Time-penalized trees (TpT): a new tree-based data mining algorithm for time-varying covariates

Mathias Valla

Time-penalized trees (TpT): a new tree-based data mining algorithm for time-varying covariates

Mathias Valla*[1,2]

[1] *Univ Lyon, Université Claude Bernard Lyon 1, Institut de Science Financière et d'Assurances (ISFA), Laboratoire SAF EA2429, F-69366, Lyon, France.*
[2] *Faculty of Economics and Business, KU Leuven, Belgium.*

Abstract

This article proposes a new decision tree algorithm that accounts for time-varying covariates in the decision-making process. Traditional decision tree algorithms assume that the covariates are static and do not change over time, which can lead to inaccurate predictions in dynamic environments. Other existing methods suggest workaround solutions such as the pseudo-subject approach, discussed in the article. The proposed algorithm utilizes a different structure and a time-penalized splitting criterion that allow a recursive partitioning of both the covariates space and time. Relevant historical trends are then inherently involved in the construction of a tree, and are visible and interpretable once it is fit. This approach allows for innovative and highly interpretable analysis in settings where the covariates are subject to change over time. The effectiveness of the algorithm is demonstrated through real-world data analysis, highlighting its potential applications in various fields, including healthcare, finance, insurance, environmental monitoring, and data mining in general.

Key words: decision tree, time-varying covariate, data mining, longitudinal study, algorithm

1 Introduction

Decision trees are a popular machine learning tool for data mining as well as classification and regression predictions. Growing such a tree is a data-driven process based on a set of input covariates and a target variable. The most famous decision tree algorithm is arguably Classification and Regression Trees (CART), introduced by Breiman et al. (1984). CART constructs a binary tree by recursively partitioning the feature space into smaller and smaller subsets, based on a splitting criterion that maximizes the separation between the target variable's values in each subset. However, traditional decision tree algorithms like CART assume that the input features or covariates are static and do not change over time. In many real-world settings, this assumption is unrealistic, and the time-dynamic nature of the covariates is highly informative and should be included in the tree construction. In such settings, not accounting for dynamic features results in information loss, hence a loss of accuracy and richness of analysis.

Other data-driven approaches can already efficiently seize the time dimension of features in prediction and data-mining settings. One can think of neural networks (See Mena et al. (2023) or Wong et al. (2022) for instance). Yet conventional parametric statistical models or machine learning approaches such as logistic regression or most tree-based models cannot handle time-varying covariates straightforwardly. They assume that individual observations are independently and identically distributed. Because of the longitudinal structure of a time-varying dataset (see Section 2.2 for more details), this independence hypothesis cannot be

*Email: mathias.valla@univ-lyon1.fr , URL: https://mathias-valla.com

met: different observations of a single subject are naturally strongly correlated. To address this limitation, some existing tree-based methods suggest workarounds such as the pseudo-subject approach in survival trees (Fu and Simonoff (2016)), which create artificial left-truncated and right-censored subjects by pooling observations over time, or the inclusion of a mixed effect model structure around a tree-based core (Hajjem et al. (2011b); Sela and Simonoff (2012); Hajjem et al. (2014)). Such computationally intensive methods proved to yield competitive results in many prediction frameworks, yet we argue in the following sections that they are not entirely satisfying in terms of interpretability.

In this article, we propose `Time-penalized Tree` (`TpT`), a new decision tree algorithm that accounts for time-varying covariates in the decision-making process. Our algorithm utilizes a different structure and a time-penalized splitting criterion that allows for recursive partitioning of both the covariates space and time. We detail the algorithm and show simulated real data-mining and visualization applications but do not discuss the theoretical convergence guarantees or the ways it can be used as a predictive tool. Such work is of primary interest to us and will constitute forthcoming studies; in that sense, any collaboration on an optimized implementation or on the study of the theoretical properties of `TpT` in regression and survival settings is more than welcomed.

The rest of this paper is structured as follows. We recall the basics about classification and regression trees as well as time-varying covariates analysis in Section 2, we also briefly present existing approaches and frame their interpretability flaws. Then we detail the specificities of `TpT` in Section 3 and explain its benefits, which is the main contribution of this work. In Section 4, we show a concrete application of our framework on a real-world life-insurance dataset, with visuals and illustration work, demonstrating the interpretability properties of `TpT`. Eventually, Section 5 concludes this paper and details future works.

2 Preliminaries

2.1 Classification and regression trees

In this section, we briefly describe the mechanisms of a simple yet powerful data-mining and prediction model: decision trees, and more specifically, Classification and Regression trees or CART (Breiman et al. (1984)). Here, we assume that all covariates are time-independent. Let $\mathcal{D} = (\boldsymbol{x^{(i)}}, y^{(i)})_{i=1}^{N}$ be a dataset of N individuals with $\boldsymbol{x^{(i)}} = \left(x_1^{(i)}, \ldots, x_p^{(i)}\right)$, the vector of p covariates and $y^{(i)}$ the target variable for the i-th subject. The covariates and target spaces are respectively denoted $\mathcal{X}$ and $\mathcal{Y}$. Decision trees create a recursive partition of $\mathcal{X}$ based on binary decision rules. This partitioning can be visualized directly in the case where there are two covariates x_1 and x_2 (see Figure 1). In that case, individual observations are represented as dots that are eventually clustered into n_L distinct, non-overlapping regions of $\mathcal{X}$ denoted $(L_1, \ldots, L_{n_L})$.

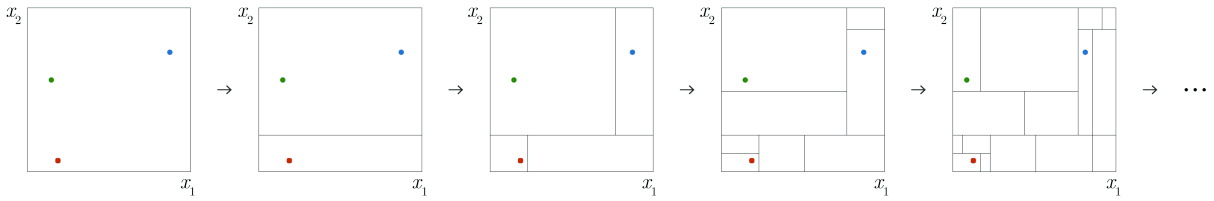


Figure 1: Decision tree recurcive partitioning

More generally it can be visualized as a tree (see Figure 2), with yes/no questions within each node and terminal nodes - or leaves - corresponding to the n_L regions of the covariates space. Because the regions defined by leaves are non-overlapping, every individual i belongs to a single leaf $L^{(i)}$, and a prediction is made for all individuals falling in the leaf. More generally, let g be a node, at g, we define $\mathcal{D}(g) = (x^{(i)}, y^{(i)}) \in \mathcal{D}$ where $(x^{(i)}, y^{(i)}) \in g$, the set of observations in the node g. The quantity $\mathcal{N}(g) = |\mathcal{D}(g)|$ is then the number of individuals in the node.

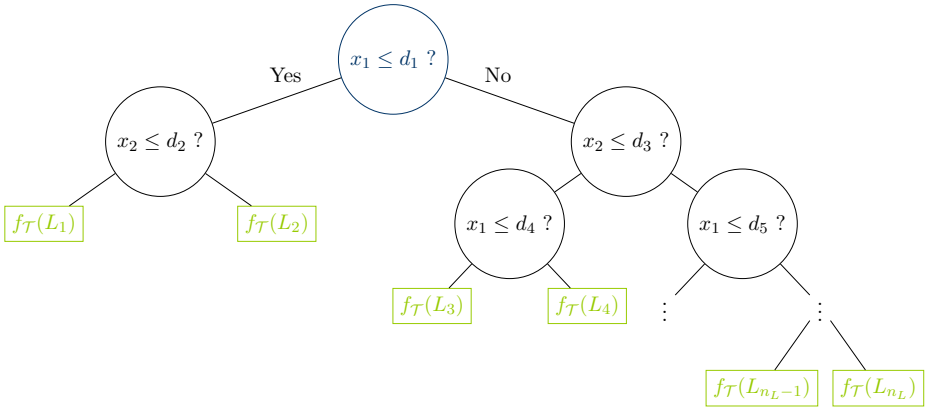


Figure 2: Decision tree example

In a classification context, the label given by the tree $\mathcal{T}$ for subject i is given by

$$f_{\mathcal{T}}(\boldsymbol{x^{(i)}}) = mode\left(\{y^{(j)}, \forall j \mid x^{(j)} \in L^{(i)}\}\right) = f_{\mathcal{T}}(L^{(i)}).$$

In a regression context, the label given by the tree $\mathcal{T}$ for subject i in leaf L_j is given by

$$f_{\mathcal{T}}(\boldsymbol{x^{(i)}}) = mean\left(\{y^{(j)}, \forall j \mid x^{(j)} \in L^{(i)}\}\right) = f_{\mathcal{T}}(L^{(i)}).$$

In both cases, a decision tree yields a single constant label[1] for any entire region: its mode or mean. The accuracy of a tree is then based on its ability to minimize the error it commits when assigning labels. Among all possible trees - thus, all possible partitions of $\mathcal{X}$ - the optimal one should be the one maximizing the label assignment accuracy. Such a tree theoretically exists but cannot generally be found in a computationally reasonable time. Therefore algorithms like CART use a top-down greedy approach: they start from an initial node - the root - containing all observations in $\mathcal{D}$. Then they find the covariate x_j and the threshold d[2] such that they optimize a splitting criterion. The root is then split into those two child nodes for which the same splitting process is repeated until a stopping criterion is triggered. Once grown, this tree is called *maximal tree*. Such a tree overfits the data, and predictions made on observations that were not used to grow the tree are usually inaccurate. That is why a last step is required: the maximal tree is pruned to a subtree that has better generalization abilities. From an algorithmic perspective, growing a CART can be summarized as such:

Input : $\mathcal{D}$, g
Output: CART

Consider the current node g, if no current node exists, create a new tree $\mathcal{T}$ with a single initial node g.
if `StoppingRules(`g`)` **then**
  Let g be a leaf with the prediction $f_{\mathcal{T}}(g)$.
else
  For all possible covariates and threshold find the pair (x_k, d) that obtain the best `SplittingCriterion(`$\mathcal{D}, x_k, d$`)`.
  Split the node g along covariate x_k at threshold d into two child nodes g_r and g_l.
  `Grow(`$\mathcal{D}(g_r)$, g_r`)`.
  `Grow(`$\mathcal{D}(g_l)$, g_l`)`.
return `Prune(`$\mathcal{T}$`)`

Algorithm 1: `Grow` algorithm for CART

A decision tree is therefore defined by its splitting criterion (`SplittingCriterion`), its stopping rule(s) (`StoppingRules`), and its pruning process (`Prune`).

2.1.1 Splitting Criterion

Originally, CART produces, at every node, a split that minimizes the heterogeneity regarding the target variable within each child node. Equivalently, the optimal split is to maximize the loss of heterogeneity between the considered node and its child nodes: the so-called *goodness-on-split*. Therefore, measures of heterogeneity are needed when the target variable is categorical - for classification tasks - and when it is numerical - for regression tasks -.

Classification: In classification, let's define p_k, $k \in \{1, \ldots, P\}$ as the proportion of observations of class k in $\mathcal{D}$. We extend this idea by defining $p_k(g)$ as the proportion of observation of class k in $\mathcal{D}(g)$. An impurity function ϕ, is a function measuring the heterogeneity, defined for p_k, $k \in \{1, \ldots, P\}$, with $p_k \geq 0$ and $\sum_k p_k = 1$ such that:

[1] or prediction, in such contexts
[2] The set of classes for categorical covariates

- $\phi(p_1, \dots, p_K) \geq 0$,
- The minimum of ϕ is reached whenever any of the $p_k = 1$, then $\phi(p_1, \dots, p_K) = 0$,
- The maximum of ϕ is reached for $\phi(\frac{1}{K}, \dots, \frac{1}{K})$,
- ϕ is symmetric with regard to its arguments.

For CART, usual classification impurities are Gini ($\phi(p_1, \dots, p_K) = -\sum_i p_i log(p_i)$), the entropy ($\phi(p_1, \dots, p_K) = \frac{1}{2}\sum_i p_i(1 - p_i)$) or the twoing measure. For our purposes, no further specificities are needed and in all generality, the impurity - or heterogeneity - of node g is measured by $I(g) = \phi(p_1(g), \dots p_K(g))$. At each node of a CART, the optimal split is chosen as the split that reduces the impurity the most. That is to say, the split that maximizes the following gain function by splitting the parent node g_p into the two child nodes g_l and g_r is

$$G(g_p; g_l, g_r) = I(g_p) - \left(\frac{\mathcal{N}(g_l)}{\mathcal{N}(g_p)} I(g_l) + \frac{\mathcal{N}(g_r)}{\mathcal{N}(g_p)} I(g_r) \right). \tag{1}$$

Of course, various other criteria and ideas for splitting exist. This paper does not aim to review all of them but we refer the astute reader to such comparisons of splitting methods (see Mingers (1989), Buntine and Niblett (1992), Breiman (1996), Shih (1999) or Drummond and Holte (2000) for instance). The efficacy of each splitting criterion has been discussed but no definitive consensus over which one is the finest exists. All measures prove desirable properties in particular scenarios while demonstrating drawbacks in others.

Regression: In a regression context, the best split can be chosen with the target variable empirical variance or mean squared error, a natural choice of heterogeneity measure. We define $MSE(g)$ the mean squared error at node g, as

$$\text{MSE}(g) = \begin{cases} \text{MSE}(g) = \sum_{i \in g} (\hat{y}_g - y_i)^2 \\ \hat{y}_g = \frac{1}{\mathcal{N}(g)} \sum_{i \in g} y_i. \end{cases} \tag{2}$$

Then, the gain function to maximize when splitting the parent node g_p into the two child nodes g_l and g_r is obviously

$$G(g_p; g_l, g_r) = MSE(g_p) - \left(\frac{\mathcal{N}(g_l)}{\mathcal{N}(g_p)} MSE(g_l) + \frac{\mathcal{N}(g_r)}{\mathcal{N}(g_p)} MSE(g_r) \right). \tag{3}$$

Even if technically, MSE is not an impurity function, we clearly see that Equation 3 is the perfect regression equivalent of Equation 1.
Thus in the following sections, we use the general notations of equation 1 with $I(g) \equiv MSE(g)$ when the target variable is numerical.

2.1.2 Stopping rules

Stopping rules can be specified. In that case, the growing phase continues until one of them is met. First of all, a node will not split any further if all observations it contains have the same target variable value. Then, a minimum improvement in the splitting criterion (for statistical test-based splitting criteria for instance), a maximum depth of the tree (parameter: `maxdepth`), a minimum number of observations in a node (parameter: `minsplit`), or a minimum number of observations in the hypothetical child nodes that would result from a new split.

2.1.3 Tree Pruning

The stopping rules affect the size of the maximal tree. No or weak stopping rules will generate a high-variance/low-bias over-fitted tree whereas constraining ones will lead to smaller, more interpretable low-variance/high-bias under-fitted trees. The idea of cost-complexity pruning developed by Breiman emerged from the need to find a compromise between the two extremes.

The main idea behind cost-complexity pruning is to consider sub-trees of the maximal tree and evaluate them with a cost function that increases as the error rate rises and decreases as the number of leaves drops. When a tree is pruned at a node, the weighted summed error of the leaves increases while the number of leaves reduces, thus a pruned sub-tree is selected only if the error gain is counter-balanced by the complexity loss. The cost of a tree $\mathcal{T}$ is given by:

$$C_\alpha(\mathcal{T}) = R(\mathcal{T}) + \alpha\psi(n_L), \qquad \alpha \geq 0, \tag{4}$$

where $R(\mathcal{T})$ is the sum of all errors or impurities of the leaves of $\mathcal{T}$, weighted by the number of individuals they represent. The function ψ is an increasing function of n_L, it is originally set to $\psi(n_L) = n_L$ in Breiman et al. (1984), but as demonstrated relevant properties when set to $\psi(n_L) = \sqrt{n_L}$ in classification settings (see Appendix A.1 for more details and references). The penalty α is the complexity parameter: the higher it is, the smaller the pruned tree. With a reasonable choice of ψ, the interest of α is that for a fixed complexity parameter value, there exists a unique smallest subtree $\mathcal{T}$ of the maximal tree $\mathcal{T}_{max}$ that minimizes $C_\alpha(\mathcal{T})$. Thus by decreasing α, we can construct a sequence of pruned optimal subtrees $[\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_{max}]$ of different sizes. This tree sequence is such that $\mathcal{T}_1$ is the root node, $\mathcal{T}_2$ a sub-tree of $\mathcal{T}$ with more leaves and accuracy than $\mathcal{T}_1$ and so on until $\mathcal{T}_{max}$, the unpruned maximal tree. With Breiman's notation, we have

$$\mathcal{T}_{max} \supseteq \cdots \supseteq \mathcal{T}_2 \supseteq \mathcal{T}_1.$$

The optimal complexity parameter value, hence the best tree in the sequence is usually selected using cross-validation.

2.2 Longitudinal notations

This paper aims to enrich the growing process of decision trees in the presence of time-varying covariates. To do so, let us introduce some notations borrowed from the existing longitudinal literature including Rizopoulos (2012) or Yao et al. (2022). Let us assume a very general setting where we want to build a dataset $\mathcal{D}_{long}$, encompassing the time-varying features of N subjects, which are repeatedly measured over time. In all generality, let us assume that among the p covariates, p_{TV} of them are time-varying and p_{TI} others are time-invariant. At time t, the set of covariates is given by $\boldsymbol{X}(t) = (X_1, X_2, \ldots, X_{p_{TI}}, X_{p_{TI}+1}(t), \ldots, X_p(t))$. In order to simplify the notations, we consider all constant features as a special case of time-varying covariates, with $\boldsymbol{X}(t) = (X_1(t), X_2(t), \ldots, X_p(t))$ with $X_k(t) = X_k$, $\forall t$ and $\forall k \in [1, \ldots, p_{TI}]$. Let $n^{(i)}$ be the number of distinct times $t_j^{(i)}, j = 0, 1, \ldots, n^{(i)} - 1$ at which subject i has been observed. At time $t_j^{(i)}$, subject i has a vector of covariates denoted $\boldsymbol{x}_j^{(i)} = \left(x_{j,1}^{(i)}, \ldots, x_{j,p}^{(i)}\right)$.

Classical longitudinal setting: For a given subject i, covariates are stored in rows, one row per observation window $[t_j^{(i)}, t_{j+1}^{(i)})$. Each row contains the unique $\left(t_j^{(i)}, t_{j+1}^{(i)}, \boldsymbol{x}_j^{(i)}, y_j^{(i)}\right)$ elements, with $y_j^{(i)}$ the target variable observed at time $t_j^{(i)}$. They are completed by the subject unique identifier i. Each row represents what we will now call an *observation*. We build $\mathcal{D}_{long}$

as the collection of all observations structured longitudinally :

$$\mathcal{D}_{long} = \left\{ \left(i, \left\{ t_j^{(i)}, t_{j+1}^{(i)}, \boldsymbol{x}_j^{(i)}, y_j^{(i)} \right\}_{j=0}^{n^{(i)}-1} \right) \right\}_{i=1}^{N}$$

Or, if displayed in a table:

ID	Time window Start	Time window End	Covariate 1	...	Covariate p	Target variable
1	$t_0^{(1)}$	$t_1^{(1)}$	$x_{0,1}^{(1)}$	...	$x_{0,p}^{(1)}$	$y_0^{(1)}$
1	$t_1^{(1)}$	$t_2^{(1)}$	$x_{1,1}^{(1)}$	...	$x_{1,p}^{(1)}$	$y_1^{(1)}$
1	$t_2^{(1)}$	$t_3^{(1)}$	$x_{2,1}^{(1)}$	...	$x_{2,p}^{(1)}$	$y_2^{(1)}$
1	$t_3^{(1)}$	$t_4^{(1)}$	$x_{3,1}^{(1)}$	...	$x_{3,p}^{(1)}$	$y_3^{(1)}$
2	$t_0^{(2)}$	$t_1^{(2)}$	$x_{0,1}^{(2)}$	...	$x_{0,p}^{(2)}$	$y_0^{(2)}$
3	$t_0^{(3)}$	$t_1^{(3)}$	$x_{0,1}^{(3)}$	...	$x_{0,p}^{(3)}$	$y_0^{(3)}$
3	$t_1^{(3)}$	$t_2^{(3)}$	$x_{1,1}^{(3)}$	...	$x_{1,p}^{(3)}$	$y_1^{(3)}$
3	$t_2^{(3)}$	$t_3^{(3)}$	$x_{2,1}^{(3)}$	...	$x_{2,p}^{(3)}$	$y_2^{(3)}$
...	...	...	...	...	...	...

