.t } [x,y] R_X [w,z] \text{ and } T, [w,z] \Vdash \psi; \\
T, [x,y] \Vdash \langle \bar{X} \rangle \psi & \text{if there is } [w,z] \text{ s.t } [x,y] R_{\bar{X}} [w,z] \text{ and } T, [w,z] \Vdash \psi.
\end{array}
$$

HS is a very general interval temporal language and its satisfiability problem is undecidable [20]. Our purpose here, however, is to study the problem of formula *induction* in the form of decision trees, and not of formula *deduction*, and therefore the computational properties of the satisfiability problem can be ignored at this stage.

3 Motivations

In this section, we present some realistic scenarios in which learning a temporal decision tree may be convenient, and, then, we discuss aspects of data preprocessing related to the temporal component.

Learning. There are several application domains in which learning a temporal decision tree may be useful. Consider, for example, a medical scenario in which we consider a data set of classified patients, each one characterized by its medical history, as in Fig. 2, top. Assume, first, that we are interested in learning a *static* (propositional) classification model. The history of our patients, that is, the collection of all relevant pieces of information about tests, results, symptoms, and hospitalizations of the patient that occurred during the entire observation period, must be processed so that temporal information is subsumed in propositional letters. For instance, if some patient has been running a fever during the observation period, we may use a proposition $fever$, with positive values for those patient that have had fever, and negative values for the others (as in Fig. 2, bottom, left). Depending on the specific case, we may, instead, use the actual temperature of each patient, and a static decision tree learning system may split over $fever < t$, for some threshold temperature t, effectively introducing a new propositional letter, and therefore a binary split. Either way, the temporal information is lost in the preprocessing. For example, we can no longer take into account whether $fever$ occurred before, after, or while the patient was experiencing headache ($head$), which may be a relevant information for a

[3] We deliberately use the symbol T to indicate both a timeline and an instance in a data set.

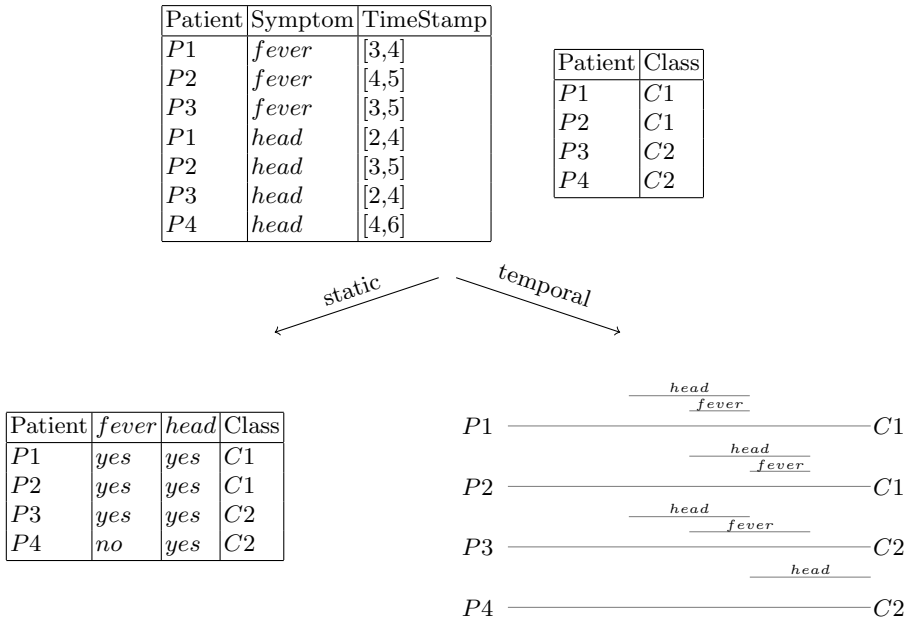


Fig. 2. Example of static and temporal treatment of information in the medical domain.

classification model. By generating, instead, the timeline of each patient (as in Fig. 2, bottom, right), we keep all events and their relative qualitative relations. By learning a decision tree on a preprocessed data set such as the one in Fig. 2 (bottom, left), we see that the attribute *head* has zero variance, and therefore zero predictive capabilities; then, we are forced to build a decision tree using attribute $fever$ alone, which results in a classifier with 75% accuracy. On the contrary, by using the temporal information in the learning process, we are able to distinguish the two classes: $C1$ is characterized by presenting both *head* and $fever$, but not overlapping, and this classifier has, in this toy example, 100% accuracy. In this example, the term *accuracy* refers to the *training set accuracy* (we do not consider independent train and test data), that is, the ability of the classification system to discern among classes on the data used to train the system itself; it should not be confused with *test set accuracy*, which measures the real classification performances that can be expected on future, real-life examples.

