MOX-Report No. 72/2020

Measure Inducing Classification and Regression Trees for Functional Data

Belli E.; Vantini S.

Measure Inducing Classification and Regression Trees for Functional Data

Edoardo Belli
edoardo.belli@polimi.it

Simone Vantini
simone.vantini@polimi.it

MOX - Modeling and Scientific Computing, Department of
Mathematics, Politecnico di Milano, Italy

Abstract

We propose a tree-based algorithm for classification and regression problems in the context of functional data analysis, which allows to leverage representation learning and multiple splitting rules at the node level, reducing generalization error while retaining the interpretability of a tree. This is achieved by learning a weighted functional L^2 space by means of constrained convex optimization, which is then used to extract multiple weighted integral features from the input functions, in order to determine the binary split for each internal node of the tree. The approach is designed to manage multiple functional inputs and/or outputs, by defining suitable splitting rules and loss functions that can depend on the specific problem and can also be combined with scalar and categorical data, as the tree is grown with the original greedy CART algorithm. We focus on the case of scalar-valued functional inputs defined on unidimensional domains and illustrate the effectiveness of our method in both classification and regression tasks, through a simulation study and four real world applications.

1 Introduction

The idea of making predictions using hierarchical, binary if-else decision rules is naturally appealing to rational human agents, as it mimics our way of thinking on a simplified and abstract level. This is also reflected in the way we program our machines, where most programming languages make use of some type of conditional branching. In the artificial intelligence community, human defined rules where used to build expert systems that where able to leverage large knowledge

bases, with the objective to aid or even replace decision makers. With the increase in computational power, memory capacity and data collection, it became clear that the approach of learning those rules was far superior than designing them. A major contribution in this direction, was the introduction of decision tree learning, and in particular the classification and regression trees (CART) algorithm (Breiman et al., 1984), which sprouted a whole new literature on tree-based methods in machine learning. The success of the CART framework was mainly driven by its flexibility and interpretability, as it provided a non-parametric tool with nonlinear decision boundaries for both classification and regression problems, which was also able to handle categorical and numerical variables, while at at the same time retaining the intuition of binary splitting rules. From the generalization perspective, CART always suffered from instability issues, where small variations in the input data could lead to inconsistent predictions. Successful techniques like bagging (Breiman, 1996) random forest (RF) (Breiman, 2001) and gradient boosted trees (GBT) (Friedman, 2001) were introduced to reduce variance and bias, improving performance at the expenses of interpretability.

In the last decade, data has become more complex and new tools are needed to extract meaningful and reliable information from it. When sample sizes are large and the focus is on prediction, deep learning (DL) has proven to be the best approach, with an overwhelming success in multiple applications that deal with complex data types, like computer vision and natural language processing. Thanks to their scalable and modular architectures, DL methods are able to exploit huge sample sizes to learn representations that capture the inner structure of these data and generalize well in different settings. However, when data is scarce and interpretability is sought after, DL is not an option and we need an additional modeling effort to be able to handle complex data. Functional Data Analysis (FDA) can be seen as a data model for smooth and high dimensional data, which often arises in applications where high-frequency sampling devices are involved, like in electrical engineering, chemometrics and in the biomedical field. The typical instance of functional data is a scalar quantity measured in time, but in general we are dealing with multidimensional objects defined on multidimensional domains, like RGB images or volumes. From a practical standpoint, functional data is clearly discrete and finite dimensional, as it is collected by sensors and stored in memory. This however should not preclude the opportunity to take advantage of the continuous nature of the phenomena that generate these data, like in the case of physical quantities. A naive approach is to look at the raw data as p evaluations of a function on a discretized grid over the domain, which does not take into account the fact that different samples could have been generated by grids with different spacing, even if they have the same number of points p. Also, there could be multiple missing values and each evaluation could be affected by noise induced by the sensors used for collecting the data, including small time discrepancies between different readings. Therefore, the true underlying functions are expressed as a finite basis expansion and are usually recovered by means of smoothing or functional principal components, depending on the application and on the amount of missing values. Another

common practice is to align the input functions to further reduce the phase variability and retain only the information given by the amplitude variability (Sangalli et al., 2010). Once the basis and/or the coefficients are estimated in this preprocessing step, the functions are evaluated on the same dense grid and known multivariate methods are employed for the numerical estimation of a classifier/regressor, with the limitations imposed by the intrinsic high dimensionality, which are not only related to model fitting and generalization error. In the case of CART, it is well known that in high dimensions the resulting trees can be prohibitively complex for any intuitive and significant interpretation, not only for the excessive height of the trees, as in some cases an intricate prediction rule may be considered more plausible (Fürnkranz et al., 2020), but especially for the difficulty of assigning a clear meaning to each split, defeating one of the main advantages of the method. Even in the case of axis-parallel splits, if for instance we consider a classification tree that has been trained on dense spectrometric input data, it may be contrived to justify selecting a specific wavelength as the splitting variable, while at the same time binning the domain in arbitrary subintervals may be too simplistic. In the FDA literature, much attention has been devoted to linear models, dimensionality reduction and inference (Ramsay and Silverman, 2005; Horváth and Kokoszka, 2012). Regarding supervised learning, nonparametric methods with functional covariates have also been studied extensively (Ferraty and Vieu, 2006), often in conjunction with other approaches like SVMs (Rossi and Villa, 2006), domain segmentation (Li and Yu, 2008) and distance learning (Alonso et al., 2012; Galeano et al., 2015). In the case of tree-based methods, most works focus either on functional inputs or responses, as the two aspects are in general decoupled, given that the response is related to the specific loss function for the task. The problem of dealing with functional covariates is often tackled by splitting each node with respect to the distance from a chosen template function, and we will discuss this more in detail in the next section. As data belongs to an infinite dimensional functional Hilbert space, rather than treating it as multiple evaluations over a dense grid, another approach is to consider each function as a whole unit (Epifanio, 2008). This opens up the possibility of extracting global information from the functions and their derivatives, like integral features, without relying exclusively on the pointwise evaluations, which are high dimensional and could be affected by measurement error or extreme fluctuations of the underlying phenomena, as the true functions are never available but are always recovered through some estimation process. Following this approach, we focus on problems with with functional inputs and introduce a tree-based algorithm that extends the standard CART algorithm to deal with (multiple) functional covariates, with the objective of improving both prediction and interpretability, while at the same time keeping all the advantages of decision trees, like the option of including other categorical/scalar covariates, different pruning schemes and the choice of any loss function. This is achieved by learning multiple weight functions for each internal node, inducing a set of measures that are used to extract multiple integral features from the input data, effectively learning representations that are weighted L^2 spaces, as we will elaborate in section three. Finally, in section

four we will analyze the performance in classification and regression problems with both simulations and real world applications.

2 Related Work

Starting from the early works that appeared more than three decades ago (Breiman et al., 1984; Quinlan, 1986), the literature about classification and regression trees has kept growing and these methods are still relevant nowadays, with developments that encompass multiple elements of the framework. In this section, we will focus on the three main aspects that we think are at the core of tree-based learning, in order to highlight the differences with respect to our proposal. It is clear that while presented separately, all three aspects are related to each other in many different ways, and we do not aim to provide an exhaustive analysis of the subject.

2.1 Training

The first aspect is the training process, which is related to the size and the architecture of the tree, that has to be controlled through hyperparameter optimization and/or model search. The original CART uses a greedy algorithm that is based on locally optimal binary partitions of the input space. This allows to avoid fixing a priori the height of the tree, which is restricted by means of pruning, either while growing the tree (pre-pruning) or after the tree has been fully grown (post-pruning). Empirical evidence shows that pre-pruning often leads to simpler trees with the same accuracy as post-pruned ones (Martin, 1997), where a typical approach is to control the growth of the tree by either limiting the maximum height or by fixing the minimum number of elements in a leaf. Permutation tests have also been used as an unbiased stopping criteria (Hothorn et al., 2006), with a p-value threshold as the hyperparameter to tune. If instead we accept to fix the height of the tree a priori, alternative approaches have been proposed in which the tree is trained for global optimality with different algorithms, like expectation maximization (Jordan and Jacobs, 1994), stochastic gradient descent with backpropagation (Norouzi et al., 2015; Kontschieder et al., 2015) or by solving a mixed integer optimization problem (Bertsimas and Dunn, 2017). In the case of our proposed method, the tree is trained with the traditional greedy algorithm, which also depends on the weight functions that are learned for each internal node. Regarding this aspect, our method can be seen as a generalization of the original CART, as it does not prevent the usage of the univariate and multivariate splitting rules that are known in the literature, nor it forces the employment of any specific stopping or pruning scheme, while adding the option to deal with functional data at the node level without relying exclusively on the single evaluations.