Table 1: A longitudinally structured dataset

2.3 Existing longitudinal tree-based algorithms

The problem when splitting time-varying covariates: Whether they are designed for survival analysis or not, longitudinal tree-based models exist and propose various methods to include time-varying covariates that cannot naturally fit in the tree-growing algorithm described in Algorithm 1. As an illustrative example, let $x_1(t)$ be a numerical time-varying covariate. At each node, a splitting rule of the form "$x_1(t) \leq s$"[3] should be able to split subjects into two child nodes. A subject for which this rule is true at all observed times will go in one child node without any ambiguity. On the other hand, the general case where the rule is true for some periods but false for anywhere else is unclear and needs to be addressed.

The "eventually not longitudinal" methods: The most naïve model would be a regular CART, trained on all observations in the longitudinal dataset without taking the correlation between observations of the same subject into account. As stated by Segal (1992), this would simply ignore the capital aspect of dealing with longitudinal data: *"The covariation induced by making several observations of some continuous response on the same unit, as occurs with repeated measures designs, cluster designs, and longitudinal studies, poses data analytic problems. Analysis of such designs that ignore the covariance structure are known to produce incorrect variance estimate."*. Other naïve attempts consist of summarising the longitudinal trajectories of time-varying covariates with a small number of parameters. For instance, one could think of only keeping the mean value of every trajectory, the median, its final slope, the baseline value, or the most recent one, ignoring all the remaining information. This leads to a loss of precious information. A similar idea is to regress every longitudinal covariate against

[3]Note that the same reasoning can be applied to categorical time-varying covariates as well.

time and possibly other features, within-subjects to include the parameters of the regression - intercept and slope - as baseline covariate. It can be argued that if the longitudinal covariates are all strongly linearly associated with time, which is rarely the case in practice, this kind of alternative solution can be relevant. Eo and Cho (2014) proposed a model called mixed-effects longitudinal tree (MELT) able to handle longitudinal response by fitting a mixed-effect model at each node of the tree. Subjects are then split based on the heterogeneity of their slopes. Kundu and Harezlak (2019) extended this idea of resuming information contained in the longitudinal covariates by a combination of splits on baseline covariates and implemented it in the R package `LongCART`. Other approaches such as Ritschard and Oris (2005) and more recently Moradian et al. (2021) designed longitudinal trees that use lagged response values as potential predictors, but still do not treat either the outcome or the covariates as inherently dynamic with time. Overall, in these methods, information is lost during the process, and the number of measurements per subject in real datasets can be too small to obtain satisfying regression parameters.

The "CART-extended" methods: Segal (1992) and De'Ath (2002) proposed independently the first applications that clearly define an extension to the CART method and directly account for correlation in the response variable. They both suffered limitations as they were designed for a longitudinal response but time-fixed covariates where all the subjects were measured at the same observation times, with the same interval between them. Segal's regression tree consisted of imputing a covariance structure in the split procedure. This led to many theoretical questions about defining that covariance structure as well as practical ones regarding the complexity of the computations. On the other hand, De'ath procedure simply modified the CART algorithm by allowing it to consider an entire matrix containing all the observations for one subject as a single observation. Allowing that was done by using the gain of MSE as a splitting criterion, and replacing the 1-dimensional mean in the MSE with a multi-dimensional mean modified with a covariance structure; the prediction given by the tree would then be the multi-dimensional mean of the observation in the terminal nodes. In both cases, those methods can be seen as fitting a model to the longitudinal outcome at every node as part of the splitting criterion. Segal's procedure has never been publically available but De'Ath's algorithm of the tree was published as the R package `mvpart` (see De'Ath (2006)) which is no longer maintained and accessible today. More recent works by Larsen and Speckman (2004) as well as Hsiao and Shih (2007) followed and improved the idea of De'ath, by redefining the node impurity measure with the Mahalanobis distance and estimating the covariance matrix from the whole data set. It is worth mentioning that other articles extended the idea of Segal, to binary responses and classification trees (Zhang (1998)), in a clustering context using deviance as a goodness-of-fit criterion for partitioning (Abdolell et al. (2002)) and then to every type of longitudinal response - not only continuous or binary - using Generalized estimating equations (Lee (2005); Lee et al. (2005); Lee (2006)). Such models show advantages in terms of predictive ability and interpretability but do not handle time-varying covariates.

The "state-of-the-art" methods: In Hajjem et al. (2011a), Sela and Simonoff (2012) and their respective extensions (see Hajjem et al. (2014); Capitaine et al. (2021)[4] and Fu and Simonoff (2015)), a general mixed-effect model is assumed for the longitudinal outcome. The tree-based part only predicts fixed effects whereas individual estimated parameters account for all the time-varying effects. Such approaches can estimate longitudinal outcomes but the

[4]Louis Capitaine also worked on a promising generalization of decision trees and forests that must be acknowledged. We refer to Appendix A.2 for further details.

inclusion of time-varying covariates is handled via the pseudo-subject workaround detailed in the next paragraph. It relies on the assumption that all the dependency between several observations of the same subject is captured by the random effect of the mixed model. In a survival setting, Fu and Simonoff (2016) and its extensions (see Yao et al. (2020)) proposed a model based on those ideas: they allowed subjects to be divided into pseudo-subjects and used an adjusted log-rank test in the splitting procedure to accommodate for left truncation and ensure that the independence implicit assumption does not lead to biased results. We strongly refer the astute reader to the works mentioned in this paragraph as we consider they are the most advantageous approaches today. The algorithms corresponding to their respective work are the R packages `REEMtree`, `LongituRF`, `LTRCtrees` and `LTRCforests`, the R function `REEMctree` and the Python library `MERF`.

Pseudo-subjects For LTRC and mixed-effect tree-based models, the time dependency is dealt with by design and assumptions: every observation is considered independent regarding the splitting process of the tree. They consider the unmodified $\mathcal{D}_{long}$ as an input and run through a CART-like growing process, finding optimal binary decision rules at each node of the tree. Whenever a split produces an ambiguity as described in paragraph 2.3, the periods $\left[t_j^{(i)}, t_{j+1}^{(i)}\right)$ where their splitting rule "$x_1(t) \leq s$" is true would go to the left node, and the other would go to the right node, thus dividing one subject into several pseudo-subjects. It cleverly addresses the time-handling issue when the bias that comes with correlated right-censored and left-truncated (LTRC) observations is neutralized otherwise. In such models, any individual can be spread in many different tree leaves - even if, at any fixed time, any individual will have a single observation that will fall into a unique one. It is undoubtedly an unbiasing and desirable property in terms of prediction but it may be at the expense of interpretability. In our opinion, none of these procedures can inherently handle time-varying covariates in a satisfying way for data-mining and visualization purposes. Indeed, following CART's example, a unique trajectory per subject would ensure a clear visualization of the data: the algorithm should be designed to separate **individuals** whose features are significantly diverging regarding the target variable rather than **pseudo-subjects**.

3 Time penalized trees

We present here the building blocks of a new way to think about decision trees in the presence of time-varying covariates: time-penalized trees or `TpT`. Let $\mathcal{D}_{long}$ be a longitudinal dataset, and $\mathfrak{T} = [0; \max_{i,j}(t_j^{(i)})]$ be the continuous observation interval of time. We define $\mathcal{D}(t)$ as the dataset containing, for every subject i, her unique observation with the maximal observation time $t_j^{(i)}$ such that $t_j^{(i)} \leq t$ and $T^{(i)} \geq t$, where $\boldsymbol{x^{(i)}}(t) \in \mathcal{X}(t)$ is the vector of covariates and $y^{(i)}(t) \in \mathcal{Y}(t)$ the target variable at time t. Eventually, $\mathcal{N}(t) = |\mathcal{D}(t)|$ is the total number of subjects under study at time t. Let g be a node, which is also identified with a subregion of $\mathcal{X} \otimes \mathfrak{T}$ it represents. We thus define $\mathcal{D}(g)$, the set of observations in the node g and $\mathcal{N}(g) = |\mathcal{D}(g)|$ the number of subjects it contains.

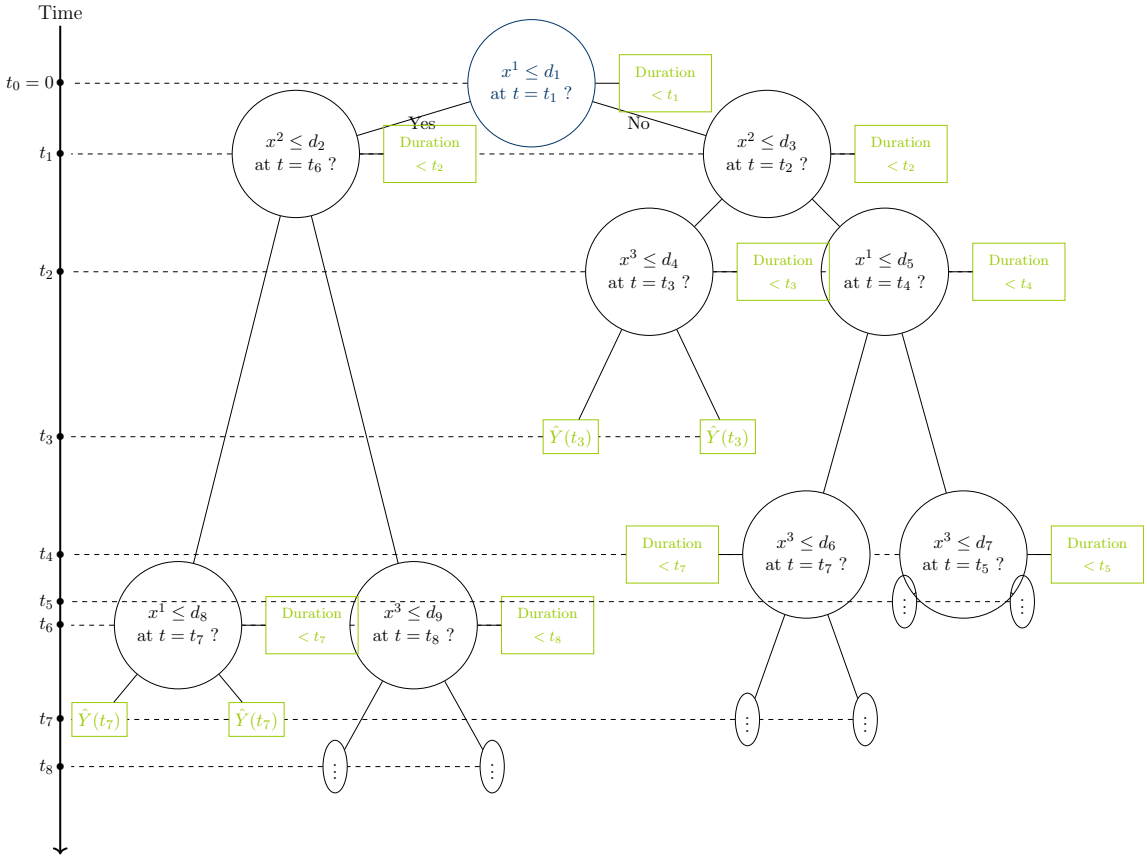
The idea behind `TpT` is to build a tree that benefits from all the longitudinal information available and where the concept of time is central: at each node, we chose to split along covariates and time. As stated in Section 2.1, a tree-growing algorithm is defined by its splitting criterion, stopping rule(s), and pruning process. This applies to `TpT` and the algorithm we suggest can be described as:

Input : $\mathcal{D}_{long}$, g_p
Output: `TpT`

Consider the current parent node g_p, if no current node exists, create a new tree $\mathcal{T}$ with a single initial node g_p at time $t_p = 0$.
if `StoppingRules(`g_p`)` **then**
  Let g_p be a leaf.
else
  For all possible covariates x_k, threshold d and time point $t_c >= t_p$ find the triplet (x_k, d, t_c) that obtain the best `SplittingCriterion(`$\mathcal{D}_{long}(t_c), x_k, d$`)`.
  Split the node g_p: all subjects with $T^{(i)} < t_c$ go to a "duration leaf" g_t. All other subjects - with $T^{(i)} \geq t_c$ - are split along covariate x_k at threshold d into two child nodes g_r and g_l.
  Iteratively run `Grow(`$\mathcal{D}_{long}(t_c)$, g_r`)`.
  Iteratively run `Grow(`$\mathcal{D}_{long}(t_c)$, g_l`)`.
return `Prune(`$\mathcal{T}$`)`

Algorithm 2: `Grow` algorithm for `TpT`

In the end, a final Time-penalized Tree would look like that:

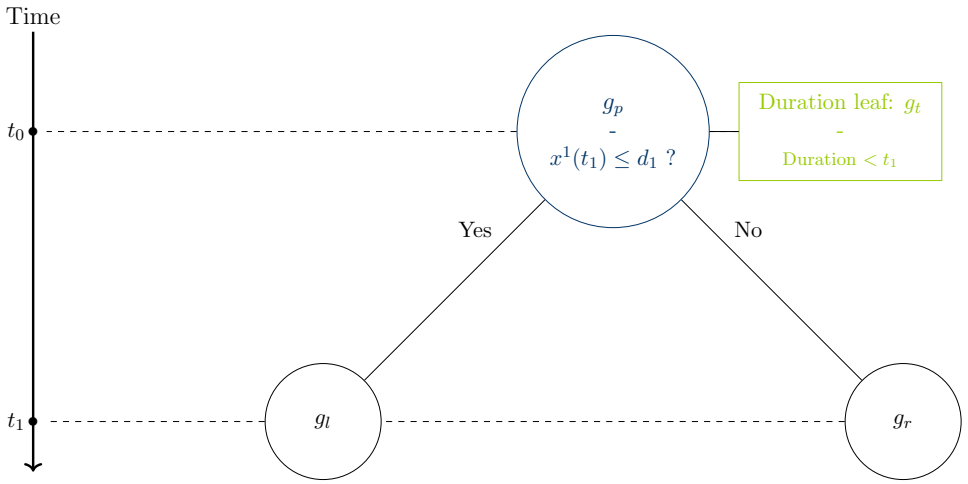


With the root node appearing in blue, leaf nodes appearing in green (*duration leaves* displayed horizontally from every node and *terminal leaves* at the very end of every branch) and the timeline being depicted on the left side of the figure.

Defining `TpT` stopping rules is exactly similar to CART (see Section 2.1.2). Its splitting criterion and pruning process, meanwhile, are modified and discussed in the sections below.

3.1 `TpT` splitting criterion

The split function for `TpT` is the critical contribution of the algorithm but it is rather straightforward. We want to select the split on a covariate, at a threshold and a time that will maximize a time-penalized split criterion. A single split of a node g_p would look like this:



To obtain such a split, we have to define a time-penalized split criterion, as

$$G_\gamma(g_p; g_l, g_r, g_t) = \left[I(g_p) - \left(\frac{\mathcal{N}(g_l)}{\mathcal{N}(g_p)} I(g_l) + \frac{\mathcal{N}(g_r)}{\mathcal{N}(g_p)} I(g_r) + \frac{\mathcal{N}(g_t)}{\mathcal{N}(g_p)} I(g_t) \right) \right] \cdot e^{-\gamma \cdot (t_c - t_p)},$$

With $\gamma \in \mathbb{R}^+$.

With $I(g)$ an impurity or MSE function as described in Section 2.1.1, t_p and t_c the respective times of the parent node and child nodes and γ the penalty parameter. We can immediately see that this is simply the classical CART splitting criterion with an additional exponential penalty term, depending on how distant in time the considered split is. The exponential penalty that we propose induces that the more time distance there is between a parent node and its potential child node, the more penalized the split. In other words, splits are chosen where covariates AND time points are informative about the target variable; we first try to find close splits if they can detect heterogeneity but distant splits will be considered if they are very informative. We can find examples of this type of exponential consideration of time in time series analysis with exponential smoothing (see Brown (1956); Holt (2004)), where exponential functions are used to assign exponentially decreasing weights over time. We only referenced two rather distant instances of tree-based modified splitting criteria where exponential weights were introduced. First in Section 5.5 of Bremner (2004) thesis that used localized splitting criteria that are based on local averaging in regression trees or local proportions in classification trees, weighted by exponential weights. The weights have no link to time or a measure of distance from the previous node. Goldstein (2014) also suggested using exponential weights in tree-based algorithms to promote splits on covariates that were already used in previous splits over others.

The partitioning procedure of `TpT` can also be visualized similarly to Figure 1. Consider two time-varying covariates $x_1(t)$ and $x_2(t)$: at depth 0 and $t = 0$, the tree is only a root and $\mathcal{D}(0)$ is not partitionned. We can see that the first split, at $t = t_0$ creates a division of $\mathcal{D}(t_1)$ such as

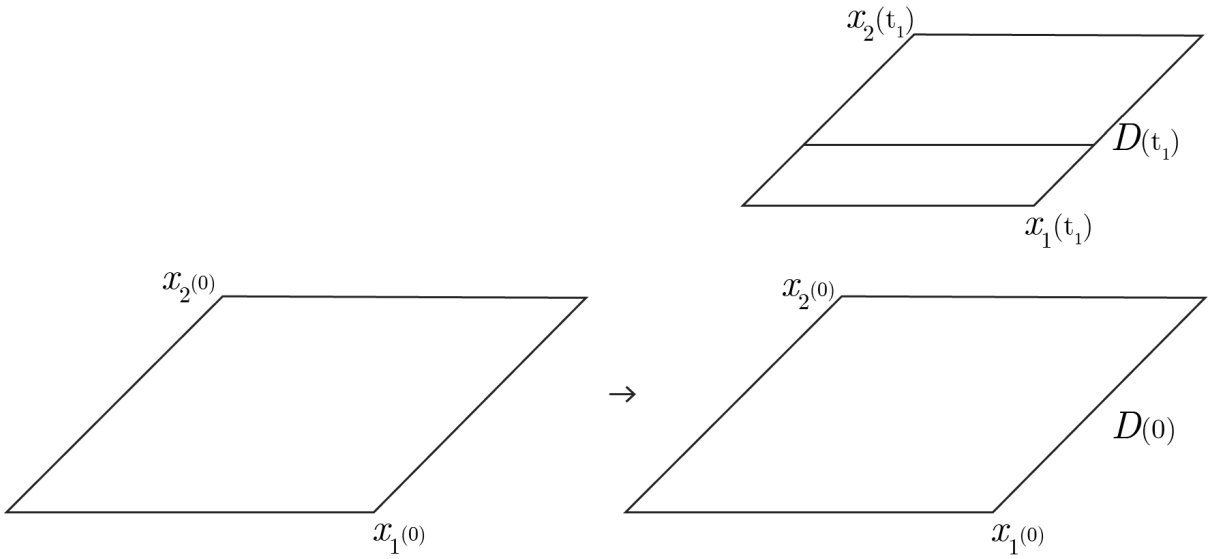


Figure 3: `TpT` 1-depth recursive partitioning

At depth 2 and $t = t_1$, all subjects that have been observed up to $t = t_2$ within each partition, are once again split into two subgroups. This creates a division of $\mathcal{D}(t_2)$ such as

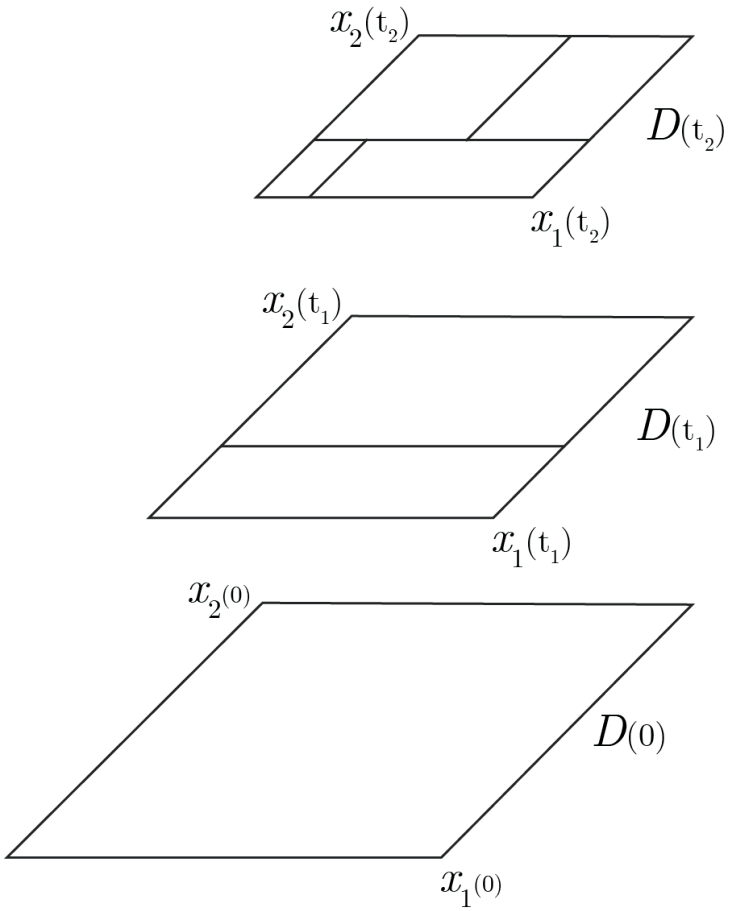


Figure 4: `TpT` 2-depth recursive partitioning

Eventually, the complete partitioning procedure can be visualized as in Figure 5. It is the representation of a classical binary split procedure, with the inclusion of a time dimension. The routes of all subjects can be displayed in that representation: the red, blue and green paths in Figure 5 are examples of such individual trajectories.