Alternatively, consider a problem in the natural language processing domain. In this scenario, a timeline may represent a *conversation* between two individuals. It is known that, in automatic processing of conversations, it is sometimes interesting to label each interval of time with one or more *context*, that is, a particular topic that is being discussed [2, 5, 25], in order to discover the existence of unexpected or interesting temporal relations among them. Suppose, for example, that a certain company wants to analyze conversations between selling

agents and potential customers: the agents contact the customers with the aim of selling a certain product, and it is known that certain contexts, such as the price of the product (*price*), its known advantages (*advantages*) over other products, and its possible minor defects (*disadvantages*) are interesting. Assume that each conversation has been previously classified between those that have been successful and those that ended without the product being acquired. Now, we want to learn a model able to predict such an outcome. By using only static information, nearly every conversation would be labelled with the three contexts, effectively hiding the underlying knowledge, if it exists. By keeping the relative temporal relations between contexts, instead, we may learn useful information, such as, for example, *if price and disadvantages are not discussed together, the conversation will be likely successful*.

Preprocessing. Observe, now, how switching from static to temporal information influences data preparation. First, in a context such as the one described in our first example, numerical attributes may become less interesting: for instance, the information on *how many times* a certain symptom occurred, or its *frequency*, are not needed anymore, considering that each occurrence is taken into account in the timeline. Moreover, since the focus is on attributes *relative temporal positions*, even categorical attributes may be ignored in some contexts: for instance, in our scenario, we may be interested in establishing the predictive value of the relative temporal position of $fever$ and *head* regardless of the sex or age of the patient. It is also worth underlying that propositional attributes over intervals allow us to express a variety of situations, and sometimes propositional labelling may result in *gaining* information, instead of loosing it. Consider, again, the case of fever, and suppose that a certain patient is experiencing low fever in an interval $[x, y]$, say, a given day, and that during just one hour of that day, that is, over the interval $[w, z]$ strictly contained in $[x, y]$, he/she has an episode of high fever. A natural choice is to represent such a situation by labelling the interval $[x, y]$ with lo and its sub-interval $[w, z]$ with hi. On the other hand, representing the same pieces of information as three intervals $[x, w], [w, z], [z, y]$ respectively labelled with lo, hi, and lo, which would be the case with a point-based representation (or with an interval-based representation under the homogeneity assumption), would be unnatural, and it would entail hiding a potentially important information such as: "*the patient presented low fever during the entire day, except for a brief episode of high fever*". Building on such considerations, our approach in the rest of this paper is based on propositional, non-numerical attributes only.

4 Learning Interval Temporal Logic Decision Trees

In this section we describe a generalization of the algorithm ID3 that is capable of learning a binary decision tree over a temporal data set, as in the examples of the previous section; as in classical decision trees, every branch of a temporal decision tree can be read as a logical formula, but instead of classical logic we use the temporal logic HS. To this end, we generalize the notion of information gain,

while, at this stage, we do not discuss pre-pruning, post-pruning, and purity degree of a sub-tree [9, 27].

Data preparation and presentation. We assume that the input data set contains timelines as instances. For the sake of simplicity, we also assume that all timelines are based on the same finite domain $\mathbb{D}$ of length N (from 0 to $N-1$). The data set $\mathcal{T}$ can be seen as an array of n structures; $\mathcal{T}[j]$ represents the j-th timeline of the data set, and it can be thought of as an interval model. Given a data set $\mathcal{T}$, we denote by $\mathcal{AP}$ the set of all propositional letters that occur in it.