2.2 Node Splitting

The second and most discussed aspect is how to perform the split inside a single node, which comprises both the problems of selecting a suitable splitting variable and computing the split itself. While CART is mostly known for axis parallel splits, which selects a single variable and therefore limits the shape of the decision boundary that can be learned, the original algorithm already allowed for oblique splits (CART-LC), by splitting with respect to a linear combination of the input features that was computed with an heuristic based on deterministic coordinate search. In the following years, multiple approaches have been proposed to find good and possibly sparse oblique split, including recursive least squares with feature elimination (Brodley and Utgoff, 1995), stochastic global optimization (Heath et al., 1993; Murthy et al., 1994; Cantú-Paz and Kamath, 2003) and GLMs (Gama, 2004). Other works focused exclusively on classification, inducing the splitting rule by the use of linear programming and SVMs (Bennett and Mangasarian, 1992; Bennett et al., 2000; Tibshirani and Hastie, 2007), logistic regression (Landwehr et al., 2005), budget-aware classifier (Liu and Tsang, 2017) and different types of discriminant analysis (Loh and Vanichsetakul, 1988; Gama, 1999). A known issue of CART is the one of selection bias, where variables with many possible splitting values are preferred during the selection process, favouring categorical variables and variables with fewer missing values. Therefore, the problem of unbiased split variable selection has been tackled by means of statistical hypothesis testing for both classification (Loh and Shih, 1997; Kim and Loh, 2001; Strobl et al., 2007; Loh, 2009) and regression (Loh, 2002) trees, where the splitting variable with the lowest p-value is chosen, and then the split can be computed with any given approach. In the context of time series and FDA, most of the attention has been devoted to classification with one functional input, where for each internal node of the tree, one or more template functions are chosen and then a univariate split is performed with respect to the distance from such functions. The selection of the template functions has been implemented using exhaustive search (Yamada et al., 2003; Douzal-Chouakria and Amblard, 2012), binary clustering (Yamada et al., 2003; Balakrishnan and Madigan, 2006) or by taking the mean function by class (Möller and Gertheiss, 2018). This template technique has also been used together with recursive binary domain segmentation and multiple distance metrics (Douzal-Chouakria and Amblard, 2012). The domain segmentation approach has further been used in a RF fashion for classification and regression with one functional input (Möller et al., 2016), where each tree of the forest is a standard CART that is fitted on a different preprocessed bootstrap subset of the inputs. This preprocessing step consists in randomizing a set of possibly overlapping intervals over the domain of the functions and taking the mean of the functions over these intervals, which is not related to the fact that the classifiers are trees and could be used in conjunction with other ensemble methods. We propose a node splitting procedure that can be used for both classification and regression, which is based on locally learning a weighted L^2 space by means of constrained convex optimization, that allows to extract multiple linear and

nonlinear weighted integral features from the input functions. The concept of node-wise representation learning is not new, as there are works in computer vision that train a multilayer perceptron for each node of the tree (Bulò and Kontschieder, 2014; Kontschieder et al., 2015). While a deep neural network learns a hierarchical and nonlinear representation of the input data, our representations are linear and not hierarchical and we need to specify the features that we want to extract for each node. For this reason, this can be seen as a model-based way of learning representations, which may lack the expressive power of more abstract ones, but has the advantage of retaining the interpretability, since the learned weight function is a measure of importance of the subsets of the domain of the input functions, and the extracted features can be nonlinear and have a clear mathematical meaning. Moreover, while the focus of this work is on the splitting rules for functional data, it is worth to note that as our rule fits in the original CART framework, in principle one could implement our approach to deal with the functional covariates, mapping each function to a real value, while at the same time including any other data source like categorical variables. This would introduce again the problem of biased variable selection, which in turn can be solved by resorting to any of the selection rules based on statistical hypothesis testing.

2.3   Input and Response Data

The third aspect is the type of task that the tree can solve and therefore the nature of the input and response data. The original CART is suited for both classification and regression problems with multivariate and categorical input data, and it has been extended to multivariate regression (Segal, 1992), longitudinal responses (Loh and Zheng, 2013; Pande et al., 2017) and other regression methods in the leaves (Torgo, 1997), multitask learning (Caruana, 1997; Zhang, 1998; Noh et al., 2004; Lee and Shih, 2006) with hierarchical labels (Vens et al., 2008), high label dimensionality (Liu and Tsang, 2017) and responses in the form of class probabilities (Smyth et al., 1995). In the case of a single response that is a probability density function, the purity of the split has been evaluated by computing the sum of the f-divergences between the responses that correspond to the inputs in the left and right nodes (Nerini and Ghattas, 2007), while smoothing and principal component analysis have been used to reduce the dimensionality of the response as a preprocessing step (Yu and Lambert, 1999). Tree-based methods have also been employed for conditional quantile estimation (Chaudhuri and Loh, 2002; Meinshausen, 2006) and conditional density estimation (Cousins and Riondato, 2019). As previously noted, the objective of this work is to introduce a splitting approach for functional inputs in the context of the original CART algorithm, without specific attention to the type of response or any additional categorical and/or scalar data. In particular, we focus on the case of (multiple) scalar-valued functional covariates defined on a one-dimensional domain, but in principle our approach could be extended and tailored to specific applications with more complex data, like vector-valued functions defined on multidimensional domains.

3   The Measure Inducing CART

In this section, we describe our method with respect to functional data, but in general it can be applied to any Hilbert data, like in the usual multivariate case. The way our tree is trained is by recursive binary splitting of the input space, as in the original CART. For this reason, we will focus on the node splitting procedure. Regarding pruning, we control the growth of the tree by limiting the minimum number of elements in a leaf, without any post pruning technique like cost complexity pruning and weakest link pruning. This is just an implementation choice and we are not restricted to the use of any patricular pruning technique. In order to underline the differences with the functional case and establish notation, we briefly recall the multivariate CART algorithm for splitting a node.

Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ be the training set with $x_i \in \mathbb{R}^p$ and $y_i \in \{1,..,K\}$ for classification or $y_i \in \mathbb{R}$ for regression. Let $j = 1,..,p$ and $s \in \mathbb{R}$ a splitting threshold, split the input space into two subregions such that

$$\mathcal{R}_1(j,s) = \{x | x_j \le s\} \qquad \mathcal{R}_2(j,s) = \{x | x_j > s\}$$

where j and s are chosen by solving the following optimization problem:

$$\min_{j,s} \left[\mathcal{L}\left(\mathcal{R}_1(j,s)\right) + \mathcal{L}\left(\mathcal{R}_2(j,s)\right) \right] \tag{1}$$

and $\mathcal{L}$ is the loss function of the node which depends on the task, for example gini index for classification and mean-square error for regression. To compute the optimal regions $\mathcal{R}_1(j^*, s^*)$ and $\mathcal{R}_2(j^*, s^*)$, we can find j^* and s^* by enumerating through the features for $j = 1,..,p$ and through the the splits $s_i = x_{ij}$ for $i = 1,..,N$, evaluating the loss function for each couple (j, s_i) and selecting the one with the lowest value of the loss. Problem (1) can be solved in $O(pN)$ time, assuming that for each j the splits s_i have been presorted before fitting the tree, which can be done in $O(pNlogN)$, and also that the cost of evaluating the loss function is $O(N)$, which holds for the most common losses used in both classification and regression. The overall cost of fitting a balanced tree is $O(pNlogN)$, while in the general case of unbalanced trees it becomes $O(pN^2)$ (Hastie et al., 2009).

When the input data belongs to a functional Hilbert space, like the L^2 space of square integrable functions, the multivariate approach of splitting the node by iterating through all the p dimensions cannot be used directly, as we first need a preprocessing step in order to estimate the true functions and evaluate them on the same p-dimensional grid. In particular, we express $x \in L^2(I)$ as a finite expansion on a suitable basis $x(t) = \sum_{j=1}^J \xi_j \psi_j(t)$, where the coefficients ξ_j, the basis functions ψ_j and the number of basis J are estimated from the raw data. If the number of missing values is not excessive, each function can be recovered individually by fixing a basis or using free knots regression splines.

Otherwise, mixed-effects models (James et al., 2000) or local smoothing (Yao et al., 2005) can be used with functional principal components in order to augment each individual sample with dataset-wide information, compensating for the unobserved parts of the domain of the individual sample. Regardless of the estimation approach that is chosen, this is already a viable option that properly manages the problems of noise and missing values, but it's not directly tied to the CART algorithm. While this decoupling can be beneficial is some cases, a tighter integration with the classification/regression methods could lead to better generalization. The tree structure is a natural candidate for this effort, as it is composed of multiple simple elements (the nodes), but at the same time allows for extreme flexibility in dealing with different types of inputs and responses (different splitting rules and loss functions). While a straightforward approach could be repeating smoothing and/or alignment in each node, this would not alleviate the issues related to high dimensional splits and interpretability. Following the idea of looking at the single functional sample as a whole unit, we recover the input functions once before fitting the tree, and instead propose a splitting rule that acts as a variable selection mechanism over subsets of the domain, avoiding to rely exclusively on the single evaluations and therefore reducing the number of splitting dimensions. In particular, by means of functional linear/logistic regression, we estimate multiple data-driven measures that will be used to compute a set of weighted integral features for each input function, with the tree splitting process that is based on such features instead of the single evaluations. As the infinite dimensional functional linear model is ill posed, we adopt the approach of fixing a rich enough basis with a suitable penalty as an identifiability constraint (Cardot et al., 2003, 2007; Crambes et al., 2009; Yuan and Cai, 2010). In particular, we employ the following simple grid basis for the weight function w inside each node:

$$w(t) = \sum_{j=1}^{p} w_j \phi_j(t) \qquad \phi_j(t) = \begin{cases} 1 & \text{if } \frac{j-1}{p} < t \le \frac{j}{p} \\ 0 & \text{otherwise} \end{cases}$$