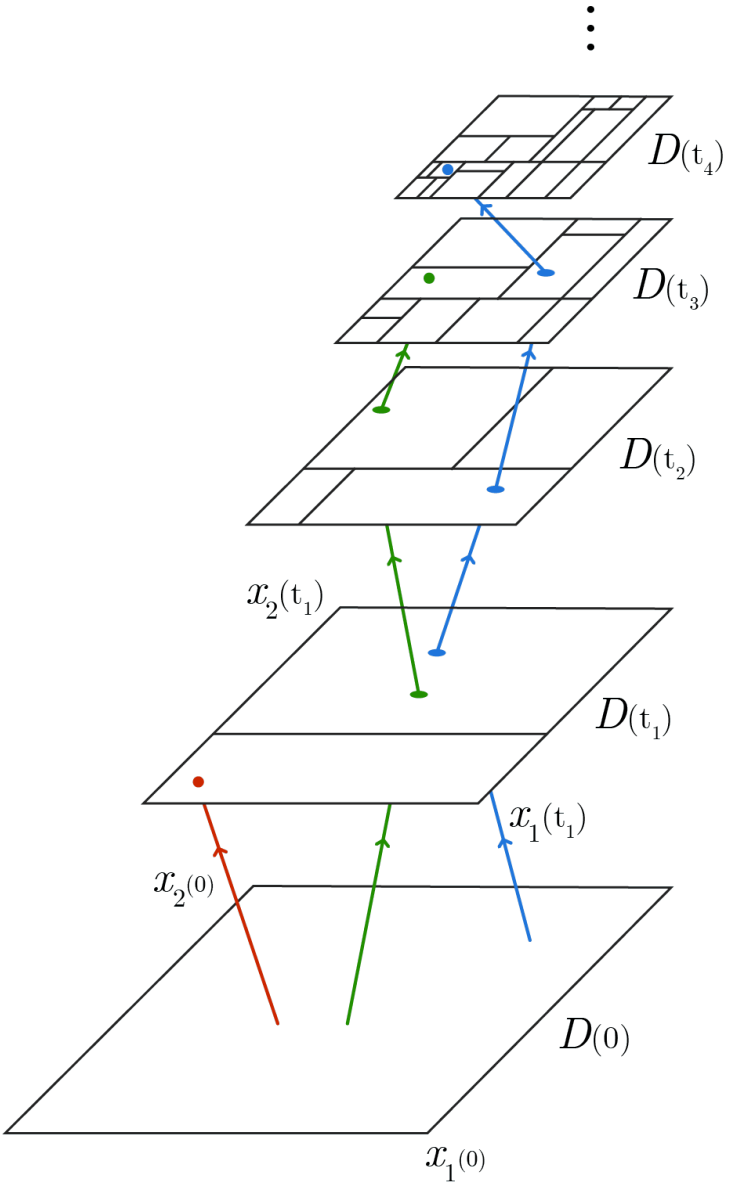


Figure 5: TpT recursive partitioning and individual trajectories

Remark 1

Several things need to be noted regarding the partitioning illustration depicted in Figure 5. First, duration leaves are not represented here: the red trajectory for instance, does not split after time $t = t_1$ because the policy it represents has a seniority inferior to t_2. Its course after time t_1 being unknown, it stops and this region of $\mathcal{D}(t_1)$ is a duration leaf for similar subjects. Secondly, this illustration shows that divisions of early steps transpose into continuous partitions in further steps: this is not true in general. The two groups formed by the partition of $\mathcal{D}(t_1)$ may not be represented by a unique region of $\mathcal{D}(t_2)$, split at a constant threshold over one covariate. For instance, the set of all individuals with a policy face amount (FA) $\leq 10,000$ at time t_1 is

composed of individuals without any observation at time t_2, of policyholders with an FA $> 10,000$ and ones with an FA $\leq 10,000$. Eventually, it is also to be noted that not every disjoint region splits at every step of the partitioning. There are times when several splits occur, others where only one region is partitioned, and others where none. All those points are not depicted in Figure 5 for simplicity's sake.

We can already foresee that higher values of γ ensure that the next optimal split is more likely to be close in time to the previous node (a distant split is to be chosen only if it is very interesting). The produced `TpT` will be close to a CART with all longitudinal covariates values blocked at $t = t_0$. And it can be easily proven that

$$TpT(\mathcal{D}_{long}) \underset{\gamma \to +\infty}{\longrightarrow} CART(\mathcal{D}_{long}(t_0)). \tag{5}$$

It allows a `TpT` to explore the covariates space but prevents it from exploring the time dimension. On the contrary, lower values of γ are more likely to produce distant splits and the constructed `TpT` will show similarities with a CART where all longitudinal covariates values rapidly approach their final value. It allows a `TpT` to split along the time dimension quickly but prevents it from exploring the covariates space at any given time.

Remark 2
Because the impurities of the parent and child nodes can be computed at different time points, it can happen that $G_\gamma < 0$. Such cases imply that a specific stopping rule must be enforced for `TpT`: G_γ must be positive for a node to split. Otherwise, it would allow ineffective splits.

3.2 `TpT` Pruning process

For a `TpT`, the penalty parameter γ affects the tree's dimensions (depth and number of leaves, see Section 4 for an analysis on the matter). An optimal γ that minimizes the impurity of the tree (the weighted sum of all leaves impurities) can be chosen but it is not a pruning process comparable to cost-complexity pruning. For a given γ, a maximal `TpT` can be grown and may overfit the data. It is obvious in a prediction context but also true in a data-mining framework. To control for bias and overfitting, various pruning strategies can be considered. First, Breiman's cost-complexity pruning is still well-defined under the `TpT` framework, for a given γ, and can be applied as long as all term nodes - denoted as g_t in previous illustrations and algorithm - are considered as leaves. We suggest a slightly different adaptation of this pruning strategy to select both α and γ simultaneously. It consists of selecting the pair (α, γ) that minimizes $C_\alpha(\mathcal{T})$, the cost of the tree. Simpler pruning strategies such as Reduced Error Pruning (see Quinlan (1987)) can also be used. Their advantages and flaws are discussed in Esposito et al. (1997) as well as their tendency to over/under-prune.

3.3 Time-to-event

Survival analysis can also be carried out directly under the `TpT` framework. Indeed, for data mining purposes, subjects are distributed in the final tree considering their last observed time $T^{(i)}$. Censorship and event occurrences are visible in the term nodes g_t. Adapting `TpT` for prediction tasks would require additional work to account for censorship (see Vock et al. (2016)), but this specific topic is not in the scope of our paper.

4 Applications

We decided to test `TpT` on real datasets. Such a longitudinal data mining algorithm can prove useful in various fields (medicine, sports analytics, taxonomy, biology), here we applied it to a life insurance customer segmentation analysis. For that purpose, we use a real-world dataset of 983 policyholders (PH) and we investigate the link between the PH's characteristics through time and the final outcome of their policies. Throughout the lifetime of such insurance policies, a series of events can occur. Firstly, one policyholder's coverage can be increased with premium payments that are highly flexible, both in terms of amount and frequency, and are adjusted according to the policyholder's financial circumstances and preferences. Additionally, policyholders may decide to reduce their coverage by withdrawing a portion of their policy. We refer to these events as partial lapses, enabling PHs to adjust their coverage to better align with their changing needs. Other financial operations can occur, such as the payment of interest or profit sharing from the insurer to the PH, and the payment of fees from the PH to the insurer. Insurance companies' information systems are usually designed to keep track of those operations at the policy level, thus actuaries and life insurers often have access to the complete history of their policyholders as the information system is updated in real-time. Eventually, one's insurance plan ends whenever the PH dies or decides to terminate it by lapsing. In the end, the timeline of such insurance policies can be illustrated below:[5]

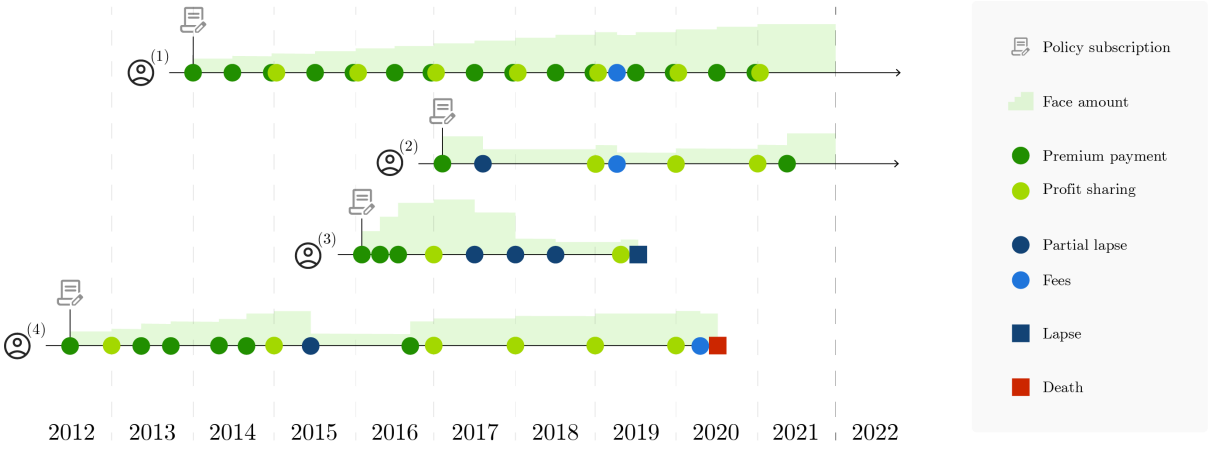


Figure 6: Example of policyholders timelines

At any given time a policy is either active, has been lapsed by the policyholder, or has ended because of her death. Among all insurance plans subscribed between 1998 and 2019, in our dataset, 57.4% are active, 22.8% ended with the death of the policyholders, and 19.8% lapsed. We only consider uncensored observations here, we thus have 55% of churned policies and 45% that ended with death. For this application, our data mining goal is to gain insights into the PH's pathways that lead to these different outcomes. We want to find time-dependent clusters of individuals with similar timelines and outcomes at a given time. This is thus a time-dependent classification problem, where the target variable is the final outcome of the policies, the tree grows with the survival time and splits on potentially time-varying covariates such as age, rate, Customer Lifetime Value (CLV), face amount (FA) or gender.

[5]Illustration taken from **?**

4.1 Properties of TpT for the maximal tree

First of all, Table 2 below displays the results obtained by TpT with various choices for the time penalty parameter γ. It shows the dimensions of TpTs (depth and number of leaves), their global impurities and costs, the highest time point when a split occurred, and the average time at which any subject is split. Graphs of those results can be found in Figure 7. Here we considered unpruned trees using the time-penalized version of the Gini impurity measure as a splitting rule and without any stopping criterion. For this application, we computed the cost of the tree with a choice of $\alpha = \sqrt{\frac{3 \log 2}{2N}}$, suggested by Scott and Nowak (2006). The pruning process then only consist of selecting γ as the solution of $\underset{\gamma}{\text{argmin}} C_\alpha(\mathcal{T})$.

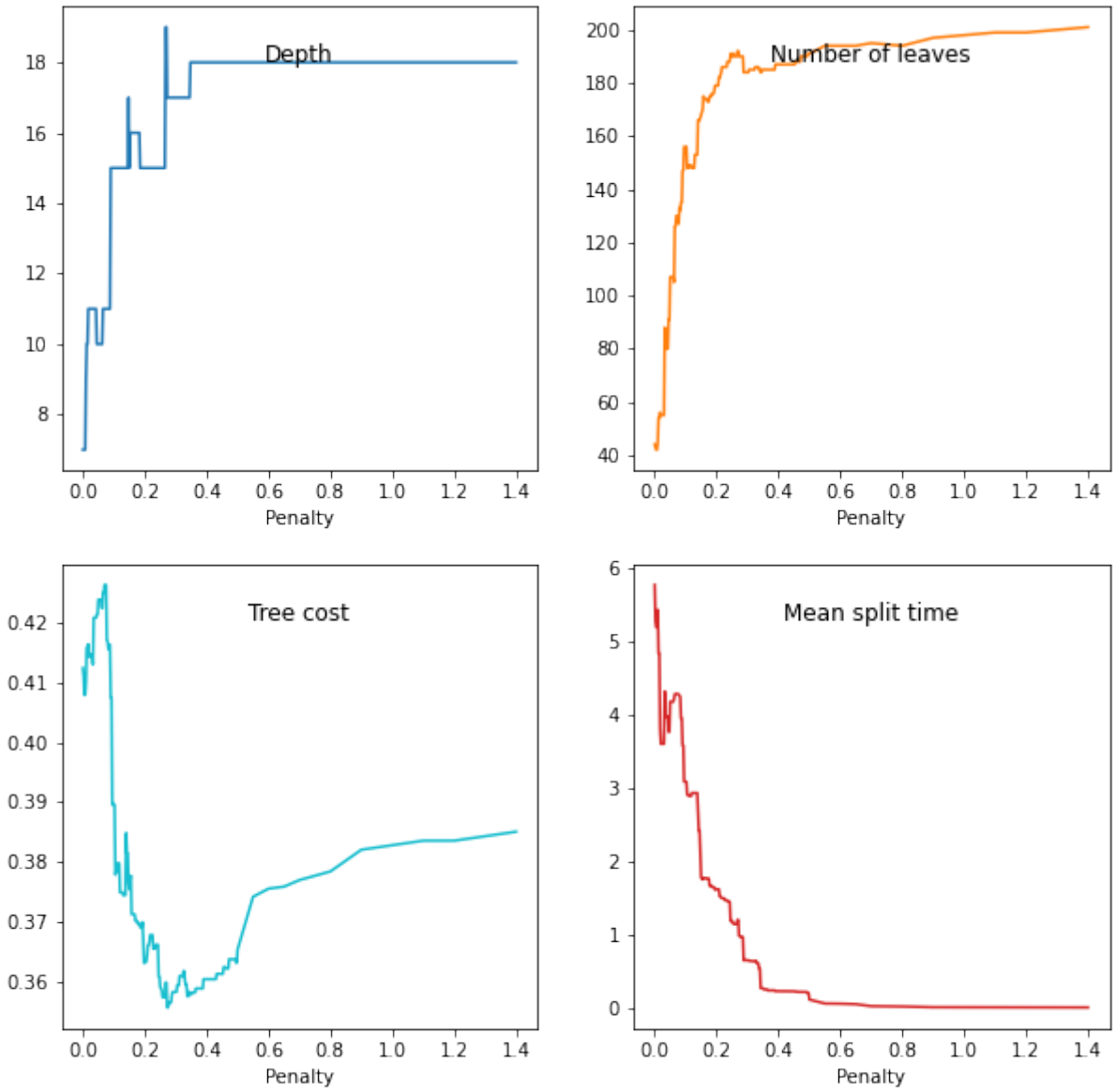


Figure 7: Characteritics of unstopped and unpruned TpT depending on the time penalty

More results and graphs obtained with this setting, but also with different impurity measures are displayed in Appendix A.3.

Time penalty γ	Runtime	Depth	# of terminal leaves	# of duration leaves	Total # of leaves	Tree cost	Max of split times	Mean of split times
0.0000	719.07	7	27	17	44	0.412	20.0	5.768
0.0025	733.95	7	27	16	43	0.411	20.0	5.276
0.0050	732.73	7	27	15	42	0.408	20.0	5.192
0.0100	725.95	9	30	15	45	0.410	20.0	5.429
0.0125	795.26	10	35	19	54	0.416	20.0	4.832
0.0175	862.99	11	35	21	56	0.416	20.0	3.81
0.0200	883.2	11	35	20	55	0.414	20.0	3.599
0.0300	868.98	11	35	20	55	0.415	20.0	3.601
0.0325	1007.65	11	55	33	88	0.413	20.0	4.315
0.0350	970.33	11	53	27	80	0.421	20.0	3.977
0.0450	1046.33	10	61	30	91	0.421	20.0	3.763
0.0500	1062.24	10	71	36	107	0.424	20.0	4.175
0.0625	1069.06	10	69	36	105	0.422	20.0	4.209
0.0650	1141.95	11	79	47	126	0.425	18.0	4.277
0.0700	1137.4	11	81	49	130	0.426	18.0	4.282
0.0775	1117.22	11	83	44	127	0.417	18.0	4.272
0.0800	1122.76	11	87	46	133	0.417	18.0	4.26
0.0825	1133.39	11	87	45	132	0.416	18.0	4.238
0.0850	1165.53	11	89	46	135	0.416	18.0	3.947
0.0900	1287.23	15	98	49	147	0.408	18.0	3.579
0.0950	1350.29	15	105	51	156	0.390	18.0	3.085
0.1025	1340.77	15	106	50	156	0.389	18.0	3.083
0.1050	1346.53	15	104	44	148	0.378	18.0	2.906
0.1075	1349.57	15	104	44	148	0.379	18.0	2.898
0.1125	1351.12	15	105	44	149	0.380	18.0	2.888
0.1200	1347.56	15	105	43	148	0.375	18.0	2.929
0.1300	1378.96	15	112	41	153	0.374	16.0	2.929
0.1400	1561.86	15	118	48	166	0.385	16.0	2.645
0.1425	1559.3	15	122	44	166	0.381	16.0	2.416
0.1475	1685.14	17	124	43	167	0.377	16.0	2.129
0.1500	1712.25	15	128	40	168	0.375	15.0	1.778
0.1525	1677.49	16	130	39	169	0.377	15.0	1.76
0.1550	1709.18	16	131	39	170	0.378	15.0	1.749
0.1575	1692.19	16	137	38	175	0.371	14.0	1.773
0.1700	1686.58	16	137	36	173	0.370	14.0	1.762
0.1775	1687.16	16	140	35	175	0.370	13.0	1.663
0.1850	1695.6	15	143	33	176	0.369	13.0	1.647
0.1925	1687.49	15	144	33	177	0.370	13.0	1.64
0.1950	1704.68	15	146	33	179	0.367	13.0	1.608
0.1975	1704.85	15	146	33	179	0.363	13.0	1.618
0.2075	1679.89	15	149	33	182	0.364	12.0	1.617
0.2100	1721.42	15	150	32	182	0.366	12.0	1.521
0.2125	1741.25	15	152	31	183	0.366	12.0	1.519
0.2150	1740.3	15	155	29	184	0.367	12.0	1.502
0.2175	1746.67	15	157	29	186	0.368	12.0	1.492
0.2275	1730.43	15	158	28	186	0.365	12.0	1.465
0.2350	1717.29	15	160	28	188	0.366	12.0	1.457
0.2375	1724.14	15	161	27	188	0.366	12.0	1.455
0.2500	1796.99	15	165	25	190	0.359	14.0	1.184
0.2525	1809.4	15	168	23	191	0.359	13.0	1.158
0.2550	1810.04	15	169	22	191	0.359	13.0	1.143
0.2575	1832.64	15	169	21	190	0.357	13.0	1.142
0.2675	1854.5	19	169	23	192	0.360	11.0	1.198
0.2725	1850.47	17	173	17	190	0.356	10.0	0.981
0.2775	1857.1	17	174	16	190	0.356	10.0	0.965
0.2875	2025.16	17	173	11	184	0.358	10.0	0.645
0.2900	2045.07	17	174	10	184	0.358	10.0	0.648
0.3050	1976.4	17	175	10	185	0.359	10.0	0.639
0.3125	1969.5	17	176	9	185	0.361	10.0	0.635
0.3250	1960.95	17	176	10	186	0.362	10.0	0.64
0.3300	1949.79	17	176	10	186	0.360	10.0	0.606
0.3325	1976.88	17	177	9	186	0.359	10.0	0.604
0.3375	1965.69	17	176	9	185	0.357	11.0	0.534
0.3425	1953.18	17	179	5	184	0.358	6.0	0.267
0.3475	1947.54	18	181	4	185	0.358	5.0	0.265
0.3650	1962.15	18	181	4	185	0.359	5.0	0.236
0.3900	1918.64	18	183	4	187	0.360	5.0	0.226
0.4325	1809.19	18	183	4	187	0.361	5.0	0.223
0.4625	1683.22	18	185	3	188	0.362	5.0	0.216
0.4725	1651.89	18	187	3	190	0.364	5.0	0.213
0.4950	1620.57	18	188	3	191	0.363	5.0	0.199
0.5000	1626.91	18	189	2	191	0.365	5.0	0.111
0.5500	1636.21	18	193	1	194	0.374	5.0	0.057
0.6000	1629.73	18	193	1	194	0.375	5.0	0.055
0.7000	1572.03	18	194	1	195	0.377	5.0	0.021
0.8000	1566.33	18	194	0	194	0.378	5.0	0.016
0.9000	1571.42	18	197	0	197	0.382	5.0	0.007
1.0000	1575.33	18	198	0	198	0.383	5.0	0.006
1.1000	1549.67	18	199	0	199	0.384	5.0	0.005
1.3000	1509.43	18	200	0	200	0.384	5.0	0.004
1.4000	1511.22	18	201	0	201	0.385	5.0	0.003