Temporal information. We are going to design the learning process based on the same principles of classical decision tree learning. This means that we need to define a notion of splitting as well as a notion of information conveyed by a split, and, to this end, we shall use the truth relation as defined in Section 2 applied to a timeline. Unlike the atemporal case, splits are not performed over attributes, but, instead, over propositional letters. Splitting is defined *relatively to an interval* $[x, y]$, and it can be local, if it is applied on $[x, y]$ itself, or temporal, in which case it depends on the existence of an interval $[z, t]$ related to $[x, y]$ and the particular relation R_X such that $[x, y]R_X[z, t]$ (or the other way around). A *local split* of $\mathcal{T}$ into $\mathcal{T}_1$ and $\mathcal{T}_2$, where $[x, y]$ is the reference interval of $\mathcal{T}$, and p is the propositional letter over which the split takes place is defined by:

$$\begin{array}{l} \mathcal{T}_1 = \{T \in \mathcal{T} \mid T, [x, y] \Vdash p\}, \\ \mathcal{T}_2 = \{T \in \mathcal{T} \mid T, [x, y] \Vdash \neg p\}. \end{array} \tag{1}$$

On the contrary, a *temporal split*, in the same situation, over the temporal relation R_X, is defined by:

$$\begin{array}{l} \mathcal{T}_1 = \{T \in \mathcal{T} \mid T, [x, y] \Vdash \langle X \rangle p\}, \\ \mathcal{T}_2 = \{T \in \mathcal{T} \mid T, [x, y] \Vdash [X] \neg p\}. \end{array} \tag{2}$$

Consequently, the *local information gain* of a propositional letter p is defined as:

$$LocalGain(p, \mathcal{T}) = Info(\mathcal{T}) - \left((\frac{|\mathcal{T}_1|}{|\mathcal{T}|}) Info(\mathcal{T}_1) + (\frac{|\mathcal{T}_2|}{|\mathcal{T}|}) Info(\mathcal{T}_2) \right),$$

where $\mathcal{T}_1$ and $\mathcal{T}_2$ are defined as in (1), while the *temporal information gain* of a propositional letter p is defined as:

$$TemporalGain(p, \mathcal{T}) = Info(\mathcal{T}) - \min_{X \in \mathcal{A}} \left\{ (\frac{|\mathcal{T}_1|}{|\mathcal{T}|}) Info(\mathcal{T}_1) + (\frac{|\mathcal{T}_2|}{|\mathcal{T}|}) Info(\mathcal{T}_2) \right\},$$

where $\mathcal{T}_1$ and $\mathcal{T}_2$ are defined as in (2) and depend on the relation R_X. Therefore, the information gain of a propositional letter becomes:

$$Gain(p, \mathcal{T}) = \max\{LocalGain(p, \mathcal{T}), TemporalGain(p, \mathcal{T})\}.$$

At each step, we aim to find the letter that maximizes the gain. Moreover, temporal splits entail associating every $T \in \mathcal{T}_1$ with a new reference interval:

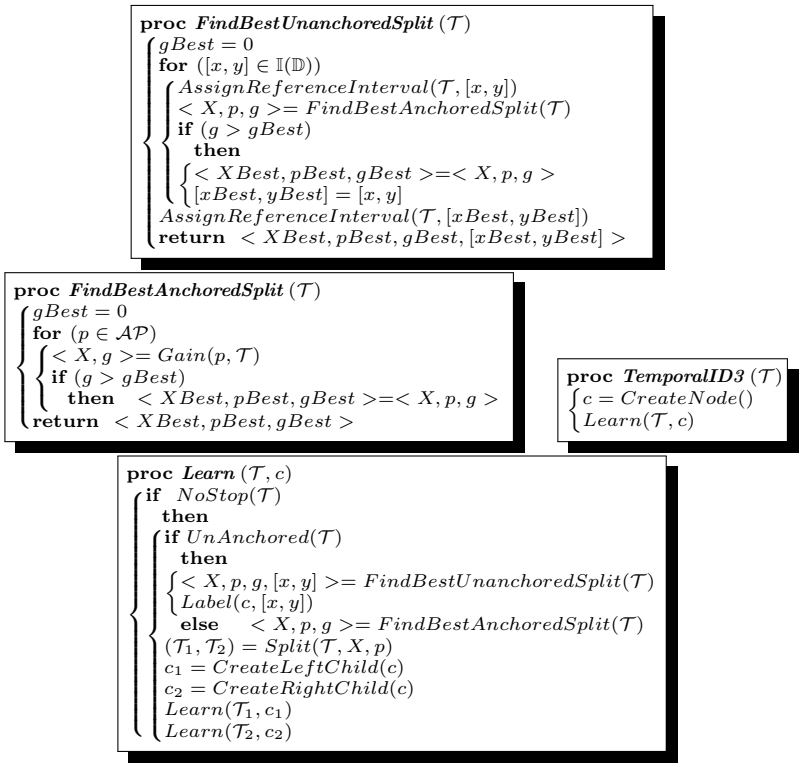


Fig. 3. The algorithm Temporal ID3.