where p is the length of the estimation grid of the smoothed inputs and is fixed before fitting the tree. Consider the training set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ with random functions $x_i \in L^2(I)$ and responses $y_i \in \{1, .., K\}$ for classification or $y_i \in \mathbb{R}$ for regression. Let $w : I \to \mathbb{R}$ be a weight function and let $s \in \mathbb{R}$ be a splitting threshold. Let $f : L^2(I) \to \mathbb{R}$ be a feature extractor chosen from a predefined set $\{f_1, .., f_M\}$. Split the input space into two subregions

$$\mathcal{R}_1^f(w, s) = \{x | f(x, w) \le s\} \qquad \mathcal{R}_2^f(w, s) = \{x | f(x, w) > s\}$$

that are chosen by solving the following optimization problem:

$$\min_{f, w, s} \left[\mathcal{L}(\mathcal{R}_1^f(w, s)) + \mathcal{L}(\mathcal{R}_2^f(w, s)) \right] \tag{2}$$

where $\mathcal{L}$ is the loss function of the node which depends on the task (classification or regression). There is no restriction on the number of splitting rules that can be used, as one can always iterate through M feature extractors and choose the corresponding rule that results in the best split with respect to the purity criteria of the node. Moreover, the splitting rules can also depend on the task and on the nature of the input data, like in the presence of any combination of categorical, numerical or functional inputs. Motivated by FDA, we propose the following splitting rules for both classification and regression with one functional input:

$$\mathcal{R}_1^{\mu}(w, s) = \left\{ x \big| \tfrac{1}{|I|} \int_I x(t)w(t)dt \leq s \right\}$$
$$\mathcal{R}_2^{\mu}(w, s) = \left\{ x \big| \tfrac{1}{|I|} \int_I x(t)w(t)dt > s \right\}$$

$$\mathcal{R}_1^{\sigma^2}(w, s) = \left\{ x \big| \int_I (x(t) - \bar{x})^2 w(t)dt \leq s \right\}$$
$$\mathcal{R}_2^{\sigma^2}(w, s) = \left\{ x \big| \int_I (x(t) - \bar{x})^2 w(t)dt > s \right\}$$

$$\mathcal{R}_1^{cos\theta}(w, s) = \left\{ x | \rho_w(x, \bar{x}_{node}) \leq s \right\}$$
$$\mathcal{R}_2^{cos\theta}(w, s) = \left\{ x | \rho_w(x, \bar{x}_{node}) > s \right\}$$

where $\bar{x} \in \mathbb{R}$ is the mean of the function $x \in L^2(I)$ with respect to the measure induced by w, so that the first rule splits the node depending on the weighted mean of the function over its domain, while the second rule splits the node with respect to the function's weighted variance over the domain. The third rule is based on the the weighted functional cosine similarity

$$\rho_w(x, z) = \frac{\int_I x(t)z(t)w(t)dt}{\sqrt{\int_I x^2(t)w(t)dt}\sqrt{\int_I z^2(t)w(t)dt}}$$

with $\bar{x}_{node} \in L^2(I)$ being the mean function over the functions in the current node of the tree. In the case of classification with K classes, we add K additional cosine splitting rules, by taking $\bar{x}_k \in L^2(I)$ as the mean function of the class k inside the current node, instead of $\bar{x}_{node}$. Regardless of the splitting rule(s) that we choose to use, the objective function of Problem (2) is nonconvex and piecewise constant. For a given w, we can solve this problem by enumeration as in the original CART, by computing the splitting thresholds $s_i = f(x_i, w)$ for $i = 1, .., N$, $f \in \{f_1, .., f_M\}$ and choosing the split with the highest purity. However, minimizing with respect to w yields a global optimization problem, as it involves finding the optimal oblique split. Given the greedy nature of the tree building procedure, we might not need to find a solution that is close to the true unknown global optimum of the node. Nevertheless, solving this problem with a derivative free approach is computationally prohibitive, especially for deeper trees. To avoid solving Problem (2) directly, we propose to learn the weight function first and then compute the optimal weighted split for the current node. This means that each node of the tree will be associated with a data-driven

weighted L^2 space, where the weight function that induces the measure of the space is learned in a supervised fashion, and such (possibly signed) measure is used to compute the integral features that will determine the optimal splitting threshold. As for the M feature extractors $\{f_1, .., f_M\}$, there is no restriction on the number of weight functions $\{w_1, .., w_D\}$ that we can compute in each node, as the optimal one will be selected by iterating over all the MD combinations in Problem (2). It is known that sign constraints can be a powerful tool in linear models (Meinshausen, 2013; Slawski and Hein, 2013), and for this reason we choose to learn three distinct weight functions $\{w_{pos}, w_{neg}, w_{sgn}\}$, with positive, negative and no sign constraints, by solving three convex optimization problems that differ by their respective set of constraints. For the task of classification, we solve the following problem:

$$\min_{w_0, w} - \sum_{i=1}^{N_{node}} \left[y_i \left(w_0 + \int_I x_i(t)w(t)dt \right) - \ln \left(1 + e^{w_0 + \int_I x_i(t)w(t)dt} \right) \right] + \lambda \int_I w^2(t)dt \tag{3}$$

$$s.t. \quad \int_I w(t)dt = |I| \quad or \quad \begin{cases} \int_I w(t)dt = |I| \\ w(t) \geq 0 \end{cases} \quad or \quad \begin{cases} \int_I w(t)dt = -|I| \\ w(t) \leq 0 \end{cases}$$

where $w_0 \in \mathbb{R}$ is the intercept, N_{node} is the number of curves in the current node and $y_i \in \{0, 1\}$ is the binary class label. For multiclass classification with K classes, we select the modal class inside the node and relabel the data as in a one-versus-rest (OvR) setting, but only for the purpose of this problem. Note that we do not resort to the standard OvR scheme of solving $K-1$ problems, as we do not use the linear model for making predictions, but only for learning the weight function of the space, with the tree splitting mechanism that provides for class discrimination. While it is true that solving all the $K-1$ problems could lead to better splits and therefore purer nodes at shallower depths, this would further increase the computational burden, without any guarantee of global optimality for the fully grown tree. The hyperparameter $\lambda \geq 0$ that controls the amount of shrinkage is selected by cross-validation, and in our implementation it is set before fitting the tree, while in principle one could select the optimal one inside each node, without any guarantee that this would increase generalization. In order to have a meaningful comparison between the different regions of the domain, the input curves are standardized over the whole dataset, again only for this step. Analogously, for the task of regression we solve:

$$\min_{w_0, w} \sum_{i=1}^{N_{node}} \left[y_i - w_0 - \int_I x_i(t)w(t)dt \right]^2 + \lambda \int_I w^2(t)dt \tag{4}$$

$$s.t. \quad \int_I w(t)dt = |I| \quad or \quad \begin{cases} \int_I w(t)dt = |I| \\ w(t) \geq 0 \end{cases} \quad or \quad \begin{cases} \int_I w(t)dt = -|I| \\ w(t) \leq 0 \end{cases}$$

with $y_i \in \mathbb{R}$. Therefore, the difference between Problems (3) and (4) is in the first term of the objective function, which is either a functional logistic loss or a functional ordinary least squares. In the context of FDA, the usual choice is often a roughness penalty rather than the ridge penalty that we employ, as the intent is to estimate a smoother linear model. However, we do not employ these models to directly perform the split, but rather use them to learn the measure associated with the weighted L^2 space of the node, with the sets of constraints that are such that $\mu(I) = \nu(I)$, with μ the Lebesgue measure and ν the measure induced by the resulting weight function. In particular, the weight function has to be positive in order to define a weighted L_2 space and have statistically well posed feature extractors f_m. For this reason, we take the absolute value of the weight function w_{neg}, which does not change the relative importance of the different parts of the domain I. Unfortunately, this interpretation breaks down in the case of the signed measure w_{sgn}, where the weighted cosine similarity is not defined, but we still retain the first two splitting rules, which are both linear models. Without solving an additional optimization problem, we also include the uniform weight function in order to extract the unweighted features, as in principle there is no guarantee that the learned representations will be beneficial. On a side note, the type of penalty in the optimization problem could either be treated as an hyperparameter to be selected by cross-validation, or multiple penalties could be included inside each node, further increasing the number of learned weight functions D, together with the training time. Overall, our approach could be seen as an oblique tree where the linear combination determines the regions of the domain I that are most informative with respect to the task, and the splitting values are obtained as a weighted function of the input curves. Another analogy could be drawn with neural networks, as our approach is also similar to having a single neuron for each node of the tree, with one of the feature extractors as the adaptively chosen activation function.

From the computational standpoint, suppose that the true input functions have been estimated and the evaluation grid is of length p. For each node of the tree, we first need to solve either Problem (3) for classification or Problem (4) for regression, once for each of the D sets of constraints. Both problems are convex with linear equality and inequality constraints and can be solved with interior point methods in $O(\sqrt{p})$ iterations (Gondzio, 2012), using Newton's method at a cost of $O(p^3)$ for a single iteration. To compute the split, for each of the D resulting weight functions ($D + 1$ if we include the uniform one), we first need to compute the M weighted features for the N_{node} curves in the current node at a cost of $O(pDMN)$, and then we solve Problem (2), by sorting the splits in $O(DMNlogN)$ time and finding the optimal one in $O(DMN)$, again assuming that the loss function can be evaluated in $O(N)$. As the splits depend on the current w_d in each node, we cannot presort them once before fitting the tree. Given that the number of features M and the number of weight functions D

are independent from p and N (and usually much smaller), the asymptotic cost of splitting a single node is $O(NlogN + p^{3.5})$. In the case of balanced trees, the asymptotic cost of fitting a tree is $O(Nlog^2N + p^{3.5}logN)$, which becomes $O(N^2logN + Np^{3.5})$ in the unbalanced case.