Table 2: Characteritics of `TpT` depending on the time penalty

We can observe that the depth and number of leaves grow with γ. This was to be expected, as a `TpT` that does not penalize time-distant splits will quickly find high impurity-gain splits at distant times thus preventing the exploration of less distant time periods. Conversely, the same phenomenon explains that the average time when splits occur is a decreasing function of γ. As the penalty parameters get high values, any future split is heavily penalized and can not compete with splits at time t_0, regardless of their potential unpenalized gain. Eventually and very interestingly, we observe that the unpenalized `TpT`, as well as the heavily penalized one, are not optimal in terms of global impurity. There exists an optimal choice of γ that generates a `TpT` minimizing the sum of its leaves impurities. This tree has a penalty parameter of 0.2725, a depth of 17 and a number of leaves of 190 - 173 terminal leaves and 17 duration leaves - and is displayed in Appendix A.3.

Such trees, without stopping criterion and post-pruning are useful to discuss the properties of `TpT`s but do not yield immediate insights on our dataset. Nevertheless, there is one statistic that proves to be insightful: the distribution of times when splits occur. Obviously, with an exponentially penalized splitting criterion, the more distant from its parent time t_p a split time t_c is, the more penalized it is and the less likely it is to be selected. The a priori probability for time to be selected as a split time is $\propto e^{-\gamma \cdot (t_p - t_c)}$ Thus, by weighting this distribution with an exponential factor, we balance this bias and retrieve the importance of the time periods. In the optimal unpruned and unstopped tree, the splitting time points are distributed as such:

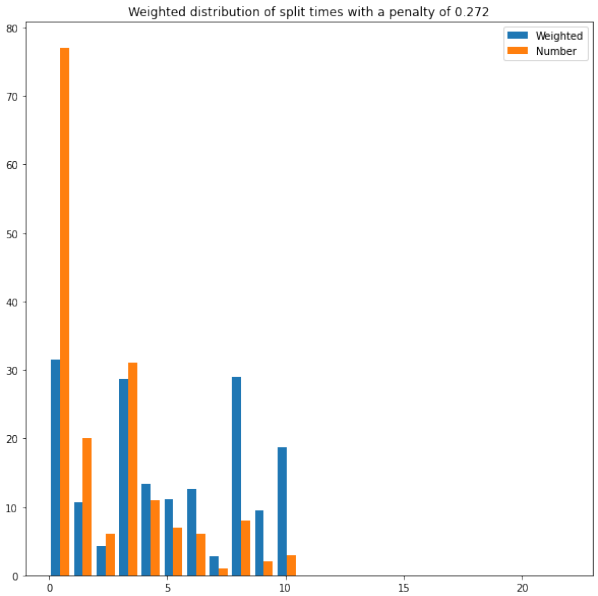


Figure 8: Split times distribution for the optimal unstopped and unpruned `TpT`

In the weighted histogram, we can clearly see that some periods seem to be critical split points that differentiate active policies from lapsers or policies that are likely to end with the policyholder's death. Interestingly, we see that $t = 0$ and $t = 8$ are particularly important in terms of differentiation between policies' outcomes. For $t = 0$, the insight is clear: most of the information that separates the churners from policies that end with the PH's death can be retrieved from the baseline covariates: for instance, the age at subscription seems to be very informative - older PHs are more exposed to the mortality risk - and thus is selected at baseline. Regarding the important splits at $t = 8$, they correspond to splits on age, CLV, or

FA. CLV is highly dependent on both age and FA, thus we could argue that age and FA are the most informative covariates at $t = 8$. By law, French life insurance plans ensure that when a given policy is at least eight years old, the policyholder can lapse without any surrender penalty. This is a clear incentive not to churn before one's policy reaches 8 years of seniority. It seems consistent to observe that this threshold is pointed out in our analysis. The third year of seniority comes right after $t = 0$ and $t = 8$ in terms of temporal importance, which does not have any obvious business justification. However, every split at $t = 3$ is either a split on CLV or the FA of the policy, thus we can argue that the final outcome of a policy seems dependent on its FA 3 years after subscription. Similarly, the unpenalized suboptimal `TpT` with $\gamma = 0$, depicted in Figure 11 only splits at times $t = 0$, $t = 3$ or $t \geq 8$, with respectively 1, 1 and 24 splits.

This application has also been tried on a larger longitudinal dataset, containing the 119,431 observations of 9,873 PHs. Characteritics of `TpT`s grown with various γ are discribed in Figure 9 It gives the split times distribution in Figure 10. Due to the heavy computation time, all other analysis are carried out on the smaller dataset.

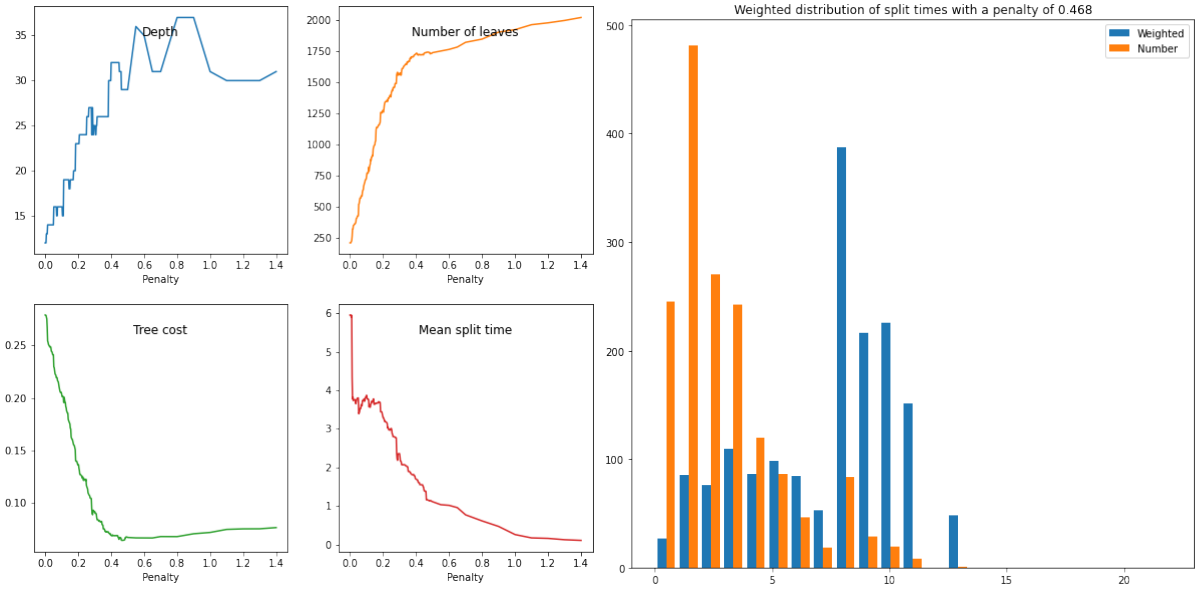


Figure 9: Characteritics of unstopped and unpruned `TpT`, trained on 9,873 PHs with various time penalties

Figure 10: Split times distribution for the optimal unstopped and unpruned `TpT`, trained on 9,873 PHs

In all following vizualisations, all leaves or regions that contain a majority of policies that ended with the PH's death are labelled "**D**", and all those that contain a majority of policies that ended with lapse (or churn) are labelled "**C**". In terms of colors, the proportion of each class is represented by a nuance between (for a 100% proportion of "**D**") and (for a 100% proportion of "**C**"). For example, a leaf or region with 50% of churners is represented by the color .

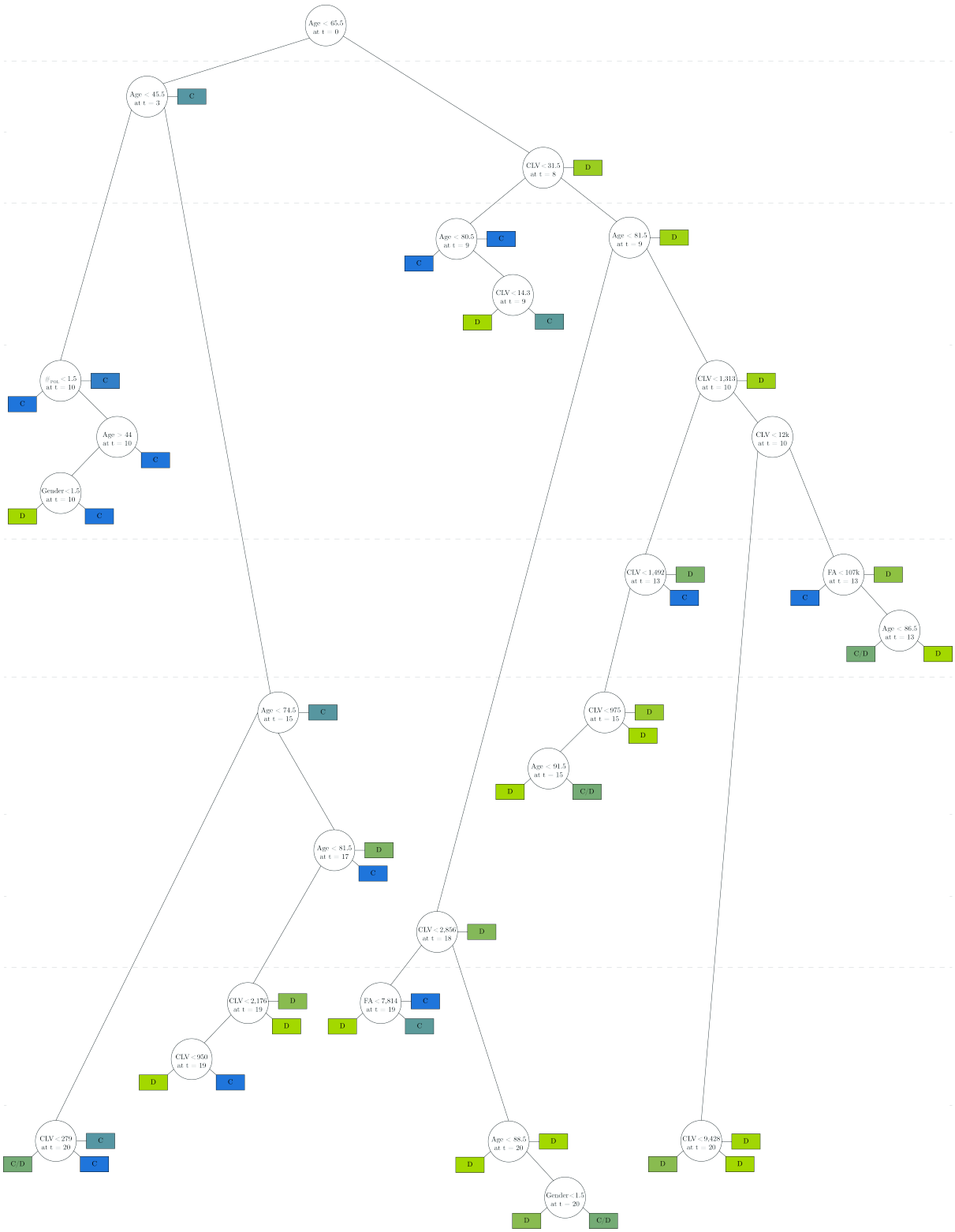


Figure 11: Overfitted unpenalized, unstopped and unpruned `TpT`

4.2 Use-case with minsplit

A clear strength of decision trees is their interpretability. Obviously, trees with hundreds of leaves each containing a handful of subjects can not be interpreted. Here we decided to inves-

tigate the results obtained by `TpT`s with various γ, using the time-penalized version of the Gini impurity measure as a splitting rule and including a stopping criterion: any leaf must contain at least 50 individuals otherwise it does not split. This choice of stopping rule is not really close to the default value for the `minsplit` parameter in most CART implementations, but it will generate shorter, less overfitted `TpT`s, better suited for direct interpretability and data analysis. Here are the results for `TpT` on our longitudinal dataset, with *minsplit*= 50. Graphs of those results can be found in Figures 12.

A METTRE A JOUR !!!!!!

Time penalty γ	Runtime	Depth	# of terminal leaves	# of duration leaves	Total # of leaves	Tree cost	Max of split times	Mean of split times
0.0000	587.03	4	9	7	16	0.319	15.0	4.309
0.0025	736.31	6	11	6	17	0.306	15.0	2.134
0.0075	730.35	6	11	5	16	0.306	15.0	2.102
0.0200	726.84	6	11	4	15	0.306	15.0	1.97
0.0275	789.2	6	13	6	19	0.300	8.0	2.203
0.0325	784.67	6	13	5	18	0.300	8.0	2.131
0.0350	817.64	6	14	4	18	0.296	9.0	1.762
0.0650	840.38	7	15	3	18	0.297	9.0	1.129
0.0925	841.11	7	15	2	17	0.299	8.0	0.88
0.1100	850.24	7	15	1	16	0.300	8.0	0.564
0.1250	873.15	7	16	1	17	0.298	5.0	0.423
0.1375	873.64	6	15	2	17	0.297	5.0	0.303
0.1900	899.41	6	15	1	16	0.297	3.0	0.157
0.2050	886.07	6	15	1	16	0.298	3.0	0.125
0.7000	752.84	6	15	0	15	0.299	1.0	0.032
0.8000	743.89	6	15	0	15	0.300	0.0	0.0

Table 3: Characteritics of `TpT` (`minsplit`: 50) depending on the time penalty

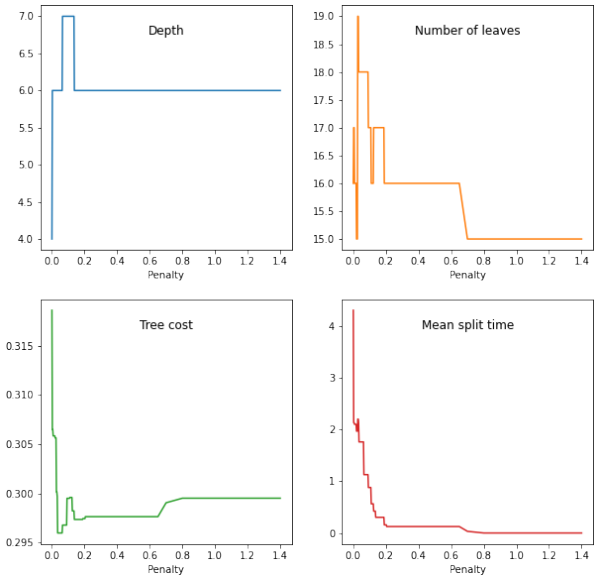


Figure 12: Characteritics of `TpT` (`minsplit`: 50) depending on the time penalty

In this table are a few remarkable `TpT`. First of all, we also see here that the trees with $\gamma = 0$ and $\gamma \to \infty$ are not the best in terms of global cost. This is a critical result: $\gamma = 0$ is the case where the last observed observation points are quickly considered whereas early periods are not really considered, and high γ represents the case where a tree is grown only one the baseline values of all time-varying covariates. Thus, `TpT` shows that considering time in the splitting process improves the global purity of the tree, it better differentiates between individuals with different outcomes and trajectories. In terms of interpretability, Figure 13 shows that the optimal `TpT`

is a compromise between small TpTs with time-distant splits and a large baseline tree without any temporal information. Whole-page versions of those trees can be found in Appendix A.3.3.

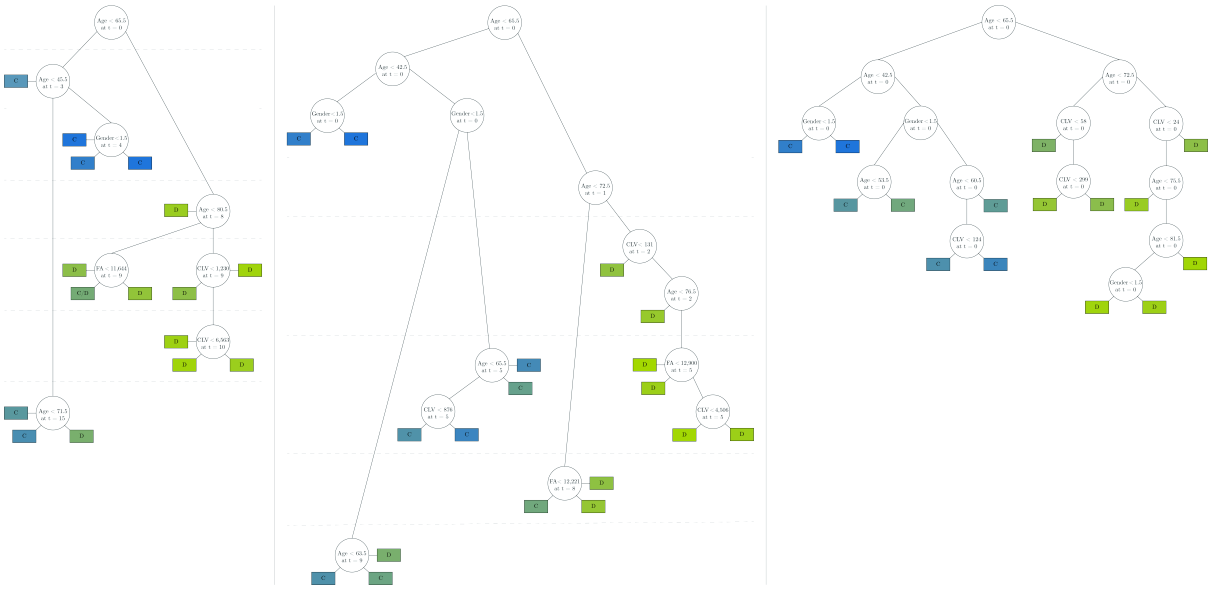


Figure 13: TpTs with $\gamma = 0$, $\gamma = 0.035$ and $\gamma \to \infty$, respectively

An important temporal dependence that can be learned from the tree is the fact that there exists an incentive not to lapse before eight years of seniority. It is clearly depicted in the optimal TpT - $\gamma = 0.035$ - as the duration leaves generated by splits occurring at times ≥ 8 contain a majority of policyholders that did not lapse. It means that regardless of their age, subjects with a seniority ≤ 8 years do not lapse. The TpT with no time penalty - $\gamma = 0$ - can capture the same temporal dependence for splits that occur immediately after 8 years for older PH but fails to do so for younger ones. This is explained by the fact that for the latter, the unpenalized TpT quickly finds an excellent split at time $t = 15$, which prevents splits around 8 years from being found. This is a compelling argument in favor of a time penalty. Furthermore, the TpT with a very high time penalty produces a tree that only splits at time $t = 0$, thus no temporal insights can be found with it. If we were to conclude from such a tree, we could say that Age is the most important covariate, and allow for a good partitionning of $\mathcal{D}$ but we cannot have any temporal analysis. This is an argument in favor of TpT and the suggested γ selection process.