they all satisfy $\langle X\rangle p$ on $[x, y]$, and, for each one of them, there is a possibly distinct witness. For example, T_1, T_2 are in $\mathcal{T}_1$ because satisfy $\langle A\rangle p$ on $[x, y]$, it could be the case that $T_1, [y, z_1] \Vdash p$ and $T_2, [y, z_2] \Vdash p$ with $z_1 \neq z_2$. So, T_1 and T_2 would have two different reference intervals, namely $[y, z_1]$ and $[y, z_2]$. In this learning model, all splits are binary. During the tree learning process, we can label the 'lefthand' edges of a split with the chosen $\langle X\rangle p$ (or just p, when the split is local), the corresponding 'righthand' edges with $[X]\neg p$ (or just $\neg p$), and we can label the node with a new reference interval if the current data set does not have one, yet.

The algorithm. Let us analyze the code in Fig. 3. At the beginning, the timelines in $\mathcal{T}$ are not assigned any reference interval, and we say that the data set is *unanchored*. The procedure *FindBestUnanchoredSplit* systematically explores every possible reference interval of an unanchored data set, and, for each one of them, calls *FindBestAnchoredSplit*, which, in turn, tries every propositional letter (and, implicitly, every temporal relation) in search of the best split. This procedure returns the best possible triple $< X, p, g >$, where X is an interval relation, if the best split is temporal, or it has no value, if the best split is local, p is a propositional letter, and g is the information gain. *TemporalID3* first

creates a root node, and, then, calls *Learn*. The latter, in turn, first checks possible stopping conditions, and, then, finds the best split into two data sets $\mathcal{T}_1$ and $\mathcal{T}_2$. Of these, the former is now *anchored* (to the reference interval returned by *FindBestUnanchoredSplit*), while the latter is still unanchored. During a recursive call, when $\mathcal{T}_1$ is analyzed to find its best split, the procedure for this task will be *FindBestAnchoredSplit*, called directly, instead of passing through *FindBestUnanchoredSplit*.

Analysis. We now analyze the computational complexity of Temporal ID3. To this end, we first compute the cost of finding the best splitting. Since the cardinality of the domain of each timeline is N, there are $O(N^2)$ possible intervals. This means that, fixed a propositional letter and a relation R_X, computing $\mathcal{T}_1$ and $\mathcal{T}_2$ costs $O(nN^2)$. Therefore, the cost of *FindBestAnchoredSplit* is obtained by multiplying the cost of a single (tentative) splitting by the number of propositional letters and the number of temporal relations (plus one, to take into account the local splitting), which sums up to $O(13nN^2|\mathcal{AP}|)$. The cost of *FindBestUnanchoredSplit* increases by a factor of N^2, as the **for** cycle ranges over all possible intervals, and therefore it becomes $O(13nN^4|\mathcal{AP}|)$. We can increase the efficiency of the implementation by suitably pre-compute the value of $\langle X\rangle p$ for each temporal relation, each propositional letter, and each interval, thus eliminating a factor of N^2 from both costs.