4 Experimental Results

In the following sections, we show a classification simulation study with two functional covariates and multiple real world applications for both multiclass classification and regression. In order to validate the performance of our approach (μCART), we compare it against known tree-based methods like CART, RF and GBT. For datasets that are not pre-smoothed, we recover the true functions by means of free knots regression splines and evaluate them on the same p-dimensional equispaced grid. This preprocessing step is a standard practice in FDA and is shared between all methods, allowing for a meaningful comparison. Moreover, we also test CART, RF and GBT with an additional preprocessing approach (FE), which consists in extracting the same features that we implement in our nodes, with the objective of showing the effectiveness of learning node-wise representations. The code has been implemented in python3 and all the known methods and cross-validation routines rely on Scikit-learn (Pedregosa et al., 2011). The optimization problems have been modeled with Pyomo (Hart et al., 2017) and solved with Ipopt (Wächter and Biegler, 2006). All the computation has been done on a desktop CPU with 4 cores at 4.4GHz and 16GB of RAM.

4.1 Classification

For the task of classification, we first show a synthetic problem with two functional covariates and class membership that depends on their shape in a subset of the domain. The second case study deals with infrared spectra classification for product authentication in chemometrics, while the third application is from the medical field and is about (multiclass) disease classification from gut microbiota. The growth of all trees is controlled by limiting the minimum number of samples required to be at a leaf node, with Gini index as the function to measure the quality of the split for CART, μCART and RF, while for GBT the loss function is the deviance and the quality of the split is measured by the improved least squares criterion (Friedman, 2001). The number of trees for the ensemble methods has been fixed to 150, while for RF in particular, the bootstrap sampling is class balanced and the maximum number of features is chosen adaptively between $\{all, log, sqrt\}$. For all case studies, we report the accuracy on the test set with the corresponding standard deviation, averaged over five random repetitions (only one for the simulation) of 5-fold cross-validation with 3-fold cross-validation for grid search hyperparameter optimization. Moreover, we also report the average tree height over the folds but only for μCART and CART, as the other methods are not easily interpretable and the mean tree

height of an ensemble of trees is arguably not as informative.

4.1.1 Simulation Study

This simulation is a binary classification problem with two functional covariates, in which the information to discriminate between the classes is localized in the first part of the domain of the first covariate, while the second one is not informative, as shown in Figure 1. Let $i = 1, .., N$ with N the sample size (each sample is a curve) and let $j = 1, .., p$ with p the number of points of the discrete evaluation grid over the domain. Regarding the first covariate, the base functional model is a sine curve evaluated in $t_j \in [0, 2\pi]$, with a curve dependent amplitude term $b_{1i} \sim \mathcal{N}(0, 1)$ and a curve dependent phase term $\phi_{1i} \sim \mathcal{N}(0, \frac{\pi}{75})$. For $t_j \in (\frac{3}{8}\pi - \phi_{1i}, \frac{5}{8}\pi - \phi_{1i})$, we add to the base model another sine that is multiplied by a curve dependent term $\alpha_i \sim \mathcal{N}(0, 0.3)$, with the sign of α_i determining the class. The second covariate is a cosine evaluated in $t_j \in [0, 2\pi]$, with a curve dependent amplitude term $b_{2i} \sim \mathcal{N}(0, 1)$ and a curve dependent phase term $\phi_{2i} \sim \mathcal{N}(0, \frac{\pi}{75})$, but no information on the class membership. The simulation has been repeated with $N=\{100, 150, 200\}$ and the functions have been evaluated in $p=200$ linearly spaced points in $[0, 2\pi]$.

$$
\begin{cases}
y_i = sgn(\alpha_i) \\
x_{1i}(t_j) = b_{1i} + \sin(t_j + \phi_{1i}), & t_j \in [0, \frac{3}{8}\pi - \phi_{1i}] \cup [\frac{5}{8}\pi - \phi_{1i}, \pi] \\
x_{1i}(t_j) = b_{1i} + \sin(t_j + \phi_{1i}) + \alpha_i \sin(8(t_j + \phi_{1i})), & t_j \in (\frac{3}{8}\pi - \phi_{1i}, \frac{5}{8}\pi - \phi_{1i}) \\
x_{2i}(t_j) = b_{2i} + \cos(t_j + \phi_{2i}), & t_j \in [0, 2\pi]
\end{cases}
$$

Table 1 shows the classification results for all combinations of methods, preprocessing approaches and sample sizes that we tested, together with the average tree heights when applicable. Our proposed tree μCART has the highest accuracy, showing that the method is able to learn local representations that are effective for the task at hand, as extracting the same features on the whole domain does not allow the other classifiers to have significantly better scores than random guessing, which is intended by design. Using the discretized function evaluations allows the other methods to at least scale with the increasing sample size, but even with double the amount of data and ensembles, the highest accuracy is still lower than the one resulting from μCART with $N=100$. Moreover, our method has the lowest average tree height, which can be beneficial for interpretability, as shown in Figure 2, where a fully grown tree is depicted, together with the (rescaled) learned measures in the inner nodes. In particular, both inner nodes split with respect to the first functional covariate, the root selects w_{sgn} with the weighted variance as splitting feature (f_{σ^2}), while the other inner node also splits with respect to f_{σ^2} but selects w_{pos}.

13

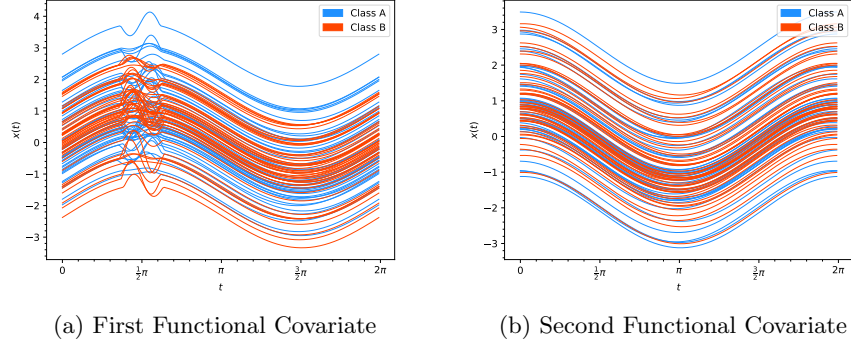(a) First Functional Covariate

(b) Second Functional Covariate

Figure 1: Simulated data, classification

Table 1: Simulation results: classification accuracy and tree height

	Accuracy %			Tree Height		
	N=100	N=150	N=200	N=100	N=150	N=200
CART	72.2 ± 6	80.1 ± 6	84.1 ± 6	4.4 ± 1	5.2 ± 1.2	6 ± 0.6
CART+FE	53.9 ± 10	53.7 ± 13	57.5 ± 7	5.2 ± 3.3	2.2 ± 0.7	9.8 ± 3.9
RF	76 ± 7	82.1 ± 5	83.6 ± 5	$-$	$-$	$-$
RF+FE	54.1 ± 7	55.6 ± 11	56 ± 9	$-$	$-$	$-$
GBT	80.8 ± 7	84.1 ± 6	85.6 ± 3	$-$	$-$	$-$
GBT+FE	53.7 ± 4	52.1 ± 12	60 ± 3	$-$	$-$	$-$
μCART	85.9 ± 18	97.4 ± 4	98 ± 1	1.4 ± 0.5	1.6 ± 0.8	1.8 ± 0.4