4.3 Pathways vizualizations

In terms of data mining and clustering, let us focus on the optimal TpT obtained in the previous section and depicted in Figure 14. In the same way a decision tree is a representation of all observations in a cross-sectional dataset, a TpT is a complete representation of a longitudinal one and we can highlight the pathway of any given policyholder in the tree. Unlike any other longitudinal tree-based model, any individual has a unique continuous trajectory in the tree. The pathways of five policyholders selected at random from our dataset are represented in the following TpT.

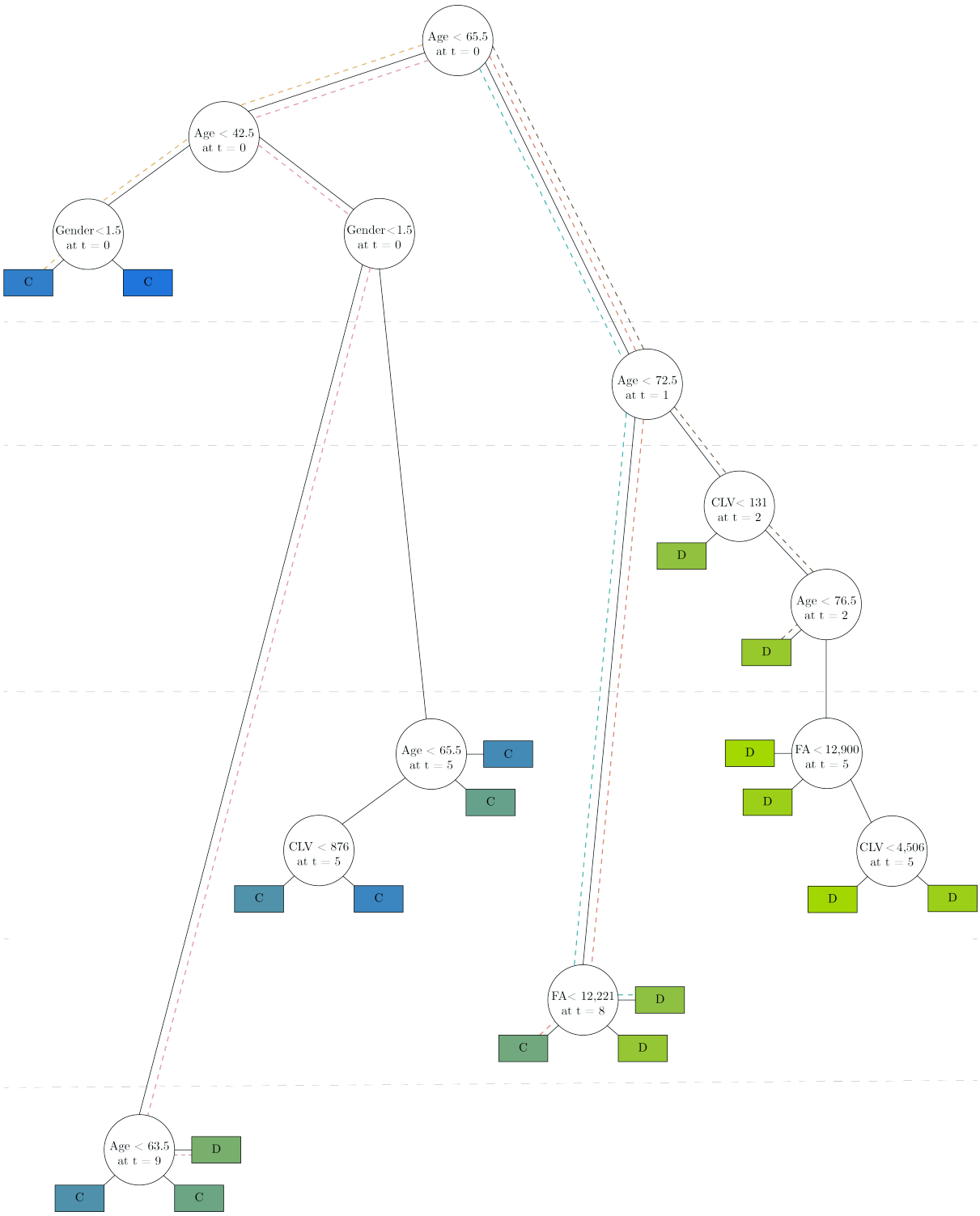


Figure 14: Individual longitudinal trajectories in the optimal `TpT` (`minsplit` = 50)

Thus, the longitudinal dataset and all individual timelines can be easily represented as a partitionning. All PH are represented on the y-axis and the region of the covariate space where they belong changes as a function of time, on the x-axis. In this example, the 18 leaves of Figure 14 correspond to the 18 final regions of Figure 15, as $t \to \infty$.

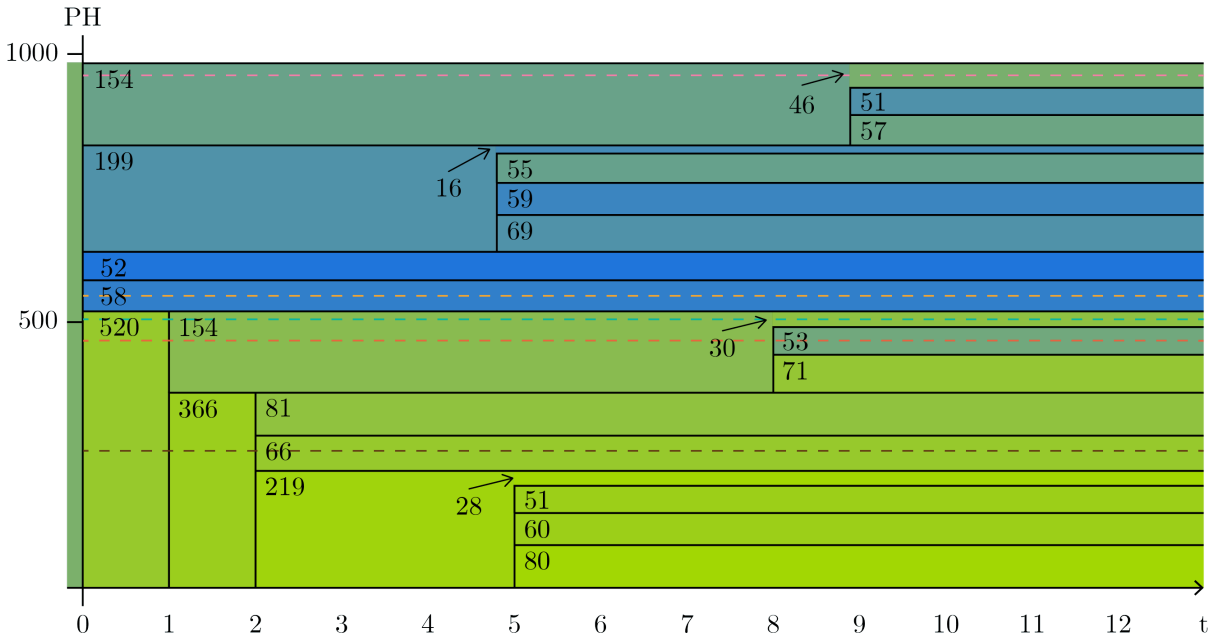


Figure 15: Global timeline and individual longitudinal trajectories

The numbers in each regions, as well as their hights is the number of PH they contained, and the five individual trajectories represented as pathways in the tree correspond to the five horizontal lines within the global timeline. This type of visualization allow to better interpret the periods where changes in the outcome can occur.

5 Conclusion, limitations and future work

5.1 Conclusion

This paper exhibits `TpT`, a new tree-based data-mining algorithm that accounts for time-varying covariates through time-penalized splitting criteria. As it is the first article on the subject, it is meant to be a mere introduction of the algorithm with few applications on real data. Our methods handle time-varying covariates as well as a longitudinal target variable inherently. Contrary to existing approaches, it does not need workaround strategies such as the pseudo-subject method and provides a tree that separates "complete individuals", as each subject covariates trajectory corresponds to a single unique trajectory in the final tree. Pruning strategies were proposed and tested with real datasets and illustrative examples. The algorithm proves to have appealing data-mining and visualization potential in various fields that could be explored more deeply in the future.

5.2 Limitations and future work

For data-mining purposes and with the algorithm as it is defined, many improvement paths can be considered.

First of all, the introduction of a penalized splitting criterion, and thus a penalty parameter could be discussed more thoroughly. The current multiplicative exponential form of penalization has been duly justified but one could explore the effects of different approaches. Other distributions of the future time cut-off penalties such as Gamma (with parameters $\alpha < 1, \beta \geq 1$ or $\alpha = 1, \beta > 2$), Pareto, Weibull (with parameter $k < 1$) or Log-logistic ($\beta \leq 1$) could be justified on concrete examples. Moreover, in the algorithm as it stands, every point in time can be considered for a potential cut-off; some time-horizon limit where distant splits would simply be ignored would have an impact on the shape of the final tree. Eventually, an additive time penalty similar to a Lasso or Ridge regression penalty term is yet to be explored. In all those scenarios, the penalty parameter affects the width and length of the final tree and can even be interpreted as a pre-pruning parameter. The properties of that pre-pruning as well as the choice of an optimal γ are yet to be discussed. On a final note, we do not know if any technical properties (see Breiman (1996); Buntine and Niblett (1992)) of the penalized splitting criterion still hold. That knowledge will certainly not affect the concrete applications of `TpT` but is more of a theoretical interest.

Secondly, we showed in illustrative applications that time-outliers can be easily miscategorized as the `TpT` can send them early in one direction of the tree from which they will not escape. In addition to that, those observations are likely to end up being isolated in a leaf if the stopping rules allow it. First of all, it is not clear if this constitutes a flaw of `TpT`. On the one hand, it forces observation into an early path that may not be consistent with later observations. On the other hand, this behavior is linked to an abrupt change of the covariates and target variable trajectories in time, which is a discriminating feature that can justify that such subjects end up in a specific leaf. We see two ways to handle this specific property:

- A first idea would be to modify the `TpT` algorithm to make it less greedy. Instead of choosing the best split at each node, we could consider finding the best sequence of several consecutive splits. This *N-step ahead* strategy would ensure that abrupt changes in covariates in the future are anticipated in early splits. In cases where the penalty

parameter is low, this approach also ensures that the `TpT` does not grow too fast with time.

- Another innovative solution is to introduce the possibility for an outlier in a node to teleport into another one, at a similar depth/split time in the tree. For instance, if it so happens that a subject trajectory suddenly becomes significantly different from the other ones in the same node, it can be clever to acknowledge that it is no longer consistent to keep it in the said node. This solution has drawbacks: it requires testing for outliers in every node, at every step. If one is found, it can only be teleported if another node within which the subject would not be an outlier is found. Moreover, it is a straightforward solution for data mining but other adaptations are necessary if `TpT` is to be used for predictive tasks. Despite this, it would still ensure individual trajectories for every subject in the tree and it would consolidate the global within-node homogeneity.

Then, our last point raises another capital question: the applications shown in Section 4 only exhibit the potential of `TpT` in a data mining setting; can it be adapted to prediction tasks? In our context, a prediction is an estimation of a subject's target variable $y^{(i)}$ at a time t, given its covariate history up to t. An obvious research path in that direction is to mimic the example of CART. For a subject in node g, predicting the mean of the target variable at time t of all subjects emerging from node g is to be tried. If the target variable is a time-to-event and the observations are censored, it could be weighted by the inverse probabilities of censoring weights (IPCW). It perfectly translates in terms of interpretability: the prediction of an outcome at time t for individual i is the mean of the outcomes at time t of all subjects taking the same trajectory in the tree. There are several obvious drawbacks to this approach: there needs to exist observations of other subjects at time t. And even if some exist, the variance of the prediction is directly linked to the number of such subjects. Exploring the properties and predictive performance of this approach is left as future work and other methods such as fitting a longitudinal model[6] at every node, not for splitting but for prediction purposes are also studied.

Eventually, if prediction is made possible in the future, exploring the performance of ensemble methods for `TpT` looks like a reasonable next step. Such approaches are in contradiction with the research of interpretability that motivated `TpT`, but competitive predictive performance could justify them.

In conclusion, we are sincerely aware of all limitations and future theoretical and applied work to be done on the subject and simply wanted to rigorously lay the foundations of `TpT` to allow forthcoming studies to start from an existing base. This paper can also prove to be useful if any collaboration and improvements on the subject - whether is it on a more efficient implementation, different applied examples, or new theoretical insights - are to result from it.

Acknowledgments and competing interests

Work(s) conducted within the Research Chair DIALog under the aegis of the Risk Foundation, an initiative by CNP Assurances.

[6] A Mixed effect model in regression or a Joint model in survival setting for instance

References

M. Abdolell, M. LeBlanc, D. Stephens, and R.V. Harrison. Binary partitioning for continuous longitudinal data: categorizing a prognostic variable. *Statistics in Medicine*, 21:3395–3409, 2002.

Andrew R. Barron. *Complexity Regularization with Application to Artificial Neural Networks*, pages 561–576. Springer Netherlands, Dordrecht, 1991. ISBN 978-94-011-3222-0. doi: 10.1007/978-94-011-3222-0_42. URL https://doi.org/10.1007/978-94-011-3222-0_42.

Gilles Blanchard, Christin Schäfer, and Yves Rozenholc. Oracle bounds and exact algorithm for dyadic classification trees. In John Shawe-Taylor and Yoram Singer, editors, *Learning Theory*, pages 378–392, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg. ISBN 978-3-540-27819-1.

L. Breiman, J. Friedman, C.J. Stone, and R.A. Olshen. *Classification and Regression Trees*. Taylor & Francis, 1984. ISBN 9780412048418. URL https://books.google.fr/books?id=JwQx-WOmSyQC.

Leo Breiman. Technical note: Some properties of splitting criteria. *Machine Learning*, 24(1):41–47, 1996.

Alexandra P. Bremner. *Localised splitting criteria for classification and regression trees*. PhD thesis, Murdoch University, 2004.

R.G. Brown. *Exponential Smoothing for Predicting Demand*. Little, 1956. URL https://books.google.fr/books?id=Eo_rMgEACAAJ.

Wray Buntine and Tim Niblett. A further comparison of splitting rules for decision-tree induction. *Machine Learning*, 8(1):75–85, 1992.

Louis Capitaine, Jérémie Bigot, Rodolphe Thiébaut, and Robin Genuer. Fréchet random forests for metric space valued regression with non euclidean predictors, 2020.

Louis Capitaine, Robin Genuer, and Rodolphe Thiébaut. Random forests for high-dimensional longitudinal data. *Statistical Methods in Medical Research*, 30(1):166–184, 2021. doi: 10.1177/0962280220946080. URL https://doi.org/10.1177/0962280220946080. PMID: 32772626.

G. De'Ath. Multivariate regression trees: a new technique for modeling species-environment relationships. *Ecology*, 83:1105–1117, 2002.

G. De'Ath. mvpart: multivariate partitioning. r package version 1.2-4, 2006.

Chris Drummond and Robert C. Holte. Exploiting the cost (in) sensitivity of decision tree splitting criteria. In *ICML*, pages 239–246, 2000.

Soo-Heang Eo and HyungJun Cho. Tree-structured mixed-effects regression modeling for longitudinal data. *Journal of Computational and Graphical Statistics*, 23(3):740–760, 2014. doi: 10.1080/10618600.2013.794732. URL https://doi.org/10.1080/10618600.2013.794732.

F. Esposito, D. Malerba, G. Semeraro, and J. Kay. A comparative analysis of methods for pruning decision trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):476–491, 1997. doi: 10.1109/34.589207.

Wei Fu and Jeffrey S. Simonoff. Unbiased regression trees for longitudinal and clustered data. *Computational Statistics & Data Analysis*, 88:53–74, 2015. ISSN 0167-9473. doi: https://doi.org/10.1016/j.csda.2015.02.004. URL https://www.sciencedirect.com/science/article/pii/S0167947315000432.

Wei Fu and Jeffrey S. Simonoff. Survival trees for left-truncated and right-censored data, with application to time-varying covariate data. *Biostatistics*, 18(2):352–369, 12 2016. ISSN 1465-4644. doi: 10.1093/biostatistics/kxw047. URL https://doi.org/10.1093/biostatistics/kxw047.

Alex Lauf Goldstein. *Topics in Tree-Based Methods*. Doctoral dissertation, University of Pennsylvania, 2014. URL https://repository.upenn.edu/dissertations/AAI3622048/.

Ahlem Hajjem, François Bellavance, and Denis Larocque. Mixed effects regression trees for clustered data. *Statistics & Probability Letters*, 81(4):451–459, 2011a. ISSN 0167-7152. doi: https://doi.org/10.1016/j.spl.2010.12.003. URL https://www.sciencedirect.com/science/article/pii/S0167715210003433.

Ahlem Hajjem, François Bellavance, and Denis Larocque. Mixed effects regression trees for clustered data. *Statistics & Probability Letters*, 81(4):451–459, 2011b. ISSN 0167-7152. doi: https://doi.org/10.1016/j.spl.2010.12.003. URL https://www.sciencedirect.com/science/article/pii/S0167715210003433.

Ahlem Hajjem, François Bellavance, and Denis Larocque. Mixed-effects random forest for clustered data. *Journal of Statistical Computation and Simulation*, 84(6):1313–1328, 2014. doi: 10.1080/00949655.2012.741599. URL https://doi.org/10.1080/00949655.2012.741599.

Charles C. Holt. Forecasting seasonals and trends by exponentially weighted moving averages. *International Journal of Forecasting*, 20(1):5–10, 2004. URL https://EconPapers.repec.org/RePEc:eee:intfor:v:20:y:2004:i:1:p:5-10.

W.-C. Hsiao and Y.-S. Shih. Splitting variable selection for multivariate regression trees. *Statistics and Probability Letters*, 77:265–271, 2007.

Madan Gopal Kundu and Jaroslaw Harezlak. Regression trees for longitudinal data with baseline covariates. *Biostatistics & Epidemiology*, 3(1):1–22, 2019. doi: 10.1080/24709360.2018.1557797. URL https://doi.org/10.1080/24709360.2018.1557797.

D.R. Larsen and P.L. Speckman. Multivariate regression trees for analysis of abundance data. *Biometrics*, 60:543–549, 2004.

S.K. Lee. On generalized multivariate decision tree by using gee. *Computational Statistics & Data Analysis*, 49:1105–1119, 2005.

S.K. Lee. On classification and regression trees for multiple responses and its application. *Journal of Classification*, 23:123–141, 2006.

S.K. Lee, H.-C. Kang, S.-T. Han, and K.-H. Kim. Using generalized estimating equations to learn decision trees with multivariate responses. *Data Mining and Knowledge Discovery*, 11:273–293, 2005.

Yishay Mansour and David McAllester. Generalization bounds for decision trees. In *Proc. 13th Annu. Conference on Comput. Learning Theory*, pages 69–80. Morgan Kaufmann, San Francisco, 2000.