If we consider $\mathcal{AP}$ fixed, and N constant, the cost of finding the best splitting becomes $O(n)$, and, under such (reasonable) assumption, we can analyze the complexity of an execution of *Learn* in terms of the number n of timelines. Two cases are particularly interesting. In the worst case, every binary split leads to a very unbalanced partition of the data set, with $|\mathcal{T}_1| = 1$ and $|\mathcal{T}_2| = n - 1$ (or the other way around). The recurrence that describes such a situation is:

$$t(n) = t(n-1) + O(n),$$

which can be immediately solved to obtain $t(n) = O(n^2)$. In the average case, every binary split leads to a non-unbalanced partition, but we cannot foresee the relative cardinality of each side of the partition. Assuming that every partition is equally probable, the recurrence that describes this situation is:

$$t(n) = \frac{1}{n}\Sigma_{k=1}^{n}(t(k) + t(n-k)) + O(n).$$

We want to prove that $t(n) = O(n\log(n))$. To this end, we prove by induction that $t(n) \leq an\log(n) + b$ for some positive constants a, b, by separating our argument in three parts. First, we manipulate our initial expression:

$$\begin{aligned}
t(n) &= \tfrac{1}{n}\Sigma_{k=1}^{n}(t(k) + t(n-k)) + O(n) \\
&= \tfrac{2}{n}\Sigma_{k=1}^{n} t(k) + O(n) \\
&\leq \tfrac{2}{n}\Sigma_{k=1}^{n}(ak\log(k) + b) + O(n) && \text{(inductive hypothesis)} \\
&= \tfrac{2}{n}\Sigma_{k=1}^{n}(ak\log(k)) + \tfrac{2}{n}\Sigma_{k=1}^{n} b + O(n) \\
&= \tfrac{2a}{n}\Sigma_{k=1}^{n}(k\log(k)) + 2b + O(n).
\end{aligned}$$

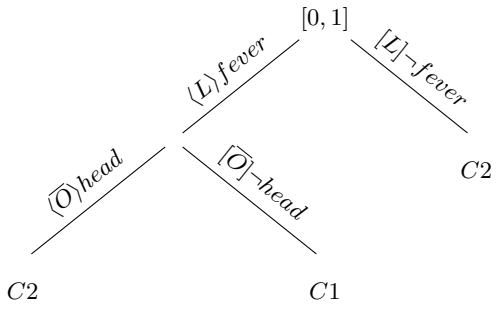


Fig. 4. A decision tree learned by Temporal ID3 on the example in Fig. 2.

Then, we focus our attention on conveniently bounding the first term of the expression that we have obtained, as follows:

$$\begin{aligned}
\Sigma_{k=1}^{n}(k\log(k)) &= \Sigma_{k=1}^{\lceil \frac{n}{2} \rceil - 1}(k\log(k)) + \Sigma_{k=\lceil \frac{n}{2} \rceil}^{n}(k\log(k)) \\
&\leq \Sigma_{k=1}^{\lceil \frac{n}{2} \rceil - 1}(k\log(\tfrac{n}{2})) + \Sigma_{k=\lceil \frac{n}{2} \rceil}^{n}(k\log(n)) \\
&= (\log(n) - 1)\Sigma_{k=1}^{\lceil \frac{n}{2} \rceil - 1}k + \log(n)\Sigma_{k=\lceil \frac{n}{2} \rceil}^{n}k \\
&= \log(n)\Sigma_{k=1}^{n}k - \Sigma_{k=1}^{\lceil \frac{n}{2} \rceil - 1}k \\
&\leq \tfrac{1}{2}\log(n)n(n+1) - \tfrac{1}{2}\tfrac{n}{2}(\tfrac{n}{2}+1) \\
&= \tfrac{1}{2}(n^2\log(n) + n\log(n)) - \tfrac{1}{8}n^2 - \tfrac{1}{4}n.
\end{aligned}$$

Finally, we plug in our bound, to obtain:

$$\begin{aligned}
t(n) &\leq \tfrac{2a}{n}(\tfrac{1}{2}(n^2\log(n) + n\log(n)) - \tfrac{1}{8}n^2 - \tfrac{1}{4}n) + 2b + O(n) \\
&= an\log(n) + 2a\log(n) - \tfrac{an}{4} - \tfrac{a}{2} + 2b + O(n) \\
&\leq an\log(n) + b,
\end{aligned}$$

for a large enough value of a such that

$$\frac{an}{4} \geq 2a\log(n) - \frac{a}{2} + b + O(n).$$

Computing the worst case has only a theoretical value; we can reasonably expect Temporal ID3 to behave like a randomized divide-and-conquer algorithm, and its computational complexity to tend towards the average case. Possible practical improvements of Temporal ID3 depend, however, on the stopping conditions, which directly influence the base case and therefore the height of the recursive call stack.