$w_{sgn} - f_{\sigma^2}$
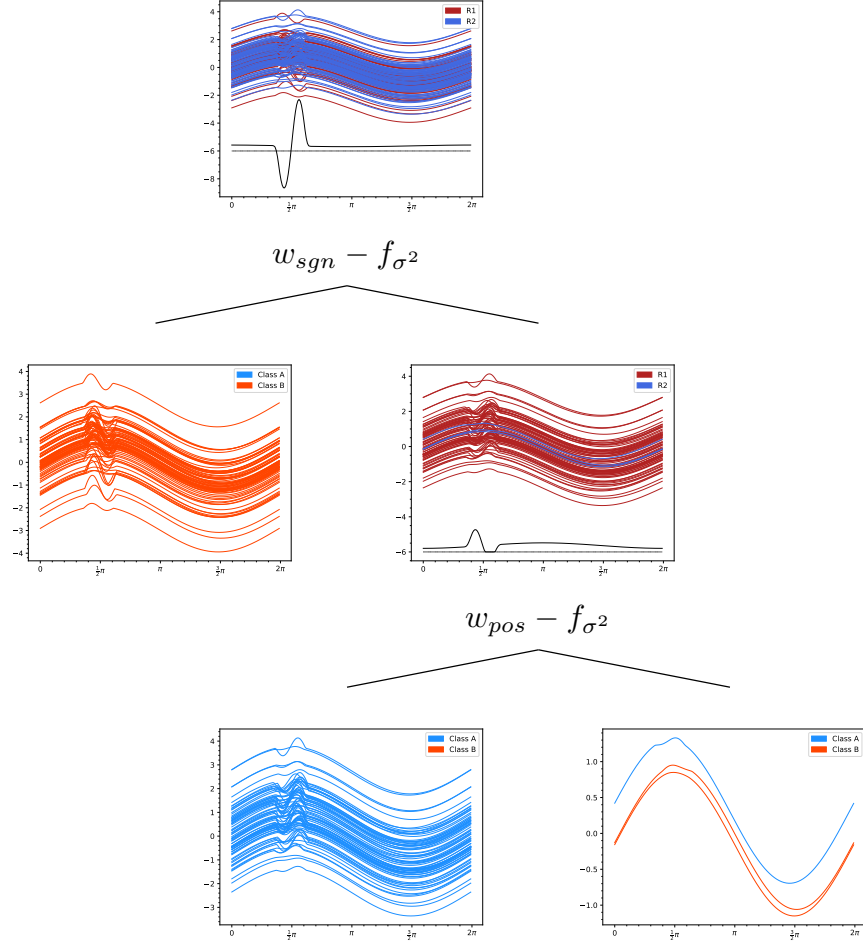
$w_{pos} - f_{\sigma^2}$
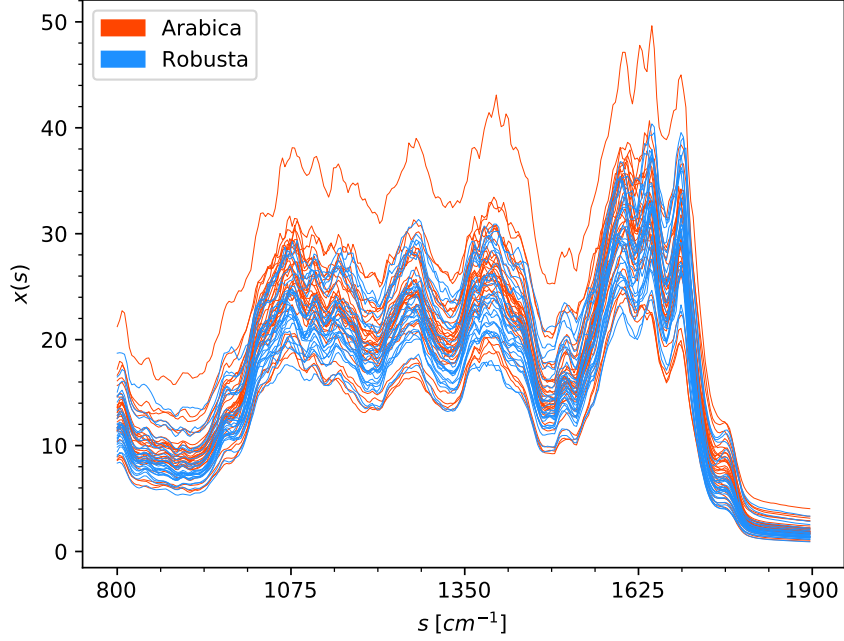
Figure 2: Simulation study: fully grown tree

Figure 3: FTIR spectra of coffee samples

4.1.2 Coffee FTIR Spectra

This dataset is composed of $N=56$ functional samples of length $p=286$ obtained by Fourier transform infrared (FTIR) spectroscopy with diffuse reflectance (DRIFT) sampling of dried instant coffee. The problem is to discriminate between two types of coffee, *Coffea Arabica* and *Coffea Canephora Robusta*, which have different trade values. The dataset is shown in Figure 3 and it is available at http://csr.quadram.ac.uk/wp-content/uploads/FTIRSpectraInstantC offee.zip. The raw data has no missing values and negligible sensor noise (Briandet et al., 1996), therefore there is no need to employ further smoothing. Table 2 shows the results and the average tree heights, where this time the FE preprocessing is more effective than the discretized function evaluations, with the weighted features of μCART providing higher accuracy and trees with a single internal node, as shown in Figure 4.

16

Table 2: Coffee FTIR Spectra results: classification accuracy and tree height

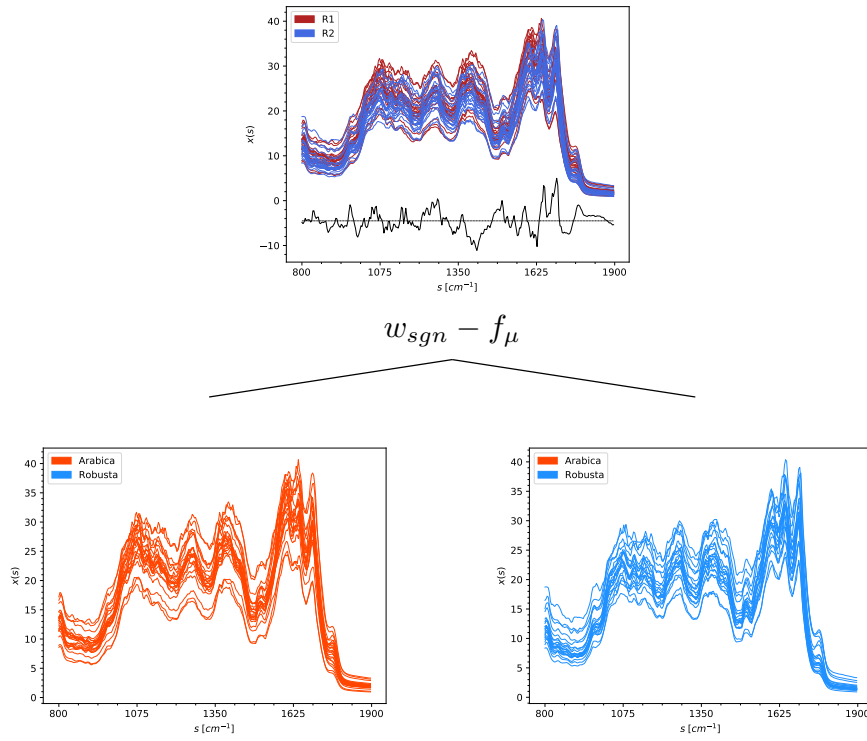|	Accuracy %	Tree Height
CART	70.4 ± 13	2.5 ± 1.1
CART+FE	82.9 ± 10	3 ± 0.7
RF	75.5 ± 14	$-$
RF+FE	87.3 ± 10	$-$
GBT	76.9 ± 10	$-$
GBT+FE	89.7 ± 7	$-$
μCART	94.3 ± 8	1.1 ± 0.1

$$w_{sgn} - f_\mu$$

Figure 4: Coffee FTIR Spectra: fully grown tree

4.1.3 Gut Microbiota DGGE

This second application is a multiclass classification problem that aims to discriminate between subjects with ulcerative colitis (UC), irritable bowel syndrome (IBS) or healthy controls, by analyzing their gut microbiota compositions, where different localized patterns seem to be associated with the different conditions. The input data is obtained through denaturing gradient gel electrophoresis (DGGE) of faecal samples, and the details of the data acquisition and processing are explained in Noor et al. (2010). The dataset is shown in Figure 5 and is composed of N=96 discretized functional samples of length p=731, presmoothed and without missing values, available at http://csr.quadram.ac.uk/wp-content/uploads/DGGE_Timepoint_1.zip. Table 3 shows the results, where for the ensemble methods the feature extraction approach is less effective, while locally learning the weights allows μCART to provide a higher score, especially if we consider the fact that this is a multiclass problem. It is also worth to note that the difference in height between the trees is lower, which could again be a consequence of how we deal with multiple classes while learning the measure in the nodes. The μCART full tree is depicted in Figure 6, where we can observe that the measure inside each decision node becomes more interpretable for deeper nodes. In particular, the deepest internal node selects w_{neg} leading to a sparse weight pattern (as previously noted, we take the absolute value of w_{neg}, therefore the baseline is zero and the weights are positive).

Table 3: Gut Microbiota DGGE results: classification accuracy and tree height

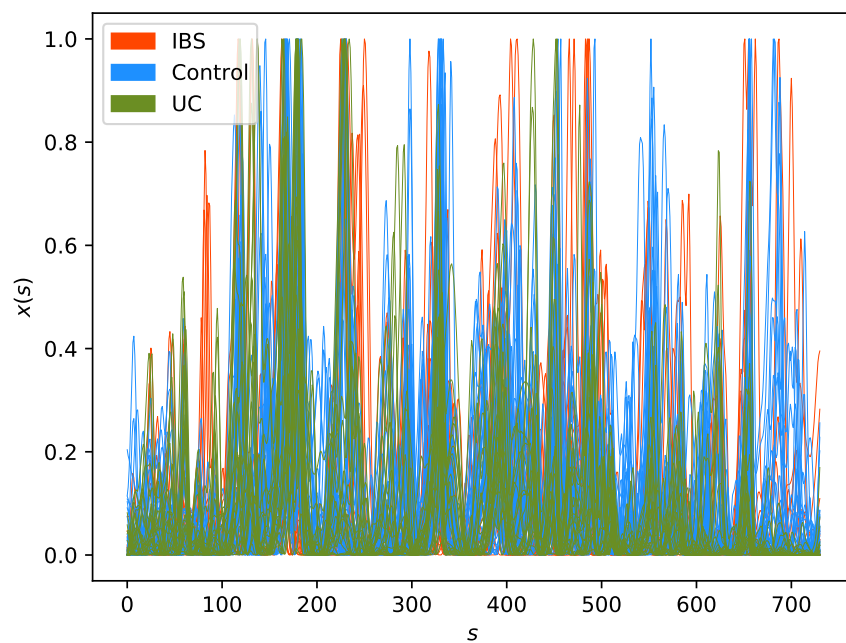|	Accuracy %	Tree Height
CART	55.3 ± 9	3.6 ± 1.3
CART+FE	58.5 ± 9	5 ± 1.6
RF	66 ± 11	$-$
RF+FE	64.4 ± 10	$-$
GBT	66 ± 10	$-$
GBT+FE	62.2 ± 11	$-$
μCART	71.2 ± 11	3.3 ± 0.9