Gary Mena, Kristof Coussement, Koen W. de Bock, Arno de Caigny, and Stefan Lessmann. Exploiting time-varying RFM measures for customer churn prediction with deep neural networks. *Annals of Operations Research*, 2023. URL https://hal.science/hal-04027550.

John Mingers. An empirical comparison of selection measures for decision-tree induction. *Machine learning*, 3(4):319–342, 1989.

Hoora Moradian, Weichi Yao, Denis Larocque, Jeffrey S. Simonoff, and Halina Frydman. Dynamic estimation with random forests for discrete-time survival data. *Canadian Journal of Statistics*, 2021.

A.B. Nobel. Analysis of a complexity-based pruning scheme for classification trees. *IEEE Transactions on Information Theory*, 48(8):2362–2368, 2002. doi: 10.1109/TIT.2002.800482.

J. Ross Quinlan. Simplifying decision trees. *Int. J. Man Mach. Stud.*, 27:221–234, 1987.

G. Ritschard and M. Oris. Life course data in demography and social sciences: statistical and data mining approaches. In R. Levy, P. Ghisletta, J.-M.Le Goff, D. Spini, and E. Widmer, editors, *Towards an interdisciplinary perspective on the life course, advances in life course research*, page 289–320. Elsevier, Amsterdam, 2005.

D. Rizopoulos. *Joint Models for Longitudinal and Time-to-Event Data, with Applications in R*. Chapman & Hall/CRC, Boca Raton, 2012.

C. Scott and R.D. Nowak. Minimax-optimal classification with dyadic decision trees. *IEEE Transactions on Information Theory*, 52(4):1335–1353, 2006. doi: 10.1109/TIT.2006.871056.

Clayton D. Scott and Robert D. Nowak. Dyadic classification trees via structural risk minimization. In *NIPS*, 2002. URL https://api.semanticscholar.org/CorpusID:14846592.

M.R. Segal. Tree-structured models for longitudinal data. *Journal of the American Statistical Association*, 87:407–418, 1992.

Rebecca Sela and Jeffrey Simonoff. Re-em trees: A data mining approach for longitudinal and clustered data. *Machine Learning*, 86:169–207, 02 2012. doi: 10.1007/s10994-011-5258-3.

Yu-Shan Shih. Families of splitting criteria for classification trees. *Statistics and Computing*, 9(4): 309–315, 1999.

David M. Vock, Julian Wolfson, Sunayan Bandyopadhyay, Gediminas Adomavicius, Paul E. Johnson, Gabriela Vazquez-Benitez, and Patrick J. O'Connor. Adapting machine learning techniques to censored time-to-event health record data: A general-purpose approach using inverse probability of censoring weighting. *Journal of Biomedical Informatics*, 61:119–131, 2016. ISSN 1532-0464. doi: https://doi.org/10.1016/j.jbi.2016.03.009. URL https://www.sciencedirect.com/science/article/pii/S1532046416000496.

Steven Y. K. Wong, Jennifer S. K. Chan, Lamiae Azizi, and Richard Y. D. Xu. Time-varying neural network for stock return prediction. *Intelligent Systems in Accounting, Finance and Management*, 29 (1):3–18, 2022. doi: https://doi.org/10.1002/isaf.1507. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/isaf.1507.

Weichi Yao, Halina Frydman, Denis Larocque, and Jeffrey S. Simonoff. Ensemble methods for survival

function estimation with time-varying covariates, 2020. URL https://arxiv.org/abs/2006.00567.

Weichi Yao, Halina Frydman, Denis Larocque, and Jeffrey S Simonoff. Ensemble methods for survival function estimation with time-varying covariates. *Statistical Methods in Medical Research*, 31(11):2217–2236, 2022. doi: 10.1177/09622802221111549. URL https://doi.org/10.1177/09622802221111549. PMID: 35895510.

H. Zhang. Classification trees for multiple binary responses. *Journal of the American Statistical Association*, 93:180–193, 1998.

A Appendix

A.1 About cost-complexity pruning

Even thougt the original cost function of the CART algorithm described by Breiman et al. (1984) is penalized proportionnaly to its number of leaves n_L, several works on the matter suggest other types of penalty. Barron (1991) shows that applying risk bounds to CART imply a penalty with $\psi(n_L) = \sqrt{n_L}$. In later works, Mansour and McAllester (2000), Nobel (2002), then Scott and Nowak (2002) also showed that risk bounds with a penalty using $\psi(n_L) = \sqrt{n_L}$ can be derived for classification trees whereas penalties proportionnal to n_L can only be derived in specific cases discusse by Blanchard et al. (2004). In summary, square-root penalties appear to have a much stronger theoretical foundation than n_L proportionnal ones in various context, notably for classification tasks.

A.2 Frechet trees

Another very interesting and general approach is Fréchet trees - and Fréchet forest - by Capitaine et al. (2020). It is a tree-building procedure that allow handling data for which input covariates and the outcome take values in general metric spaces. Concretely, it is designed to handle covariates and outcomes that can be any functions and can be, in particular, functions of time. In this article, they illustrate the prediction ability of Fréchet forests on longitudinal data and the robustness of their method to missing data and time shifts. Several limitations can be pointed out: firstly the mathematical assumption of the existence of the Fréchet mean in the output space must be verified and that mean must be approximated as precisely as possible. Another limitation is the interpretability, as it is always the case with bagging techniques, but here it is also true for individual Fréchet trees: if covariates' importance can be analyzed, relevant threshold and time points can not be easily observed. Eventually, the computational burden of this algorithm is also important. This method has been implemented in the R package `FrechForest`.

A.3 More results

A.3.1 Results without stopping criterion

The maximal unpruned and unstopped TpT, obtained with the time-penalized Gini splitting criterion and an optimal time penalty achieves a depth of 17, has 190 leaves - 173 terminal leaves, 17 duration leaves - and is too large to be fully displayed as a tree here. However, we can still represent it as a list of decisions describing how the dataset is partitionned:

```
The maximum depth achieved is 17
The number of leaves is 190
173 terminal leaves  and 17 duration leaves
The tree impurity is  : 0.06270710057403012
The penalized tree impurity is  : 0.3556124586066797
The maximum time where a split occured is 10.0
The average split  time is  0.9805922147055561

The tree is :
  depth = 0 if Age <= 65.5 at t = 0.0, samples: 983
 and no duration leaf
    then  depth = 1 if Age <= 42.5 at t = 0.0, samples: 463
     and no duration leaf
        then  depth = 2 if GENDER <= 1.5 at t = 0.0, samples: 110
         and no duration leaf
```

```
    then  depth = 3 if Age <= 30.5 at t = 0.0, samples: 58
     and no duration leaf
        then  depth = 4{value: CHURNED, samples: 29}
        else  depth = 4 if CLV <= 9.16 at t = 0.0, samples: 29
         and no duration leaf
            then  depth = 5{value: CHURNED, samples: 9}
            else  depth = 5 if FACE_AMOUNT <= 7197.19 at t = 2.0, samples: 20
             and no duration leaf
                then  depth = 6 if FACE_AMOUNT <= 3654.22 at t = 2.0, samples: 9
                 and no duration leaf
                    then  depth = 7 if Age <= 39.5 at t = 2.0, samples: 5
                     and no duration leaf
                        then  depth = 8{value: CHURNED, samples: 3}
                        else  depth = 8{value: CHURNED 0.5, samples: 2}
                    else  depth = 7{value: DEATH, samples: 4}
                else  depth = 6{value: CHURNED, samples: 11}
    else  depth = 3{value: CHURNED, samples: 52}
else  depth = 2 if GENDER <= 1.5 at t = 0.0, samples: 353
 and no duration leaf
    then  depth = 3 if Age <= 53.5 at t = 0.0, samples: 154
     and no duration leaf
        then  depth = 4 if Age <= 52.5 at t = 0.0, samples: 53
         and no duration leaf
            then  depth = 5 if CLV <= 13.11 at t = 0.0, samples: 47
             and no duration leaf
                then  depth = 6 if FACE_AMOUNT <= 3196.44 at t = 3.0, samples: 10
                 and duration leaf has 1 samples. Label is: CHURNED 1.0
                    then  depth = 7{value: CHURNED, samples: 7}
                    else  depth = 7{value: CHURNED 0.5, samples: 2}
                else  depth = 6 if CLV <= 88.4 at t = 0.0, samples: 37
                 and no duration leaf
                    then  depth = 7 if Nb_Contrats <= 1.5 at t = 0.0, samples: 14
                     and no duration leaf
                        then  depth = 8 if CLV <= 40.05 at t = 0.0, samples: 12
                         and no duration leaf
                            then  depth = 9 if Age <= 45.5 at t = 0.0, samples: 8
                             and no duration leaf
                                then  depth = 10{value: DEATH, samples: 3}
                                else  depth = 10 if CLV <= 17.03 at t = 0.0, samples: 5
                                 and no duration leaf
                                    then  depth = 11{value: DEATH, samples: 2}
                                    else  depth = 11{value: CHURNED, samples: 3}
                            else  depth = 9{value: DEATH, samples: 4}
                        else  depth = 8{value: CHURNED, samples: 2}
                    else  depth = 7 if CLV <= 591.46 at t = 0.0, samples: 23
                     and no duration leaf
                        then  depth = 8 if CLV <= 352.28 at t = 0.0, samples: 18
                         and no duration leaf
                            then  depth = 9 if CLV <= 155.3 at t = 0.0, samples: 16
                             and no duration leaf
                                then  depth = 10 if CLV <= 190.4 at t = 1.0, samples: 9
                                 and no duration leaf
                                    then  depth = 11{value: CHURNED 0.5, samples: 2}
                                    else  depth = 11{value: CHURNED, samples: 7}
                                else  depth = 10 if CLV <= 178.0 at t = 0.0, samples: 7
                                 and no duration leaf
                                    then  depth = 11{value: DEATH, samples: 2}
                                    else  depth = 11 if CLV <= 259.66 at t = 0.0, samples: 5
                                     and no duration leaf
                                        then  depth = 12{value: CHURNED, samples: 3}
                                        else  depth = 12{value: CHURNED 0.5, samples: 2}
                            else  depth = 9{value: DEATH, samples: 2}
                        else  depth = 8{value: CHURNED, samples: 5}
            else  depth = 5{value: CHURNED, samples: 6}
        else  depth = 4 if CDI_NOM_PRODUIT <= 1.5 at t = 0.0, samples: 101
         and no duration leaf
            then  depth = 5 if FACE_AMOUNT <= 10325.88 at t = 4.0, samples: 83
             and duration leaf has 3 samples. Label is: DEATH 1.0
                then  depth = 6 if CLV <= 16.28 at t = 4.0, samples: 41
                 and no duration leaf
                    then  depth = 7 if Age <= 60.5 at t = 6.0, samples: 8
                     and duration leaf has 1 samples. Label is: CHURNED 1.0
                        then  depth = 8{value: CHURNED 0.5, samples: 2}
                        else  depth = 8{value: DEATH, samples: 5}
                    else  depth = 7 if CLV <= 89.04 at t = 4.0, samples: 33
                     and no duration leaf
                        then  depth = 8{value: CHURNED, samples: 8}
                        else  depth = 8 if CLV <= 100.4 at t = 4.0, samples: 25
                         and no duration leaf
                            then  depth = 9{value: DEATH, samples: 2}
                            else  depth = 9 if CLV <= 181.96 at t = 4.0, samples: 23
                             and no duration leaf
                                then  depth = 10{value: CHURNED, samples: 5}
                                else  depth = 10 if CLV <= 310.27 at t = 5.0, samples: 18
                                 and no duration leaf
                                    then  depth = 11{value: DEATH, samples: 3}
                                    else  depth = 11 if Age <= 68.5 at t = 5.0, samples: 15
                                     and no duration leaf
                                        then  depth = 12 if FACE_AMOUNT <= 3972.54 at t = 5.0, samples: 11
                                         and no duration leaf
                                            then  depth = 13{value: DEATH, samples: 3}
                                            else  depth = 13 if CLV <= 748.3 at t = 6.0, samples: 8
                                             and no duration leaf
                                                then  depth = 14{value: CHURNED, samples: 4}
                                                else  depth = 14 if CLV <= 917.37 at t = 6.0, samples: 4
```

```
                                                and no duration leaf
                                                    then  depth = 15{value: DEATH, samples: 2}
                                                    else  depth = 15{value: CHURNED, samples: 2}
                                            else  depth = 12{value: CHURNED, samples: 4}
                    else  depth = 6 if FACE_AMOUNT <= 17894.43 at t = 4.0, samples: 39
                     and no duration leaf
                        then  depth = 7{value: DEATH, samples: 8}
                        else  depth = 7 if Age <= 65.5 at t = 5.0, samples: 31
                         and no duration leaf
                            then  depth = 8 if CLV <= 1745.92 at t = 5.0, samples: 17
                             and no duration leaf
                                then  depth = 9 if FACE_AMOUNT <= 21616.0 at t = 5.0, samples: 5
                                 and no duration leaf
                                    then  depth = 10{value: DEATH, samples: 3}
                                    else  depth = 10{value: CHURNED, samples: 2}
                                else  depth = 9 if CLV <= 3172.41 at t = 5.0, samples: 12
                                 and no duration leaf
                                    then  depth = 10{value: CHURNED, samples: 6}
                                    else  depth = 10 if FACE_AMOUNT <= 64766.89 at t = 6.0, samples: 6
                                     and no duration leaf
                                        then  depth = 11{value: DEATH, samples: 3}
                                        else  depth = 11{value: CHURNED, samples: 3}
                            else  depth = 8 if FACE_AMOUNT <= 20931.16 at t = 6.0, samples: 14
                             and duration leaf has 1 samples. Label is: DEATH 1.0
                                then  depth = 9{value: CHURNED, samples: 2}
                                else  depth = 9 if CLV <= 10948.21 at t = 9.0, samples: 11
                                 and duration leaf has 4 samples. Label is: DEATH 1.0
                                    then  depth = 10 if CLV <= 5539.86 at t = 9.0, samples: 4
                                     and no duration leaf
                                        then  depth = 11{value: DEATH, samples: 2}
                                        else  depth = 11{value: CHURNED, samples: 2}
                                    else  depth = 10{value: DEATH, samples: 3}
                else  depth = 5 if CLV <= 3.37 at t = 0.0, samples: 18
                 and no duration leaf
                    then  depth = 6{value: CHURNED 0.5, samples: 2}
                    else  depth = 6 if CLV <= 21.3 at t = 0.0, samples: 16
                     and no duration leaf
                        then  depth = 7{value: CHURNED, samples: 6}
                        else  depth = 7 if CLV <= 363.44 at t = 0.0, samples: 10
                         and no duration leaf
                            then  depth = 8{value: CHURNED 0.5, samples: 4}
                            else  depth = 8 if CLV <= 2068.21 at t = 0.0, samples: 6
                             and no duration leaf
                                then  depth = 9{value: CHURNED, samples: 4}
                                else  depth = 9{value: CHURNED 0.5, samples: 2}
else  depth = 3 if CDI_NOM_PRODUIT <= 1.5 at t = 0.0, samples: 199
 and no duration leaf
    then  depth = 4 if Age <= 60.5 at t = 0.0, samples: 167
     and no duration leaf
        then  depth = 5 if Age <= 43.5 at t = 0.0, samples: 115
         and no duration leaf
            then  depth = 6 if CLV <= 82.12 at t = 0.0, samples: 4
             and no duration leaf
                then  depth = 7{value: DEATH, samples: 2}
                else  depth = 7{value: CHURNED 0.5, samples: 2}
            else  depth = 6 if CLV <= 2145.92 at t = 3.0, samples: 111
             and duration leaf has 1 samples. Label is: DEATH 1.0
                then  depth = 7 if FACE_AMOUNT <= 28288.52 at t = 3.0, samples: 94
                 and no duration leaf
                    then  depth = 8 if CLV <= 910.67 at t = 3.0, samples: 92
                     and no duration leaf
                        then  depth = 9 if CLV <= 840.53 at t = 3.0, samples: 80
                         and no duration leaf
                            then  depth = 10 if FACE_AMOUNT <= 12512.22 at t = 6.0, samples: 78
                             and no duration leaf
                                then  depth = 11 if CLV <= 217.81 at t = 8.0, samples: 65
                                 and duration leaf has 4 samples. Label is: CHURNED 0.5
                                    then  depth = 12 if Age <= 66.5 at t = 8.0, samples: 24
                                     and no duration leaf
                                        then  depth = 13 if Age <= 54.5 at t = 8.0, samples: 20
                                         and no duration leaf
                                            then  depth = 14 if Age <= 55.5 at t = 10.0, samples: 7
                                             and duration leaf has 1 samples. Label is: CHURNED 1.0
                                                then  depth = 15{value: CHURNED, samples: 3}
                                                else  depth = 15{value: DEATH, samples: 3}
                                            else  depth = 14 if CLV <= 74.87 at t = 8.0, samples: 13
                                             and no duration leaf
                                                then  depth = 15{value: CHURNED, samples: 10}
                                                else  depth = 15{value: CHURNED 0.67, samples: 3}
                                        else  depth = 13 if Age <= 67.5 at t = 8.0, samples: 4
                                         and no duration leaf
                                            then  depth = 14{value: DEATH, samples: 2}
                                            else  depth = 14{value: CHURNED 0.5, samples: 2}
                                    else  depth = 12 if Age <= 53.0 at t = 8.0, samples: 37
                                     and no duration leaf
                                        then  depth = 13{value: CHURNED 0.5, samples: 2}
                                        else  depth = 13 if FACE_AMOUNT <= 3848.48 at t = 10.0, samples: 35
                                         and duration leaf has 5 samples. Label is: CHURNED 0.8
                                            then  depth = 14 if FACE_AMOUNT <= 3086.41 at t = 10.0, samples: 8
                                             and no duration leaf
                                                then  depth = 15{value: CHURNED, samples: 6}
                                                else  depth = 15{value: CHURNED 0.5, samples: 2}
                                            else  depth = 14{value: CHURNED, samples: 22}
                                else  depth = 11 if Age <= 60.5 at t = 7.0, samples: 13
                                 and duration leaf has 1 samples. Label is: DEATH 1.0
```