Example of execution. Consider our initial example of Fig. 2, with four timelines distributed over two classes. Since this is a toy example, there are many different combination of interval, relation, and propositional letter that give the same information gain. Fig. 4 gives one possible outcome, which seems to indicate that, looking at the entire history, the class $C2$ is characterized by presenting headache and overlapping fever, or no fever at all.

5 Conclusions

Among all classification models, decision trees are still very popular. Their success is due to the computational efficiency of the learning process, the interpretability of tree models, and their wide versatility. Classical decision trees are designed to interpret categorical and numerical attributes, and, in both cases, every node of a tree can be seen as a propositional letter. Therefore, a decision tree can be seen as a structured set of propositional logic rules, the right-hand part of which is a class. Since classifying based on the static aspects of data is not always adequate, and considered that decision tree learning cannot deal with temporal knowledge in an explicit manner, building on the logical interpretation of a decision tree, we considered the problem of learning a classification model capable to combine propositional knowledge with qualitative temporal information. We presented, first, a very well-known temporal propositional logic, that is, HS. Then, we showed how temporal data can be prepared in a optimal way for a temporal decision tree to be learned, and we presented a generalization of the most classical decision tree learning algorithm, that is, ID3, that is able to split the data set based on temporal, instead of static, information. We analyzed the computational complexity of the proposed algorithm, and we showed a complete example of execution. Future work include testing our method on real data, and improving the capabilities of Temporal ID3 by enriching the underlying language and studying the effect of different pruning and stopping conditions. Real data testing is not an easy task: data that contain temporal information, such as medical data (as per our example) are usually not prepared for timeline analysis. Therefore, a non-trivial algorithmic effort must be devoted to data preparation, with particular attention to correct and meaningful labeling of intervals.

Machine learning is generically focused on a non-logical approach to knowledge representation. However, when learning should take into account temporal aspects of data, a logical approach can be associated to classical methods, and interval temporal logics has been already proposed as a possible tool in this sense. Besides temporal decision tree learning, temporal logic can be also used for rule extraction [17]. Combining these approaches with fragments of interval temporal logics whose satisfiability problem is decidable (and tractable) may result into an integrated systems that pairs induction and deduction of formulas, intelligent elimination of redundant rules, and automatic verification of inducted knowledge against formal requirement. Also, using a logical approach to describe learned knowledge may require non-standard semantics for logical formulas (e.g., fuzzy semantics, or multi-valued propositional semantics); these, in turn, pose original and interesting questions on the theoretical side concerning the computational properties of the problems associated with these logics (i.e., satisfiability or model checking), generating, *de facto*, a cross-feeding effect on the two fields.