Figure 5: Gut Microbiota DGGE by class: ulcerative colitis (UC), irritable bowel syndrome (IBS) and healthy controls
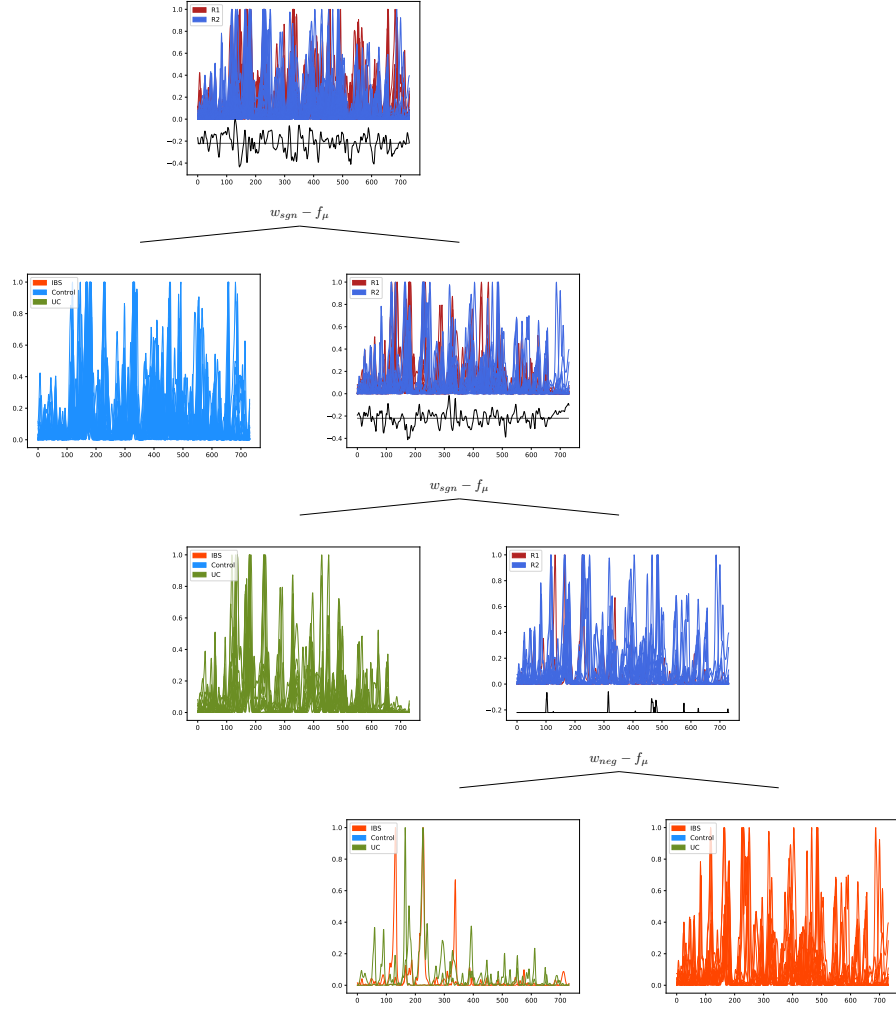
Figure 6: Gut Microbiota DGGE: fully grown tree

4.2 Regression

For the task of regression, we show two spectroscopy applications with the objective to predict the moisture content in either wheat or biscuit dough samples. We again control the growth of all trees by limiting the minimum number of samples required to be at a leaf node, but with mean-square error (mse) instead of Gini index for CART, μCART and RF, and least squares loss instead of deviance loss for GBT. The number of trees for the ensembles is still fixed to 150, with the same grid search scheme for chosing the maximum number of features for RF. Together with the average tree height, we report the mse on the test set with the corresponding standard deviation, averaged over five random repetitions of 5-fold cross-validation with 3-fold cross-validation for grid search hyperparameter optimization.

4.2.1 Wheat NIR Spectra

This dataset is a collection of near-infrared reflectance (NIR) spectra extracted from N=100 wheat samples, measured between $1100nm$ and $2500nm$ (Kalivas, 1997), while the response variable is the moisture content of the sample, with values over the whole dataset of 14.7 ± 1.37 ($\mu \pm \sigma$). The raw data is available in the CRAN package *fds* (Shang and Hyndman, 2018) and some curves present constant artifacts that may be the result of imputation of missing values. For this reason, we smooth the raw data individually by means of penalized free knots regression splines, with the strength of the penalization that is manually tuned in order to avoid over/under smoothing. This preprocessing is done once before fitting the trees, and therefore all methods are compared on the same data, obtained by evaluating the recovered functions on a grid of length p=150, as shown in Figure 7. Table 4 shows the results and the tree heights. Like in the previous case study, the chosen features are less effective than directly using all the function evaluations, suggesting that over the whole domain this approach is not favoured for this problem. On the contrary, by extracting those same features on a suitable weighted subset of the domain, μCART is able to provide the lowest mse at the price of growing more complex trees, as the difference in tree heights between the methods is significant. We omit reporting the fully grown μCART tree as it would not reasonably fit into a single page.
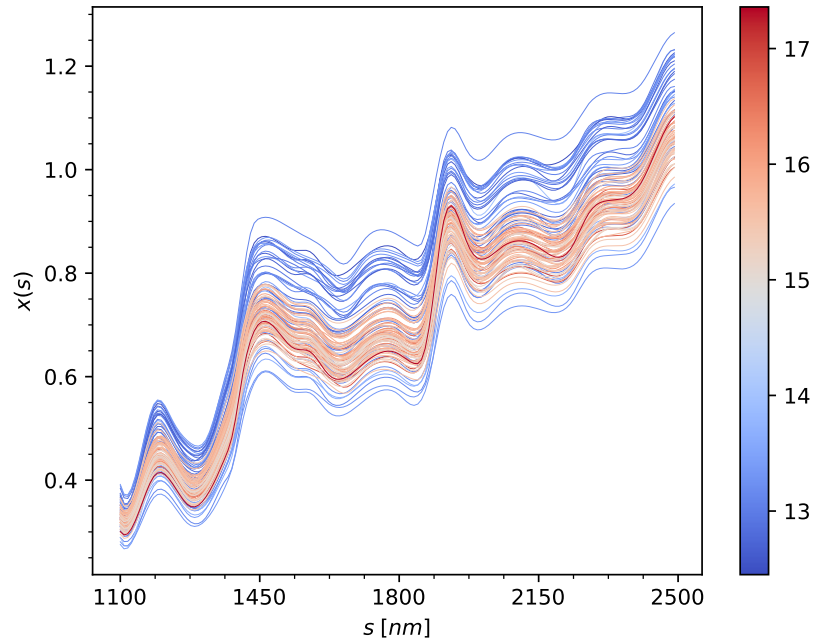
Figure 7: Wheat NIR spectra

Table 4: Wheat NIR spectra results: regression error (mse) and tree height

	mse	Tree Height
CART	$1.31 \pm .53$	4.4 ± 1.9
CART+FE	$1.57 \pm .54$	5.4 ± 1.9
RF	$0.95 \pm .46$	$-$
RF+FE	$1.33 \pm .48$	$-$
GBT	$0.93 \pm .37$	$-$
GBT+FE	$1.53 \pm .43$	$-$
μCART	$0.70 \pm .54$	7.1 ± 0.5

4.2.2 Biscuit Dough NIR Spectra

The second regression application deals with predicting the moisture content of biscuit dough (14.2 ± 1.47), from NIR spectra measured between $1100nm$ and $2498nm$. The raw data is also available in the CRAN package *fds* (Shang and Hyndman, 2018), and is comprised of N=72 curves evaluated in p=700 equi-spaced wavelengths, without missing values or conspicuous noise, as shown in Figure 8. This dataset was originally studied in a Bayesian wavelet framework (Brown et al., 2001), with a different preprocessing in order to reduce compute time, we instead use the evaluations on the whole domain. Table 5 shows the results with the corresponding tree heights. The relative errors between the methods and preprocessing approaches are similar to the previous case study, suggesting that locally weighted features are an effective approach for moisture prediction in the $1100nm$-$2498nm$ band. The tree heights are instead comparable between the methods, with Figure 9 showing the fitted μCART tree. As observed in the gut microbiota application, the learned representations become more interpretable with the depth of the nodes, and in particular we find a similarly sparse pattern in the right branch, while the left one selects the uniform weights with the cosine distance as feature extractor.
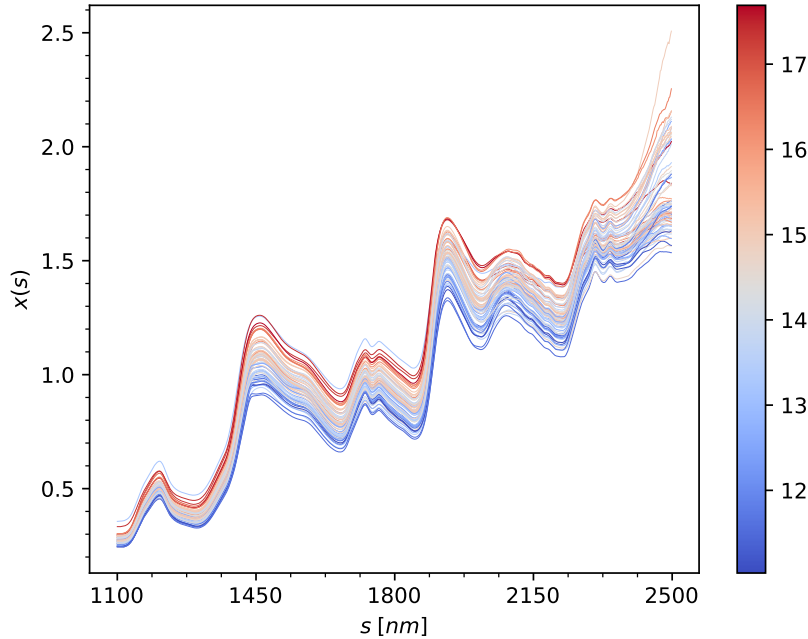