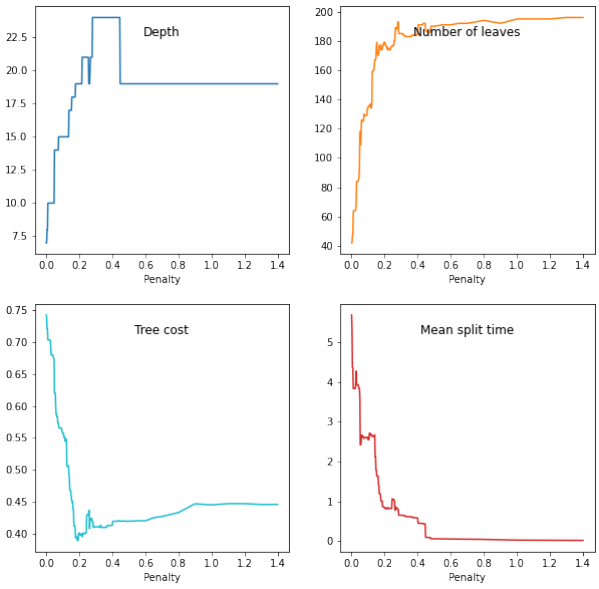
```
                                                then  depth = 12 if FACE_AMOUNT <= 16037.28 at t = 8.0, samples: 8
                                                 and no duration leaf
                                                    then  depth = 13{value: DEATH, samples: 3}
                                                    else  depth = 13 if Age <= 57.5 at t = 8.0, samples: 5
                                                     and no duration leaf
                                                        then  depth = 14{value: DEATH 0.67, samples: 3}
                                                        else  depth = 14{value: CHURNED, samples: 2}
                                                else  depth = 12{value: CHURNED, samples: 4}
                                            else  depth = 10{value: DEATH, samples: 2}
                                        else  depth = 9{value: CHURNED, samples: 12}
                                    else  depth = 8{value: DEATH, samples: 2}
                                else  depth = 7{value: CHURNED, samples: 16}
                            else  depth = 5 if Age <= 64.5 at t = 0.0, samples: 52
                             and no duration leaf
                                then  depth = 6 if CLV <= 251.48 at t = 0.0, samples: 43
                                 and no duration leaf
                                    then  depth = 7 if FACE_AMOUNT <= 73.34 at t = 3.0, samples: 29
                                     and no duration leaf
                                        then  depth = 8{value: DEATH, samples: 3}
                                        else  depth = 8 if CLV <= 119.45 at t = 3.0, samples: 26
                                         and no duration leaf
                                            then  depth = 9 if CLV <= 54.46 at t = 4.0, samples: 11
                                             and no duration leaf
                                                then  depth = 10{value: CHURNED 0.67, samples: 3}
                                                else  depth = 10{value: CHURNED, samples: 8}
                                            else  depth = 9 if FACE_AMOUNT <= 3331.54 at t = 3.0, samples: 15
                                             and no duration leaf
                                                then  depth = 10{value: DEATH, samples: 2}
                                                else  depth = 10 if CLV <= 445.55 at t = 3.0, samples: 13
                                                 and no duration leaf
                                                    then  depth = 11{value: CHURNED, samples: 3}
                                                    else  depth = 11 if CLV <= 709.34 at t = 3.0, samples: 10
                                                     and no duration leaf
                                                        then  depth = 12 if CLV <= 622.04 at t = 3.0, samples: 4
                                                         and no duration leaf
                                                            then  depth = 13{value: CHURNED 0.5, samples: 2}
                                                            else  depth = 13{value: DEATH, samples: 2}
                                                        else  depth = 12 if FACE_AMOUNT <= 16863.76 at t = 3.0, samples: 6
                                                         and no duration leaf
                                                            then  depth = 13{value: CHURNED, samples: 3}
                                                            else  depth = 13{value: DEATH 0.67, samples: 3}
                                    else  depth = 7 if Nb_Contrats <= 1.5 at t = 0.0, samples: 14
                                     and no duration leaf
                                        then  depth = 8 if FACE_AMOUNT <= 22229.94 at t = 1.0, samples: 11
                                         and no duration leaf
                                            then  depth = 9{value: DEATH 0.67, samples: 3}
                                            else  depth = 9{value: DEATH, samples: 8}
                                        else  depth = 8{value: CHURNED, samples: 3}
                                else  depth = 6 if CLV <= 633.18 at t = 0.0, samples: 9
                                 and no duration leaf
                                    then  depth = 7{value: CHURNED, samples: 7}
                                    else  depth = 7{value: CHURNED 0.5, samples: 2}
                            else  depth = 4 if CLV <= 2081.39 at t = 0.0, samples: 32
                             and no duration leaf
                                then  depth = 5 if Age <= 63.5 at t = 0.0, samples: 30
                                 and no duration leaf
                                    then  depth = 6 if FACE_AMOUNT <= 37433.94 at t = 3.0, samples: 26
                                     and duration leaf has 2 samples. Label is: CHURNED 1.0
                                        then  depth = 7{value: CHURNED, samples: 21}
                                        else  depth = 7{value: CHURNED 0.67, samples: 3}
                                    else  depth = 6 if Age <= 64.5 at t = 0.0, samples: 4
                                     and no duration leaf
                                        then  depth = 7{value: CHURNED 0.5, samples: 2}
                                        else  depth = 7{value: CHURNED, samples: 2}
                                else  depth = 5{value: CHURNED 0.5, samples: 2}
    else  depth = 1 if Age <= 72.5 at t = 0.0, samples: 520
     and no duration leaf
        then  depth = 2 if CLV <= 2.73 at t = 0.0, samples: 180
         and no duration leaf
            then  depth = 3 if CLV <= 99.7 at t = 2.0, samples: 13
             and no duration leaf
                then  depth = 4 if CLV <= 11.97 at t = 3.0, samples: 11
                 and no duration leaf
                    then  depth = 5{value: CHURNED, samples: 9}
                    else  depth = 5{value: CHURNED 0.5, samples: 2}
                else  depth = 4{value: DEATH, samples: 2}
            else  depth = 3 if CLV <= 2982.24 at t = 0.0, samples: 167
             and no duration leaf
                then  depth = 4 if Nb_Contrats <= 2.5 at t = 0.0, samples: 165
                 and no duration leaf
                    then  depth = 5 if CDI_NOM_PRODUIT <= 1.5 at t = 0.0, samples: 159
                     and no duration leaf
                        then  depth = 6 if CLV <= 153.51 at t = 0.0, samples: 146
                         and no duration leaf
                            then  depth = 7 if Nb_Contrats <= 1.5 at t = 1.0, samples: 61
                             and no duration leaf
                                then  depth = 8 if Age <= 70.5 at t = 1.0, samples: 58
                                 and no duration leaf
                                    then  depth = 9 if GENDER <= 1.5 at t = 1.0, samples: 30
                                     and no duration leaf
                                        then  depth = 10 if CLV <= 152.49 at t = 1.0, samples: 17
                                         and no duration leaf
                                            then  depth = 11{value: DEATH, samples: 10}
                                            else  depth = 11 if CLV <= 212.33 at t = 1.0, samples: 7
                                             and no duration leaf
```

```
                                then  depth = 12{value: CHURNED 0.5, samples: 2}
                                else  depth = 12{value: DEATH, samples: 5}
                            else  depth = 10 if FACE_AMOUNT <= 3475.12 at t = 1.0, samples: 13
                             and no duration leaf
                                then  depth = 11 if FACE_AMOUNT <= 1499.92 at t = 1.0, samples: 7
                                 and no duration leaf
                                    then  depth = 12{value: DEATH, samples: 2}
                                    else  depth = 12 if Age <= 69.5 at t = 1.0, samples: 5
                                     and no duration leaf
                                        then  depth = 13{value: CHURNED, samples: 3}
                                        else  depth = 13{value: CHURNED 0.5, samples: 2}
                                else  depth = 11 if CLV <= 195.51 at t = 1.0, samples: 6
                                 and no duration leaf
                                    then  depth = 12{value: DEATH, samples: 4}
                                    else  depth = 12{value: CHURNED 0.5, samples: 2}
                        else  depth = 9 if FACE_AMOUNT <= 2307.93 at t = 1.0, samples: 28
                         and no duration leaf
                            then  depth = 10 if CLV <= 35.73 at t = 1.0, samples: 6
                             and no duration leaf
                                then  depth = 11{value: DEATH, samples: 4}
                                else  depth = 11{value: CHURNED 0.5, samples: 2}
                            else  depth = 10{value: DEATH, samples: 22}
                    else  depth = 8{value: CHURNED 0.67, samples: 3}
                else  depth = 7 if CLV <= 161.62 at t = 0.0, samples: 85
                 and no duration leaf
                    then  depth = 8{value: CHURNED, samples: 2}
                    else  depth = 8 if CLV <= 1185.78 at t = 0.0, samples: 83
                     and no duration leaf
                        then  depth = 9 if CLV <= 1072.6 at t = 0.0, samples: 69
                         and no duration leaf
                            then  depth = 10 if Nb_Contrats <= 1.5 at t = 0.0, samples: 67
                             and no duration leaf
                                then  depth = 11 if CLV <= 296.66 at t = 0.0, samples: 61
                                 and no duration leaf
                                    then  depth = 12 if CLV <= 240.56 at t = 0.0, samples: 22
                                     and no duration leaf
                                        then  depth = 13 if CLV <= 396.72 at t = 1.0, samples: 14
                                         and no duration leaf
                                            then  depth = 14 if CLV <= 342.37 at t = 1.0, samples: 7
                                             and no duration leaf
                                                then  depth = 15{value: CHURNED 0.5, samples: 2}
                                                else  depth = 15{value: DEATH, samples: 5}
                                            else  depth = 14 if CLV <= 428.3 at t = 1.0, samples: 7
                                             and no duration leaf
                                                then  depth = 15{value: CHURNED, samples: 3}
                                                else  depth = 15 if GENDER <= 1.5 at t = 1.0, samples: 4
                                                 and no duration leaf
                                                    then  depth = 16{value: DEATH, samples: 2}
                                                    else  depth = 16{value: CHURNED 0.5, samples: 2}
                                        else  depth = 13{value: DEATH, samples: 8}
                                    else  depth = 12 if CLV <= 522.24 at t = 0.0, samples: 39
                                     and no duration leaf
                                        then  depth = 13 if CLV <= 388.65 at t = 0.0, samples: 24
                                         and no duration leaf
                                            then  depth = 14 if Age <= 70.5 at t = 0.0, samples: 12
                                             and no duration leaf
                                                then  depth = 15 if CLV <= 308.23 at t = 0.0, samples: 8
                                                 and no duration leaf
                                                    then  depth = 16{value: CHURNED 0.5, samples: 2}
                                                    else  depth = 16{value: DEATH, samples: 6}
                                                else  depth = 15 if CLV <= 320.33 at t = 0.0, samples: 4
                                                 and no duration leaf
                                                    then  depth = 16{value: CHURNED, samples: 2}
                                                    else  depth = 16{value: CHURNED 0.5, samples: 2}
                                            else  depth = 14 if CLV <= 427.18 at t = 0.0, samples: 12
                                             and no duration leaf
                                                then  depth = 15{value: CHURNED, samples: 4}
                                                else  depth = 15 if CLV <= 507.19 at t = 0.0, samples: 8
                                                 and no duration leaf
                                                    then  depth = 16 if GENDER <= 1.5 at t = 0.0, samples: 6
                                                     and no duration leaf
                                                        then  depth = 17{value: DEATH, samples: 3}
                                                        else  depth = 17{value: CHURNED 0.67, samples: 3}
                                                    else  depth = 16{value: CHURNED, samples: 2}
                                        else  depth = 13 if CLV <= 735.13 at t = 0.0, samples: 15
                                         and no duration leaf
                                            then  depth = 14{value: DEATH, samples: 8}
                                            else  depth = 14 if CLV <= 767.92 at t = 0.0, samples: 7
                                             and no duration leaf
                                                then  depth = 15{value: CHURNED, samples: 2}
                                                else  depth = 15 if FACE_AMOUNT <= 47527.88 at t = 1.0, samples: 5
                                                 and no duration leaf
                                                    then  depth = 16{value: DEATH, samples: 3}
                                                    else  depth = 16{value: CHURNED 0.5, samples: 2}
                                else  depth = 11{value: DEATH, samples: 6}
                            else  depth = 10{value: CHURNED, samples: 2}
                    else  depth = 9 if Age <= 71.0 at t = 0.0, samples: 14
                     and no duration leaf
                        then  depth = 10{value: DEATH, samples: 11}
                        else  depth = 10{value: DEATH 0.67, samples: 3}
else  depth = 6 if FACE_AMOUNT <= 7905.44 at t = 2.0, samples: 13
 and no duration leaf
    then  depth = 7 if FACE_AMOUNT <= 1385.41 at t = 3.0, samples: 8
     and duration leaf has 2 samples. Label is: CHURNED 1.0
        then  depth = 8{value: DEATH, samples: 2}
```

```
                        else  depth = 8{value: CHURNED, samples: 4}
                    else  depth = 7{value: DEATH, samples: 5}
                else  depth = 5 if Nb_Contrats <= 4.5 at t = 1.0, samples: 6
                 and no duration leaf
                    then  depth = 6{value: CHURNED, samples: 4}
                    else  depth = 6{value: DEATH, samples: 2}
            else  depth = 4{value: CHURNED, samples: 2}
    else  depth = 2 if CLV <= 24.19 at t = 0.0, samples: 340
     and no duration leaf
        then  depth = 3 if CLV <= 23.77 at t = 0.0, samples: 70
         and no duration leaf
            then  depth = 4 if Age <= 81.5 at t = 0.0, samples: 68
             and no duration leaf
                then  depth = 5 if Age <= 76.5 at t = 0.0, samples: 53
                 and no duration leaf
                    then  depth = 6 if CDI_NOM_PRODUIT <= 1.5 at t = 0.0, samples: 32
                     and no duration leaf
                        then  depth = 7 if CLV <= 1.72 at t = 0.0, samples: 24
                         and no duration leaf
                            then  depth = 8 if CLV <= 2.86 at t = 1.0, samples: 7
                             and no duration leaf
                                then  depth = 9{value: DEATH, samples: 3}
                                else  depth = 9{value: CHURNED 0.5, samples: 4}
                            else  depth = 8{value: DEATH, samples: 17}
                        else  depth = 7 if GENDER <= 1.5 at t = 4.0, samples: 8
                         and duration leaf has 2 samples. Label is: CHURNED 0.5
                            then  depth = 8{value: CHURNED, samples: 2}
                            else  depth = 8 if CLV <= 56.71 at t = 4.0, samples: 4
                             and no duration leaf
                                then  depth = 9{value: CHURNED 0.5, samples: 2}
                                else  depth = 9{value: DEATH, samples: 2}
                    else  depth = 6 if CLV <= 1.5 at t = 0.0, samples: 21
                     and no duration leaf
                        then  depth = 7{value: CHURNED, samples: 3}
                        else  depth = 7 if CLV <= 101.49 at t = 3.0, samples: 18
                         and duration leaf has 1 samples. Label is: DEATH 1.0
                            then  depth = 8 if Age <= 79.5 at t = 3.0, samples: 13
                             and no duration leaf
                                then  depth = 9{value: CHURNED, samples: 2}
                                else  depth = 9 if GENDER <= 1.5 at t = 3.0, samples: 11
                                 and no duration leaf
                                    then  depth = 10 if CLV <= 24.93 at t = 3.0, samples: 4
                                     and no duration leaf
                                        then  depth = 11{value: DEATH, samples: 2}
                                        else  depth = 11{value: CHURNED, samples: 2}
                                    else  depth = 10 if Age <= 80.5 at t = 3.0, samples: 7
                                     and no duration leaf
                                        then  depth = 11{value: DEATH 0.67, samples: 3}
                                        else  depth = 11{value: DEATH, samples: 4}
                            else  depth = 8{value: CHURNED, samples: 4}
                else  depth = 5{value: DEATH, samples: 15}
            else  depth = 4{value: CHURNED, samples: 2}
    else  depth = 3 if CDI_NOM_PRODUIT <= 1.5 at t = 0.0, samples: 270
     and no duration leaf
        then  depth = 4 if Age <= 74.5 at t = 0.0, samples: 240
         and no duration leaf
            then  depth = 5 if CLV <= 303.99 at t = 0.0, samples: 41
             and no duration leaf
                then  depth = 6 if Age <= 73.5 at t = 0.0, samples: 23
                 and no duration leaf
                    then  depth = 7 if CLV <= 391.43 at t = 2.0, samples: 6
                     and no duration leaf
                        then  depth = 8{value: CHURNED 0.5, samples: 2}
                        else  depth = 8{value: DEATH, samples: 4}
                    else  depth = 7{value: DEATH, samples: 17}
                else  depth = 6 if CLV <= 334.97 at t = 0.0, samples: 18
                 and no duration leaf
                    then  depth = 7{value: CHURNED, samples: 3}
                    else  depth = 7 if CLV <= 1380.47 at t = 0.0, samples: 15
                     and no duration leaf
                        then  depth = 8{value: DEATH, samples: 13}
                        else  depth = 8{value: CHURNED 0.5, samples: 2}
            else  depth = 5 if Age <= 89.5 at t = 0.0, samples: 199
             and no duration leaf
                then  depth = 6 if FACE_AMOUNT <= 65229.84 at t = 3.0, samples: 192
                 and duration leaf has 2 samples. Label is: DEATH 1.0
                    then  depth = 7 if FACE_AMOUNT <= 5858.16 at t = 3.0, samples: 160
                     and no duration leaf
                        then  depth = 8 if FACE_AMOUNT <= 5693.4 at t = 4.0, samples: 27
                         and duration leaf has 3 samples. Label is: DEATH 1.0
                            then  depth = 9{value: DEATH, samples: 22}
                            else  depth = 9{value: CHURNED, samples: 2}
                        else  depth = 8 if Age <= 78.5 at t = 3.0, samples: 133
                         and no duration leaf
                            then  depth = 9 if FACE_AMOUNT <= 14620.96 at t = 3.0, samples: 13
                             and no duration leaf
                                then  depth = 10 if CLV <= 743.21 at t = 3.0, samples: 5
                                 and no duration leaf
                                    then  depth = 11{value: DEATH, samples: 3}
                                    else  depth = 11{value: CHURNED 0.5, samples: 2}
                                else  depth = 10{value: DEATH, samples: 8}
                            else  depth = 9 if Age <= 81.5 at t = 3.0, samples: 120
                             and no duration leaf
                                then  depth = 10 if Age <= 80.5 at t = 3.0, samples: 49
                                 and no duration leaf
```

```
                                    then  depth = 11{value: DEATH, samples: 33}
                                    else  depth = 11 if CLV <= 1337.19 at t = 3.0, samples: 16
                                     and no duration leaf
                                        then  depth = 12 if CLV <= 1236.12 at t = 3.0, samples: 7
                                         and no duration leaf
                                            then  depth = 13{value: DEATH, samples: 5}
                                            else  depth = 13{value: CHURNED 0.5, samples: 2}
                                        else  depth = 12{value: DEATH, samples: 9}
                                else  depth = 10{value: DEATH, samples: 71}
                        else  depth = 7 if Age <= 79.5 at t = 3.0, samples: 30
                         and no duration leaf
                            then  depth = 8 if CLV <= 6469.44 at t = 3.0, samples: 6
                             and no duration leaf
                                then  depth = 9{value: CHURNED, samples: 2}
                                else  depth = 9{value: DEATH, samples: 4}
                            else  depth = 8 if CLV <= 4697.78 at t = 4.0, samples: 24
                             and duration leaf has 2 samples. Label is: DEATH 1.0
                                then  depth = 9{value: CHURNED 0.5, samples: 2}
                                else  depth = 9{value: DEATH, samples: 20}
                    else  depth = 6 if GENDER <= 1.5 at t = 0.0, samples: 7
                     and no duration leaf
                        then  depth = 7{value: CHURNED 0.5, samples: 2}
                        else  depth = 7{value: DEATH, samples: 5}
            else  depth = 4 if Age <= 80.5 at t = 0.0, samples: 30
             and no duration leaf
                then  depth = 5 if GENDER <= 1.5 at t = 0.0, samples: 11
                 and no duration leaf
                    then  depth = 6 if Age <= 75.5 at t = 0.0, samples: 4
                     and no duration leaf
                        then  depth = 7{value: CHURNED 0.5, samples: 2}
                        else  depth = 7{value: DEATH, samples: 2}
                    else  depth = 6 if Age <= 79.5 at t = 0.0, samples: 7
                     and no duration leaf
                        then  depth = 7{value: CHURNED, samples: 4}
                        else  depth = 7{value: CHURNED 0.67, samples: 3}
            else  depth = 5{value: DEATH, samples: 19}
```

The unstopped and unpruned `TpTs`, obtained with the time-penalized entropy splitting criterion, and various time penalties yields the following results:



A.3.2 Results with `minsplit` = 25

The maximal unpruned and unstopped `TpTs`, obtained with the time-penalized entropy splitting criterion, `minsplit`= 25, and various time penalties yields the following results:

Time penalty γ	Runtime	Depth	# of terminal leaves	# of duration leaves	Total # of leaves	Tree cost	Max of split times	Mean of split times
0.0000	671.79	7	26	16	42	0.743	20.0	5.689
0.0025	703.08	7	27	15	42	0.740	20.0	5.306
0.0050	796.89	8	30	17	47	0.721	20.0	4.366
0.0100	939.1	10	40	23	63	0.704	20.0	3.835
0.0125	937.04	10	40	24	64	0.704	20.0	3.837
0.0250	940.39	10	41	24	65	0.702	20.0	3.933
0.0275	944.95	10	43	25	68	0.692	20.0	4.277
0.0300	1034.06	10	53	31	84	0.680	20.0	3.933
0.0425	1054.6	10	54	32	86	0.677	20.0	3.875
0.0450	1058.25	10	55	32	87	0.675	20.0	3.845
0.0475	1056.82	10	59	34	93	0.674	20.0	3.84
0.0500	1246.9	14	73	45	118	0.620	19.0	3.613
0.0525	1354.92	14	69	40	109	0.620	20.0	2.419
0.0575	1383.59	14	77	41	118	0.598	20.0	2.484
0.0600	1389.81	14	82	44	126	0.587	19.0	2.666
0.0650	1383.68	14	82	43	125	0.583	19.0	2.639
0.0700	1379.63	14	83	43	126	0.574	19.0	2.605
0.0750	1427.5	15	86	44	130	0.569	19.0	2.581
0.0775	1427.62	15	87	42	129	0.566	19.0	2.606
0.0950	1421.46	15	91	42	133	0.561	19.0	2.591
0.0975	1421.71	15	92	42	134	0.560	19.0	2.606
0.1000	1420.78	15	93	42	135	0.557	19.0	2.549
0.1075	1447.24	15	95	41	136	0.551	19.0	2.712
0.1150	1438.93	15	97	40	137	0.545	19.0	2.688
0.1175	1447.51	15	97	37	134	0.549	19.0	2.657
0.1200	1458.82	15	98	38	136	0.547	19.0	2.674
0.1250	1502.87	15	114	45	159	0.506	18.0	2.643
0.1300	1504.6	15	115	45	160	0.506	18.0	2.628
0.1375	1504.56	17	119	45	164	0.494	15.0	2.647
0.1400	1647.49	17	121	46	167	0.488	15.0	2.672
0.1425	1724.77	17	124	43	167	0.468	15.0	2.119
0.1475	1750.56	17	125	44	169	0.467	17.0	1.787
0.1500	1756.44	17	132	44	176	0.464	17.0	1.78
0.1525	1766.5	17	135	44	179	0.455	15.0	1.63
0.1550	1791.99	18	135	37	172	0.452	15.0	1.633
0.1575	1798.21	18	136	35	171	0.449	15.0	1.638
0.1600	1800.5	18	136	34	170	0.450	15.0	1.636
0.1625	1844.87	18	143	28	171	0.438	15.0	1.415
0.1650	1825.93	18	143	29	172	0.436	15.0	1.416
0.1675	1884.66	18	154	23	177	0.412	11.0	1.187
0.1750	1889.17	18	155	22	177	0.407	11.0	1.158
0.1775	1910.45	19	156	18	174	0.394	11.0	1.003
0.1825	1901.92	19	157	17	174	0.395	11.0	1.003
0.1875	1940.97	19	164	13	177	0.390	11.0	0.865
0.1950	1938.27	19	166	13	179	0.398	11.0	0.846
0.1975	1909.02	19	166	13	179	0.401	11.0	0.843
0.2050	1931.62	19	165	12	177	0.398	11.0	0.809
0.2100	1921.77	19	165	11	176	0.399	11.0	0.807
0.2175	1938.66	21	165	9	174	0.395	12.0	0.844
0.2200	1940.89	21	166	9	175	0.401	12.0	0.828
0.2250	1946.0	21	166	8	174	0.400	12.0	0.818
0.2375	1941.67	21	167	9	176	0.402	12.0	0.819
0.2450	1939.31	21	160	16	176	0.428	14.0	1.059
0.2475	1917.16	21	161	16	177	0.430	14.0	1.05
0.2575	1920.18	19	162	18	180	0.435	14.0	1.013
0.2600	1916.36	19	163	17	180	0.437	14.0	1.007
0.2625	1968.98	19	172	10	182	0.408	11.0	0.771
0.2650	1965.17	20	172	17	189	0.422	13.0	0.849
0.2675	1969.52	21	173	16	189	0.423	13.0	0.85
0.2700	1960.23	21	174	14	188	0.421	13.0	0.85
0.2725	1997.83	21	175	14	189	0.424	13.0	0.817
0.2775	1977.08	21	176	13	189	0.420	13.0	0.816
0.2800	2194.84	24	182	11	193	0.419	13.0	0.791
0.2850	2167.05	24	179	6	185	0.410	7.0	0.649
0.2925	2138.61	24	179	6	185	0.411	7.0	0.647
0.3125	2115.89	24	179	5	184	0.411	7.0	0.641
0.3225	2088.42	24	178	5	183	0.410	7.0	0.626
0.3300	2076.5	24	178	5	183	0.413	7.0	0.612
0.3650	2027.6	24	180	4	184	0.412	7.0	0.596
0.3750	2051.34	24	180	4	184	0.413	7.0	0.584
0.4000	2079.19	24	181	4	185	0.413	7.0	0.56
0.4025	2015.83	24	186	5	191	0.419	7.0	0.444
0.4325	1864.67	24	187	5	192	0.420	7.0	0.431
0.4475	1757.62	19	182	5	187	0.420	7.0	0.095
0.4575	1731.83	19	183	3	186	0.420	4.0	0.092
0.4800	1577.62	19	187	3	190	0.420	4.0	0.057
0.5500	1577.43	19	188	3	191	0.420	4.0	0.053
0.6500	1577.71	19	189	3	192	0.425	4.0	0.048
0.7000	1573.9	19	190	2	192	0.427	4.0	0.045
0.8000	1578.4	19	193	1	194	0.433	4.0	0.04
0.9000	1589.0	19	192	0	192	0.447	3.0	0.032
1.0000	1592.38	19	195	0	195	0.445	3.0	0.022
1.1000	1576.86	19	195	0	195	0.447	2.0	0.02
1.3000	1576.14	19	196	0	196	0.446	2.0	0.015

Table 4: Characteritics of unstopped and unpruned entropy TpTs depending on the time penalty

Time penalty γ	Runtime	Depth	# of terminal leaves	# of duration leaves	Total # of leaves	Tree cost	Max of split times	Mean of split times
0.0000	668.38	5	14	8	22	0.664	15.0	5.722
0.0025	690.38	5	14	7	21	0.664	15.0	5.172
0.0050	782.22	6	17	8	25	0.638	15.0	4.346
0.0100	840.15	7	16	8	24	0.646	15.0	3.552
0.0150	914.61	8	17	8	25	0.637	15.0	3.301
0.0250	912.15	8	18	9	27	0.636	15.0	3.307
0.0275	917.44	8	18	7	25	0.640	15.0	3.033
0.0300	978.72	8	18	8	26	0.637	15.0	2.347
0.0400	1085.14	8	25	10	35	0.618	15.0	1.953
0.0475	1102.22	8	27	12	39	0.613	15.0	1.959
0.0500	1216.25	8	31	14	45	0.586	9.0	1.578
0.0575	1213.44	8	31	15	46	0.584	9.0	1.568
0.0675	1213.17	8	31	13	44	0.585	9.0	1.5
0.0850	1205.48	9	31	15	46	0.581	10.0	1.52
0.1000	1196.86	9	32	14	46	0.571	10.0	1.434
0.1125	1194.82	9	32	13	45	0.569	10.0	1.418
0.1175	1195.85	8	30	11	41	0.581	11.0	1.244
0.1200	1253.33	11	33	9	42	0.562	9.0	0.823
0.1350	1242.11	11	33	9	42	0.561	9.0	0.807
0.1625	1345.51	11	34	9	43	0.562	9.0	0.714
0.1700	1321.99	11	34	8	42	0.562	9.0	0.708
0.1925	1288.5	11	34	9	43	0.563	9.0	0.689
0.1950	1279.26	11	34	8	42	0.567	9.0	0.651
0.2000	1304.58	11	34	5	39	0.565	9.0	0.38
0.2050	1323.32	11	34	4	38	0.563	8.0	0.316
0.2400	1333.81	11	34	3	37	0.567	6.0	0.275
0.2525	1340.74	11	33	3	36	0.568	6.0	0.242
0.3150	1358.63	11	32	1	33	0.570	3.0	0.173
0.3725	1310.12	11	32	0	32	0.574	2.0	0.045
0.3750	1350.37	11	32	0	32	0.575	2.0	0.014

Table 5: Characteritics of Entropy `TpTs` (`minsplit`=25) depending on the time penalty

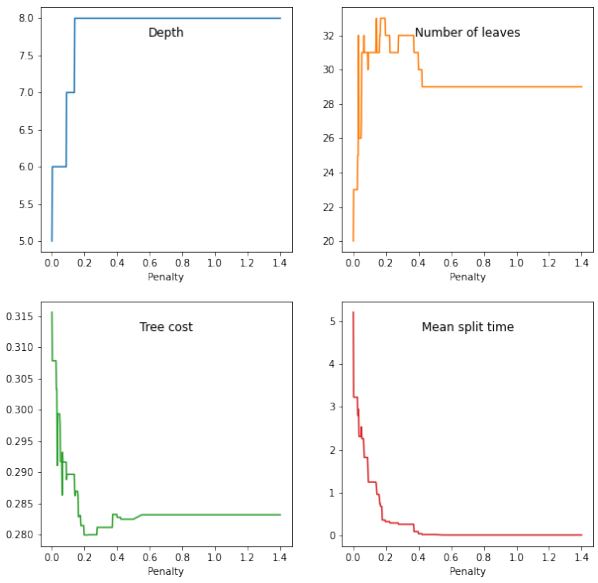


Figure 16: Characteritics of Entropy `TpTs` (`minsplit`=25) depending on the time penalty

A.3.3 Results with `minsplit` = 50

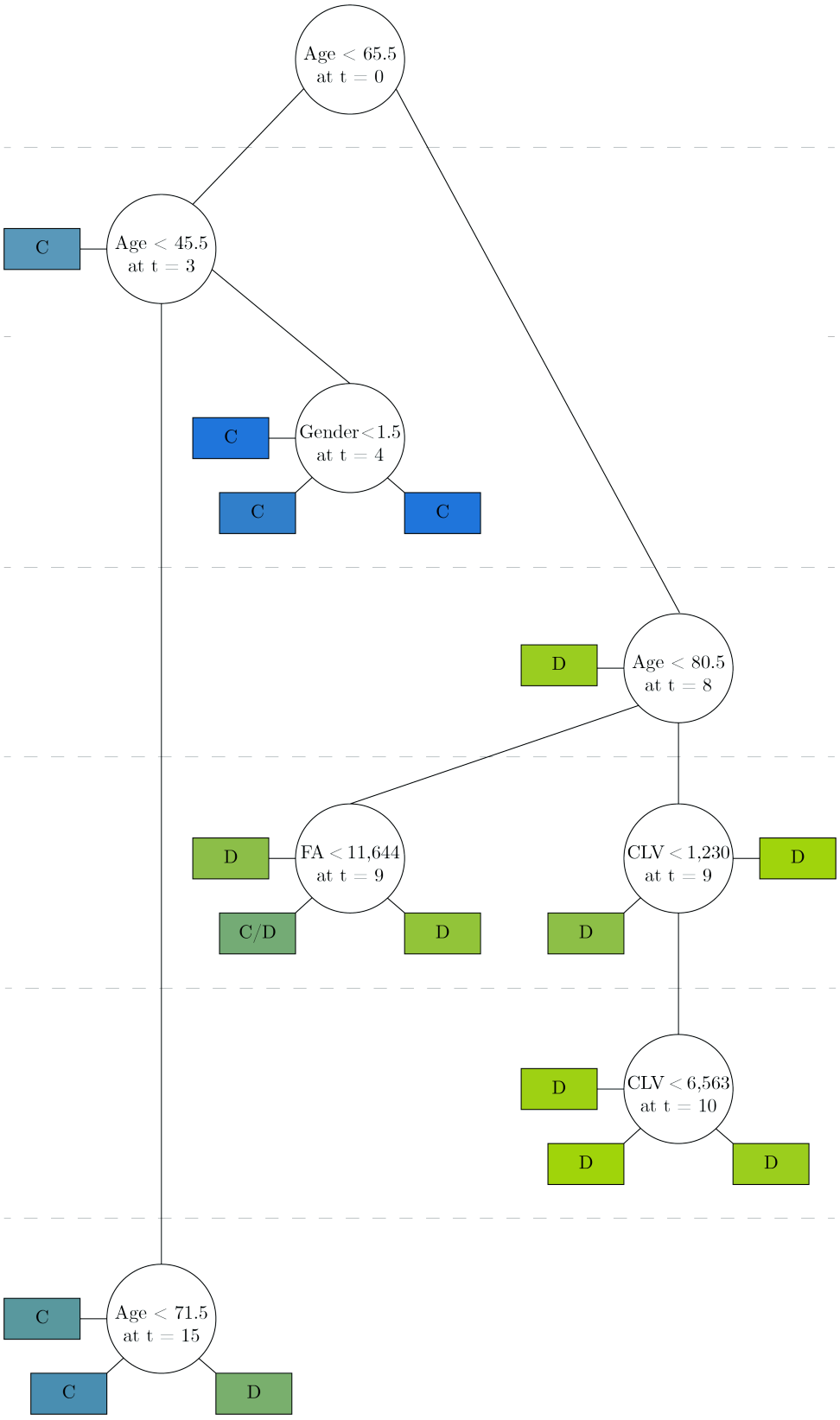


Figure 17: Gini `TpT` (`minsplit`=50) with $\gamma = 0$

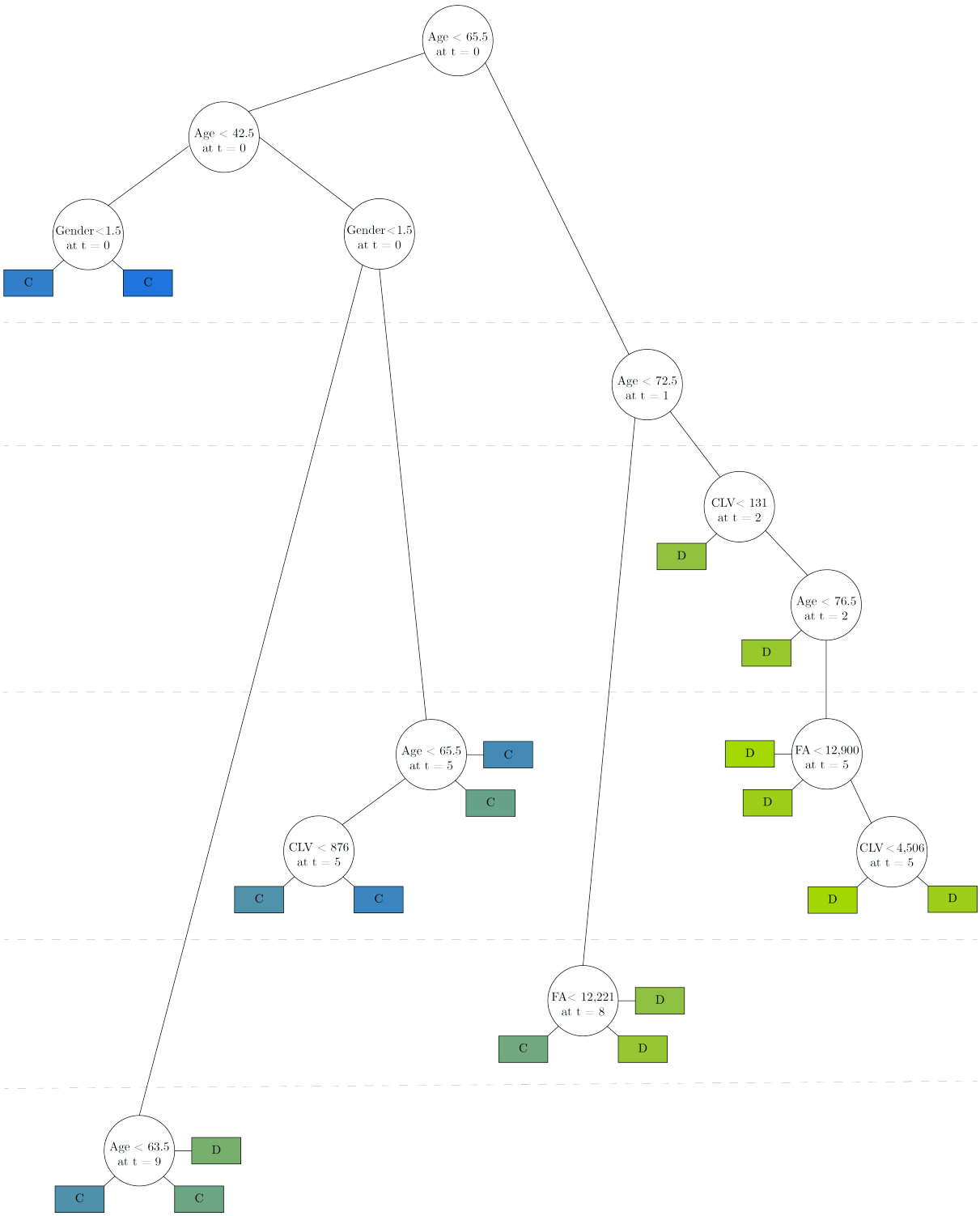


Figure 18: Gini `TpT` (`minsplit`=50) with the optimal time penalty

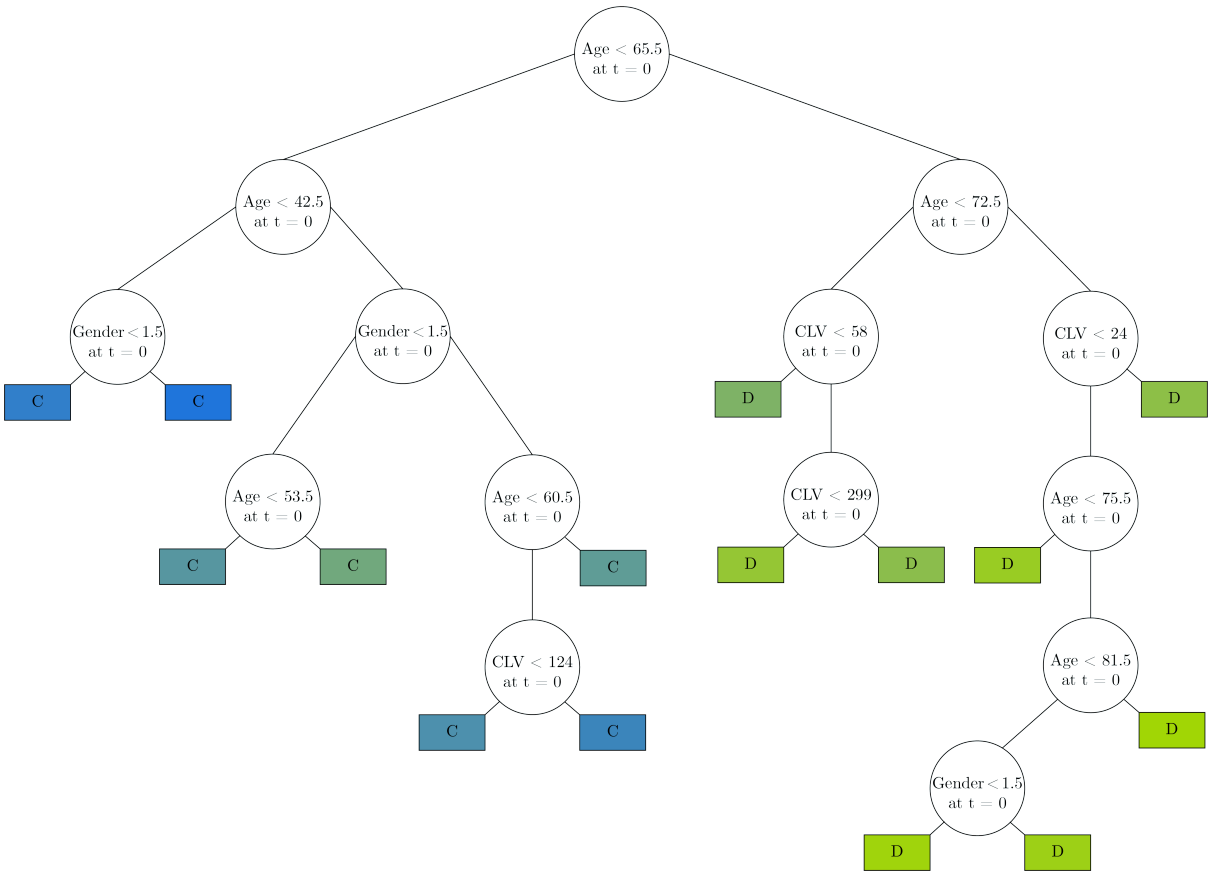


Figure 19: Gini `TpT` (`minsplit`=50) with $\gamma \to \infty$