References

1. Allen, J.F.: Maintaining knowledge about temporal intervals. Communications of the ACM **26**(11), 832–843 (1983)

2. Alluhaibi, R.: Simple interval temporal logic for natural language assertion descriptions. In: Proc. of the 11th International Conference on Computational Semantics (IWCS). pp. 283–293 (2015)
3. Angluin, D.: Learning regular sets from queries and counterexamples. Information and Computation **75**(2), 87–106 (1987)
4. Antipov, S., Fomina, M.: A method for compiling general concepts with the use of temporal decision trees. Scientific and Technical Information Processing **38**(6), 409–419 (2011)
5. Baeza-Yates, R.: Challenges in the interaction of information retrieval and natural language processing. In: Proc. of the 5th International on Computational Linguistics and Intelligent Text Processing (CICLing). pp. 445–456 (2004)
6. Bartocci, E., Bortolussi, L., Sanguinetti, G.: Data-driven statistical learning of temporal logic properties. In: Proc. of the 12th International Conference on Formal Modeling and Analysis of Timed Systems. pp. 23–37 (2014)
7. Blockeel, H., De Raedt, L.: Top-down induction of first-order logical decision trees. Artificial Intelligence **101**(1-2), 285–297 (1998)
8. Bombara, G., Vasile, C., Penedo, F., Yasuoka, H., Belta, C.: A decision tree approach to data classification using signal temporal logic. In: Proc. of the 19th International Conference on Hybrid Systems: Computation and Control. pp. 1–10 (2016)
9. Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and regression trees. Wadsworth and Brooks, Monterey, CA (1984)
10. Bresolin, D., Della Monica, D., Goranko, V., Montanari, A., Sciavicco, G.: Metric propositional neighborhood logics on natural numbers. Software and System Modeling **12**(2), 245–264 (2013)
11. Bresolin, D., Monica, D.D., Montanari, A., Sala, P., Sciavicco, G.: Interval temporal logics over strongly discrete linear orders: Expressiveness and complexity. Theoretical Computer Science **560**, 269–291 (2014)
12. Bresolin, D., Sala, P., Sciavicco, G.: On begins, meets, and before. International Journal of Foundations of Computer Science **23**(3), 559–583 (2012)
13. Brunello, A., Marzano, E., Montanari, A., Sciavicco, G.: J48S: A sequence classification approach to text analysis based on decision trees. In: Proc. of the 24th International Conference on Information and Software Technologies, (ICIST). pp. 240–256 (2018)
14. Bufo, S., Bartocci, E., Sanguinetti, G., Borelli, M., Lucangelo, U., Bortolussi, L.: Temporal logic based monitoring of assisted ventilation in intensive care patients. In: Proc. of the 6th International Symposium on Leveraging Applications of Formal Methods, Verification and Validation. pp. 391–403 (2014)
15. Cheng, H., Yan, X., Han, J., Hsu, C.: Discriminative frequent pattern analysis for effective classification. In: Proc. of the 23rd International Conference on Data Engineering (ICDE). pp. 716–725 (2007)
16. Console, L., Picardi, C., Dupré, D.: Temporal decision trees: Model-based diagnosis of dynamic systems on-board. J. Artif. Intell. Res. **19**, 469–512 (2003)
17. Della Monica, D., de Frutos-Escrig, D., Montanari, A., Murano, A., Sciavicco, G.: Evaluation of temporal datasets via interval temporal logic model checking. In: Proc. of the 24th International Symposium on Temporal Representation and Reasoning (TIME). pp. 11:1–11:18 (2017)
18. Fan, W., Zhang, K., Cheng, H., Gao, J., Yan, X., Han, J., Yu, P., Verscheure, O.: Direct mining of discriminative and essential frequent patterns via model-based search tree. In: Proc. of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 230–238 (2008)

19. Fournier-Viger, P., Gomariz, A., Šebek, M., Hlosta, M.: VGEN: fast vertical mining of sequential generator patterns. In: Proc. of the 16th International Conference on Data Warehousing and Knowledge Discovery (DaWaK). pp. 476–488 (2014)
20. Halpern, J., Shoham, Y.: A propositional modal logic of time intervals. Journal of the ACM **38**(4), 935–962 (1991)
21. Karimi, K., Hamilton, H.J.: Temporal rules and temporal decision trees: A C4.5 approach. Tech. Rep. CS-2001-02, Department of Computer Science, University of Regina (2001)
22. Lin, W., Orgun, M.: Temporal data mining using hidden periodicity analysis. In: Proc. of the 12th International Symposium on Foundations of Intelligent Systems (ISMIS). pp. 49–58 (2000)
23. Mballo, C., Diday, E.: Decision trees on interval valued variables. Symbolic Data Analysis **3**(1), 8–18 (2005)
24. Nguyen, L., Kapinski, J., Jin, X., Deshmukh, J., Butts, K., Johnson, T.: Abnormal data classification using time-frequency temporal logic. In: Proc. of the 20th International Conference on Hybrid Systems: Computation and Control. pp. 237–242 (2017)
25. Pratt-Hartmann, I.: Temporal prepositions and their logic. Artificial Intelligence **166**(1–2), 1–36 (2005)
26. Quinlan, J.: Induction of decision trees. Machine Learning **1**, 81–106 (1986)
27. Quinlan, J.: Simplifying decision trees. International Journal of Human-Computer Studies **51**(2), 497–510 (1999)
28. Rajan, A.: Automated requirements-based test case generation. SIGSOFT Software Engeneering Notes **31**(6), 1–2 (2006)
29. Vagin, V., Morosin, O., Fomina, M., Antipov, S.: Temporal decision trees in diagnostics systems. In: 2018 International Conference on Advances in Big Data, Computing and Data Communication Systems (icABCD). pp. 1–10 (2018)
30. Witten, I., Frank, E., Hall, M., Pal, C.: Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann (2016)