Figure 8: Biscuit dough NIR spectra

Table 5: Biscuit dough NIR spectra results: regression error (mse) and tree height

	mse	Tree Height
CART	$1.24 \pm .49$	3.4 ± 1.9
CART+FE	$1.58 \pm .52$	3.6 ± 1.4
RF	$0.90 \pm .41$	$-$
RF+FE	$1.32 \pm .42$	$-$
GBT	$1.07 \pm .49$	$-$
GBT+FE	$1.52 \pm .49$	$-$
μCART	$0.74 \pm .38$	3.8 ± 0.4

$w_{sgn} - f_{\mu}$

$w_{sgn} - f_{\sigma^2}$

$w_{sgn} - f_{\mu}$

$w_{uni} - f_{\cos\theta}$
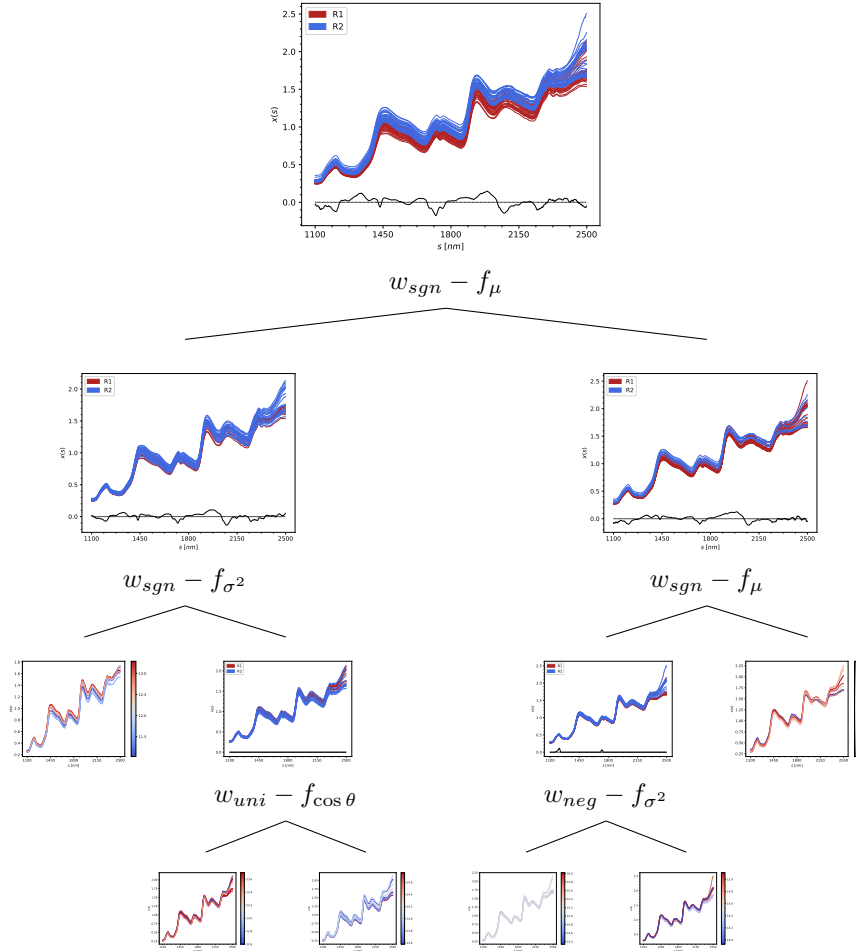
$w_{neg} - f_{\sigma^2}$

Figure 9: Biscuit dough NIR spectra: fully grown tree

5 Conclusion

In this work we have proposed a novel nonparametric tree-based tool for classification and regression in the context of functional and high dimensional smooth data. In particular, we introduced a set of splitting rules that are able to leverage the functional data model at the node level, while at the same time maintaining the capability of handling multiple and heterogeneous data sources, as the growing procedure fits within the known CART greedy algorithm. By leveraging the concept of weighted L^2 spaces, it is possible to learn meaningful representations for each internal node of the tree, extracting locally informative features that allow for an interpretable method with low generalization error and variable selection properties. The input functions are therefore seen as a single unit, for instead of relying on the single evaluations to perform the split, we select between a set of user defined adaptively weighted features. Our splitting rule can be seen as a specific instance of oblique splitting, and given that a direct optimization of the node is intractable, we proposed to learn the split by means of constrained convex optimization with interior point methods, modeling the linear combination as the weight function that induces the measure of the weighted L^2 space in the current node. The effectiveness of our method has been shown in both classification and regression tasks with one or multiple scalar-valued functional covariates defined on unidimensional domains, through simulation studies and real world spectrometry applications. In particular, when the chosen features are in fact informative for the task at hand, our method can have lower error than ensemble methods like random forest and gradient boosted trees. As our work focused only on simpler functional covariates, further extensions can be developed for more complex data like vector-valued functions defined on multidimensional domains. Other interesting aspects could be bagging and randomized selection of the splitting variable/feature extractor.

References

A. M. Alonso, D. Casado, and J. Romo. Supervised classification for functional data: A weighted distance approach. *Computational Statistics & Data Analysis*, 56(1):2334–2346, 2012.

S. Balakrishnan and D. Madigan. Decision trees for functional variables. In *International Conference on Data Mining*, 2006.

K. P. Bennett and O. Mangasarian. Robust linear programming discrimination of two linearly inseparable sets. *Optimization Methods and Software*, 1(1): 23–34, 1992.

K. P. Bennett, N. Cristianini, J. Shawe-Taylor, and D. Wu. Enlarging the margins in perceptron decision trees. *Machine Learning*, 41(3):295–313, 2000.

D. Bertsimas and J. Dunn. Optimal classification trees. *Machine Learning*, 106 (7):1039–1082, 2017.

L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and regression trees*. Wadsworth and Brooks, Belmont, CA, USA, 1984.

R. Briandet, E. K. Kemsley, and R. H. Wilson. Discrimination of arabica and robusta in instant coffee by fourier transform infrared spectroscopy and chemometrics. *Journal of Agricultural and Food Chemistry*, 44(1):170–174, 1996.

C. E. Brodley and P. E. Utgoff. Multivariate decision trees. *Machine Learning*, 19(1):45–77, 1995.

P. J. Brown, T. Fearn, and M. Vannucci. Bayesian wavelet regression on curves with application to a spectroscopic calibration problem. *Journal of the American Statistical Association*, 96(454):398–408, 2001.

S. R. Bulò and P. Kontschieder. Neural decision forests for semantic image labelling. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

E. Cantú-Paz and C. Kamath. Inducing oblique decision trees with evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 7(1):54–68, 2003.

H. Cardot, F. Ferraty, and P. Sarda. Spline estimators for the functional linear model. *Statistica Sinica*, 13(3):571–591, 2003.

H. Cardot, C. Crambes, A. Kneip, and P. Sarda. Smoothing splines estimators in functional linear regression with errors-in-variables. *Computational Statistics & Data Analysis*, 51(10):4832–4848, 2007.

R. Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

P. Chaudhuri and W.-Y. Loh. Nonparametric estimation of conditional quantiles using quantile regression trees. *Bernoulli*, 8(5):561–576, 2002.

C. Cousins and M. Riondato. CaDET: interpretable parametric conditional density estimation with decision trees and forests. *Machine Learning*, 108 (8-9):1613–1634, 2019.

C. Crambes, A. Kneip, and P. Sarda. Smoothing splines estimators for functional linear regression. *The Annals of Statistics*, 37(1):35–72, 2009.

A. Douzal-Chouakria and C. Amblard. Classification trees for time series. *Pattern Recognition*, 45(3):1076–1091, 2012.

I. Epifanio. Shape descriptors for classification of functional data. *Technometrics*, 50(3):284–294, 2008.

F. Ferraty and P. Vieu. *Nonparametric Functional Data Analysis: Theory and Practice*. Springer, New York, NY, USA, 2006.

J. H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232, 2001.

J. Fürnkranz, T. Kliegr, and H. Paulheim. On cognitive preferences and the plausibility of rule-based models. *Machine Learning*, 109(4):853–898, 2020.

P. Galeano, E. Joseph, and R. E. Lillo. The mahalanobis distance for functional data with applications to classification. *Technometrics*, 57(2):281–291, 2015.

J. Gama. Discriminant trees. In *International Conference on Machine Learning*, 1999.

J. Gama. Functional trees. *Machine Learning*, 55(3):219–250, 2004.

J. Gondzio. Interior point methods 25 years later. *European Journal of Operational Research*, 218(3):587–601, 2012.

W. E. Hart, C. D. Laird, J.-P. Watson, D. L. Woodruff, G. A. Hackebeil, B. L. Nicholson, and J. D. Siirola. *Pyomo–optimization modeling in python*. Springer Science & Business Media, New York, NY, USA, 2017.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, New York, NY, USA, 2009.

D. Heath, S. Kasif, and S. Salzberg. Induction of oblique decision trees. In *International Joint Conference on Artificial Intelligence*, 1993.

L. Horváth and P. Kokoszka. *Inference for functional data with applications*. Springer, New York, NY, USA, 2012.

T. Hothorn, K. Hornik, and A. Zeileis. Unbiased recursive partitioning: A conditional inference framework. *Journal of Computational and Graphical Statistics*, 15(3):651–674, 2006.

G. M. James, T. J. Hastie, and C. A. Sugar. Principal component models for sparse functional data. *Biometrika*, 87(3):587–602, 2000.

M. I. Jordan and R. A. Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural Computation*, 6(2):181–214, 1994.

J. H. Kalivas. Two data sets of near infrared spectra. *Chemometrics and Intelligent Laboratory Systems*, 37(2):255–259, 1997.

H. Kim and W.-Y. Loh. Classification trees with unbiased multiway splits. *Journal of the American Statistical Association*, 96(454):589–604, 2001.

P. Kontschieder, M. Fiterau, A. Criminisi, and S. R. Bulò. Deep neural decision forests. In *IEEE International Conference on Computer Vision*, 2015.

N. Landwehr, M. Hall, and E. Frank. Logistic model trees. *Machine Learning*, 59(1-2):161–205, 2005.

T.-H. Lee and Y.-S. Shih. Unbiased variable selection for classification trees with multivariate responses. *Computational Statistics & Data Analysis*, 51 (2):659–667, 2006.

B. Li and Q. Yu. Classification of functional data: A segmentation approach. *Computational Statistics & Data Analysis*, 52(1):4790–4800, 2008.

W. Liu and I. W. Tsang. Making decision trees feasible in ultrahigh feature and label dimensions. *Journal of Machine Learning Research*, 18(81):1–36, 2017.

W.-Y. Loh. Regression trees with unbiased variable selection and interaction detection. *Statistica Sinica*, 12(2):361–386, 2002.

W.-Y. Loh. Improving the precision of classification trees. *Annals of Applied Statistics*, 3(4):1710–1737, 2009.

W.-Y. Loh and Y.-S. Shih. Split selection methods for classification trees. *Statistica Sinica*, 7(4):815–840, 1997.

W.-Y. Loh and N. Vanichsetakul. Tree-structured classification via generalized discriminant analysis. *Journal of the American Statistical Association*, 83 (403):715–725, 1988.

W.-Y. Loh and W. Zheng. Regression trees for longitudinal and multiresponse data. *The Annals of Applied Statistics*, 7(1):495–522, 2013.

J. K. Martin. An exact probability metric for decision tree splitting and stopping. *Machine Learning*, 28(2-3):257–291, 1997.

N. Meinshausen. Quantile regression forests. *Journal of Machine Learning Research*, 7(Jun):983–999, 2006.

N. Meinshausen. Sign-constrained least squares estimation for high-dimensional regression. *Electronic Journal of Statistics*, 7(1):1607–1631, 2013.

A. Möller and J. Gertheiss. A classification tree for functional data. In *International Workshop on Statistical Modeling*, 2018.

A. Möller, G. Tutz, and J. Gertheiss. Random forests for functional covariates. *Journal of Chemometrics*, 30(1):715–725, 2016.

S. K. Murthy, S. Kasif, and S. Salzberg. A system for induction of oblique decision trees. *Journal of Artificial Intelligence Research*, 2(1):1–32, 1994.

D. Nerini and B. Ghattas. Classifying densities using functional regression trees: Applications in oceanology. *Computational Statistics & Data Analysis*, 51(10): 4984–4993, 2007.

H. G. Noh, M. S. Song, and S. H. Park. An unbiased method for constructing multilabel classification trees. *Computational Statistics & Data Analysis*, 47 (1):149–164, 2004.

S. O. Noor, K. Ridgway, L. Scovell, E. K. Kemsley, E. K. Lund, C. Jamieson, I. T. Johnson, and A. Narbad. Ulcerative colitis and irritable bowel patients exhibit distinct abnormalities of the gut microbiota. *BMC Gastroenterology*, 10(134), 2010.

M. Norouzi, M. D. Collins, M. Johnson, D. J. Fleet, and P. Kohli. Efficient non-greedy optimization of decision trees. In *Advances in Neural Information Processing Systems*, 2015.

A. Pande, L. Li, J. Rajeswaran, J. Ehrlinger, U. B. Kogalur, E. H. Blackstone, and H. Ishwaran. Boosted multivariate trees for longitudinal data. *Machine Learning*, 106(2):277–305, 2017.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct): 2825–2830, 2011.

J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

J. O. Ramsay and B. W. Silverman. *Functional Data Analysis*. Springer, New York, NY, USA, 2005.

F. Rossi and N. Villa. Support vector machine for functional data classification. *Neurocomputing*, 69(1):730–742, 2006.

L. M. Sangalli, P. Secchi, S. Vantini, and V. Vitelli. k-mean alignment for curve clustering. *Computational Statistics & Data Analysis*, 54(5):1219–1233, 2010.

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

H. L. Shang and R. J. Hyndman. fds: Functional data sets. https://CRAN.R -project.org/package=fds, 2018. R package version 1.8.

M. Slawski and M. Hein. Non-negative least squares for high-dimensional linear models: consistency and sparse recovery without regularization. *Electronic Journal of Statistics*, 7(1):3004–3056, 2013.

P. Smyth, A. Gray, and U. M. Fayyad. Retrofitting decision tree classifiers using kernel density estimation. In *International Conference on Machine Learning*, 1995.

C. Strobl, A.-L. Boulesteix, and T. Augustin. Unbiased split selection for classification trees based on the gini index. *Computational Statistics & Data Analysis*, 52(1):483–501, 2007.

R. Tibshirani and T. Hastie. Margin trees for high-dimensional classification. *Journal of Machine Learning Research*, 8(Mar):637–652, 2007.

L. Torgo. Functional models for regression tree leaves. In *International Conference on Machine Learning*, 1997.

C. Vens, J. Struyf, L. Schietgat, S. Džeroski, and H. Blockeel. Decision trees for hierarchical multi-label classification. *Machine Learning*, 73(2):185–214, 2008.

A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–37, 2006.

Y. Yamada, E. Suzuki, H. Yokoi, and K. Takabayashi. Decision-tree induction from time-series data based on a standard-example split test. In *International Conference on Machine Learning*, 2003.

F. Yao, H. Müller, and J. Wang. Functional data analysis for sparse longitudinal data. *Journal of the American Statistical Association*, 100(470):577–590, 2005.

Y. Yu and D. Lambert. Fitting trees to functional data, with an application to time-of-day patterns. *Journal of Computational and Graphical Statistics*, 8 (4):749–762, 1999.

M. Yuan and T. T. Cai. A reproducing kernel Hilbert space approach to functional linear regression. *The Annals of Statistics*, 38(6):3412–3444, 2010.

H. Zhang. Classification trees for multiple binary responses. *Journal of the American Statistical Association*, 93(441):180–193, 1998.

MOX Technical Reports, last issues

Dipartimento di Matematica
Politecnico di Milano, Via Bonardi 9 - 20133 Milano (Italy)

67/2020 Caramenti, L.; Menafoglio, A.; Sgobba, S.; Lanzano, G.
Multi-Source Geographically Weighted Regression for Regionalized Ground-Motion Models

68/2020 Galvani, M.; Torti, A.; Menafoglio, A.; Vantini S.
FunCC: a new bi-clustering algorithm for functional data with misalignment

69/2020 Galvani, M.; Torti, A.; Menafoglio, A.; Vantini S.
FunCC: a new bi-clustering algorithm for functional data with misalignment

70/2020 Belli E.
Smoothly Adaptively Centered Ridge Estimator

71/2020 Belli E; Vantini S.
Ridge regression with adaptive additive rectangles and other piecewise functional templates

66/2020 Didkovsky, O.; Ivanov, V.; Papini, M.; Longoni, L.; Menafoglio, A.
A comparison between machine learning and functional geostatistics approaches for data-driven analyses of solid transport in a pre-Alpine stream

65/2020 Di Gregorio, S.; Vergara, C.; Montino Pelagi, G.; Baggiano, A.; Zunino, P.; Guglielmo, M.; Fu
Prediction of myocardial blow flow under stress conditions by means of a computational model

64/2020 Fiz, F.; Viganò, L.; Gennaro, N.; Costa, G.; La Bella, L.; Boichuk A.; Cavinato, L.; Sollini, M.;
Radiomics of Liver Metastases: A Systematic Review

63/2020 Tuveri, M.; Milani, E.; Marchegiani, G.; Landoni, L.; Torresani, E.; Capelli, P.; Sperandio, N.;
HEMODYNAMICS AND REMODELING OF THE PORTAL CONFLUENCE IN PATIENTS WITH CANCER OF THE PANCREATIC HEAD: A PILOT STUDY

62/2020 Massi, M. C.; Ieva, F.
Representation Learning Methods for EEG Cross-Subject Channel Selection and Trial Classification